ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 5, May 2018

High Performance Packet Capturing Using Data Plane Development Kit (DPDK)

Sonali Argade[1], Hardika Shah[2], Niranjan Kulkarni[3], Sayali Divekar[4]

Department of Computer Engineering, RMDSSOE, Pune, India

ABSTRACT: Huge packets drop, more time required to capture the packets, frequent system calls, etc. are the major obstacles in fast packet capturing. There are many available packet capturing frameworks in the market, but all of them have these as major problems. In data plane packets are transferred to user level through kernel space. The Data Plane Development Kit (DPDK) software reduces these problems by bypassing the kernel level. High performance packet capturing tool (PackCap) is proposed here, which enhances the performance characteristics of packet capturing by increasing speed at 1 GBps. With PackCap packets are captured with the maximum speed and with less packet drops. PackCap ports TCPdump on DPDK by socket creation between the two. The kernel level is bypassed in this PackCap tool. PacCap only considers data plane packets, other packets like control plane packets are not considered. PackCap can be used by network administrator to inspect packets, to solve network problems and to determine whether network security policies are being followed.

KEYWORDS: DPDK, EAL, NFV, TCPdump

I. INTRODUCTION

Packet capturing tools are used over worldwide for security purposes. Packet capturing tools are used greatly now-a-days to know how the packets are traversed in the network. Recent advances of server-virtualization technology and cloud computing technology allows many services, such as web, database, and application, run with virtualized computing resources in a data center[7]. Virtual machines need to communicate and access peripherals, which for systems used as servers mostly means disks and network interfaces[5].The network interface (NIC) is normally able to manage circular lists (called NIC rings) of memory buffers, and move packets between the physical links and these buffers without CPU intervention.[6] Packet processing at a switch consists of: (a) packet reception, (b) packet header parsing, (c) packet classification, (d) flow-table lookup to determine outgoing port, (e) QoS mechanisms such as policing on the ingress and scheduling on the egress, and (f) packet transmission [10]. The primary goal of Network Function Virtualization (NFV) is the migration of physical network functions to software versions running on virtual machines (VM) in cloud computing environments[13].

II. RELATED WORK

In the 2008 Nick McKeown, et al [1] proposed the Run experiments on heterogeneous switches and no need for vendors to expose but reuse of controllers is avoided.In the year 2009 Steen Larsen, et al [2] proposed End to end latency and Adaptive interrupt moderation but Variability and I/O performance on virtual machine is limited.
In 2010 Binbin Zhang, et al [3] developed I/O performance and Optimization of Kernel Virtual Machine but guest OS is simple.
In the year 2010 Sangjin Han, et al [4] proposed that the system Outperforms software routers by more than a factor of four and Minimized per-packet processing overhead but the system is not available to public.
In 2012 L. Rizzo, et al [5] developed the system with highspeed communication but not compatible with better hypervisor.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 5, May 2018

In the year 2014 Y. Nakajima, et al [7] proposed that use of DPDK accelerates network I/O performance by bypassing packet processing in OS kernel and direct access to NIC packet buffer from data-plane program but in case of long packet, the switch performs 10-Gbps wirerate.
In 2015 Michio Honda, et al [9] stated that A flexible software switch providing high throughput, low CPU utilization and which can switch packets at rates of hundreds of gigabits per second but it Receives per packet semantics instead of batches of packet decreases the performance. In the year 2015 Tom Barbette, et al [10] proposed that DPDK FastClick implementations removes the important overhead for small packets, mostly by removing the Click wiring cost by using batches and reducing the cost of the packet pool, using I/O batching and FastClick improved abstraction in packet processing but the flow capabilities of fastClick is not good for the development of Middleboxes.
In 2015 Sebastian Gallenmüller, et al [11] stated that DPDK seems to be superior to netmap. It uses well known OS interfaces and modified system calls for packet IO, leading to increased performance but the high data rate can be achieved by modifying the interfaces he limitation is that this system requires appplication to be ported to one of the frameworks.

In the year 2016 Reza Rahimi, et al [14] proposed that loop-count variable is used to control packet batching. Packet drop rate could be non-zero when the OpenFlowtable size could be large but there is need for careful calibration andplanning parallelization and a software switch should be calibrated for the highest packet arrival rate that it can handle with zero packet drops.The authors of the accepted manuscripts will be given a copyright form and the form should accompany your final submission.

III. PRESENT SYSTEM

The input to the PackCap tool is the data plane packets in the network. Packets are captured by the TCPdump through IPstack and using libpcap libraries. TCPdump is ported on the interface of DPDK i.e dpdk0 interface. With the help of the socket created between DPDK and the TCPdump, packets are then transferred to DPDK where these packets are directly entered in user space and the kernel level is bypassed. This approach reduces frequent system calls, context switching between processes, etc. Using PackCap, packets are captured at the speed of 1 GBps.

IV. ALGORITHM APPROACH

Algorithm 1:

pcap_loop(pcap_t,*p,intcnt, pcap_handler callback, u_char *user)

1.Use integer to store register
2.use for loop
3. if (p->rfile != NULL)
4. n = pcap_offline_read(p, cnt, callback, user);
}
Else
5. Use do while
6. register = p->read_op(p, cnt, callback, user);
7.while (register == 0);
8. if (register <= 0)
9. return (register);
10. if (!PACKET_COUNT_IS_UNLIMITED(cnt))
11. cnt - = register;
12. if (cnt<= 0)
13. return (0);

Algorithm 2:

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 5, May 2018

pcap_read_dpdk_mmap_v2(pcap_t*handle, intmax_packets, pcap_handlercallback,u_char *user)

1. Use structure structpcap_dpdk*handlep = handle->priv;
2. union thdr h;
3. Define packets as 0.
4. return packets
5. h.raw = RING_GET_CURRENT_FRAME(handle);
   use if loop
6. if (h.h2 ->tp_status == TP_STATUS_KERNEL)
7. 7.ret = pcap_wait_for_frames_mmap(handle);
8. if (ret)
9. return ret;

V. SYSTEM ARCHITECTURE

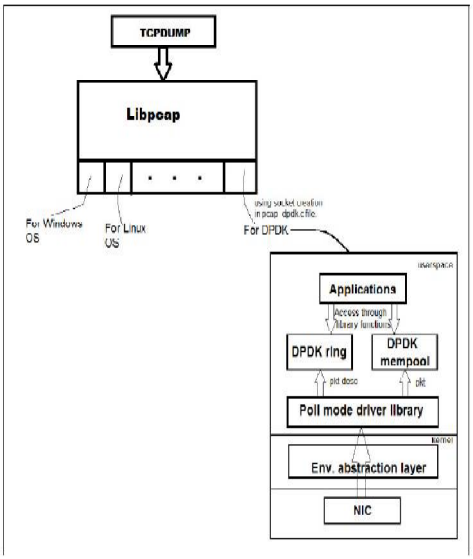

Fig 1: System Architecture

VI. PRESENT SYSTEM WORKING

The DPDK structure makes a course of action of libraries for specific programming circumstances through the arrangement of an Environment Abstraction Layer (EAL).The EAL covers the characteristic specific and gives a standard programming interface to libraries, open hardware stimulating operators and other hardware and working system (Linux, FreeBSD) segments. Once the EAL is made for a specific area, originators associate with the library to make their applications. For instance, EAL gives the frameworks to help Linux, FreeBSD, Intel IA-32 or 64-bit, ARM 32-or 64-bit.The EAL in like manner gives additional organizations including time references, non particular transport get to, take after and research limits and alert operations.The DPDK executes a low overhead rushed to-completing model for brisk data plane execution and gets to devices through looking over to take out the execution overhead of meddle with processing.The DPDK is moreover amid the time spent including the event based programming model for fast data plane processing.The DPDK similarly joins programming representations that element best practices for programming

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 5, May 2018

building, tips for data structure layout and limit, application profiling and execution tuning utilities and tips that address ordinary framework execution inadequacies.

VII. WORK FLOW DIAGRAM

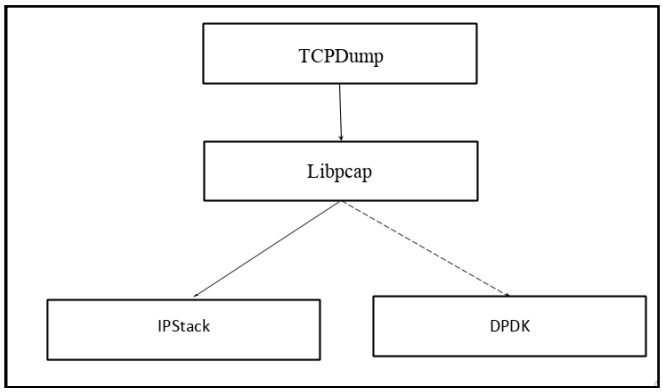

Fig 2: Work flow

VIII. EXPERIMENTAL RESULT

As we have done our venture our outcome can be stated as usage of bundle catching utilizing DPDK instrument with parcel sending in NFV. What's more, result comes to catching of parcels by DPDK structure fruitful execution of Tcp-dump and zero bundle drop.

IX. CONCLUSION

The paper describes quick CPU is essential for productive and unsurprising execution. DPDK is best execution unsurprising methodology for bundle sending in a network.SR-IOV ought to be stayed away from underway systems in light of the execution qualities. Bundle sending execution of NFV-hubs is under the consideration particularly in the fields of transporter and datacenter systems where the idea of NFVhas just been sent. NFV acquires bunches of advantages in system and administration, yet low and shaky execution nature of NFV-hubs anticipates full-scale sending of NFV. Incalculable execution change strategies have been proposed up until now, and system specialists and heads have different choices for NFV-hub frameworks.

REFERENCES

[1] Nick McKeown, Tom Anderson, HariBalakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner, "OpenFlow: Enabling Innovation in Campus Networks", ACM SIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 69-74, April 2008.
[2] Steen Larsen, ParthasarathySarangam, Ram Huggahalli, "Architectural Breakdown of End-to-End Latency in a TCP/IP Network", IEEE, 2009.
[3] Binbin Zhang, Xiaolin Wang, Rongfeng Lai, Liang Yang, ZhenlinWang,YingweiLuo, Xiaoming Li, "Evaluating and Optimizing IOVirtualization in Kernel-based Virtual Machine (KVM)", IFIInternational Conference on Network and Parallel Computing (NPC), pp. 220–231, 2010.
[4] Sangjin Han, Keon Jang, KyoungSoo Park, Sue Moon, "PacketShader: a GPU-Accelerated Software Router", ACM SIGCOMM 2010, pp. 195–206, Delhi, India, Sep. 2010.
[5] L. Rizzo and G. Lettieri, "VALE, a Switched Ethernet for. Virtual Machines", 8th International Conference on emerging Networking Experiments and Technologies (CoNEXT'12), pp. 61–72, 2012.
[6] L. Rizzo, M. Carbone, and G. Gatalli, "Transparent acceleration of software packet forwarding using netmap", IEEE INFOCOM, pp. 2471–2479, 2012.
[7] Y. Nakajima, T. Hibi, H. Takahashi, H. Masutani, K. Shimano, and M. Fukui, "Scalable, High-performance, Elastic Software OpenFlow Switch in Userspace for Wide-area Network", Open Networking Summit (ONS 2014), Santa Clara, CA, March 2014.

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 7, Issue 5, May 2018

[8] R. Bonafiglia, I. Cerrato, F. Ciaccia, M. Nemirovsky, F. Risso, "Assessing the Performance of Virtualization Technologies for NFV: a Preliminary Benchmarking.", Fourth European Workshop on Software Defined Networks (EWSDN), pp. 67–72, Bilbao, Spain, 2015.
[9] MichioHonda , Felipe Huici, Giuseppe Lettieri, Luigi Rizzo, "mSwitch: A Highly-Scalable, Modular Software Switch", ACM SIGCOMM Symposium on Software Defined Networking Research (SOSR), June 2015.
[10] Tom Barbette, Cyril Soldani, Laurent Mathy, "Fast Userspace Packet Processing", ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), pp. 5–16, Oakland, CA, May 2015.
[11] Sebastian Gallenmuller, Paul Emmerich, Florian Wohlfart, Daniel Raumer, and Georg Carle, "Comparison of Frameworks for High-Performance Packet IO", ACM/IEEE Symposium on Architectures for Networking and Commu- nications Systems (ANCS), pp. 29–38, Oakland, CA, May 2015.
[12] Paul Emmerich, Sebastian Gallenmuller, Daniel Raumer, Florian Wohlfart, and Georg Carle, "MoonGen: A Scriptable High-Speed Packet Generator", ACM Conference on Internet Measurement Conference (IMC), pp. 275–287, 2015.
[13] Michail-AlexandrosKourtis,GorgiosXilouris,VincenzoRiccobene,MichaelJ.Mcgarth, Giuseppe Petralia, "Enhancing VNF performance by exploiting SR –IOV and DPDK Packet Processing Acceleration", IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), pp. 74–78, San Francisco Nov. 2015.
[14] Reza Rahimi& M. Veeraraghavan ,Y. Nakajima & H. Takahashi Y. Nakajima, S. Okamoto & N. Yamanaka, "A High-Performance OpenFlow Software Switch", IEEE 17th International Conference on High Performance Switching and Routing (HPSR), pp. 93–99, June 2016.
[15] Ryota Kawashima, Tsunemasa Hayashi, Hiroshi Matsuo, "Evaluation of Forwarding Efficiency in NFV-nodes toward Predictable Service Chain Performance", IEEE transaction of network services and management, 2017.