Unveiling Performance of NFV Software Dataplanes

Zhixiong Niu
City Univeristy of Hong Kong

Hong Xu
City Univeristy of Hong Kong

Libin Liu
City Univeristy of Hong Kong

Yongqiang Tian
City University of Hong Kong

Peng Wang
City Univeristy of Hong Kong

Zhenhua Li
Tsinghua University

ABSTRACT

The key technology of NFV is software dateplane, which has attracted much attention in both academia and industry recently. Yet, in practice, there is very little understanding about its performance till now. We make a comprehensive measurement study of NFV software dataplanes in terms of packet processing throughput and latency, the most fundamental performance metrics. Specifically, we compare two state-of-the-art open-source NFV dataplanes, BESS and ClickOS, using commodity 10GbE NICs under various typical workloads. Our key observations are that (1) both dataplanes have performance issues processing small (≤128B) packets; (2) it is not always the best to colocate all VMs of a service chain on one server due to NUMA effect. We propose resource allocation strategies to remedy the problems, including carefully adding vNIC queues and CPU cores to vNFs, and distributing VNFs of a service chain to separate servers. To essentially address these problems and scale their performance, software dataplanes need to improve the support for NIC queues and multiple cores.

CCS CONCEPTS

• **Networks → Network performance analysis**; **Network measurement**;

KEYWORDS

NFV; Measurement; Software Dataplanes; DPDK

ACM Reference Format:
Zhixiong Niu, Hong Xu, Libin Liu, Yongqiang Tian, Peng Wang, and Zhenhua Li. 2017. Unveiling Performance of NFV Software Dataplanes. In *CAN '17: Cloud-Assisted Networking Workshop, December 12, 2017, Incheon, Republic of Korea.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3155921.3158430

1 INTRODUCTION

Middleboxes are ubiquitous in today's networks and provide important network functions to operators [18]. Traditionally, middleboxes are deployed as dedicated proprietary hardware. Network function virtualization (NFV) is now emerging to replace hardware boxes with virtual software instances running on commodity servers. NFV holds great promises to improve flexibility and efficiency of network function management. Thus it is quickly gaining momentum in the industry [4].

A key enabling technology to NFV is software dataplane. It provides a virtualized platform for hosting software middleboxes with high performance packet I/O to userspace. A number of NFV dataplanes have been developed recently. For example BESS [6] and NetVM [8] use KVM for virtualization and Intel DPDK for packet I/O. ClickOS [12] relies on Xen and netmap [15] instead for virtualization and packet I/O, respectively.

Despite the progress, there is a lack of understanding on NFV dataplane performance in the community.

A software dataplane desires high-performance packet processing, flexible programming interfaces, security/isolation between colocating VNFs, and so forth [8, 12]. Among these performance metrics, packet processing capability and latency are fundamental and critical since they determine the basic usability of a software dataplane. Hence it is the focus of our study. Specifically, we ask the questions, *how well do these software dataplanes perform packet processing in practice? More importantly, what are the areas of improvement to make software dataplanes be more competent against hardware so they can be widely adopted by the industry?*

Existing work and their evaluation do not address these questions well. Most work (e.g. BESS [6]) focuses on raw packet I/O without software middleboxes running as VMs on top. Some (e.g. ClickOS [12]) report performance of different software middleboxes only when deployed individually. More importantly, no performance comparison is done across NFV dataplanes under the same environment. Thus, it is unclear whether these NFV dataplanes can achieve line rate with different packet processing logic in software middleboxes, what their bottlenecks are if any, what the latency would look like, and how they would perform against each other in various settings such as NF chaining and colocation.

In this paper, we present the first measurement study of NFV dataplanes that provides answers to the above questions. We strategically choose two popular open source NFV dataplanes, BESS [6] and ClickOS [12], that differ widely in virtualization and packet I/O technologies. As a first step we focus on their packet processing throughput and latency running on commodity servers and 10GbE NICs. We use two basic virtual network functions (VNFs): L3 forwarding and firewall.

Our measurements reveal several major findings:

(1) Both BESS and ClickOS can achieve line rate with medium to large packets (>128B), even when CPU is clocked down to 1.2GHz. In a practical setting with mixed packet sizes and low utilization, both dataplanes can handle typical traffic.
(2) Both dataplanes cannot achieve line rate processing small packets (≤ 128B) by a 2.6GHz CPU core. We observe that

CAN '17, December 12, 2017, Incheon, Republic of Korea

ACM ISBN 978-1-4503-5423-3/17/12…$15.00
https://doi.org/10.1145/3155921.3158430

adding more vNIC (queues) and correspondingly more vCPUs achieves line rate for 64B packets. For ClickOS we believe the bottleneck is its high CPU usage, which cannot be resolved due to its lack of SMP support.

(3) Performance also degrades in the NF chaining scenario, with ClickOS being more sensitive to chain length. Perhaps surprisingly, placing all VNFs of the chain on the same server does not necessarily lead to best performance, because the NUMA effect may further degrade performance when there are too many VNFs to be put to the same CPU socket. In this case simply assigning VNFs to different servers and using NICs to chain them can eliminate the NUMA effect.

(4) The end-to-end per-packet latency is small for both software dataplanes. Out of all components, the NF processing delay is the most significant source of latency. Batching helps reduce per-packet latency at high packet sending rates, and its configuration has salient impact on latency performance. Finally, latency increases with a longer NF chain and ClickOS suffers more due to the inefficient netmap VALE switch that does not use zero-copying.

The results provide useful implications for efficient resource management of NFV deployment in practice. For a telecom or ISP that deploys NFV to run her middleboxes, our results suggest that a dynamic resource allocation strategy can be adopted to opportunistically adjust the CPU speed or number of cores of the VNF and save energy without sacrificing performance. Most production networks are mildly utilized, suggesting that significant savings of electricity cost can be realized using this approach. Our results on NF chaining also shed light on VNF placement, an important management task of an NFV cluster. We show that it is better to place VNFs on separate servers (on the same CPU socket) and chain them up using NICs in order to eliminate the NUMA effect, when it is impossible to assign them to one CPU socket.

Our study also provides helpful implications for the research community on performance optimization of software dataplane. The results consistently suggest that an important research direction is to configure multiple cores and NIC queues, which can fundamentally scale the performance of software dataplane in demanding scenarios, especially as the network evolves to 100G and beyond. We also find that the NFs need to be carefully written in order to reduce latency, because they are usually optimized only for throughput with aggressive batching that hurts latency especially at low packet sending rates. We comment that performance of current software dataplanes may not be solid enough for production use. Yet by further optimizing the architecture, they have strong potential to deliver competent performance similar to hardware middleboxes in the near future.

2 BACKGROUND

We start by providing background of BESS and ClickOS.

2.1 BESS

The BESS software dataplane composes of three components: Intel DPDK [3] as the high-performance userspace packet I/O framework, BESS [6] as the programmable dataplane, and KVM as the hypervisor to isolate the VNFs.

DPDK. The Intel DPDK framework allows applications to poll data directly from the NIC without kernel involvement, thus providing high-performance userspace network I/O. To achieve line rate, a DPDK process occupies the CPU core and constantly polls the NIC for packets.

BESS. BESS [2], previously known as SoftNIC [6], is a programmable dataplane abstraction layer that allows developers to flexibly build software that leverages NIC features with minimal performance loss. One can develop her own packet processing pipeline with a series of *modules*. A module can interact with a physical NIC (pNIC) and/or a vNIC of a VM. When two modules are connected in a pipeline, a *traffic class* (TC) is created. A TC is assigned with a unique *worker* thread running on a dedicated core to move packets between the modules. A worker may be assigned to multiple TCs.

KVM. BESS provides a backend vNIC driver based on `vhost-net` which allows it to interact with KVM. We thus choose KVM as the hypervisor environment for it.

2.2 ClickOS

ClickOS. ClickOS [12] is another popular NFV platform. It composes of netmap [15] and VALE [16] as the packet I/O framework, Click [10] as the programmable dataplane, and Xen as the hypervisor. By redesigning the virtual network drivers in Xen, ClickOS achieves very high packet processing performance. Meanwhile, by leveraging Click users can flexibly build software middleboxes.

VALE and netmap. VALE is a virtual software switch for packet I/O based on netmap [15]. ClickOS modifies VALE to support pNIC directly. A key difference between VALE and TCs in BESS (or DPDK based software dataplanes) is that VALE does not use dedicated threads/cores to move packets between modules; the sending thread does the work of copying packets into the Rx queue.

3 METHODOLOGY

We explain our measurement methodology in detail here.

3.1 Hardware Setup

We conduct our measurements using both *physical* machines rented from Aptlab[1] and our own servers. We use two c6220 nodes in Aptlab with 2 Xeon E5-2650v2 processors (8 cores each, 2.6Ghz), 64GB DDR3 1.86GHz Memory and an Intel X520 10GbE PCIe dual port NIC to do all throughput experiments. For delay experiments, we use our own testbed: two servers with 1 Xeon E5-2640v2 processors (8 cores each, 2.0Ghz), 32GB DDR3 1.3GHz Memory and an Intel X520 10GbE PCIe dual port NIC. We use two types of servers since Aptlab provides more powerful machines for high throughput, and our servers provide better latency as they are connected back-to-back without any hardware switch or network virtualization. Note the two types of servers we use are very similar in terms of their CPU, memory, and NIC. We confirm experimentally that when Aptlab server CPU is downclocked to 2.0Ghz, the results are quantitatively the same with our servers.

For most experiments, one node runs a packet generator to send packets of different sizes to the other node, which acts as the hypervisor hosting VNFs to process packets. Packets are sent back through another NIC of the hypervisor to the first node, which is also our vantage point.

3.2 Software Dataplane Settings

For BESS, we use Linux kernel 3.19.0, QEMU/KVM 2.2.1, DPDK 2.20, and the latest BESS source code [2]. For ClickOS, we use Linux kernel 3.9.10, Xen 4.4.2, ClickOS 0.1 and netmap commit `3ccdada` respectively. We use an older version of netmap that works with the modified Xennet library provided by ClickOS for Xen backend and frontend NIC drivers.

Figure 1 illustrates the packet I/O pipeline in our measurements with a single VNF. Each VNF has two vNICs, vNIC0 as the ingress NIC and vNIC1 as the egress NIC to its next hop. In general, for both BESS and ClickOS, first a packet is moved by the pNIC0 driver to the backend of vNIC0, which then sends it to the frontend driver in the VNF. After being processed by the VNF, the packet is sent to the frontend and then backend of vNIC1. Finally it is sent to an output NIC in the hypervisor.

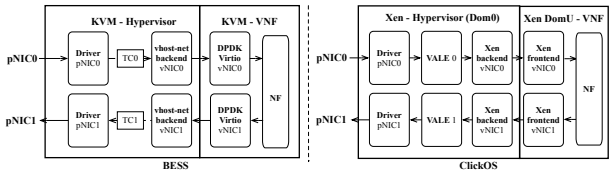

Figure 1: Packet I/O pipeline with a single VNF.

3.3 Network Functions

We use two NFs in this study, L3 forwarding (L3FWD), and firewall. We cannot use generic software such as Snort or Bro because they are not written with DPDK or Click to exploit software dataplane capability. Our measurements with simple NFs serve as the upper bound of performance, because more complicated NFs cost more CPU cycles and cannot enjoy better performance. We assign 1 vCPU and 1GB memory to each VNF unless stated otherwise.

L3FWD. We use the `ip_pipeline` example code provided by DPDK as the L3FWD implementation in BESS. The ClickOS implementation is done by concatenating `FromDevice`, `StaticIPLookup`, and `ToDevice` elements. In both implementations we insert 10 entries to the routing table.

Firewall. We build the firewall based on the same `ip_pipeline` in BESS. In ClickOS, the implementation uses `FromDevice`, `IPFilter`, and `ToDevice` elements. We use 10 rules to filter packets.

We also use the simple L2 forwarding in the NF chaining experiment only though. We do not consider it a NF as it does not have any packet processing logic.

L2FWD. We directly use the L2FWD provided out-of-the-box from `l2fwd` of DPDK for BESS. In ClickOS, we implement L2FWD by connecting the `FromDevice` and `ToDevice` elements between vNICs.

4 THROUGHPUT RESULTS

We investigate the throughput performance of BESS and ClickOS in different scenarios in this section. The thesis of the evaluation is simple: can these NFV dataplanes achieve line rate, and if not, what are the bottlenecks? Is it possible to improve? We first look at the baseline scenario with a single software middlebox, running different NFs with varying CPU speed (§4.1). Based on the results we analyze and identify performance bottlenecks of both dataplanes (§4.2). We then deploy multiple NFs in a scenarios that is commonplace in practice: NF chaining where packets go through the middleboxes sequentially for processing (§4.3).

4.1 Baseline Performance

We start with just a single VNF. Since software packet processing is CPU-intensive, we want to see if CPU speed is the bottleneck here. In this set of experiments, we vary the CPU speed from the configurable range of 1.2GHz to 2.6GHz for our CPU without Turbo Boost, and investigate the throughput with different packet sizes.

Figure 2 demonstrates the performance of L3FWD. We observe the following. First, performance increases with CPU speed for small packets (64B–256B), which is expected—a faster CPU can process more instructions and thus more packets. For 64B and 128B packets, performance improvement is commensurate with CPU speed-up between 1.2GHz to 2.4GHz. With 64B packets for instance, at 1.2GHz throughput of BESS and ClickOS is 4.31Mpps and 2.10Mpps, respectively, while at 2.4GHz it roughly doubles at 8.60Mpps and 4.26Mpps, respectively. The improvement is smaller at 2.6GHz. Second, both NFV dataplanes achieve line rate for packets bigger than 128B, but have problems dealing with smaller packets even at 2.6GHz. BESS can achieve 10Gbps with 128B packets at 2.4GHz, and ClickOS can process 256B packets at 10Gbps at 2.6GHz. Yet for 64B packets, even at 2.6GHz, neither achieves line rate: BESS tops at 9.34Mpps and ClickOS 5.34Mpps. Third, BESS outperforms ClickOS in all cases, especially for small packets. We compare our results with those reported in the ClickOS original paper [12] and confirm they are similar. For example, in Figure 13 of [12], throughput of L3FWD and Firewall with 64B packets are 4.25Mpps and 5.40Mpps, respectively, and ours are 5.34Mpps and 5.03Mpps, respectively. Finally, we find that the performance difference between L3FWD and firewall is minimal, as shown in Table 1. We thus only show L3FWD results hereafter for brevity.

Table 1: Throughput of different NFs with varying CPU speed. Packet size is 64 bytes.

	BESS (Mpps)		ClickOS (Mpps)	
CPU	L3FWD	Firewall	L3FWD	Firewall
1.2GHz	4.31	4.45	2.10	1.98
1.6GHz	5.73	5.92	2.82	2.75
2.0GHz	7.19	7.38	3.44	3.27
2.4GHz	8.60	8.87	4.26	4.24
2.6GHz	9.34	9.59	5.34	5.03

We also use an empirical packet size distribution from Facebook's web server cluster [17] to see how the software dataplanes perform in a practical environment. The median packet size is ~120B, and most packets are less than 256B. We configure `pkt-gen` to sample the trace and generate packets first at 10Gbps, and observe that the average throughput in BESS is 8.662Mpps. However a production network is rarely fully utilized. Facebook reports their median link utilization is 10%–20% [17]. This implies that the median packet processing requirement is 0.87Mpps–1.73Mpps. Both BESS and ClickOS are able to provide such capability in lowest CPU frequency of 1.2GHz.

These observations have interesting implications for NFV resource allocation. They suggest that there are ample opportunities

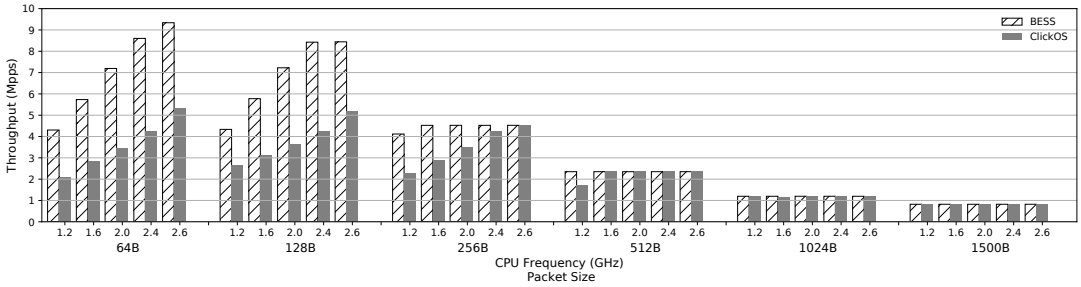

Figure 2: Throughput with different CPU speeds and packet sizes. We show throughput in both Mpps and Gb/s.

for the operator to downclock the CPU in order to save energy and electricity cost in the average case. Care has to be taken though, of course, to ensure performance does not suffer when there are sudden bursts of small packets. This may be a useful resource allocation strategy for operators as well as meaningful research directions to look into for the networking community. Our observations also motivate us to identify performance bottlenecks in both NFV dataplanes for small packets, which we explain next.

4.2 Performance Bottlenecks

One may argue that the performance deficiency of software dataplanes in small packet regime is acceptable in practice, since small packets may be less common. However, these systems may not be able to achieve line rates in the emerging 40G or 100G networks [9], even for large packets. Therefore we believe it is important for us to understand the performance bottleneck and improve performance.

To identify bottlenecks, we conduct the following analysis. For BESS, we observe from using `monitor port` command that about 5Mpps 64B packets are lost in the pipeline between pNIC0 and vNIC0. To verify the analysis, we conduct another experiment by adding a round-robin module (RR) and another two vNICs (which is similar to add NIC queues) to the L3FWD VM. Traffic is evenly split between the two input vNICs. This time throughput reaches line rates for all packet sizes. We believe the single CPU core is bottleneck of BESS. To further increase the performance, it is necessary add more vNICs (queues) and CPU cores to scale up the processing rate.

For ClickOS, we analyze the CPU utilization of the L3FWD instance with the CPU at the highest 2.6GHz. We found higher clocked CPU cores can achieve higher throughput. This implies that more CPU resource may be needed here. However, ClickOS currently does not have SMP support [12], preventing us from adding more cores to the VM. This also means adding more vNICs does not help without more CPU. Another possible solution is to use multiple VNFs working in parallel. This naturally requires a load balancer (LB) to split the traffic. VALE is a simple L2 switch without any load balancing capability [16]. Adding a LB VM does not work either since the LB itself becomes the bottleneck. Therefore we are unable to resolve the bottleneck without modification to ClickOS itself.

To summarize, the results here verify that BESS's bottleneck is the single CPU core. This can be resolved by sending traffic to two vNICs of one VNF in parallel to fully utilize multiple vCPUs. We also present evidence to suggest that ClickOS should add SMP support that allows it to utilize multiple CPU cores. In any case, we note that it is imperative for the NFV software dataplane architecture to provide horizontal scaling of its performance, in order to better utilize multiple cores and physical NIC queues. We believe this is an interesting open research area as the NICs evolves to 40Gbps and beyond.

4.3 NF Chaining

It is common to deploy multiple software middleboxes on the same machine. In this section we look into the NF chaining scenario, where the processing pipeline consists of a chain of different middleboxes. We are interested to see if the performance of an NF chain can match that of just a single NF. We compose chains of different lengths: the 1-NF chain uses only a L3FWD; the 2-NF chain uses a firewall followed by a L3FWD; and the 3-NF chain adds a L2FWD to the end of the 2-NF chain.

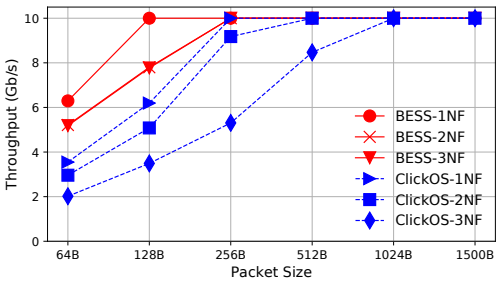

Figure 3: Throughput with varying length of NF chain.

The results are shown in Figure 3 with all vCPUs running on the same physical CPU. The performance of BESS suffers mild degradation for 64B–128B packets, as can be seen from the overlapping lines of 2-NF chain and 3-NF chain. We suspect there is a bottleneck in chaining the vNICs of different VMs because both firewall and L3FWD can achieve higher performance individually as shown in §4.1. Performance of ClickOS also degrades as the chain grows, especially for small packets. A ClickOS L3FWD achieves line rate with 256B packets, but a 2-NF chain or 3-NF chain cannot. A 3-NF chain cannot even reach line rate with 512B packets. Note that here we use multiple VALE switches with independent vCPUs pinning to different cores to chain the VMs as suggested by [12]. We believe the overhead of copying packets in VALE attributes to the performance penalty.

When deploying a NF chain, an important factor we must consider is the affinity of vCPUs and the effect of NUMA. We can pin each vCPU to the same physical CPU, or pin them to CPUs in different sockets. The latter is unavoidable sometimes as the commodity CPUs have limited cores per CPU, and DPDK-based NFV dataplanes like BESS require many dedicated cores as mentioned in §2.1 and §3.2.

We perform another measurement to evaluate the effect of NUMA on NF chaining. Figure 4 shows the result for BESS as a case study. NUMA has a significant impact on performance. For the 2-NF chain, assigning two vCPUs and TCs to different sockets cuts the throughput of 64B packet by nearly half. For the 3-NF chain (the third VNF runs in a different socket than the first two), line rate is only reached for 512B and larger packets. The performance discrepancy is mainly because operations between different NUMA sockets can cause cache misses and ping pong effect [11]. To mitigate NUMA effect, we attempt to bridge NFs in a chain via NICs across servers. For example, in a 3-NF chain, NF1 and NF2 are located on server A on the same NUMA socket, and NF3 is located in server B. The two servers are connected by 10GbE NIC. We observe that this eliminates the NUMA effect: throughput of the chain is identical to the case when all 3 NFs share the same CPU socket as shown in Figure 4.

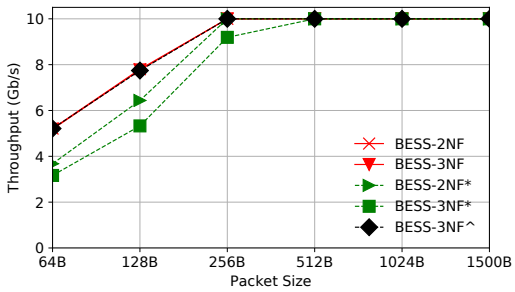

Figure 4: Throughput of NF chaining with the NUMA effect. "*" here denotes the case when the vCPUs belong to different sockets. "^" here denotes the case when NFs are chained up via NICs of different servers to avoid penalty from NUMA.

To summarize, the results here show that BESS works adequately with small performance drop in a NF chain, while ClickOS's throughput becomes lower with longer chains. They also demonstrate the importance of carefully assigning cores to VMs of the chain due to NUMA, which implies that it is not always best to colocate VNFs of a chain on the same server. A practical strategy is to place them on different servers to avoid NUMA effect. These observations are useful for real NFV deployment.

5 LATENCY RESULTS

We now investigate another important performance metric—latency—in different scenarios in this section. We use timestamps on packets to measure the end-to-end latency. The end-to-end latency includes transmission delay between servers, software dataplanes packet I/O latency between pNICs and vNICs, NF packet I/O latency between vNICs, and NF processing latency.

We first look at the impact of packet sending rate on latency (§5.1). Then, we investigate the impact of NF and length of service chains (§5.2).

5.1 Impact of Sending Rate

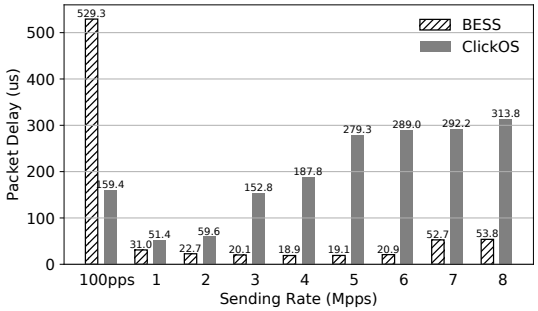

Figure 5: Latency with varying sending rates (128B packets).

In this section, we investigate the latency performance using various packet sending rates to see if software dataplanes can offer consistent low latency. Figure 5 shows the results with sending rates ranging from 1Mpps to 8Mpps. We also include the result when packets are sent at 100pps, i.e. every 10 ms. This corresponds to short transactions for websites and distributed applications (with a few packets), which are common in practice [20]. We fix packet size to be 128B here, similar to the average packet size in Facebook's production data center ~120B as discussed in §4.1. We observe similar results for other packet sizes.

We make several interesting observations here. First, latency at 100pps is significantly larger than other sending rates for both software dataplanes. For BESS, the latency is even an order of magnitude larger. The reason for this discrepancy is that at such low speed, packets are processed individually rather than in a batch. Packets are fetched and sent in a batch basis to optimize for high throughput. Thus the NFs send the processed packets out only when the queue reaches a burst threshold or a timer expires.

Second, in BESS, per-packet end-to-end latency has an interesting trend of decreasing first and then increasing when sending rate is over 7Mpps. This can be explained as follows. The initial decreasing trend is a clear result of BESS's batching optimization. However, when the sending rate hits the processing capacity of the dataplanes, the queuing delay at pNIC Rx queue dominates. For our own servers with a slower CPU the maximum throughput of BESS is ~7Mpps. Hence, latency of BESS starts to increase at 7Mpps. ClickOS does not exhibit this pattern, most likely because its batching is not as aggressive, and latency increases with sending rates even before the queueing delay kicks in.

Based on the results here, we suggest the operators should carefully load balance the traffic in their NFV deployments. If traffic to an NF is too low, latency may be high without batching; if traffic exceeds the capacity of the VM, it is crucial to shift traffic away to other servers before latency spikes. Meanwhile, setting the burst threshold and timer value have great impact on latency and throughput of software dataplanes. This is an open research question.

5.2 Impact of NF Chaining

Figure 6 depicts the latency in NF chaining scenario. Note that the NFs has the same configuration with §4.3 with all VMs running on the same CPU. As expected, latency increases as the chain gets longer. BESS's latency of a 2NF chain is 34.326 µs, less than the

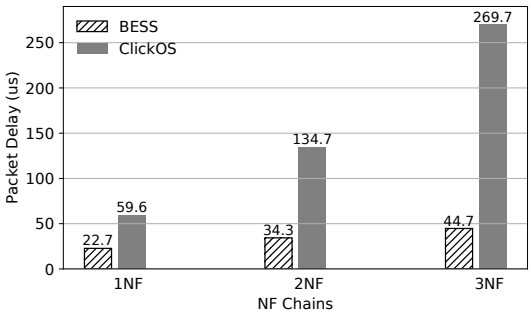

Figure 6: Average latency of NF chains of various lengths. Packet size is 128B; sending rate is 2Mpps.

combined latency of firewall and L3FWD (45.982 µs) since there is no redundant packet I/O on pNICs. However, we find that ClickOS's latency of a chain is actually larger than the combined latency of individual NFs. This is because likely due to the netmap VALE switch used which does not use zero-copy to share packets among the NFs.

6 RELATED WORK

We introduce related work on NFV dataplane other than BESS and ClickOS now. NetVM [8] and OpenNetVM[22] are NFV platforms based on DPDK and KVM/Docker, similar to BESS. It provides high-speed inter-VM communication with zero-copy through shared huge pages. Systems such as ptnetmap [5] and mSwitch [7] based on netmap address efficient transfer between VMs in a single server. E2 [13] is a general NFV management framework focusing on NF placement, scheduling, scaling, etc. Its dataplane uses BESS. NetBricks [14] and Flurries [21] are NFV platform based on different virtualization technologies and abstraction levels for VNF. We plan to study their performance in the future.

Our measurement study provides performance comparison across solutions with actual VNFs and complements existing work that evaluates their own system with mostly L2 forwarding. There is little measurement study on NFV in general. Wu et al. design PerfSight [19] as a diagnostic tool for extracting comprehensive low-level information regarding packet processing performance of the various elements. It focuses on virtualization layer (KVM) without integrating with any NFV dataplane such as BESS and ClickOS.

7 CONCLUSION

In this paper, we conducted a measurement study on the performance of BESS and ClickOS. Both dataplanes are capable of achieving 10G line rate with medium and large packets. They have performance issues in the, small packet regime and NF chaining scenario, which may become more severe in high speed networks. We proposed to fundamentally address the limitation by architecting the software dataplane for horizontal performance scaling, in order to better utilize multiple cores and NIC queues. Moreover, NFs are the major source of latency compared to dataplane, and need to be further optimized for latency while maintaining high throughput. Adaptive batching according to packet sending rate may be helpful here.

Our study can be extended in many directions. One possibility is to consider more complex NFs, such as NAT, VPN, etc, and practical application traffics. We also plan to further investigate the chaining scenario and identify ways to improve performance.

8 ACKNOWLEDGMENT

The project is supported in part by the Hong Kong RGC GRF-11216317, CRF-C7036-15G, High-Tech Research and Development Program of China ("863-China Cloud" Major Program) under grant 2015AA01A201 and the National Natural Science Foundation of China (NSFC) under grants 61471217, 61432002 and 61632020.

We are thankful to members of BESS and ClickOS teams, especially to Sangjin Han from UC Berkeley, and Filipe Manco from NEC Laboratories Europe, for help in setting up the NFV platforms on our testbed.

REFERENCES

[1] [n. d.]. APTLAB. http://aptlab.net/. ([n. d.]).
[2] [n. d.]. BESS: Berkeley Extensible Software Switch. https://github.com/NetSys/bess, commit b43bbdd. ([n. d.]).
[3] [n. d.]. DPDK. http://dpdk.org/. ([n. d.]).
[4] European Telecommunications Standards Institute. [n. d.]. Network Functions Virtualisation: Introductory White Paper. http://portal.etsi.org/NFV/NFV_White_Paper.pdf. ([n. d.]).
[5] Stefano Garzarella, Giuseppe Lettieri, and Luigi Rizzo. 2015. Virtual device passthrough for high speed VM networking. In *Proc. ACM/IEEE ANCS*.
[6] Sangjin Han, Keon Jang, Aurojit Panda, Shoumik Palkar, Dongsu Han, and Sylvia Ratnasamy. 2015. *SoftNIC: A Software NIC to Augment Hardware.* Technical Report. UC Berkeley.
[7] Michio Honda, Felipe Huici, Giuseppe Lettieri, and Luigi Rizzo. 2015. mSwitch: A highly-scalable, modular software switch. In *Proc. ACM SOSR*.
[8] Jinho Hwang, K. K. Ramakrishnan, and Timothy Wood. 2014. NetVM: High Performance and Flexible Networking Using Virtualization on Commodity Platforms. In *Proc. USENIX NSDI*.
[9] Lavanya Jose, Lisa Yan, Mohammad Alizadeh, George Varghese, Nick McKeown, and Sachin Katti. 2015. High Speed Networks Need Proactive Congestion Control. In *Proc. ACM HotNets*.
[10] Eddie Kohler. 2001. *The Click Modular Router.* Ph.D. Dissertation. MIT.
[11] David Levinthal. 2009. Performance analysis guide for intel core i7 processor and intel xeon 5500 processors. *Intel Performance Analysis Guide* 30 (2009).
[12] Joao Martins, Mohamed Ahmed, Costin Raiciu, Vladimir Olteanu, Michio Honda, Roberto Bifulco, and Felipe Huici. 2014. ClickOS and the Art of Network Function Virtualization. In *Proc. USENIX NSDI*.
[13] S. Palkar, C. Lan, S. Han, K. Jang, A. Panda, and S. Ratnasamy. 2015. E2: A Framework for NFV Applications. In *Proc. ACM SOSP*.
[14] Aurojit Panda, Sangjin Han, Keon Jang, Melvin Walls, Sylvia Ratnasamy, and Scott Shenker. 2016. NetBricks: Taking the V out of NFV.. In *Proc. USENIX OSDI*.
[15] Luigi Rizzo. 2012. netmap: A Novel Framework for Fast Packet I/O. In *Proc. USENIX ATC*.
[16] Luigi Rizzo and Giuseppe Lettieri. 2012. VALE, a Switched Ethernet for Virtual Machines. In *Proc. ACM CoNEXT*.
[17] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C. Snoeren. 2015. Inside the Social Network's (Datacenter) Network. In *Proc. ACM SIGCOMM*.
[18] Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. 2012. Making Middleboxes Someone else's Problem: Network Processing As a Cloud Service. In *Proc. ACM SIGCOMM*.
[19] Wenfei Wu, Keqiang He, and Aditya Akella. 2015. PerfSight: Performance Diagnosis for Software Dataplanes. In *Proc. ACM IMC*.
[20] Kenichi Yasukata, Michio Honda, Douglas Santry, and Lars Eggert. 2016. StackMap: Low-Latency Networking with the OS Stack and Dedicated NICs. In *Proc. USENIX ATC*.
[21] Wei Zhang, Jinho Hwang, Shriram Rajagopalan, K.K. Ramakrishnan, and Timothy Wood. 2016. Flurries: Countless Fine-Grained NFs for Flexible Per-Flow Customization. In *Proc. ACM CoNEXT*.
[22] Wei Zhang, Guyue Liu, Wenhui Zhang, Neel Shah, Phil Lopreiato, Gregoire Todeschi, KK Ramakrishnan, and Timothy Wood. 2015. OpenNetVM: A Platform for High Performance Network Service Chains. In *Proc. ACM HotMiddlebox*.