Demystifying the Performance Interference of Co-located Virtual Network Functions

Chaobing Zeng Fangming Liu* Shutong Chen Weixiang Jiang Miao Li
Key Laboratory of Services Computing Technology and System, Ministry of Education,
School of Computer Science and Technology, Huazhong University of Science and Technology

Abstract— **Network function virtualization (NFV) decouples network functions from the dedicated hardware and enables them running on commodity servers, facilitating widespread deployment of virtualized network functions (VNFs). Network operators tend to deploy VNFs in virtual machines (VMs) due to VM's ease of duplication and migration, which enables flexible VNF placement and scheduling. Efforts have been paid to provide efficient VNF placement approaches, aiming at minimizing the resource cost of VNF deployment and reducing the latency of service chain. However, existing placement approaches may result in hardware resource competition of co-located VNFs, leading to performance degradation. In this paper, we present a measurement study on the performance interference among different types of co-located VNFs and analyze how VNFs' competitive hardware resources and the characteristics of packet affect the performance interference. We disclose that the performance interference between co-located VNFs is ubiquitous, which causes the performance degradation, in terms of VNFs' throughput, ranging from 12.36% to 50.3%, and the competition of network I/O bandwidth plays a key role in the performance interference. Based on our measurement results, we give some advices on how to design more efficient VNF placement approaches.**

I. Introduction

As a flexible and economical technology for network function deployment, network function virtualization (NFV) draws more and more attention nowadays. It decouples network functions (NFs) from the dedicated hardware and enables them running on commodity servers, showing great potential for reducing capital expenditure of NF deployment and providing network operators an ability of on-demand scaling [1], [2].

Leveraging the virtualization technology, virtual network functions (VNFs) run in virtual machines (VMs) [3]. Unlike traditional middleboxes with fixed locations [4], VNFs could be deployed on any commodity server in the network by migrating VMs. However, this usage model raises new challenges. Firstly, a packet may need to go through a set of NFs. Network operators should carefully decide where these VNF instances should be placed among lots of candidate servers, to reduce end-to-end latency [5]. Secondly, to improve the resource utilization, VNF placement should keep pace with flow fluctuations by dynamically scaling the number of VNF instances. Finally, existing VNF instances may need to be consolidated to another server to free and shut down some servers for saving energy. Existing studies are devoted to tackling the above challenges and offering efficient VNF placement approaches. For instance, Li et al. design a real-time system, NFV-RT, to dynamically provision resources to meet traffic demand [6]. Eramo et al. propose a migration policy to tell network operators when and where they should migrate VNFs for saving energy [7]. And Mehraghdam et al. provide different VNF placement approaches to satisfy changeable requirements [8].

However, all of the existing VNF placement approaches have not taken the performance interference among co-located VNFs into account. Although the virtualization technology provides a level of performance isolation among VMs, there is still significant performance interference among VNFs running on one shared hardware infrastructure [9]. As evidenced by our measurement in Sec. II-B, the performance of flow monitor, in terms of throughput in our work, decreases by 38.47% when the flow monitor runs with an intrusion detection system simultaneously. Performance interference among VNFs is brought by inherent resource sharing principle across VMs. A VNF has to compete with other co-located VNFs for resources, including network I/O bandwidth, CPU, memory, etc. [10]. The more intense the resource competition is, the more severe the performance interference is [11]. However, to maximize resource utilization and reduce power consumption, network operators tend to place VNF instances on the same physical servers as much as they can, resulting in fierce resource competition and severe performance interference [12]. An in-depth understanding of performance interference among co-located VNFs is significant for efficient VNF management.

To address the issues above, in this paper, we classify VNFs into different types according to their packet operations, as summarized in Table. II, and measure the performance interference among different types of co-located VNFs. Firstly, we verify the existence and severity of the performance interference by measuring the performance of each VNF with different VNF combinations. Next, we analyze the resource usages of each VNF when VNFs run in pairs to explore the relation between resource and performance interference. Finally, we vary the level of packet characteristics, e.g., packet size and transmission rate, to evaluate their impacts on performance interference. Our main findings are as follows:

- The performance interference among VNFs is ubiquitous

*This work was supported in part by the National Key Research & Development (R&D) Plan under grant 2017YFB1001703, in part by NSFC under Grant 61722206 and 61761136014 (NSFC-DFG) and 61520106005, in part by the National 973 Basic Research Program under Grant 2014CB347800, and in part by the Fundamental Research Funds for the Central Universities under Grant 2014YQ001, 2014TS006 and 2017KFKJXX009. (Corresponding author: Fangming Liu)

and severe, causing up to 50.3% of throughput degradation, due to competitions of network I/O bandwidth, CPU, cache, and memory.
- For different co-located VNFs, the performance interference is different. The reason is that the competitive resources and competitive levels are different for different VNF combinations. For example, VNFs with packet-header-reading operations and VNFs with packet rewriting operations suffer little performance interference, because there is little resource competition between them.
- Packet operations of VNFs can be sorted in descending order of CPU resource usage: after packet rewriting would come link status updating, then table look-up, then packet reading.
- In general, the performance interference among VNFs becomes intenser when packet size rises. This is because of longer I/O processing time and more occupied memory pages.
- For most of VNFs, the higher the transmission rate is, the severer performance interference the co-located VNFs have, because of the intense competition for network I/O bandwidth, CPU, and cache.

The performance interference measurement also provides some insights on the VNF placement:

- The resource allocation mechanism should assign different resources to VNFs in different scenarios. For example, it is more necessary to allocate CPU resources to VNFs with packet rewriting operations than to other types of VNFs.
- The location of VNFs has significant effect on the performance. For instance, when selecting where to deploy VNF with packet rewriting operations, it is better to choose the server where VNF with packet-header-reading operations runs, since there is little performance interference between them.
- During VNF placement, network operators should over-provision VNF instances to offset performance degradation caused by co-located VNFs.

The remainder of this paper is structured as follows. We first discuss the background of performance interference and the motivation in Sec. II. Next, Sec. III discusses the measurement setup. And Sec. IV presents the details of measurement. Based on the measurement results, the insights on VNF placement mechanism are provided in Sec. V. Then, Sec. VI discusses the related works. Finally, Sec. VII concludes the paper.

II. Background and Motivation

In this section, we prove the existence of performance interference among VNFs from a theoretical perspective and an experimental perspective, respectively.

A. Background

Ideally, VMs' performance would be independent of the activity going-on on the hardware, which is accomplished by smart scheduling and resource allocation policies. However, it is hard to realize the performance isolation of VMs in real world. Specifically, for example, the hyper-threading technology enables one physical core to simulate two logic cores (i.e., vCPU), which means that the vCPUs allocated to different VMs may share the same physical core and caches. When setting up a VM, we can specify the number of vCPU and allocate fixed size memory and disk, however, it is technically difficult to allocate network I/O bandwidth, memory bandwidth, and disk bandwidth to different VMs [13]. The inherent resource sharing principle across VMs may cause performance interference [14]. Furthermore, the effect of resource competition may be interactional, which means the competition of one type of resource may cause other resources' competition [15].

As mentioned above, VNFs generally run in VMs and they would co-locate in the same server. Obviously, the performance interference among VNFs is inevitable and dependent on the activities/characteristics of VNFs. For example, frequent network I/O operations of VNFs may cause network I/O bandwidth competition, and packet header detecting as well as packet rewriting operations can lead to VNFs' sensitivity to CPU and cache. However, *how severe is the performance interference among VNFs*; *what is the key role in impacting VNFs' performance*; and *how do we avoid performance interference during VNF placement*? The answers still remain unknown.

B. Motivation

To reveal the severity of performance interference, we conduct a preliminary experiment with `Snort`[1] and `Pktstat`[2] which represent two types of VNFs listed in Table II. We deploy `Snort` and `Pktstat` on two co-located VMs to perform filtering and listening functions, respectively. We utilize another server to generate and send traffic to each VM. Table I shows the performance of `Snort` and `Pktstat` running separately and simultaneously. We observe that compared with the case of running separately, when running simultaneously, the processing capacities of `Snort` and `Pktstat` decrease by 35.51% and 38.47%, respectively. These results validate that the performance of VNFs will be cut down severely by the performance interference across co-located VNFs.

TABLE I
Performance comparison between Snort and Pktstat running separately and simultaneously.

	Running Separately	**Running Simultaneously**
`Snort`	955 Mbps	615.88 Mbps
`Pktstat`	169 Mbps	103.99 Mbps

We now discuss about the effect of co-located VNFs' performance interference and why we should take the performance interference into account while placing VNF instances. As illustrated in Fig. 1, there are two different types of VNFs, VNF 1 and VNF 2, each of which deals with different traffic. VNF 1 and VNF 2 are placed on server 1 and server 2, respectively.

[1]`Snort`: https://www.snort.org/
[2]`Pktstat`: https://linux.die.net/man/1/pktstat

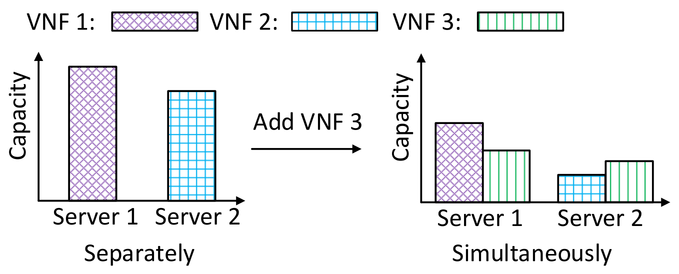

Fig. 1. A motivating example of co-located VNFs' performance interference.

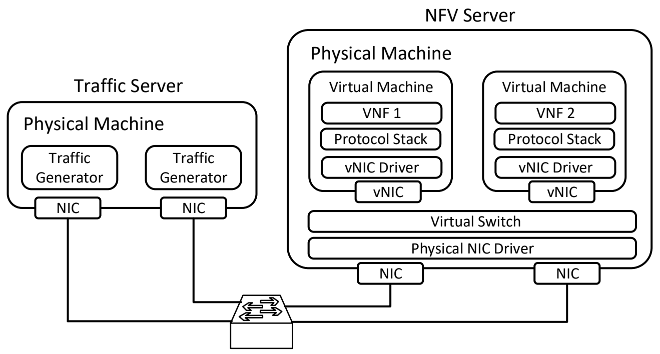

Fig. 2. The measurement platform with two servers. The left server acts as the traffic sender and the right one is the NFV server where the VNFs run.

Now, the rising of network traffic calls for an additional VNF 3. Network operators have two choices here: deploying VNF 3 on server 1 or server 2, which are marked as plan 1 and plan 2, respectively. As shown in Fig. 1, both two choices cause significant performance degradation: the processing capacities of VNF 1 and VNF 2 running with VNF 3 separately are lower than their original processing capacities. It is worth noting that the performance degradation of VNF 2 is higher than that of VNF 1, and VNF 3's processing capacity in plan 1 is higher than that in plan 2. If the network operators overlook the performance interference and select plan 2, the overall processing capacity will be lower than that in plan 1, resulting in much more profit loss. Moreover, each VNF suffers from a sudden and severe performance degradation no matter in plan 1 or plan 2 because of the interference. Thus, the added VNF 3 may not meet the requirements of the rising traffic, and the traffic processed by the original VNF may suffer from longer latency due to performance degradation, leading to unreliability. Existing VNF placement approaches, which do not take performance interference into account, cannot foresee this situation, resulting in unresponsive action. If the performance interference is taken into account, the performance degradation can be relieved by deploying VNF 3 on server 1, and VNF placement approach will prepare sufficient VNF instances to reduce latency. Above all, it is essential to attach importance to the performance interference among VNFs when adjusting the scale of VNF instances to follow the traffic fluctuation.

III. Measurement Setup

A. Measurement Platform

As shown in Fig. 2, we conduct the measurement on two commodity servers, one server is virtualized and acts as the NFV server, and the other server acts as the traffic server, sending packets to the NFV server. Each server is configured with an eight-core Intel Xeon E5-2620 v4 2.1 GHz CPU, a 64 GB memory, and two Intel I350 GbE network interface cards (NICs).

In the NFV server, each VNF instance runs on an exclusive VM and each VM instance is equipped with 1 vCPU core, 2 GB memory and 10 GB disk. We utilize the traffic server to generate and send packets with varied packet sizes, and transmission rates. The NFV server receives and forwards the packets to the corresponding VNFs. Throughput, response time, and packet loss are measured as performance metrics by `packETH`[3], `siege`[4], and `iperf`[5]. Note that throughput and latency are two important performance metrics in network, packet loss rate is consider for it can reveal the waste of resources [16]. Besides, to analyze the relationship between resource and performance, we record the CPU utilization, memory usage and cache usage using `dstat`[6], `ps` command and `perf`[7]. Each measurement is carried out 10 times independently, and we illustrate the average values as measurement results.

B. VNF Setup

From the aspect of packet processing, VNFs can be divided into six types which is shown in Table II. As depicted in Table II, these VNFs whose operations are reading packets comprise more than 75% in data centers, while the others which modify the packets account for less than 25% [17]. In the later case, Type VI is far less than Type IV and Type V, which is less than 5%. So, we choose the top five types as our measurement objects.

For each type of VNFs, we choose one common application as representative, as listed in Table II. The introduction of selected applications and their setups are discussed as follows.

- `Iptables`[8]. `Iptables` is an administration tool for IPv4 packet filtering and network address translation. `Iptables` in our measurement is equiped with an IP set which contains hundreds of IP addresses. It is used to check the IP address of each incoming packet using IP set and forward the packet to the destination server.
- `Pktstat`. `Pktstat` is a flow monitor displays a real-time summary of packet activity on an interface. By reading a packet header, `Pktstat` can get information about IP

[3]`packETH`: http://packeth.sourceforge.net/packeth/Home.html
[4]`siege`:https: //www.joedog.org/siege-home/
[5]`iperf`: https://iperf.fr/
[6]`dstat`: https://linux.die.net/man/1/dstat
[7]`perf`: https://en.wikipedia.org/wiki/Perf_(Linux)
[8]`Iptables`: https://linux.die.net/man/8/iptables

TABLE II
TYPES OF VIRTUAL NETWORK FUNCTIONS AND APPLICATIONS.

Type	VNF Examples	Ratio	Packet Processing			Application
			IP	Port	Pay-load	
I	Gateway	>75%	R	–	–	`Iptables`
II	Firewall, Monitor		R	R	–	`Pktstat`
III	NIDS, DPI, Caching		R	R	R	`Snort`
IV	LoadBlancer, Proxy	<25%	R/W	–	–	`Nginx`
V	NAT		R/W	R/W	–	`Nginx`
VI	VPN, Compression, Encryption		R/W	R/W	R/W	–

address, ports, packet type and size, and using this information to update the status of each connection.

• `Snort`. `Snort` is a network intrusion prevention system with real-time traffic analysis. `Snort` will check the header and content of each incoming packet, performing real-time monitoring of network transmission. In our measurement, we configure `Snort` with thousands of detection rules. When detecting illegal packets, `Snort` will discard these packets, send alerts to system and log anomalous actions. Such discarding operations consume the processing capacity of `Snort`, however, these processed and discarded packets cannot be measure by our measurement tools. To ensure the accuracy of measurement, `Snort` in our measurement only sends alerts and writes logs, disabling the packet discarding operation.

• `Nginx`[9]. `Nginx` is a proxy server, supporting HTTP, HTTPS, TCP, UDP, etc. In our measurement, `Nginx` acts as a load balancer. When the packets coming, it will modify the packet header and forward the packet to another server. It can also decide whether to modify the packet ports or not. Hence `Nginx` can be classed as both Type IV and Type V, as shown in Table II. In Type IV, `Nginx` will only modify the IP address of incoming packets before forwarding, while `Nginx` also modify the packet ports in Type V.

Note that the performance of NFs also depends on the characteristics of packets. For example, Snort will compare the IP address, ports and content of incoming packets with predefined rules. The cost of this table look-up operation varies with different packets. So, all the packets are generated with random addresses and ports, which approximately confirms to reality.

IV. MEASUREMENT RESULTS

In this section, we measure the performance interference of VNFs and identify the key factors in impacting the performance of VNF instances. We first investigate the impact of different VNF combinations on performance interference. Then, we collect and analyze each VM's resource utilization

[9]`Nginx`: http://nginx.org/en/

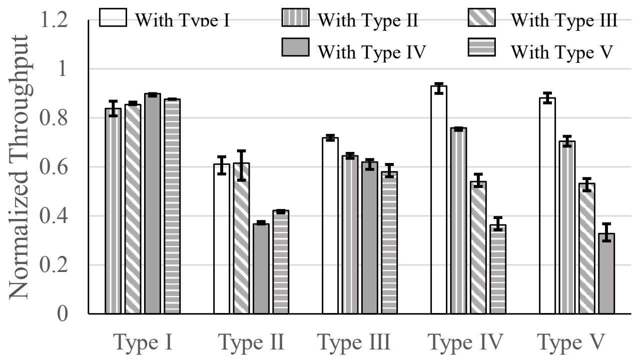

Fig. 3. Normalized throughput of each VNF in different pairs.

to identify the competitive resources on VNFs' performance interference. Finally, we proceed to examine whether the characteristics of packet can impact the performance variation of VNF instances. For simplicity, we use the value when VNF runs separately as a baseline to compute variations in other contexts.

A. The performance interference between different VNF combinations

We first investigate the impact of VNF's type on performance interference by measuring the performances of applications mentioned in Table II when they run in pairs. The performance metric used to evaluate performance interference here is "normalized throughput", the ratio of the VNF's throughput when it runs with other VNF to its throughput when it runs separately.

As depicted in Fig. 4, we find that the performance interference of co-located VNFs is ubiquitous. All types of VNFs suffer performance degradation when they run in pairs, because the normalized throughput is always less than 100%. We utilize "target VNF" to represent the VNF on the horizontal axis in Fig. 4, and the VNF running with the target VNF is called "competitor". Obviously, with different target VNF or different competitor, the normalized throughput is different. For example, running with Type III, the normalized throughput of Type I is 85.41% while the normalized throughput of Type II is 61.53%. We find that no matter what target VNF is, when Type IV or Type V is the competitor, the normalized throughput of the target VNF is almost the lowest. As a result, Type IV and Type V are two of the most aggressive among these types. We then analyze the average normalized throughput of each type, as shown in Table III. We observe that among all types, Type II is the most sensitive as it suffers the lowest normalized throughput of down to 49.69%.

TABLE III
AVERAGE NORMALIZED THROUGHPUT OF TARGET VNF.

Type	I	II	III	IV	V
Normalized Throughput	87.64%	49.69%	64.07%	64.78%	61.14%

We draw the major observations of this section as follows:

- The performance interference among VNFs is ubiquitous and severe, causing performance degradation ranging from 12.36% to 50.3%. For different co-located VNFs, the performance degradation is different.
- Type IV and Type V are two of the most aggressive among tested VNFs, because the normalized throughput of target VNF is always lowest when they are competitors.
- Type II is the most sensitive among tested VNFs, as its average normalized throughput is lowest among all VNFs.

B. Performance interference and resource competition

From the aspect of resource, the performance interference may be caused by resource competition of network I/O bandwidth, CPU, cache, and memory. In this section, firstly, we learn the effect of network I/O bandwidth competition on performance interference. Secondly, for other resources, we collect each VM's resource utilization to identify competitive resources.

Since all VNFs belong to network I/O applications, the network I/O bandwidth must be one of important competitive resources [18]. Fig. 4 shows the performance degradation when VNFs run in different pairs with network I/O bandwidth guarantee. We find that all types of VNFs suffer network I/O bandwidth competition as the normalized throughput in Fig. 4 is higher than that in Fig. 3 where VNFs run without network I/O bandwidth guarantee. The performance degradation caused by network I/O bandwidth competition accounts for half of total degradation, meaning that network I/O bandwidth competition plays an important role in performance interference between VNFs. Compared with Fig. 4, the trends of normalized throughput are same except Type III. Without network I/O bandwidth guarantee, more CPU resource is consumed when VNFs receive packets from host server. The competition of CPU resource of Type III when it runs with Type IV or Type V is more serve, resulting in much more performance degradation of Type III. We also find that the level of performance interference caused by network I/O bandwidth competition of different co-located VNF pairs is different.

We further analyze the competition of other resources by the measurement with network I/O bandwidth guarantee. We use `dstat`, `ps` command and `perf` to record the virtualization specific system-level characteristics, such as CPU utilization, resident memory, the number of cache hits per second, which help to reveal the intrinsic factors causing performance interference. In the following, we analyze the relationship between performance interference and these system status by type of VNFs.

1) Type I: As illustrated in Fig. 5, we find that the performance degradation of Type I, the packet loss rate as well as the response time are small, no matter which type of VNF it runs with. And Type I's performance is barely affected when it runs with Type IV or Type V. Among all system status, the variation of normalized throughput generally keeps pace with CPU utilization, showing that Type I is sensitive to CPU resource. We also observe that the number of cache hits has impact on the performance of co-located VNFs. Note that although the CPU utilization of Type I when it runs with Type V is higher than the utilization when it runs with Type IV, the normalized throughput is lower. The reason lies in the number of cache hits that more cache hits help to decrease the CPU utilization used to wait for packet accessing. Moreover, for Type I, resident memory is only used to store rule set rather than processing packets, as this type of VNF will discard packets immediately when its processing capacity cannot keep up with incoming packets. The resident memory of Type I is at about 980 kBytes no matter what VNF it runs with, meaning that memory resource is not the sensitive resource for Type I.

2) Type II: As shown in Fig. 6, the fluctuation of packet loss rate and response time are consistent with the trend of performance degradation. Like Type I, it is easy to conclude that Type II is sensitive to CPU resource as the variation of normalized throughput is consistent with CPU utilization when Type II runs with Type I and Type III. When running with Type IV and Type V, CPU utilization of Type II almost reaches to 100%, intense competition for CPU resource causes significant performance degradation of Type II. Furthermore, unlike Type I, the number of cache hits of Type II when it runs with Type I is higher than that when it runs with Type III, however, the trend of normalized throughput is opposite. This phenomenon implies that the impact of the number of cache hits is too limited to offset the effect of CPU resource and Type II is more sensitive to CPU rather than cache. We also observe that the variation of resident memory is limited when Type II runs with other VNFs, which means that memory resource is not the sensitive resource of Type II.

3) Type III: Fig. 7 depicts the system status of VM on which Type III runs. The packet loss rate and the response time are small. And the normalized throughput of Type III when it runs with Type IV or Type V is higher than that when it runs with Type I and Type II, which is generally consistent with CPU utilization. It means that Type III is sensitive to CPU resource. Similarly to Type I, the number of cache hits also has impact on the performance of Type III. Although the CPU utilization of Type III when it runs with Type I is higher than the utilization when it runs with Type II, the normalized throughput is lower. That is because that more cache hits mean less CPU utilization used to wait for packet accessing. Resident memory is constant with different competitors and is ten times as much as that of other VNFs, showing that Type III is insensitive and intensive to memory resource.

4) Type IV: As described in Fig. 8, the normalized throughput of Type IV takes a significant decreasing trend when it runs with different VNFs from Type I to Type V, while packet loss rate and response time are opposite. No matter which type the competitor is, the packet loss rate is less than 2% and the response time ranges from 0.1s to 0.2s. We find that the CPU utilization always reaches 100%, showing a hard competition for CPU resource. The measurement results from Fig. 5 to Fig. 9 also show that, all tested types of VNFs can be sorted in

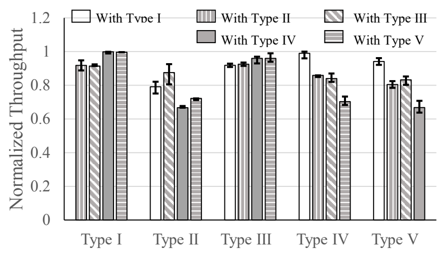

Fig. 4. Normalized throughput of each VNF in different pairs with network I/O bandwidth guarantee.

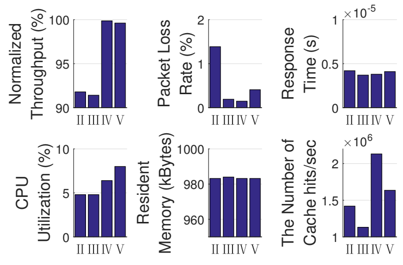

Fig. 5. VM's resource utilization where Type I runs.

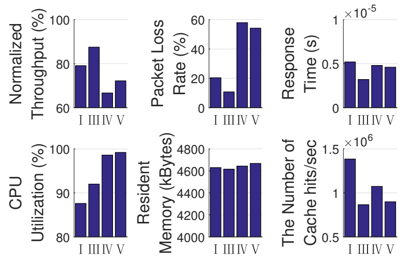

Fig. 6. VM's resource utilization where Type II runs.

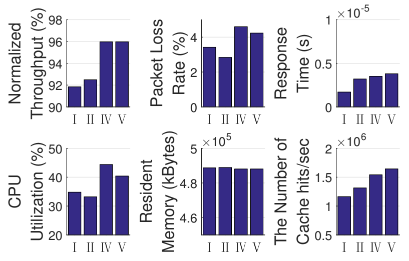

Fig. 7. VM's resource utilization where Type III runs.

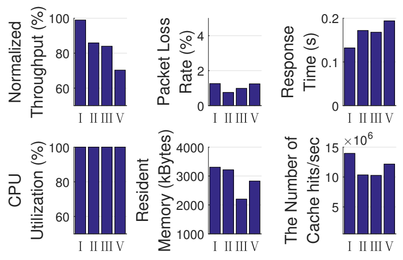

Fig. 8. VM's resource utilization where Type IV runs.

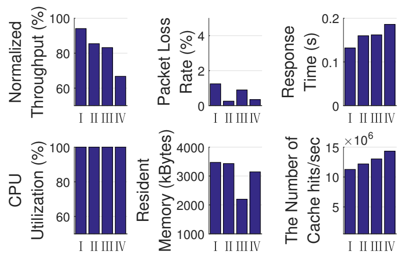

Fig. 9. VM's resource utilization where Type V runs.

descending order of CPU resource usage: after Type V would come Type IV, then Type II, then Type III, then Type I. When the competitor consumes less CPU resource, the blocking time of Type IV is shorter and Type IV can get CPU time slices more quickly, leading to better performance. Moreover, like cache hits of Type I, resident memory also has effect on the performance of Type IV: more resident memory means more space to store packets to be processed. And hence the performance of Type IV when it runs with Type II is higher than that when it runs with Type III, although Type II needs more CPU resource than Type III. Furthermore, we find a trend of decreasing normalized throughput of Type IV as the resident memory decreases except when it runs with Type V whose representative application is the same as Type IV. The reason is that Type IV and Type V own lots of sharing memory, making more resident memory. Besides, the number of cache hits has little effect on Type IV's performance, showing Type IV is insensitive to cache resource. Type IV is intensive to cache resource, as the number of its cache hits is an order of magnitude larger than other types of VNFs.

5) Type V: Fig. 9 illustrates that the trend of performance metrics and resource utilization state of Type V is almost the same with that of Type IV in Fig. 8. The difference is that the additional port-rewriting operations of Type V make the CPU resource competition severer, causing higher performance degradation.

According to these analyses, we conclude the major observations as follows:

- The performance interference among VNFs is caused by resource competition. Specially, the network I/O bandwidth competition plays a key role in performance interference.
- VNF's performance has a positive correlation with the amount of its sensitive resources. Among all tested types of VNFs, they are all sensitive to network I/O bandwidth and CPU resource. Besides, both Type I and Type III are also sensitive to cache resource; both Type IV and Type V are also sensitive to memory resource.
- Intensive resource has no effect on the performance interference. Among all tested types of VNFs, Type III is intensive to memory resource, and Type IV as well as Type V are intensive to cache resource.
- All tested types of VNFs can be sorted in descending order of CPU resource usage: after Type V would come Type IV, then Type II, then Type III, then Type I, and their corresponding operations are packet rewriting, link status updating, table look-up, and packet reading, respectively.

C. The impact of the characteristics of packet on performance interference

In this section, we proceed to investigate whether the packet characteristics impact the performance of VNF instances. As discussed in Sec. IV-B, the trends of response time and packet loss rate are generally consistent with the normalized throughput, hence we only utilize normalized throughput as the performance metric in the following analyses. Moreover, we find that the performance of Type I shows the same trend with Type III when packet characteristics vary, and they also have the same sensitive resources. The same observations hold

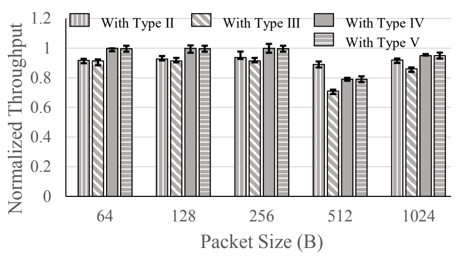

Fig. 10. Normalized throughput of Type I with different packet sizes.

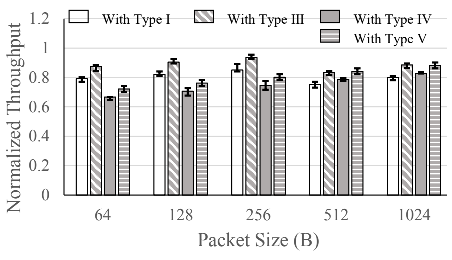

Fig. 11. Normalized throughput of Type II with different packet sizes.

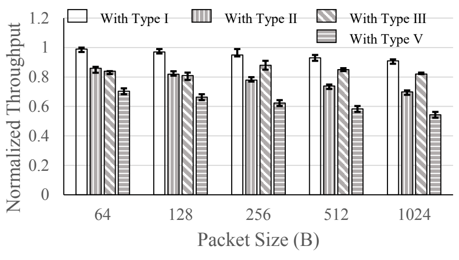

Fig. 12. Normalized throughput of Type IV with different packet sizes.

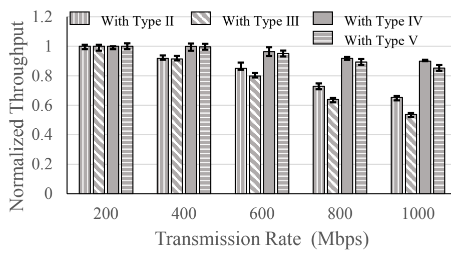

Fig. 13. Normalized throughput of Type I with different transmission rates.

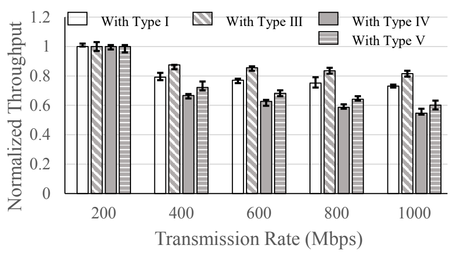

Fig. 14. Normalized throughput of Type II with different transmission rates.

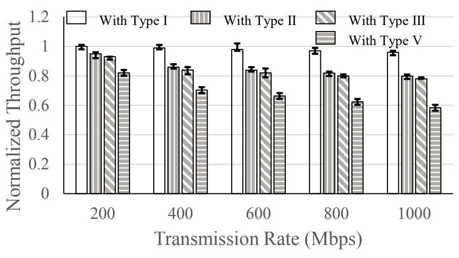

Fig. 15. Normalized throughput of Type IV with different transmission rates.

for Type IV and Type V. Thus, we select Type I, Type II, and IV as the representatives in the following measurements.

Different packet sizes. To see the impact of packet size on the performance interference among co-located VNFs, we utilize the traffic server to send packets with sizes from 64 B up to 1024 B, at the transmission rate of 1 Gbps. The results of three representative VNFs are shown in Fig. 10, Fig. 11, and Fig. 12, respectively.

As shown in Fig. 10, the normalized throughput of Type I is stable when the packet size varies from 64 B to 256 B, and it declines when the packet size is no less than 512 B. When the packet size becomes larger, each network I/O request needs to transmit more data, resulting in longer I/O processing time and more occupied memory pages, which is also called "world switch" cost. That is the reason of performance degradation of Type I. As the world switch cost of 512 B of packet size is larger than that of 1024 B of packet size, the normalized throughput of Type I rises when the packet size increases from 512 B to 1024 B [19]. We also find that the normalized throughput has the most unapparent decline when Type I runs with Type II. As we mentioned in Sec. IV-B, Type I is sensitive to cache resource. When the packet size increases under the same network transmission rate, the number of packets decreases. Less cache is needed to store packet headers during Type I's and Type II's computing, leading to less cache resource competition, and hence less performance degradation.

Now, we look at Fig. 11. In the case of Type II running with Type I or Type III, the normalized throughput raises when the packet size increases from 64 B to 256 B. This is because the cache competition between Type II and other VNFs can be mitigated when packet size rises. And then the performance decreases when the packet size further increase because of the world switch cost. However, when Type II runs with Type IV or Type V, the performance of Type II keeps growing when the packet size increases. As mentioned in Sec. IV-B, the performances of Type IV and Type V are affected by blocking time. When packet size becomes larger, Type II needs more CPU time to receive packets, leading to longer blocking time of Type IV and Type V. The performances of Type IV and Type V decline, resulting in less resources consumed by Type IV and Type V. Thus, Type II has more resources to improve performance. Moreover, when the packet size is larger than 256 B, we find that the increasing rate of Type II's normalized throughput declines, whose reason lies in the world switch cost.

Fig. 12 shows that Type IV's throughput decreases when packet size varies from 64 B to 1024 B. As mentioned above, the performance of Type IV is limited by longer blocking time when packet size increases. And the world switch cost also pays an effort to lower normalized throughput. The level of performance degradation when Type IV runs with Type II or Type V is severer than it runs with Type I or Type III, because the CPU competition is more serve when Type IV runs with VNFs which consume more CPU resource.

Different transmission rates. To further understand the impact of transmission rate on the performance interference among co-located VNFs, we utilize the traffic server to send fixed-size packets with transmission rates from 200 Mbps to up to 1 Gbps. The results of three representative VNFs are shown in Fig. 13, Fig. 14, and Fig. 15, respectively.

As shown in Fig. 13, the normalized throughput of Type I is always less than 100%, and the performance degradation becomes more and more severe when the transmission rate increases. As mentioned above, network I/O bandwidth competition plays a key role in performance interference. As the transmission rate rises, network workloads become heavier, leading to fiercer network I/O bandwidth competition. Moreover, more CPU resource is consumed because VNFs receive more packets. Meanwhile more cache resource is consumed by packets as transmission rate rises. We observe that the normalized throughput of Type I when it runs with Type II or Type III is lower than it runs with Type IV and Type V. The reason lies on the cache resource competition between Type I and Type II or Type III is severer when transmission rate increases.

From Fig. 14, we can see clearly that the normalized throughput of Type II decreases when the transmission rate increases from 200 Mbps to 1000 Mbps because of competitions of network I/O bandwidth, CPU and cache. When Type II runs with Type I or Type III, we observe that the normalized throughput of Type II has a sharp decline when the transmission rate varies from 400 Mbps to 600 Mbps because CPU resource becomes the bottleneck. As depicted in Fig. 6, the CPU utilization of Type II is higher than 80% when it runs with Type I or Type III with network I/O bandwidth guarantee, showing its strong demand for CPU resource. As transmission rate increases, more CPU resource is consumed by packet receiving. When transmission rate exceeds 400 Mbps, the limited CPU resource can no more meet Type II's resource demand.

Finally, Fig. 15 shows that due to the competition for network I/O bandwidth, CPU resource, and memory resource, the level of Type IV's throughput degradation becomes higher as the transmission rate raises. As transmission rate rises, more packets need to be stored in memory, resulting in strong memory competition. We also observe that the normalized throughput of Type IV is highest when Type I is the competitor, and it has poorest performance when Type V is the competitor no matter what transmission rate is. It implies that, Type V is the most aggressive to Type IV and Type I has the weakest aggression on Type IV among all tested VNFs.

We draw the major observations of this section as follows:

- In general, for all tested types of VNFs except Type II, the performance of VNFs becomes poorer when the packet size increases. This is because of the higher world switch cost and longer blocking time.
- When Type II runs with Type IV or Type V, its performance improves when packet size rises, due to mitigated cache resource competition and more resource it owns.
- For most of VNFs, the performance interference between co-located VNFs becomes severer when the transmission rate rises, because of the intense competition for network I/O bandwidth, CPU, and cache.

V. Discussion

The measurement results of the performance interference among co-located VNFs have many potential applications. In this section, we provide some suggestions on how to utilize our observations and analyses to provide an efficient VNF placement approach, from the perspectives of resource allocation, location selection, and dynamic scaling.

Resource Allocation. The resource allocation mechanism should assign different resources to VNFs in different scenarios. First, different VNF instances need different types of and the different number of resources. For example, as mentioned in Sec. IV-B, Type III is intensive to memory resource while Type IV and Type V are intensive to cache resource. And VNFs with packet rewriting or link status updating operations consume more CPU resource than those only with packet reading operations. Second, the level of performance interference generally depends on the allocated resources. For example, when placing Type I and Type III, who are both sensitive to cache, on a same physical server, it is better to bind each VNF with different physical core to partly avoid cache competition.

Location Selection. The location of VNFs also has significant effect on the performance. Network operators must carefully decide where VNF instances should be placed to reduce the end-to-end latency. According to our measurement results, the level of performance interference is different with different co-located VNFs, and the operators could reach higher performance goals if placing VNFs properly. For example, among the tested VNFs, Type V is the most aggressive to Type IV while Type I has the weakest aggression to Type IV. That is, it is better to place Type IV to where Type I locates, rather than to the server where Type V runs.

Dynamic Scaling. As illustrated in Sec. IV, the performance interference among VNFs is ubiquitous and severe. However, the existing VNF placement approaches are all based on an impractical hypothesis that the resource and performance isolation between co-located VMs is thorough, leading to inefficient VNF instance scaling. For example, if we need a Type II instance to deal with additional traffic, the existing approaches would like to initialize an instance whose capacity exactly meets the traffic demand. However, as mentioned in Sec. IV-A, the performance of this instance may suffer up to 50.3% of throughput degradation due to performance interference. And the actual processing capacity cannot meet the demand immediately, leading to packet loss and latency. To take another example, suppose that Type I and Type II locate in a same physical server and deal with different traffic. When the network traffic processed by Type I has a sharp decline, the existing approaches would like to recall Type I. This decision would eliminate the performance interference and improve Type II's performance, which, however, leads to resource over-provisioning and hence waste of resources.

VI. Related Work

In this section, we briefly review existing work on NFV measurement and VM interference, as well as VNF placement.

NFV measurement. Wu et al. propose PerfSight, a system detecting the root causes of performance problems in software data planes by extracting and analyzing comprehensive low-level information of the various elements (e.g., pNIC driver and virtual switches) [10]. Xu et al. take the first step to investigate the power efficiency of different NFV implementations with extensive experiments [20]. Different from these works that focus on the software data planes' performance detecting or energy efficiency of VNF, our work is devoted to investigating performance interference of co-located VNFs.

VM interference. Park et al. study the performance interference of memory thrashing in virtualized cloud environment and offer some solutions to mitigate the effects of performance interference [21]. Different from this work, our work not only pays attention to the performance interference of memory thrashing but also the performance interference brought by other reasons, such as CPU, cache. Bu et al. present a task scheduling strategy to mitigate interference for MapReduce applications in virtual clusters, which preserves task data locality at the same time [22]. Xu et al. present a heterogeneity and interference-aware VM provisioning framework named Heifer, which offers corresponding configuration of VM based on the performance predictions of MapReduce applications [23]. Nonetheless, our work focuses on the performance interference of VNFs rather than that of MapReduce applications.

VNF placement. Moens et al. present and evaluate an integer linear program named VNF-P, which focuses on allocating resources for NFV service chains in a hybrid scenario [24]. Mehraghdam et al. propose a model depicting the service chain which composes of NFs, and allocate resource with a mixed integer quadratically constrained program [8]. Sang et al. solve the problem of joint placement and allocation of VNF instances in a new NFV-enabled networking paradigm using greedy algorithms [3]. Ma et al. present a traffic-aware VNF placement approach, which also takes account of the relationships among VNFs [4]. These works, however, have not taken the performance interference among co-located VNFs into account. Our work measures the performance interference among co-located VNFs and provides insights on VNF placement, offering opportunities to improve and supplement VNF placement approaches.

VII. Conclusion

In this paper, we classify VNFs into different types according to their packet operations, and present the study on performance interference among different types of co-located VNFs. To investigate the root cause of performance interference, we measure the VNFs' performance with different combinations of co-located VNFs, varying packet sizes and transmission rates. We find that the performance interference among VNFs is ubiquitous and severe, while the interference level shows differences with different groups of co-located VNFs. The reason is that the co-located VNFs compete for resources, usually network I/O bandwidth, CPU, cache, and memory. We also observe that the performance interference will be severer when packet size and transmission rate increase. Accordingly, we give some suggestions on how to make the VNF placement approach more efficient.

References

[1] NFV white paper. [Online]. Available: https://portal.etsi.org/NFV/NFV_White_Paper.pdf

[2] NFV white paper2. [Online]. Available: https://portal.etsi.org/NFV/NFV_White_Paper2.pdf

[3] Y. Sang, B. Ji, G. R. Gupta, X. Du, and L. Ye, "Provably efficient algorithms for joint placement and allocation of virtual network functions," in *Proc. of IEEE INFOCOM*, 2017.

[4] W. Ma, O. Sandoval, J. Beltran, D. Pan, and N. Pissinou, "Traffic aware placement of interdependent NFV middleboxes," in *Proc. of IEEE INFOCOM*, 2017.

[5] Q. Zhang, Y. Xiao, F. Liu, J. C. S. Lui, J. Guo, and T. Wang, "Joint optimization of chain placement and request scheduling for network function virtualization," in *Proc. of IEEE ICDCS*, 2017.

[6] Y. Li, L. T. X. Phan, and B. T. Loo, "Network functions virtualization with soft real-time guarantees," in *Proc. of IEEE INFOCOM*, 2016.

[7] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Transactions on Networking*, pp. 1–18, 2017.

[8] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *IEEE International Conference on Cloud Networking*, 2014.

[9] F. Xu, F. Liu, L. Liu, H. Jin, B. Li, and B. Li, "iAware: Making live migration of virtual machines interference-aware in the cloud," *IEEE Transactions on Computers*, pp. 3012–3025, 2014.

[10] W. Wu, K. He, and A. Akella, "Perfsight: Performance diagnosis for software dataplanes," in *Proc. of ACM IMC*, 2015.

[11] X. Pu, L. Liu, Y. Mei, S. Sivathanu, Y. Koh, and C. Pu, "Understanding performance interference of I/O workload in virtualized cloud environments," in *IEEE International Conference on Cloud Computing*, 2010.

[12] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *IEEE International Conference on Cloud Networking*, 2015.

[13] S. Govindan, J. Liu, A. Kansal, and A. Sivasubramaniam, "Cuanta: Quantifying effects of shared on-chip resource interference for consolidated virtual machines," in *Proc. of ACM SoCC*, 2011.

[14] R. C. Chiang and H. H. Huang, "Tracon: Interference-aware schedulingfor data-intensive applicationsin virtualized environments," *IEEE Transactions on Parallel and Distributed Systems*, pp. 1349–1358, 2014.

[15] Q. Huang and P. P. Lee, "An experimental study of cascading performance interference in a virtualized environment," *SIGMETRICS Perform. Eval. Rev.*, 2013.

[16] S. G. Kulkarni, W. Zhang, J. Hwang, S. Rajagopalan, K. K. Ramakrishnan, T. Wood, M. Arumaithurai, and X. Fu, "NFVnice: Dynamic backpressure and scheduling for nfv service chains," in *Proc. of ACM SIGCOMM*, 2017.

[17] V. Sekar, N. Egi, S. Ratnasamy, M. K. Reiter, and G. Shi, "Design and implementation of a consolidated middlebox architecture," in *Proc. of USENIX NSDI*, 2012.

[18] Y. Mei, L. Liu, X. Pu, and S. Sivathanu, "Performance measurements and analysis of network i/o applications in virtualized cloud," in *IEEE International Conference on Cloud Computing*, 2010.

[19] C. Li, C. Ding, and K. Shen, "Quantifying the cost of context switch," in *Proc. of Workshop on Experimental Computer Science*, 2007.

[20] Z. Xu, F. Liu, T. Wang, and H. Xu, "Demystifying the energy efficiency of network function virtualization," in *Proc. of IEEE/ACM IWQoS*, 2016.

[21] J. Park, Q. Wang, J. Li, C. A. Lai, T. Zhu, and C. Pu, "Performance interference of memory thrashing in virtualized cloud environments: A study of consolidated n-tier applications," in *IEEE International Conference on Cloud Computing*, 2016.

[22] X. Bu, J. Rao, and C.-z. Xu, "Interference and locality-aware task scheduling for mapreduce applications in virtual clusters," in *Proc. of ACM HPDC*, 2013.

[23] F. Xu, F. Liu, and H. Jin, "Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud," *IEEE Transactions on Computers*, pp. 2470–2483, 2016.

[24] H. Moens and F. D. Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *IEEE International Conference on Network and Service Management*, 2014.