Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 9928210, 21 pages
https://doi.org/10.1155/2021/9928210

Research Article

Virtual Network Resource Optimization Model for Network Function Virtualization

Đani Vladislavić,[1] **Darko Huljenić**,[1] **and Julije Ožegović**[2]

[1]*Ericsson Nikola Tesla d.d., Zagreb, Croatia*
[2]*Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, University of Split, Croatia*

Correspondence should be addressed to Đani Vladislavić; djani.vladislavic@ericsson.com

Received 25 March 2021; Revised 15 June 2021; Accepted 21 July 2021; Published 17 August 2021

Academic Editor: Ruhui Ma

Network function virtualization (NFV) is a concept aimed at achieving telecom grade cloud ecosystem for new-generation networks focusing on capital and operational expenditure (CAPEX and OPEX) savings. This study introduces empirical throughput prediction model for the virtual network function (VNF) and network function virtualization infrastructure (NFVI) architectures based on Linux kernel. The model arises from the methodology for performance evaluation and modeling based on execution area (EA) distribution by CPU core pinning. EA is defined as a software execution unit that can run isolated on a compute resource (CPU core). EAs are derived from the elements and packet processing principles in NFVIs and VNFs based on Linux kernel. Performing measurements and observing linearity of the measured results open the possibility to apply model calibration technique to achieve general VNF and NFVI architecture model with performance prediction and environment setup optimization. The modeling parameters are derived from the cumulative packet processing cost obtained by measurements for collocated EAs on the CPU core hosting the bottleneck EA. The VNF and NFVI architecture model with performance prediction is successfully validated against the measurement results obtained in emulated environment and used to predict optimal system configurations and maximal throughput results for different CPUs.

1. Introduction

Modeling of network function (physical) service performance was a straightforward task determined by well-defined test scope during verification activities in telecom vendors' laboratories for a long time. Models coming from the labs were successfully used for system dimensioning at telecom service providers (TSPs). Determinism of the network function service performance going virtual (virtual network function (VNF)) has been lost due to variety of systems (network function virtualization infrastructures (NFVIs)) it may be deployed on. These can have different hardware (HW) capabilities, virtualization layer, acceleration technologies applied, and the possibility of multi-VNF deployments. Having all deployment options verified in telecom vendor labs (even if it would be realistic) would imply huge development costs and as such would make capital expenditure (CAPEX) cost increase inevitably. This can jeopardize the main idea of NFV, aiming for CAPEX and operational expenditure (OPEX) reduction. Identifying critical VNF and NFVI factors and bottlenecks impacting VNF service performance and providing a model for system dimensioning in specific NFVI architectures become an important step for further expansion of NFV [1].

This study proposes VNF and NFVI architecture model with performance prediction based on measurements on the commodity multicore system and by applying model calibration technique. This is the first study proposing VNF throughput prediction model validated in emulated environment. The target NFVIs in this study are based on commodity hardware and on the state of the art of generic Linux kernel processing using new API (NAPI) I/O framework and open virtual switch (OVS), providing network virtual device emulation within the kernel (vhost-net). A proposed model can be used to predict throughput for any virtual machine- (VM-) based VNF using paravirtual (virtio)

network device drivers in guest operating system (OS) and QEMU/KVM as the hypervisor in relation to target NFVI configuration. Yet, it is required that the white box view of its software architecture, system bottlenecks, and throughput requirements are known in order to be applied in the model. The representative VNF used for validation and throughput performance prediction in this study is simple kernel-based switching function (OVS).

Following related researches, the model defined in the study presumes CPU processing capacity to be the bottleneck in packet processing systems. The model arises from the novel methodology of throughput evaluation based on execution area (EA) distribution derived from general packet processing steps in operating system kernel defined in our previous work [2]. This study extends the work in [2] to consider processing in VMs. The proposed methodology, in accordance with empirical EA bottleneck determination, is used to build a set of optimal modeling system configurations for the target VNF and NFVI architectures per given number of CPU core resources. For each of the modeling system configurations, an empirical analysis of throughput performance is performed. Model calibration and validation are done against testbed measurements for all defined modeling system configurations. The defined VNF and NFVI architecture model with performance prediction is further used to predict the throughput performance and provide optimal VNF/NFVI system dimensioning for any given number of CPU cores.

2. Related Work

Performance evaluation and modeling of packet processing for virtual environments are research topics for already more than a decade. This chapter provides an overview of the known empirical and analytical models defined for packet processing systems, comparing contribution of these models for applying in modeling of VNF service performance. The related work is divided into three basic categories: mathematical models, measurements of processing steps, and combination of previous two based on empirical modeling.

In 2006, Wu et al. [3] proposed a mathematical model for packet reception in a user-space application. They described the chain of components involved in packet processing from the reception at NIC up to delegation to application. They modeled the NIC by token bucket algorithm where tokens presented the packet descriptors and the remaining elements in the chain were modeled as queuing processes [4]. Baynat et al. [5, 6] provided numerical models for virtual switching systems based on Markov chains. These models were verified in simulated environment only, and their applicability to real or emulated environment including VNFs is questionable due to basic assumptions taken (such as not considering NIC emulation shown to be the main bottleneck in the system). For these reasons, we applied combination of mathematical and measurement-based modeling in our study (empirical modeling). Further analytical models based on queuing theory and network calculus applied in software-defined networking (SDN) were presented in [7–11]. Sattar and Matrawy [12] provided an overview on the analytical models questioning their applicability due to simplification of the network elements to provide a mathematical formulation.

Bruschi and Bolla [13] in 2007 presented detailed processing steps in Linux software router implementation based on Linux kernel version 2.6. They applied RFC 2544 compliant test cases for software router performance analysis. In their work [14] in 2008, they concluded that smart CPU/core allocation to multiple NIC receiving and sending buffers gives the possibility of scaling performance of software routers. Dobrescu et al. [15] in 2009 closely reached 10 Gbps throughput for the shortest packets by revisited software router architecture. Our work assembles the knowledge of the scalable software architectures presented in these papers for building our EAs and related empirical throughput prediction model. Emmerich et al. [16] compared different forwarding engines of the Linux kernel, namely, software router, Linux bridge, and open virtual switch (OVS) [17], giving slight throughput advantage to OVS. The authors concluded that OVS throughput linearly scales with the number of flows, until the number of used cores equals the number of flows. This work also examined the throughput capabilities of a VM attached to an OVS, highlighting the huge throughput downgrade for the VM case compared to configuration without the VM. Applying the Linux profiling tool perf, the authors concluded the main reasons for the throughput downgrade are increased context switching and functions related to packet queues. Unfortunately, the work provided no details on the applied VM network I/O mechanism, only highlighting one core was used for the OVS and one for the VM. In the subsequent work [18], the same authors applied perf profiling to OVS, Linux bridge, and software router, concluding that the main performance bottleneck is the generality of operating system's network stack implementation. The authors also provided an overview of potential hardware bottlenecks (network bandwidth, NIC processing capacity, PCIe bandwidth, memory bandwidth, and CPU cache size) but concluded they are not related to packet forwarding throughput limitation for the input traffic rate not exceeding 10 Gbps. They compared the results of kernel-based forwarding engines to user-space forwarding engines (OVS DPDK), claiming a sixfold throughput increase for user space switches. Similar results were presented by Paolino et al. in the work [19] where they introduced SnabbSwitch user space virtual switch. The performance decrease due to implementation overheads in kernel was also reported in [20, 21]. The authors in [22] performed packet header analysis and modifications, reporting no major impacts on performance. Kawashima et al. [23, 24] provided a common performance benchmark for various I/O frameworks (NAPI, data plane development kit (DPDK), and netmap) and related forwarding engines (OVS, OVS DPDK, and VALE [25, 26]), focusing on latency/jitter and throughput bottlenecks in the virtual switching technologies. They derived similar conclusions for NAPI as authors in [16, 18], pointing the generality of the network stack to be the main bottleneck of the packet processing in kernel. The measurement results in this work showed the maximum throughput on 4.0 GHz processor core for OVS kernel-space forwarding to be 1.6–

2.0 Mpps and for OVS kernel-space forwarding with virtual NIC emulation (vhost-net) to be 0.5–1.0 Mpps, for any frame sizes of single constant bit rate (CBR) UDP flow. Although the authors highlighted NIC I/O (vhost-net) packet relay overhead to be the main bottleneck for the packet processing towards VMs, they provided no further system configuration details nor detailed bottleneck analysis. Our work points to different software (EA) bottlenecks in relation to applied system configurations.

Meyer et al. [27] measured and simulated the performance of software routers based on multicore systems. After calibrating and validating their model based on testbed measurements, they evaluated and predicted the performance of existing and future software router architectures. The CPU is considered to be the bottleneck of the system. From their testbed measurements, they derived the relation that the throughput of the software router follows a linear behavior, depending on the number of used CPU cores and the frame size. The model proposed in this work provides fair results for the simplistic router network function (NF) architecture, such as the one considered in the work. However, in more complex architectures, such as when the network functions are being virtualized in the context of network function virtualization (NFV), the model cannot be applied. Our model, on the other hand, has the ambition to be applied for these kinds of architectures. Our previous work [2] presented throughput evaluation methodology based on execution area (EA) distribution derived from general packet processing steps in kernel packet switching. We analyzed throughput results at different points in the system in relation to EA distribution by core pinning, which is considered as CPU resource consumption modeling parameter for the optimal throughput in different scenarios where virtual switching is applied. We presented the results for different NIC settings of interrupt coalescing (IC), and we made the relation of the observed throughput results to NAPI processing key performance indicators (KPIs). Raumer et al. [28] introduced an empirical model for calculating maximum packet rate of Linux software router, claiming to be designed in conformance with various high-speed packet forwarding frameworks. The model and its validation were focused on single core processing as software routers scale linearly with the number of cores due to optimal parallelism [16]. Same as for the model in [27], for more complex architectures in the context of NFV, this model cannot be applied. The work in [29] measured and simulated how networking software like network interface card (NIC) driver and the OS network stack influence the packet latency. They analyzed NIC driver and OS mechanisms with respect to packet processing based on commodity hardware and in respect to adaptive IC rate algorithm. They calibrated the model according to measurement and profiling results obtained in their testbed setup. Differently than this work that focuses on packet latency in relation to IC rate, our study considers IC rate in relation to packet throughput.

A detailed knowledge of the complex systems is required by the scientists aiming for low level optimizations and performance predictions of these systems. Tremendous effort has been given by the researchers in the last years for benchmarking, modeling, extending, and in general understanding virtual technologies on commodity hardware. Simplistic empirical models are shown within these researches to have better applicability to the real systems compared to analytical models existing today. Empirical model for throughput prediction and throughput-optimized environment setup is indeed the focus of this study.

3. Execution Area-Based Throughput Prediction Modeling

3.1. Approach. As highlighted by majority of benchmarking and modeling researches presented in Chapter 2, the typical and most prominent bottleneck of the packet processing systems is the software processing at CPU and the respective software design of packet processing system. These are the basic assumptions followed in throughput prediction model construction.

The elements of the NF/VNF referent system architecture are CPU exhaustive elements running in kernel space of the NF/VNF. The VNF (together with QEMU emulation) runs in the user-space of the host. The remaining elements of the VNF/NFVI referent system architecture are CPU exhaustive elements running in kernel space of the host. In order to build a throughput prediction model of this complex software architecture in relation to multicore CPU architectures, the architecture needs to be presented in a per-core executable modeling elements, simple to (re)build and measure in any multicore CPU architectures.

The previous work [2] introduced the methodology of performance evaluation of packet processing systems based on execution areas (EAs). This methodology is further used as a baseline for throughput prediction model construction. EA can be defined as a software execution unit that can run isolated on a compute resource (CPU core). EAs are derived from the elements of referent system architecture and related packet processing principles. Main criteria characteristics of an EA are as follows:

(i) It can run on an isolated core for the single flow traffic

(ii) It can run collocated with any other EA

(iii) It is CPU exhaustive execution unit influencing single flow packet processing capacity

(iv) It has well-defined input and output interfaces

(v) It is scalable with number of flows jointly with other units

The defined EAs are further used to build throughput optimized system configurations for the testbed measurements in terms of EA colocations for the given number of CPU cores and the single flow to be served. System configurations are denoted with "$x + y$", where "x" presents the number of cores used by the NFVI and "y" presents the number of cores used by the VNF. A prerequisite for each system configuration is to be scalable with the number of flows so that they can further be used to build a simple as possible throughput

prediction model. Besides by the EA colocations, each system configuration is defined by NIC IC settings limited to two options, off (favoring small latency) and set to 125 μs (optimizing the number of expensive HW IRQs towards CPU). The 125 μs constant is chosen empirically as it shows significant difference in VNF service performance comparing to IC off. The system configurations are built gradually based on bottleneck EA determination, starting from the minimal granularity system configuration ("1 + 1" + IC off/125) up to the bottleneck EA being fully isolated on the CPU resource ("5 + 2" + IC off/125). The minimal granularity system configuration assures the basic requirement of separating the VNF CPU resources from the NFVI CPU resources. The bottleneck EA determination is performed using representative kernel-based virtual switching VNF. The model can be applied to any VNF though, as long as its software architecture, system bottlenecks, and throughput requirements of the VNF within target NFVI are known. Since the CPU resources of the VNF and NFVI are separated, it is expected that identified bottleneck EAs within target NFVI remain irrespective of the VNF applied in the system. A bottleneck EA is obtained empirically for each measurement scenario and for each system configuration applied in testbed. A measurement scenario is defined by the number of constant bit rate (CBR) flows of the fixed frame size.

Since the software processing speed at CPU is assumed to be the bottleneck of the packet processing systems in general, it can be assumed the empirically determined bottleneck EAs for system configurations and measurement scenarios applied in testbed to be valid for the same system configuration in any other testbed. Under the same assumption, it can be claimed the CPU frequency and number of CPU cores are the factors to determine the throughput performance of any CBR measurement scenario, defined by the fixed frame size and number of flows, in any of the defined system configurations. The former reveals the possibility of predicting throughput performance of each system configuration individually in relation to traffic input (frame size and number of CBR flows) and CPU hardware architecture (CPU frequency and number of cores).

The throughput prediction model for each throughput-optimized system configuration in this study arises from measurement results applied in testbed based on model calibration technique. Due to <0.1% difference between the measurements of ("3 + 1," "4 + 1," "4 + 2," and "5 + 2") + IC off and ("3 + 1," "4 + 1," "4 + 2," and "5 + 2") + IC 125 pairs, the calibration is arbitrarily applied for the former and the predicted results are considered applicable for both pairs. Model calibration is performed based on measurements of cumulative packet processing cost of collocated EAs in cycles/packet for the CPU core hosting the bottleneck EA. The measurements are conducted for three times, and the mean value is taken for the model calibration. The repeated measurements are shown to have a small variation to the mean value (±1%) so the confidence bounds are omitted in this study. Building the throughput prediction model based on cost of packet processing ensures the applicability of the model to any CPU, regardless of frequency and number of cores. Modeling parameter extraction based on cumulative packet processing cost is described in Chapter 3.6. The throughput prediction model can further be used to predict the throughput for any measurement scenario and any of the defined modeling system configurations, for the given number of CPU cores and related CPU frequency. The throughput prediction model can also be used to provide an optimal system configuration for the required throughput or the optimal system configuration guaranteeing maximal throughput, for any given measurement scenario.

3.2. Software Execution Areas of the Representative NF/VNF. EAs in the NF/VNF referent architecture are depicted in Figure 1. Three CPU-exhausting EAs are extracted based on the presented EA criteria. 10 Gbps NIC in testbed processes packets on a line speed rate regardless of frame size so it cannot bottleneck the system throughput [2].

Receiving EA controls the tasks from the Rx HW IRQ handling up to delegation to forwarding function in kernel (OVS kernel data path). In a VNF case, this EA also embraces the KVM actions to kick the vhost-worker thread for informing it of the descriptors being stored in Rx available ring (depicted in Figure 2). Fwd & Snd controls the tasks for packet transmission through the qdisc and device driver for storing the descriptor in the ring buffer. In case of VNF, this EA also embraces the KVM actions to kick the vhost-worker thread for informing it of the descriptors being stored in Tx available ring and most of the transmission completions (depicted in Figure 2). Clearing performs the packet completions related to transmitted packets.

3.3. Software Execution Areas of the Target NFVI. EAs in the VNF/NFVI referent architecture are depicted in Figure 2. On top of the three EAs of the VNF presented earlier, 6 CPU-exhausting EAs are extracted in NFVI based on the presented EA criteria.

Receiving EA in NFVI embraces the same functions as Receiving EA in a NF. Fwd & Snd VM EA has the same input as Fwd & Snd in NF, but steers the packet towards the socket buffer, instead of pushing it towards qdisc. Receiving VM EA performs the ingress device emulation of the VM, including packet polling from the socket buffer, conversion to device specific virtio format, and pushing it to the ring buffer. It also includes communication towards KVM, including Rx HW IRQ (KVM irqfd) initiation towards VM.

After the VNF processing in respective EAs, Sending VM EA is kicked through the interprocess communication mechanism by KVM. This EA performs the egress VM device emulation of the VM, including ring buffer polling, conversion to kernel format, and pushing the packets in CPU backlog buffer. It also includes communication towards KVM, including Tx HW IRQ (KVM irqfd) initiation towards VM when needed. Fwd & Snd (NIC) EA polls the packets from the CPU backlog and pushes it through the qdisc and device driver to the Tx ring buffer. Clearing EA finally performs the transmission completion tasks for the sent packets.

3.4. EAs as a Server Queuing System. Figure 3 presents the previously defined EAs of the VNF/NFVI referent system architecture as a server queuing system. An overview of each

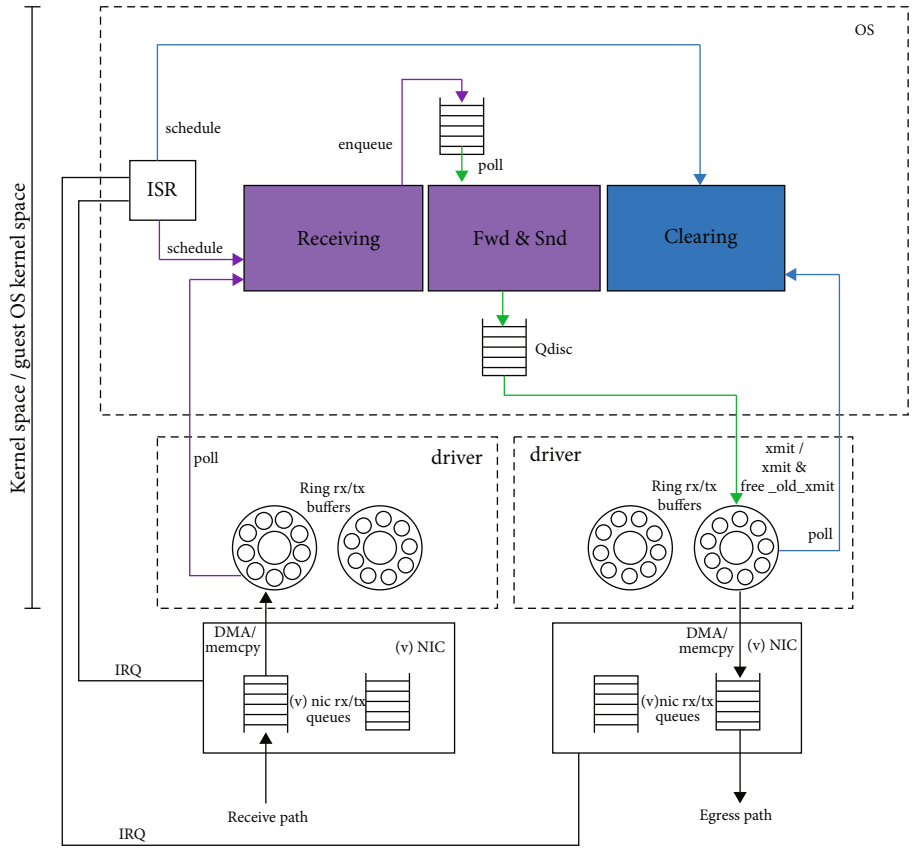


Figure 1: Execution areas in target NF/VNF.

queuing server in the context of the detailed processing steps description given previously, the notifications they send or receive and the relevant queues they handle are given below. Dotted red lines represent notifications that exist only for scenarios where the servers of the same colors are physically dislocated (to different CPU cores). The underlying marked queues exist as well only where these servers are physically dislocated.

3.5. Model Definition. Server queuing system described previously is modeled in Figure 4 showing EAs as bottlenecks. Each queuing server has an input queue it serves and an outbound queue(s) it fills to be processed by the following server in the sequence. The double B4 buffer between S3 and S4 from Figure 3 is simplified, showing one lossless buffer for completeness. The lossless double buffer B7 between B6 buffer and S7 from Figure 3 is merged into B6 buffer as it has no relevance for the throughput prediction model definition.

The modeling system configurations are linearly scalable by the factor f as $f * (x + y) <= n$, where "$x + y$" denotes the modeling system configuration irrespective of the IC time and n denotes the number of cores in the system. The simplified model of the VNF/NFVI system architecture is depicted in Figure 5. The 1st, 2nd, and f^{th}"$x + y$" represent a queuing server model depicted in Figure 4, constructed by specific modeling system configuration. Y_{in} represents the traffic input rate to the model, defined by N CBR flows of fixed frame size S and equal rate. The CBR flows are fairly distributed to f queuing server models, each with an Y_0 input rate.

Y_0 represents the traffic input rate to the queuing server model in packets per second (pps). Each server i processes packets at rate Y_i. For each packet to process, a queuing server needs a number of CPU cycles C_i to be executed per packet (cycles per packet). Packets that cannot be accepted for processing by the sever need to be dropped at the preceding server or even at prepreceding server with a packet rate Y_{i-1}^{-}, respectively, Y_{i-2}^{-}. The servers shown out of the packet processing chain do not explicitly contribute to queue entries, but they impact the processing capabilities of other servers. A specific queuing server i uses a CPU for a mean time of $\mu_i \leq 1$, determined by OS scheduler. The total throughput in packets per second that can be achieved by the specific queuing server can be defined as per

$$Y_i = \text{CPU}_{\text{freq}} * \frac{\mu_i}{C_i}. \quad (1)$$

CPU_{freq} is the static value representing the CPU frequency in cycles per second. Obviously, the queuing server limiting the throughput of each queuing server system will be the one resulting with the smallest μ_i/C_i.

As per definition, EAs (queuing servers) can be collocated on the same or dislocated to different CPU cores. The total number of packets per second that can be processed by the

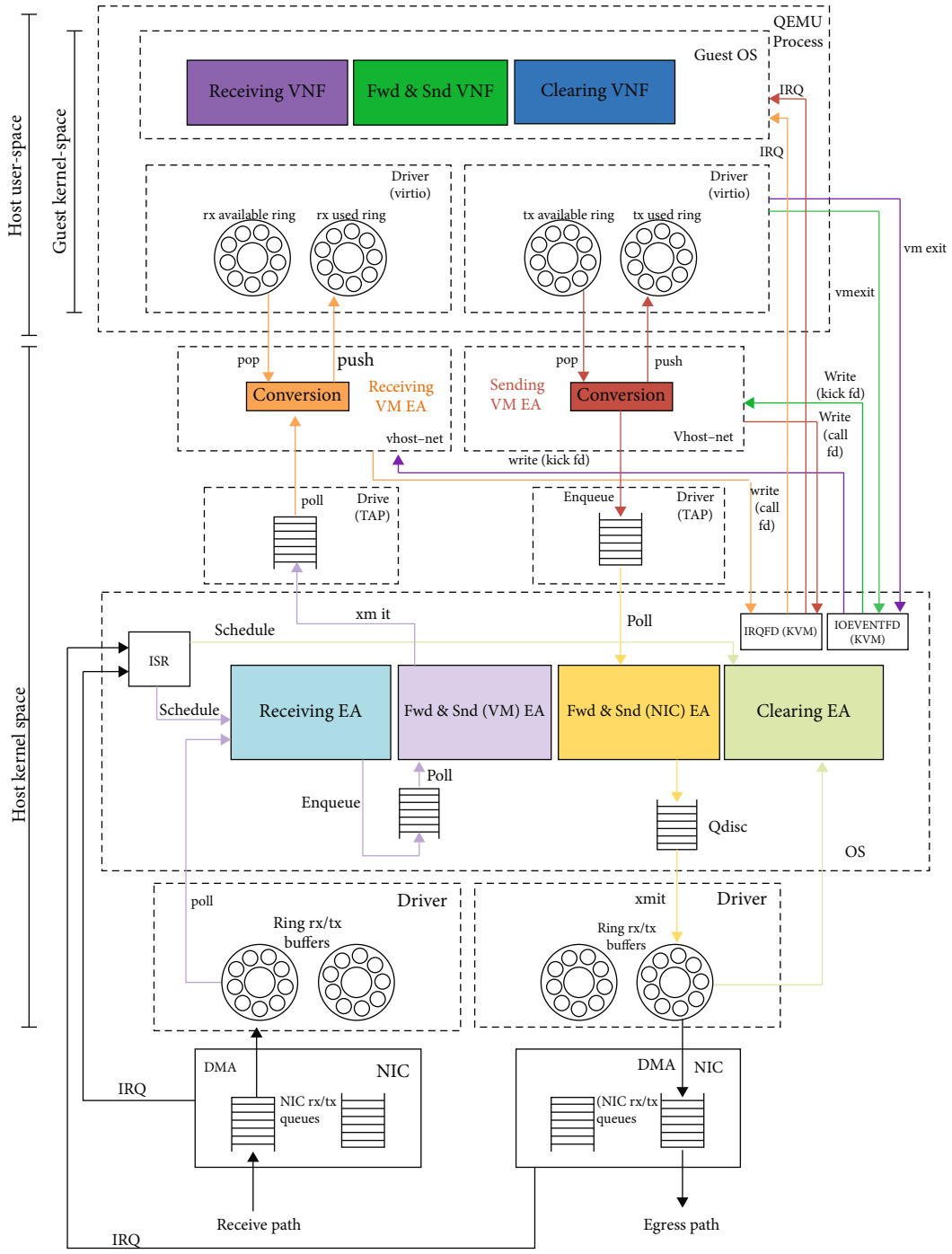


Figure 2: Execution areas in target VNF and NFVI.

specific CPU core k containing queuing server i, without experiencing packet losses, can be defined as per

$$Y_k = \frac{\text{CPU}_{\text{freq}}}{\text{Sum}(C_i)}, (1 \le i \le l). \quad (2)$$

The variable l presents the number of queuing servers allocated on core k. Since μ_i and C_i are influenced by several factors (server colocations, OS scheduler, cache, memory, etc.) leading to unpredictable values, the smallest μ_i/C_i is derived empirically based on the overflowed queues in measurements, without quantifying μ_i and C_i, assuming the heuristic to be valid in general.

Assuming the uniform traffic split into f CBR flows with constant frame size, the total throughput of the system is then defined as per

$$Y = f * \frac{\text{CPU}_{\text{freq}}}{C_f}. \quad (3)$$

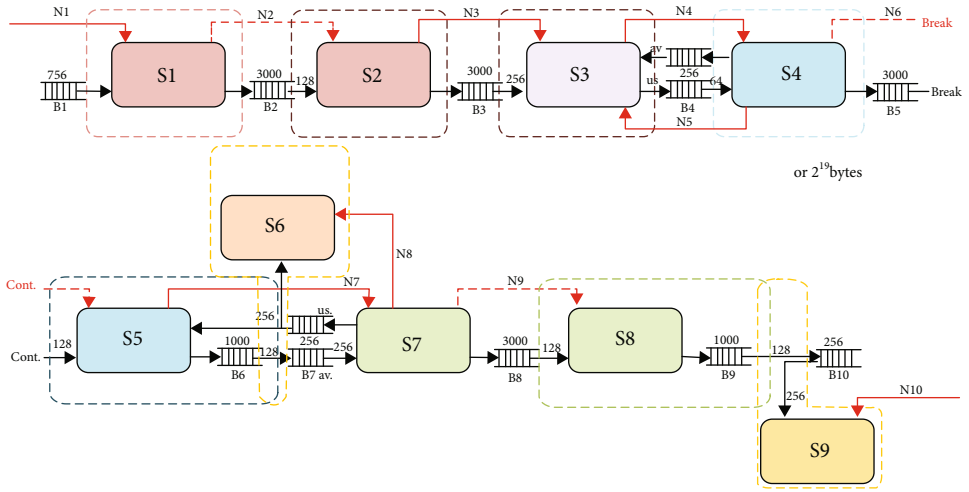


Figure 3: Server queuing system.

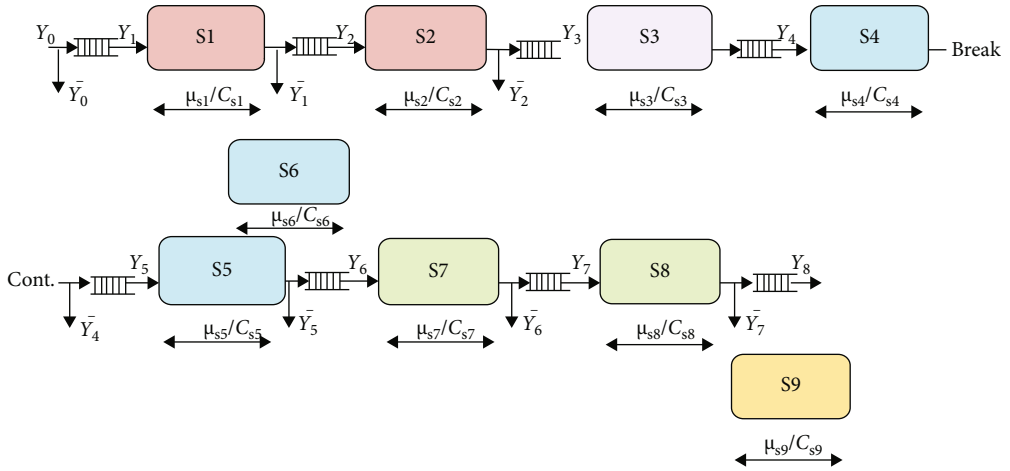


Figure 4: Queuing server model with EAs as bottlenecks.

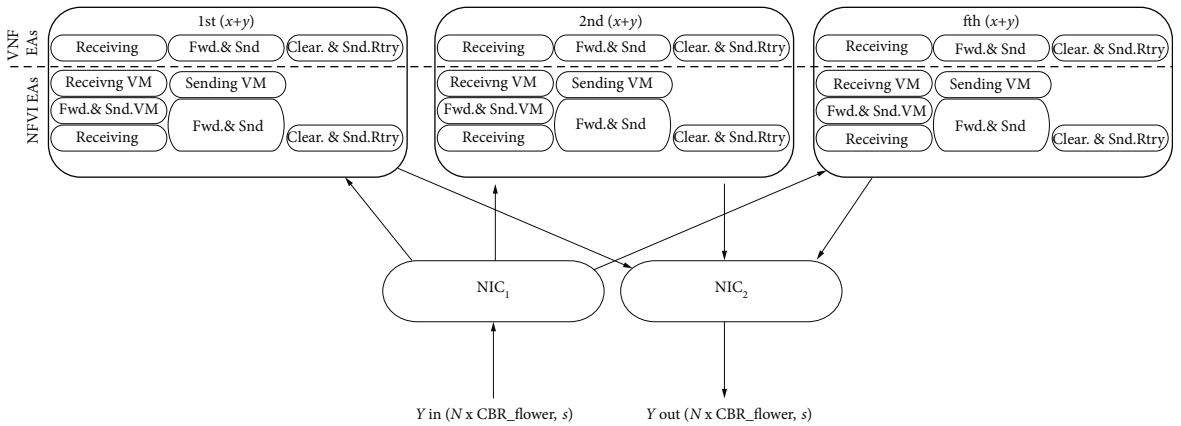
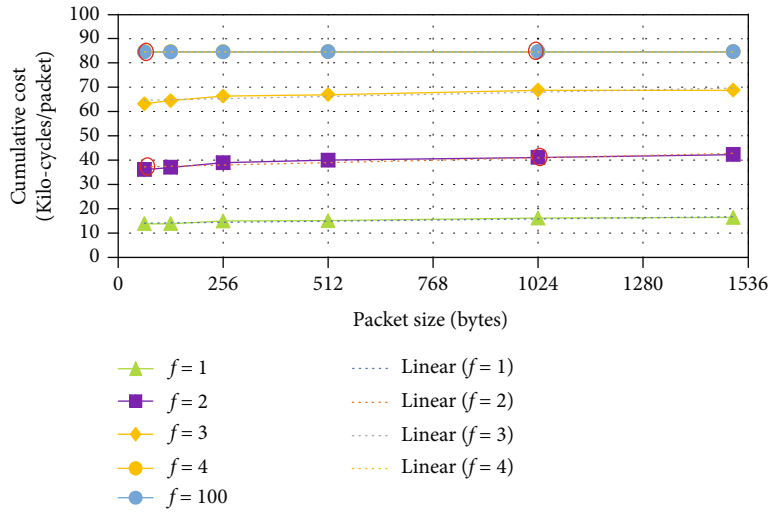
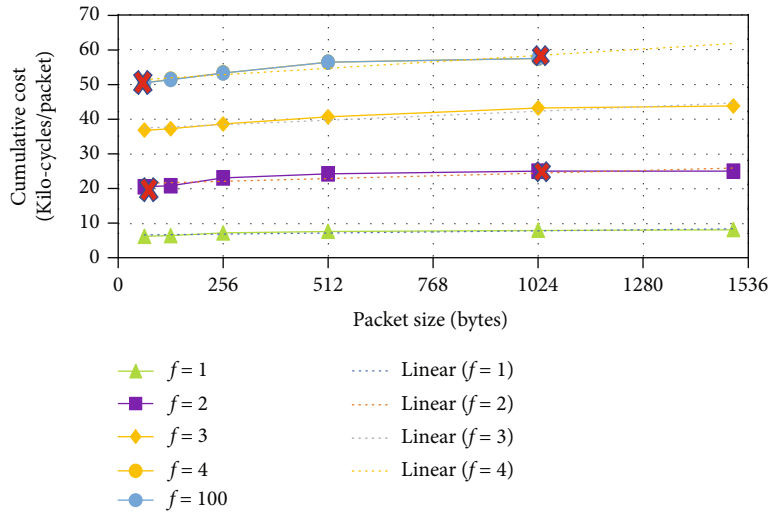


Figure 5: Simplified model of the VNF/NFVI system architecture.

$C_f = \mathrm{Sum}(C_i)\ (1 \le i \le l)$ denotes the cost in cycles per packet per each core hosting the queuing server with the smallest μ_i/C_i. Based on the observed measured costs (Figures 6 and 7), the heuristic relation is derived that cumulative cost per packet processing $C_{Cf} = f * C_f$ in cycles per packet follows the linear behavior for "1 + 1" + IC off, "1 +



(a) "1 + 1" + IC off



(b) "1 + 1" + IC 125

Figure 6: Cumulative cost: "1 + 1" + IC off and "1 + 1" + IC 125 modeling system configurations.

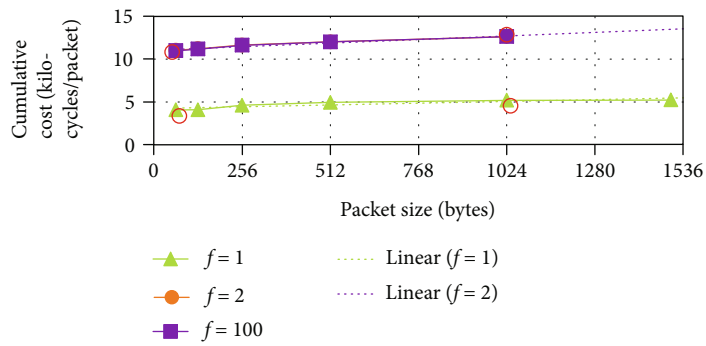


Figure 7: Cumulative cost: "3 + 1" + IC off/125 modeling system configuration.

1" + IC 125, "2 + 1" + IC off, "2 + 1" + IC 125, and "3 + 1" + IC off/125 modeling system configurations. It depends on the scale factor f and the frame size S according to

$$C_{Cf} = (a * f + a_0) * S + (b * f + b_0), \quad 1 \le f \le abs\left(\frac{n}{x+y}\right). \tag{4}$$

For the modeling system configurations requiring >8 CPU cores for calibration ("4 + 1" + IC off/125, "4 + 2" + IC off/125, and "5 + 2" + IC off/125), the assumption is taken that these modeling system configurations scale equally to "3 + 1" + IC off/125. This assumption is taken based on the fact that none of these modeling system configurations depend on the IC time when it comes to throughput, which is aligned with "3 + 1" + IC off/125. Following, it is assumed that the resulted heuristic in Equation (4) holds for arbitrary n-core CPUs if the offered load is uniformly split into f CBR flows with constant frame size S which are served by $f * (x + y) <= n$ CPU cores. The constant values for a, b, a_0, b_0 are derived from the empirical measurements through the model calibration, as described in Chapter 3.6. Based on Equations (3) and (4), the total throughput in megapackets per second (Mpps) of the system can be given as per

$$Y_{\text{Mpps}} = \min\left(\frac{f^2 * \text{CPU}_{\text{freq}} * 10^{-6}}{(a * f + a_0) * S + (b * f + b_0)}, \frac{L * 10^{-6}}{(S+20) * 8}\right), \quad 1 \le f \le abs\left(\frac{n}{x+y}\right). \tag{5}$$

L represents the link speed in bits per second. In order to calculate throughput in gigabits per second (Gbps), the Ethernet preamble (7 B), start of frame delimiter (1 B), and the interframe gap (12 B) must be considered in Equation (5) on top of the frame size, giving the following:

$$Y_{\text{Gbps}} = \min\left(\frac{f^2 \text{CPU}_{\text{freq}} * (S+20) * 8 * 10^{-9}}{(a * f + a_0) * S + (b * f + b_0)}, L * 10^{-9}\right), \quad 1 \le f \le abs\left(\frac{n}{x+y}\right). \tag{6}$$

The min calculus in Equations (5) and (6) assures the estimated throughput not to exceed the physical bandwidth link limits.

3.6. Calibration. Model calibration is the process of setting the well-defined parameters of the model with respect to a specific real system. The determination of the model parameters is based on measurement results of the modeled system. In this study, the modeling parameters are obtained based on cumulative cost (see Chapter 3.5 for cumulative cost definition). Equation (3) is used to obtain the cumulative cost from maximum throughput measurements. Only measurements with <0.5% packet loss are considered. The results comply with perf measurements for NFVI cores.

Figure 6 presents the cumulative cost C_{Cf} for the "1 + 1" + IC off and "1 + 1" + IC 125 modeling system configurations. On 8-core CPU verification can be performed for maximal scale factor $f = 4$. As it can be observed in Figure 6, there is no influence on cost variation with increasing the number of flows to higher number than applied scale factor f, as long as the load is equally distributed. This conclusion stands for all modeling system configurations. Full curve connects the points obtained by measurements. The dotted lines represent the linear regression line for the measuring points. In order to obtain the modeling parameters for the calibration (a, b, a_0, b_0), only 4 calibration points are needed. The 4 points are taken from the regression lines shown in red circles/crosses in Figure 6 for better approximation to all measured values. The modeling parameters are obtained by applying Equation (4).

The same procedure is applied for obtaining modeling parameters in the "2 + 1" + IC off and "2 + 1" + IC 125 modeling system configurations. The maximal scale factor f that can be verified in testbed for these modeling system configurations is $f = 2$. Cumulative cost C_{Cf} for the "3 + 1" + IC off/125 modeling system configuration is given in Figure 7. In this modeling system configuration, the maximal scale factor f that can be verified in testbed is also $f = 2$. For "4 + 1" + IC off/125, "4 + 2" + IC off/125, and "5 + 2" + IC off/125 modeling system configurations, the maximal scale factor f that can be verified in testbed is $f = 1$. This is insufficient to obtain modeling parameters. The model is calibrated for these modeling system configurations following assumption in throughput prediction model definition that throughput scales with f equally to "3 + 1" + IC off/125 modeling system configuration.

The calibration points for all modeling system configurations are given in Table 1. Calibration points for modeling system configurations "1 + 1" + IC off and "1 + 1" + IC 125 are taken arbitrarily for $f = 2$ and $f = 4$, while for other modeling system configurations calibration points are taken for $f = 1$ and $f = 2$. The resulting modeling parameters are given in Table 2.

Applying the throughput prediction model parameters in Table 2 to Equations (5) and (6), the maximum achievable throughput can be predicted for any number of CBR flows and any fixed frame size S for any number of CPU cores n per individual modeling system configuration.

4. Model Validation

4.1. Testbed. Figure 8 shows the testbed setup and Table 3 gives details of the hardware/software components of the testbed and used kernel/driver settings. The measurements were conducted using two physical machines. One of the machines is considered a Device under Test (DuT) and the other machine is used as traffic emulator. Both machines are equal in HW configuration. Traffic is run from the emulator node (MoonGen [30] traffic generator) towards the DuT that returns it back to the emulator. MoonGen is a

Table 1: Calibration points (S, C_{Cf}).

C_{Cf} / S (B)	"1 + 1" + IC off ($f = 2, 4$)	"1 + 1" + IC 125 ($f = 2, 4$)	"2 + 1" + IC off ($f = 1, 2$)	"2 + 1" + IC 125 ($f = 1, 2$)	"3 + 1" + IC off/125 ($f = 1, 2$)	"4 + 1" + IC off/125 ($f = 1, 2$)	"4 + 2" + IC off/125 ($f = 1, 2$)	"5 + 2" + IC off/125 ($f = 1, 2$)
64	37232	21583	5967	3978	4072	3562	3114	2757
1024	40962	24511	8809	5920	5191	3918	3989	4187
64	86898	51473	16262	12130	11044	10466	10018	9661
1024	92456	58336	18974	14800	12630	11023	11094	11292

Table 2: Throughput prediction model parameters.

Parameter	"1 + 1" + IC off	"1 + 1" + IC 125	"2 + 1" + IC off	"2 + 1" + IC 125	"3 + 1" + IC off/125	"4 + 1" + IC off/125	"4 + 2" + IC off/125	"5 + 2" + IC off/125
a	0.9521	2.0494	-0.062	1.1643	0.7817	0.7817	0.7817	0.7816
a_0	1.9813	-1.0486	2.4840	0.3936	0.0575	-0.4293	-0.1462	-0.33672
b	24772.0	14814.0	10738.9	8237.4	6854.9	6854.9	6854.9	6854.9
b_0	-12560.8	-8239.8	-4665.4	-4141.8	-2656.4	-3340.9	-3749.1	-3749.1

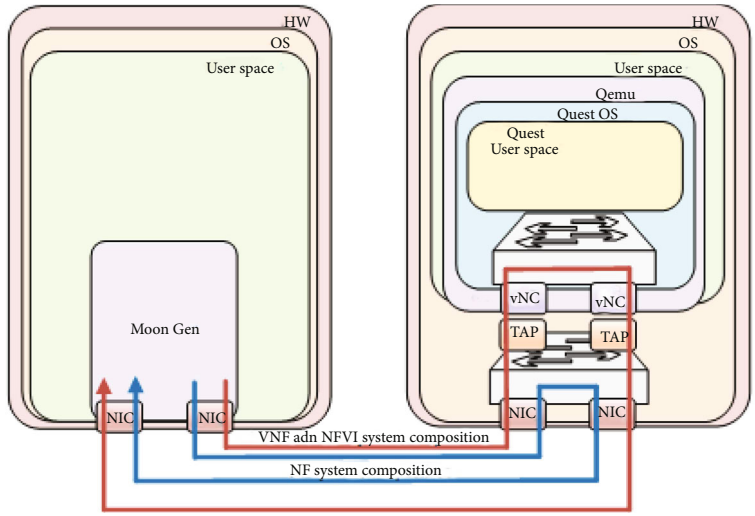


Figure 8: Testbed configuration.

DPDK traffic generator, capable of saturating 10 Gbps link even with the smallest 64 B UDP packets using single flow. The DuT runs the Linux profiling tool perf that reduces the maximum throughput for <1%.

The measurements are performed for the two types of system compositions for the DuT machine. At first, a forwarding engine (OVS) is executed on a physical machine that bridges two physical NICs. This is referred to as network function (NF) composition. At second, a forwarding engine is executed as a VM-based VNF that is connected to the host using a virtio mechanism and that bridges two virtual NICs. This is referred to as VNF and NFVI composition. In both system compositions, NF and VNF execute simple flow entry into OVS that forwards incoming packets from (v)NIC1 to (v)NIC2. OVS is configured to switch the packets from one port to another.

In NFVI, the flows are configured to forward incoming packets from NIC1 to TAP1 and from TAP2 to NIC2.

For each system composition, system configurations are applied based on throughput optimized distribution of EAs per number of available cores for single flow, each in addition considering NIC settings without interrupt coalescing (IC) and IC time arbitrary set to 125 μs. The system configuration is defined by "$x + y$" + IC settings, where "x" is the number of cores reserved for NFVI, "y" is the number of cores reserved for VNF for each flow, and IC represents one or both setting of IC time. The system configuration for single flow is further scaled linearly up to the number that is allowed by the total number of CPU cores. The allowed scale factor f is determined by $f * (x + y) <= 8$, where 8 is the total number of cores in the DuT. The empirical methodology used for building the system configuration is described in Chapter 4.2. The throughput measurements for the NF configurations published in [2] are used in this study for comparison to VNF system configurations' measurements.

For each "$x + y$" + IC system configuration in VNF and NFVI system, composition measurement scenarios are defined in terms of UDP CBR traffic of $1 - f$ and 100 flows with fixed frame sizes of 64, 128, 256, 512, 1024, and 1500 bytes and packets per second (pps) rates of 50, 200, 400, 600,…, max kpps (with packet loss < 0.5%),…, max kpps. The max kpps is derived from the frame size and the limit of physical link (10 Gbps). The same measurement scenarios, limited to single UDP CBR traffic, are executed and published in [2] for the NF system composition. The frame size is denoted as the Ethernet frame size without preamble (7 B), start of frame delimiter (1 B), and interframe gap (12 B). The frame structure is depicted in Figure 9.

Each measurement scenario was conducted for 12 minutes. Each case is repeated 3 times to conduct statistical analysis. Packet counters, used for throughput measurements and packet loss analysis per each defined EA, are extracted on the DuT before and after the test to obtain counter values relevant for the measurement scenario. In order to discover and confirm the bottleneck EA for of the particular system configuration, NAPI processing key performance indicator (KPI) (soft IRQs, IPIs, and HW IRQs) counters are extracted for each measurement scenario, also before and after measurement in order to obtain counter values relevant for the measurement scenario. Perf is applied on each core to measure the number of cycles per each CPU core relevant for the test. These measurements are required to obtain average CPU utilization and cost per packet in terms of cycles per packet for CPU core. Perf is limited for usage only in NFVI due to limited possibilities in VNF environment. For the VNF in context of this study, it is enough to understand if the CPU is fully utilized or not. The full CPU utilization for target VNF cores can be understood from NAPI KPIs.

MoonGen load generator is used to produce artificial CBR traffic at packet rates that scale up to the link speed of 10 Gbps. As explained above, the produced traffic consists of $1 - f$ and 100 flows with evenly distributed packet rates. These flows are distributed to $1 - f$ sets of "$x + y$" cores. In case of $1 - f$ flows, this effectively means that each flow is processed by a dedicated set of "$x + y$" cores. For 100 flows, Receive Side Scaling (RSS) assures fair share of the flows amongst the sets of "$x + y$" cores.

The different "$x + y$" + IC system configurations and packet flow distribution to different sets of "$x + y$" cores are

Table 3: DuT, VNF, and emulator setup.

Host (DuT)	
CPU	Intel(R) Xeon(R) W-2145 CPU @ 3.70 GHz, 8 cores, HT off, Turbo boost off, C-state disabled
RAM	Dual channel 4 × 32 GB, 2666 MHz
OS	Ubuntu 16.04.5 LTS
$NIC_{1/2}$	82599ES 10-Gigabit SFI/SFP+
Network driver	ixgbe 4.2.1
Forwarding engine	OVS 2.5.5
NF	OVS 2.5.5
I/O framework	NAPI
Hypervisor	QEMU 2.5
Kernel settings	net.core.dev_weight = 128 txqueuelen = 1000 net.core.netdev_budget = 300 netdev_max_backlog = 3000
Driver/NIC settings	NIC offload functions = off Rx/Tx-usecs = 0/125 Rx ring buffer = 256 Tx ring buffer = 256
VNF	
vCPU	2 cores, HT off
RAM	4 GB
OS	Ubuntu 16.04.5 LTS
$vNIC_{1/2}$	vhost-net
Network driver	virtio-net 1.0
VNF	OVS 2.5
I/O framework	NAPI
Kernel settings	net.core.dev_weight = 128 txqueuelen = 1000 net.core.netdev_budget = 300 netdev_max_backlog = 3000
Driver/NIC settings	NIC offload functions = off Rx/Tx-usecs = - Rx ring buffer = 256 Tx ring buffer = 256
Emulator	
CPU	Intel(R) Xeon(R) W-2145 CPU @ 3.70 GHz
RAM	Dual channel 4 × 32 GB, 2666 MHz
OS	Ubuntu 16.04.5 LTS
Traffic generator	MoonGen
$NIC_{1/2}$	82599ES 10-Gigabit SFI/SFP+
I/O framework	DPDK

achieved by configuring the features in HW NIC, Linux kernel, and virtio mechanism. The HW NIC feature used is RSS and already mentioned IC. Each queue can be attached to different cores by configuration. A flow entering the kernel for processing through physical or virtual (TAP) interface can further be pushed for kernel processing to a different core than it was processed for reception from (v)NIC. The Linux kernel feature that is used to achieve this configuration is called Receive Packet Steering (RPS). A thread isolation is another kernel feature that is used to pin kernel device emulation threads to specific cores. Finally, virtio multiqueue feature is used in the kernel device emulation and guest driver that enables parallel packet processing in VM. In order to enable a parallel processing and scale the performance, each queue requires dedicated cores for kernel device emulation and VNF (VM) processing.

4.2. Building the "x + y" + IC System Configurations. "x + y" + IC system configurations are built based on empirical bottleneck EA determination, starting from the minimal granularity ("1 + 1" + IC off/125) up to the bottleneck EA being fully isolated on the CPU core. The gradual construction of system configurations is depicted in Figure 10 from top to bottom. Bottleneck EA per "x + y" + IC system configuration are marked with a plus sign. The bottleneck EA remains the same irrespective of the number of flows. The following criteria are considered for the "x + y" + IC system configuration definition:

(i) Receiving EA and Fwd & Snd VM EA are dislocated as the last resort due to additional processing overhead introduced in dislocated system configuration

(ii) Sending VM EA and Fwd & Snd NIC EA are dislocated as the last resort due to additional processing overhead introduced in dislocated system configuration

(iii) Clearing EA is collocated either with Fwd & Snd NIC EA or Receiving EA based on the higher total throughput measurements

The maximum granularity "x + y" + IC system configuration based on this methodology is shown to be "5 + 2" + IC. Further separation provides no additional throughput increase as the bottleneck remains in highlighted EAs.

4.3. System Configuration: "1 + 1" + IC Off. Figure 11 shows the predicted and measured maximum throughput for "1 + 1" + IC off system configuration in 8-core CPU with 3.7 GHz clock. The respective system configuration scales by the factor 4 in the 8-core CPU. The *x*-axis shows the frame size in bytes. The *y*-axis represents the measured and predicted maximum throughput of the VNF in (a) Mpps and in (b) Gbps. The results reveal poor throughput performance that can be achieved by this setup in comparison to theoretical values. The reason is that the collocated EAs competing for CPU resources on NFVI core cause the full CPU utilization already for small packet rates, leading the bottleneck EA to cause buffer overflow. The throughput increases with the number of flows until the number of flows reaches scale factor number. This observation reveals the strong dependency of the maximum throughput to the scale factor and,

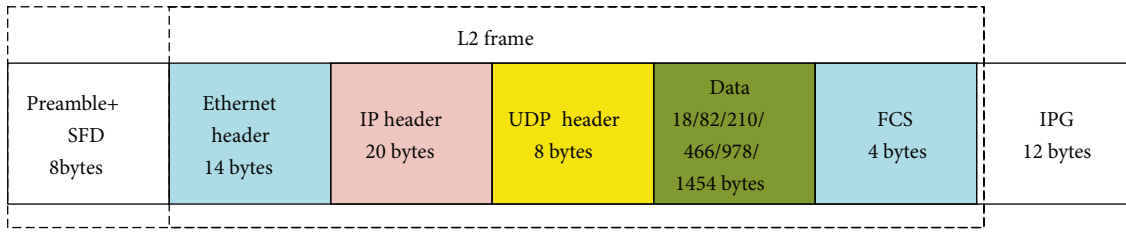


Figure 9: Ethernet frame structure.

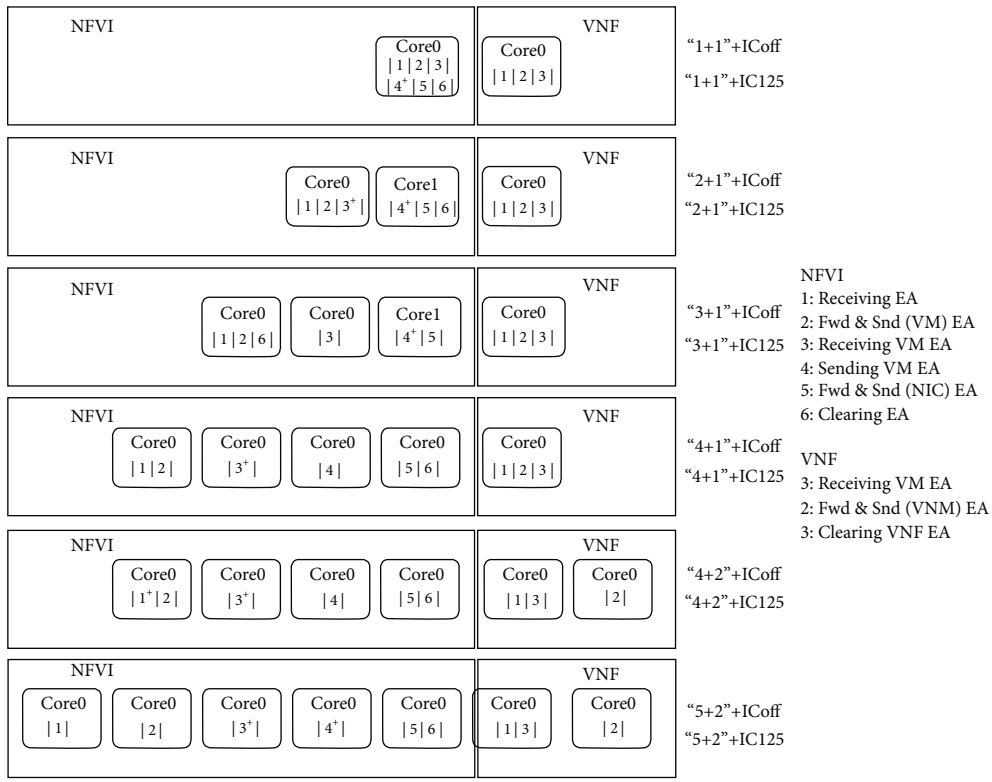


Figure 10: "$x + y$" + IC system configuration decision flow.

respectively, to the number of used cores due to parallel processing. The predicted throughput results for 100 flows are the same as for 4 flows, since the number of flows exceeding the scale factor number has no effect on the throughput prediction model. The measured results for 100 flows coincide with the measurements for the 4 flows. The maximum throughput also depends on the frame size, but to a lower magnitude than for the number of used cores. As it can be observed, predicted results highly match the measured values.

Relative error ERR in percentage as $\mathrm{ERR} = (Y_{\mathrm{sim}} - Y_{\mathrm{meas}})/Y_{\mathrm{meas}}$ reveals the worst-case prediction is for single flow where the relative error of the prediction is up to ~15%. The mean deviation is 3.55%. This result reveals a decent precision of throughput prediction model for this modeling system configuration. The confidence bounds are omitted in this study as the predictions are based on CBR flows which do not show large variance. The latter is valid for the validation process within this chapter in general.

4.4. System Configuration: "1 + 1" + IC 125. The subject system configuration differs from the previous in the value of IC time only. Figure 12 shows the predicted and measured maximum throughput for this system configuration in 8-core CPU with 3.7 GHz clock. As it can be observed, the conclusion from previous system configuration analysis related to maximum throughput dependency to the scale factor and, respectively, to the number of used cores is equally valid for this system configuration. The maximum throughput also slightly depends on the frame size, as in the previous system configuration. Although the general conclusions for the two system configurations are equal, there is a 2-fold increase in maximum throughput for the "1 + 1" + IC 125 system configuration. It is still valid for this system configuration that collocated EAs compete for CPU resources on NFVI core that causes eventually the full CPU utilization and buffer overflow at the bottleneck EA. However, less CPU cycles are consumed per packet on this core since the HW IRQs are being suppressed by IC time. This leads to lower CPU consumption

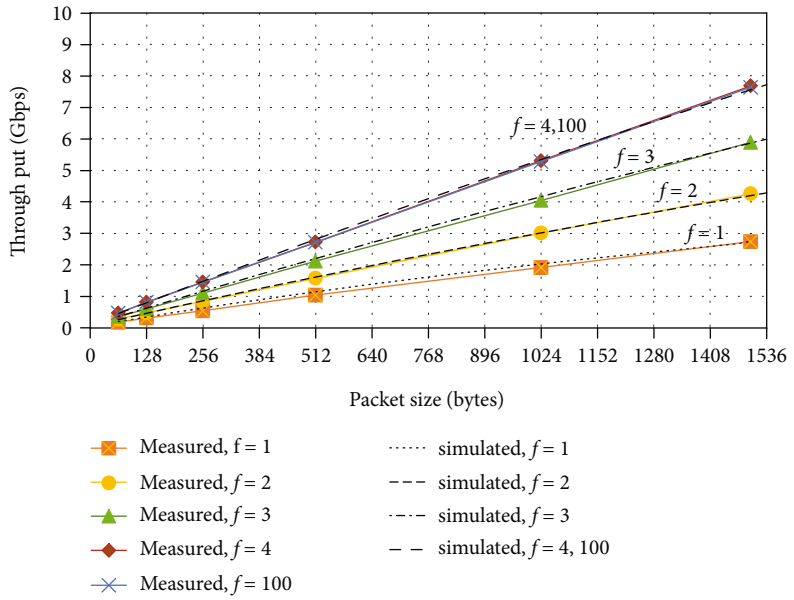


Figure 11: Predicted (simulated) and measured maximum throughput of the VNF in "1 + 1" + IC off modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

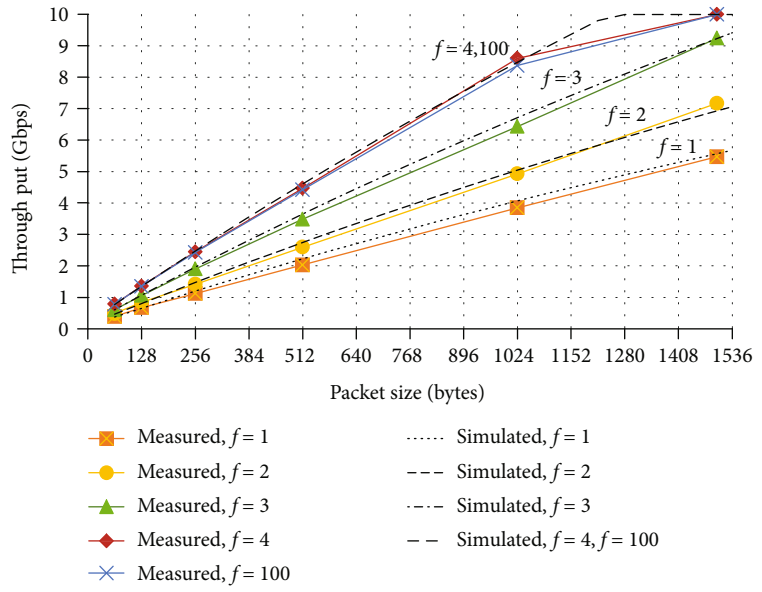


Figure 12: Predicted (simulated) and measured maximum throughput of the VNF in "1 + 1" + IC 125 modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

for the same rates, comparing to previous system configuration, leading to higher maximum throughput.

The subject system configuration enables the 10 Gbps link saturation for the <1200 B frames with minimum 4 flows according to predictions. The measurements confirmed the link to be the system bottleneck for 1500 B frames and 4 flows. As it can be observed, the predicted results highly match the measured. It can be concurred though, based on observation, that the measured results are not entirely following linear behavior. The high slope observed for the measurement curves for 4 and 100 flows between 1024 B and 1500 B is caused by not having measurement for the frame size between the two. Performing the measurements for 1152 B or 1280 B frames would probably lead to lower slopes of the measurement curves, in line with the predictions.

Relative error ERR reveals that the worst-case prediction is for single flow where the relative error of the prediction reaches 9.4%. The mean deviation is 2.88%, which points to even slightly better prediction results of the throughput prediction model for this system configuration.

4.5. System Configuration: "2 + 1" + IC Off. Figure 13 shows the predicted and measured maximum throughput for "1 + 1" + IC off system configuration in 8-core CPU with

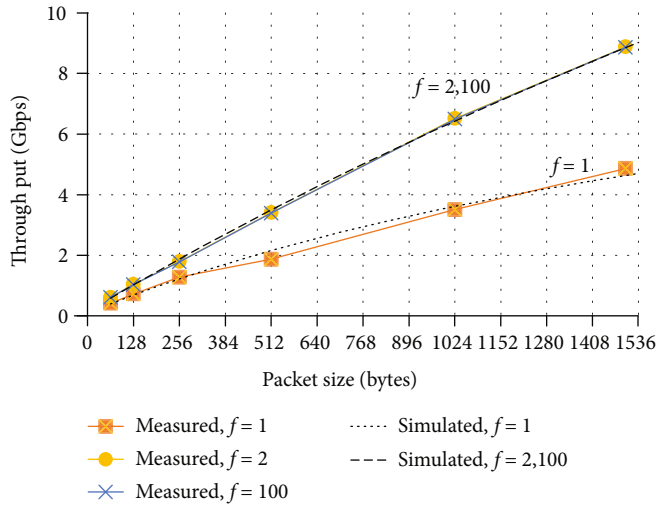


Figure 13: Predicted (simulated) and measured maximum throughput of the VNF in "2 + 1" + IC off modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

3.7 GHz clock. The respective system configuration scales by the factor 2 in the 8-core CPU. The x-axis shows the frame size in bytes. The y-axis represents the measured and predicted maximum throughput of the VNF in (a) Mpps and in (b) Gbps. The general conclusions from the previous system configuration with regard to dependency to number of used cores (i.e., scale factor) and frame size are applicable for this system configuration as well. This system configuration shows significantly higher maximum throughput for the same number of used cores in comparison to "1 + 1" + IC off system configuration (see $f = 3$ in Figure 11 and $f = 2$ in Figure 13). The reason is that the obstacle of generality of NAPI design in kernel stack that is preventing Sending VM EA to run when CPU is fully utilized is avoided by dislocating Sending VM EA and Receiving EA to different cores. Comparing the results for the same number of used cores with "1 + 1" + IC 125, it can be observed that the results match to high extent (see $f = 3$ in Figure 12 and $f = 2, 100$ in Figure 13). This shows that the interrupt suppression also mitigates the mentioned obstacle. However, this mitigation comes with the penalties of latency and jitter. Although the throughput prediction model considers maximum throughput, respectively, packet loss, the influence of system configurations on latency and jitter must be understood to properly dimension and configure the system.

Observing the predicted and measured maximum throughput in the subject system configuration, it can be concluded they coincide to high extent. However, the measurements for single flow and 512 B frames deviate from the prediction significantly. The reason for this is unknown.

Relative error ERR reveals that the worst-case prediction is for single flow where the relative error of the prediction is up to ~15%. The worst case is an extreme for the already mentioned 512 B single flow measurements. The mean deviation is 3.64%, which points to slightly worse prediction results than in formerly analyzed system configurations.

4.6. System Configuration: "2 + 1" + IC 125. The subject system configuration differs from the previous in the value of IC time only. Figure 14 shows the predicted and measured maximum throughput for this system configuration in 8-core CPU with 3.7 GHz clock. As for previously analyzed system configuration, the general dependencies of maximum throughput to a number of used cores and frame size are valid for this system configuration. As it can be observed both from predicted and measured maximum throughput results, this system configuration outperforms previous system configurations for the same number of used cores (see $f = 3$ in Figures 11 and 12 and see $f = 2, 100$ in Figures 13 and 14). The reason is that this system configuration benefits from both Receiving EA dislocation from Sending VM EA, mitigating the obstacle of generality of NAPI design in kernel stack, and from IC time, suppressing the interrupts. IC time interrupt suppression again comes with latency and jitter penalties of course.

Predicted maximum throughput points that the subject system configuration hits the physical link limit of 10 Gbps for the frames > 1280 B and minimum 2 flows. The measurements confirmed the prediction for 1500 B frame size. Observing the predicted and measured results, we concur on the slight deviation of the measured results from linear predictions. This guides on the influence of the parameters not considered by the throughput prediction model (such as cache or memory). Yet, the linear approximation used in the throughput prediction model provides satisfactory predictions.

Relative error ERR reveals the worst-case prediction is for single flow where the relative error of the prediction is up to ~6.5%. This is the lowest extreme compared to relative errors from previously analyzed system configurations. The mean deviation is 3.4%. The mean deviation and extreme deviation lead to the conclusion that most of the measurements are similarly deviating from the predicted values. This can also be concluded from observing Figure 14.

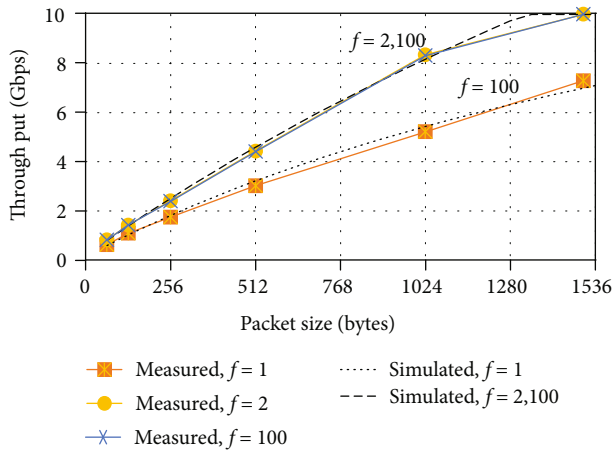


Figure 14: Predicted (simulated) and measured maximum throughput of the VNF in "2 + 1" + IC 125 modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

4.7. System Configuration: "3 + 1" + IC Off/125. Figure 15 shows the predicted and measured maximum throughput for "3 + 1" + IC off/125 system configuration in 8-core CPU with 3.7 GHz clock. The respective system configuration scales by the factor 2 in the 8-core CPU. The x-axis shows the frame size in bytes. The y-axis represents the measured and predicted maximum throughput of the VNF in (a) Mpps and in (b) Gbps. As for previously analyzed system configuration, the general dependencies of maximum throughput to a number of used cores and frame size are valid for this system configuration. It can also be observed that the measured results slightly deviate from linear predictions, but still to a level of satisfactory results of the predictions. The throughput prediction model considers no result deviation for IC off and IC 125 μs. This is supported by the measurements that show almost no deviation as well. This system configuration achieves the highest maximal throughput for the CPU under test in comparison to previously analyzed system configurations. The reason is that it completely separates the vNIC emulation (Receiving VM EA and Sending VM EA) from the EAs handling HW IRQs (Receiving EA and Clearing EA) in NFVI, completely mitigating the influence of HW IRQs to vNIC emulation. The predicted maximal throughput points that this system configuration hits the physical link limit of 10 Gbps for the frames > 1024 B and minimum 2 flows. The measurements confirmed the prediction for 1500 B frame size.

Relative error ERR reveals that the worst-case prediction is for IC 125 μs and single flow where the relative error of the prediction is up to ~5.6%. This extreme does not deviate a lot from the mean deviation that is 2.02%.

4.8. System Configuration: "4 + 1" + IC Off/125. Figure 16 shows the predicted and measured maximum throughput for "4 + 1" + IC off/125 system configuration in 8-core CPU with 3.7 GHz clock. The respective system configuration scales by the factor 1 in the 8-core CPU. The x-axis shows the frame size in bytes. The y-axis represents the measured and predicted maximum throughput of the VNF in (a) Mpps

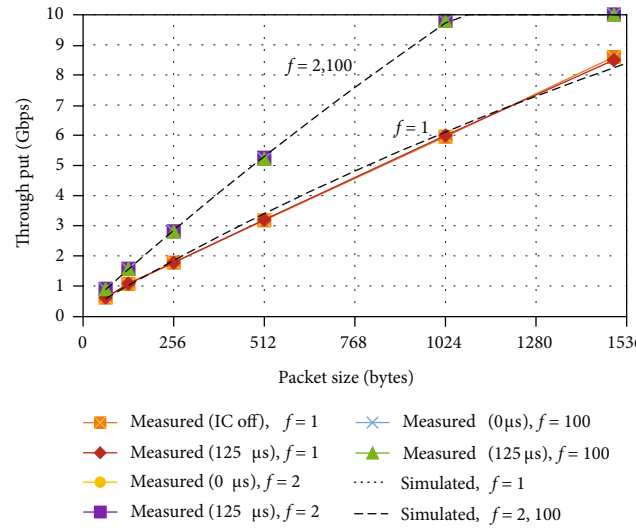


Figure 15: Predicted (simulated) and measured maximum throughput of the VNF in "3 + 1" + IC off/125 modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

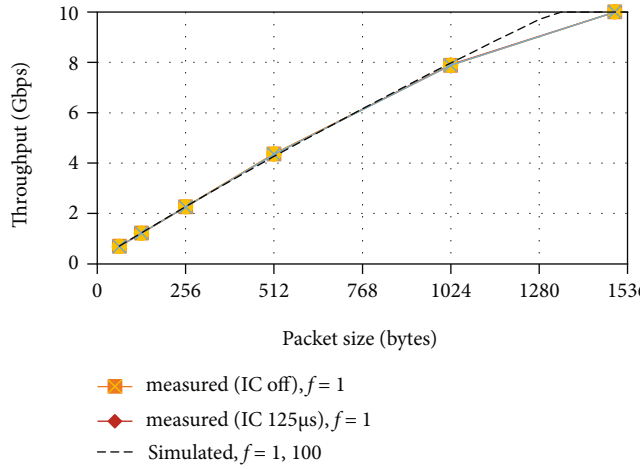


Figure 16: Predicted (simulated) and measured maximum throughput of the VNF in "4 + 1" + IC off/125 modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

and in (b) Gbps. This system configuration can saturate the 10 Gbps link already for single flow. Predictions point this can be achieved for the frame sizes > 1280 B. The measurements confirmed this is possible for 1500 B frames. In this system configuration, both Receiving VM EA and Sending VM EA are completely isolated on dedicated CPU cores. Since Receiving VM EA is the bottleneck EA for 512 B-1024 B frame size measurements (see Chapter 4.2), it can be concurred that this is the optimal configuration to achieve the highest throughput for these frame size for single flow, when applied in testbed CPU. Based on this conclusion, it can be claimed the target VNF will never be saturated for these frame size with ingress traffic since the bottleneck will always hit the Receiving VM EA (vNIC). Although this system configuration is optimal for some frame sizes applied for single flow in testbed CPU, this is not the optimal system

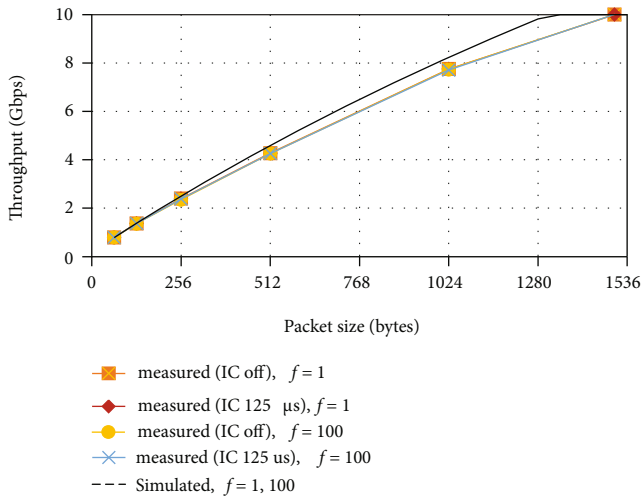


Figure 17: Predicted (simulated) and measured maximum throughput of the VNF in "4 + 2" + IC off/125 modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

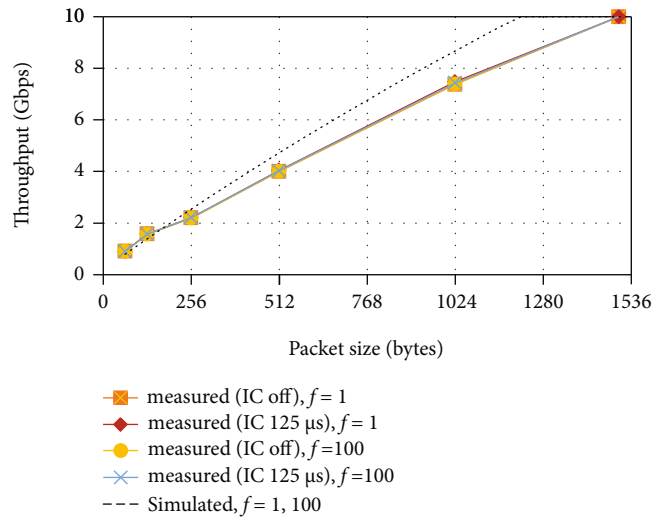


Figure 18: Predicted (simulated) and measured maximum throughput of the VNF in "5 + 2" + IC off/125 modeling system configuration on 8-core CPU with 3.7 GHz and 10 Gbps links.

configuration to achieve the highest throughput in general for the respective CPU. Higher maximal throughput can be achieved for "3 + 1" + IC off/125 (see Figure 15), applying minimum 2 flows or even "2 + 1" + IC 125, also applying 2 flows as a minimum (see Figure 14, $f = 2$).

The subject system configuration is the only system configuration where the VNF can be the system bottleneck (see Chapter 4.2), which is the case for 64 B-256 B frame size measurements. This leads to the fact that this system configuration is the only system configuration that can be used to directly compare target NF with virtual representation (VNF). Based on observation of Figure 16, it can be concluded that the predicted results highly match the measured values.

Relative error ERR reveals that the worst-case prediction is not exceeding 2.4%. The mean deviation is 0.97%. 2.4% is the lowest extreme and 0.97% is the lowest mean deviation amongst all analyzed system configurations.

4.9. System Configuration: "4 + 2" + IC Off/125. Figure 17 shows the predicted and measured maximum throughput for "4 + 2" + IC off/125 system configuration in 8-core CPU with 3.7 GHz clock. The respective system configuration scales by the factor 1 in the 8-core CPU. The x-axis shows the frame size in bytes. The y-axis represents the measured and predicted maximum throughput of the VNF in (a) Mpps and in (b) Gbps. As it can be observed, both predicted and measured results point to slight increase in maximal throughput for 64 B-256 B frames. The reason is that by providing additional core for VNF processing, the bottleneck is pushed towards Receiving EA. Although predictions show slight maximal throughput increase for the larger frames as well, the measured results for 512 B-1024 B frame sizes show actually slight degradation. It can be assumed that the slight decrease is due to cache memory handling between CPU cores that needs more frequent flushing and loading than for previous system configuration. The bottleneck EA for measurement scenarios with larger frame sizes continues to be Receiving VM EA. The measured results for small frame

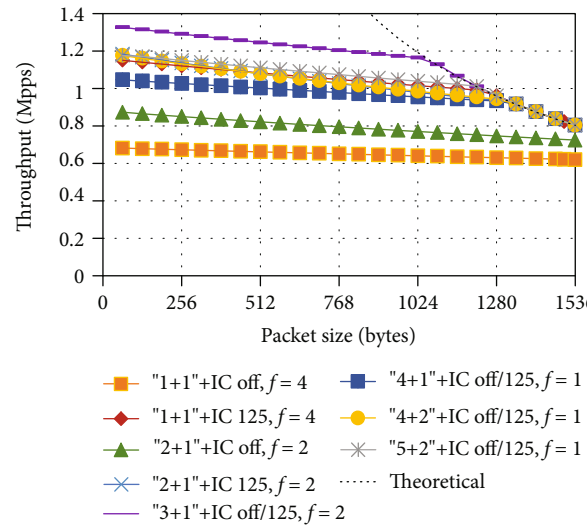
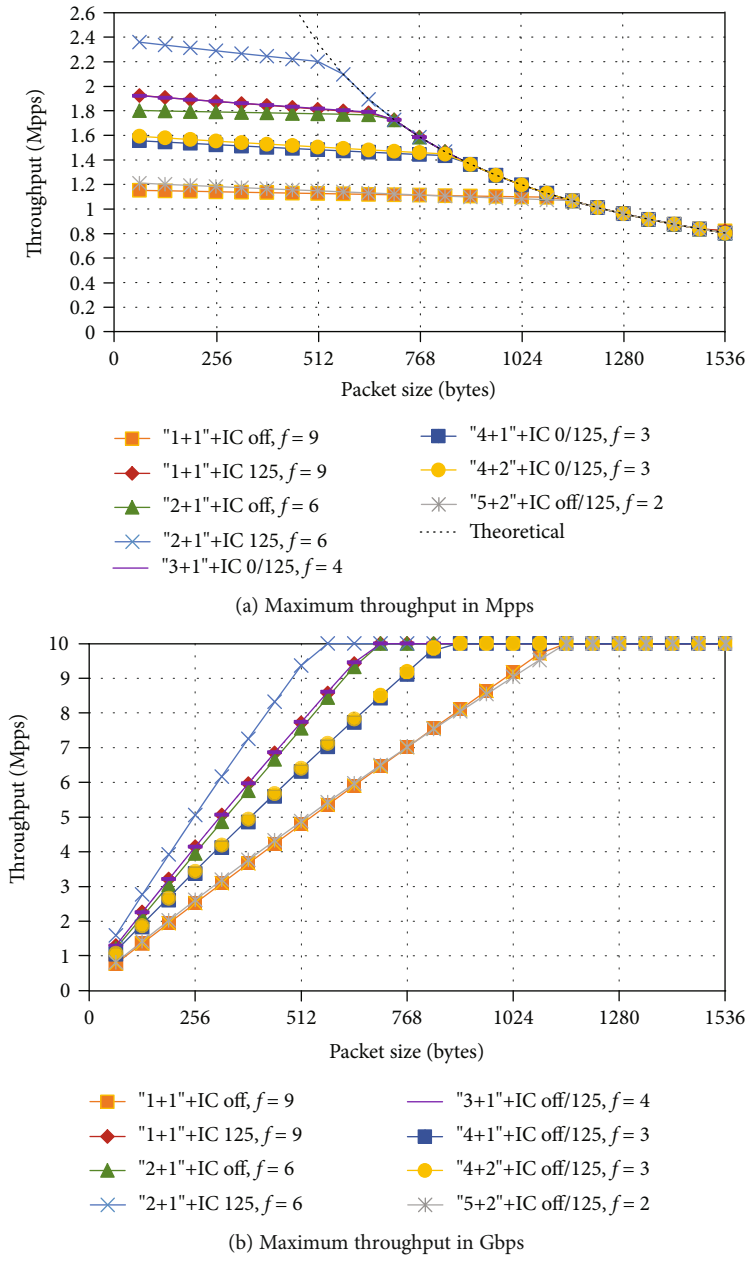


Figure 19: Maximal throughput model predictions for the VNF per system configuration in Mpps, on 8-core CPU with 3.7 GHz and 10 Gbps links.

show significantly higher maximal throughput than larger frames, causing the measurement curve to deviate from linear approximation. This leads to slightly increased deviation between predicted and measured results for large frames. The predicted results point that the single flow can saturate the 10 Gbps physical link for measurement scenarios including frames larger than 1280 B. This is confirmed for 1500 B frames by measurements.

Relative error ERR reveals that the worst-case prediction is for 512 B and 1024 B frame size measurements where the relative error of the prediction is up to ~7.8%. This extreme does not deviate a lot from the mean deviation that is 3.35%. This result reveals a decent precision of throughput prediction model for this modeling system configuration comparable to previous system configurations.



(a) Maximum throughput in Mpps

(b) Maximum throughput in Gbps

Figure 20: Maximal throughput model predictions for the VNF per system configuration, on 16-core CPU with 3.0 GHz and 10 Gbps links.

4.10. System Configuration: "5 + 2" + IC Off/125. Figure 18 shows the predicted and measured maximum throughput for "5 + 2" + IC off/125 system configuration in 8-core CPU with 3.7 GHz clock. The respective system configuration scales by the factor 1 in the 8-core CPU. The x-axis shows the frame size in bytes. The y-axis represents the measured and predicted maximum throughput of the VNF in (a) Mpps and in (b) Gbps. As it can be concluded from the measurement curve, the values for small frames (64 B-128 B) highly outperform the measured maximal throughput values for larger frames. The only interpretation of such results can be in memory handling and less costly packet relay from kernel space to user space and backwards. This relatively high deviation of the measured results from linear approximation is causing slightly underestimated predicted values for small packets, while on the other hand slightly overestimated results for larger frames in comparison to measured values. Although predicted results point to almost matching results for this system configuration in comparison to previous "4 + 2" + IC off/125, the measured results show better results in this system configuration for small frames and slightly worse measured results for larger frames. This leads to conclusion that this system configuration may be optimal only for extremely low frame traffic. The predicted results point that the single flow can saturate the 10 Gbps physical link for measurement scenarios including frames larger than 1280 B. This is confirmed for 1500 B frames by measurements.

Relative error ERR reveals that the worst-case prediction is for 512 B frame where the relative error of the prediction is up to ~14.7%. The mean deviation is 10.17%, which is the highest mean deviation compared to all other system configurations. This result reveals nonsatisfactory precision of the throughput prediction model for this modeling system configuration.

4.11. Validation Summary. The validation of the defined and calibrated throughput prediction model has proved the model applicability to the referent VNF and NFVI architecture. The mean deviation out of all measurements is 3.74%, for which it can be concurred to be a satisfactory result on a general level. The highest mean deviation is for "5 + 2" + IC off/125 system configuration as these measurements appeared to deviate from linear approximation significantly. It can be concurred that for this system configuration only, the throughput prediction model provides nonsatisfactory preciseness of maximal throughput results. Figure 19 presents maximal throughput that can be achieved on 8-core CPU with 3.7 GHz. The *y*-axis represents predicted maximum throughput of the VNF in Mpps. Prediction reveals "3 + 1" + IC off/125 to be throughput optimal system configuration for this CPU when the traffic involves 2 flows minimum. This system configuration requires all 8 CPU cores to achieve this.

By observing the predicted and measured throughput results, some general concussions are listed below:

(i) Maximum throughput highly depends on the scale factor related to system configuration on particular CPU (maximal throughput is increased until the number of flows reaches scale factor number)

(ii) Maximum throughput depends on the frame size

4.12. Prediction Results for CPU: Intel® Core™ i9-10980XE. By applying calibrated and validated VNF throughput prediction model, it is possible to forecast the maximum throughput performance for any CPU. In this chapter, we predict the maximal throughput and optimal system configuration to achieve the highest throughput for sufficiently high number of flows on the subject CPU [31] dating from q4/2019 and 10 Gbps link. The subject CPU is an 18-core CPU with 3 GHz clock. The simulation scenarios are the same as described in Chapter 4.2. Figure 20 shows the predicted results. The *y*-axis represents the predicted maximum throughput for the OVS VNF in (a) Mpps and in (b) Gbps. The physical link can be saturated with this CPU for traffic scenarios with frame sizes > 512 B according to predictions. This can be achieved for "2 + 1" + IC 125 system configuration and 6 flows minimum. Amongst the two least optimal throughput system configurations is "5 + 2" + IC off/125 according to predicted results. If the criteria would be set to consider only single flow, then this system configuration would come on top. For traffic scenarios with frame sizes < 512 B, the throughput prediction model still imposes the throughput to be strongly limited by CPU processing capacity.

5. Conclusion

This study proposed a VNF and NFVI architecture model with performance prediction and bottleneck determination for optimal virtual network function performance analysis. It is presented in the study that the throughput performance of the VNF can be modeled using software execution units (EAs) that can run isolated or collocated on a compute resource (CPU core). A prerequisite for the VNF to be applied in the model is that the white box view of its software architecture is known and that system bottlenecks and throughput requirements are already identified. The validation results of the model using representative VNF showed the mean deviation out of all measurements from predicted results to be 3.74%. Although the mean deviation points to respectable prediction results of our model, setting loose some of the prerequisites and assumptions of the model (CBR traffic, same frame size, and linearly scalable system configurations with number of flows) could lead to significantly higher deviations. In addition, the defined EA-based throughput prediction model considers only a single static value of IC against the IC off, not considering the latency/jitter dimension of the traffic that is affected by the IC. Besides CPU processing resources and IC, network functions, regardless if deployed as NF or VNF, may be impacted by other parameters, such as CPU cache memory, CPU NUMA topology, L3 cache memory sharing, or the usage of CPU hyperthreading. Therefore, it is important in the future to validate the throughput prediction results for more real-world VNFs. The future work will concentrate to carry out more fine-grained measurements, modeling, and simulation in order to extend the proposed throughput prediction model and to extend its applicability.

Data Availability

Research data is confidential.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] ETSI Industry Specification Group (ISG) NFV, "ETSI GS NFV 002 V1.2.1: network functions virtualization (NFV); architectural framework," 2014.

[2] D. Vladislavic, G. Topic, K. A. Vrgoc, J. Ozegovic, and D. Huljenic, "Throughput evaluation of kernel based packet switching in a multi-core system," in *27th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pp. 1–6, Split, Croatia, 2019.

[3] W. Wu, M. Crawford, and M. Bowden, "The performance analysis of Linux networking - packet receiving," *Computer Communications*, vol. 30, no. 5, pp. 1044–1057, 2007.

[4] A. O. Allen, *Probability, Statistics, and Queueing Theory with Computer Science Applications*, Academic Press, 2nd Edition edition, 1990, ISBN: 0-12-051051-0.

[5] G. A. Gallardo, B. Baynat, and T. Begin, "Performance modeling of virtual switching systems," in *2016 IEEE 24th*

International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 125–134, London, UK, 2016.

[6] Z. Su, B. Baynat, and T. Begin, "A new model for DPDK-based virtual switches," in *2017 IEEE Conference on Network Softwarization (NetSoft)*, pp. 1–5, Bologna, Italy, 2017.

[7] B. Xiong, K. Yang, J. Zhao, W. Li, and K. Li, "Performance evaluation of OpenFlow-based software-defined networks based on queueing model," *Computer Networks*, vol. 102, pp. 172–185, 2016.

[8] S. Azodolmolky, R. Nejabati, M. Pazouki, P. Wieder, R. Yahyapour, and D. Simeonidou, "An analytical model for software defined networking: a network calculus-based approach," in *2013 IEEE Global Communications Conference (GLOBECOM)*, pp. 1397–1402, Atlanta, US, Dec. 2013.

[9] K. Mahmood, A. Chilwan, O. Østerbø, and M. Jarschel, "Modelling of OpenFlow-based software-defined networks: the multiple node case," *IET Networks*, vol. 4, no. 5, pp. 278–284, 2015.

[10] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia, "Modeling and performance evaluation of an OpenFlow architecture," in *2011 23rd International Teletraffic Congress (ITC)*, pp. 1–7, Anaheim, US, Sept 2011.

[11] A. Bianco, R. Birke, L. Giraudo, and M. Palacin, "OpenFlow switching: data plane performance," in *2010 IEEE International Conference on Communications*, pp. 1–5, Cape Town, South Africa, 2010.

[12] D. Sattar and A. Matrawy, "An empirical model of packet processing delay of the Open vSwitch," in *2017 IEEE 25th International Conference on Network Protocols (ICNP)*, pp. 1–6, Toronto, Canada, 2017.

[13] R. Bolla and R. Bruschi, "Linux software router: data plane optimization and performance evaluation," *Journal of Networks*, vol. 2, no. 3, pp. 6–17, 2007.

[14] R. Bolla and R. Bruschi, "PC-based software routers: high performance and application service support," in *Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow*, pp. 27–32, Seattle, US, 2008.

[15] M. Dobrescu, N. Egi, K. Argyraki et al., "RouteBricks: exploiting parallelism to scale software routers," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pp. 15–28, Montana, US, 2009.

[16] P. Emmerich, D. Raumer, F. Wohlfart, and G. Carle, "Performance characteristics of virtual switching," in *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*, pp. 120–125, Luxembourg, 2014.

[17] "Open vSwitch," http://www.openvswitch.org/.

[18] P. Emmerich, D. Raumer, F. Wohlfart, and G. Carle, "Assessing software and hardware bottlenecks in pc-based packet forwarding systems," in *ICN 2015 : The Fourteenth International Conference on Networks*, p. 90, Barcelona, Spain, 2015.

[19] M. Paolino, N. Nikolaev, J. Fanguede, and D. Raho, "SnabbSwitch user space virtual switch benchmark and performance optimization for NFV," in *2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, pp. 86–92, San Francisco, US, 2015.

[20] Y. Zhao, L. Iannone, and M. Riguidel, "Software switch performance factors in network virtualization environment," in *2014 IEEE 22nd International Conference on Network Protocols*, pp. 468–470, Raleigh, US, Oct. 2014.

[21] B. Zhang, X. Wnag, R. Lai et al., "Evaluating and optimizing I/O virtualization in kernel-based virtual machine (KVM)," in *IFIP International Conference on Network and Parallel Computing*, pp. 220–231, Heidelberg, 2010.

[22] S. Lange, A. Nguyen-Ngoc, S. Gebert et al., "Performance benchmarking of a software-based LTE SGW," in *2015 11th International Conference on Network and Service Management (CNSM)*, pp. 378–383, Barcelona, Spain, Nov. 2015.

[23] R. Kawashima, H. Nakayama, T. Hayashi, and H. Matsuo, "Evaluation of forwarding efficiency in NFV-nodes toward predictable service chain performance," *Transactions of Network Services and Management*, vol. 14, no. 4, pp. 920–933, 2017.

[24] R. Kawashima, S. Muramatsu, H. Nakayama, T. Hayashi, and H. Matsuo, "A host-based performance comparison of 40G NFV environments focusing on packet processing architectures and virtual switches," in *2016 Fifth European Workshop on Software-Defined Networks (EWSDN)*, pp. 19–24, The Hague, The Netherlands, Oct. 2016.

[25] L. Rizzo and G. Lettieri, "VALE, a switched Ethernet for virtual machines," in *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, pp. 61–72, Nice, France, 2012.

[26] M. Honda, F. Huici, G. Lettieri, and L. Rizzo, "mSwitch: a highly-scalable, modular software switch," in *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, pp. 1–13, Santa Clara, US, June 2015.

[27] T. Meyer, F. Wohlfart, D. Raumer, B. Wolfinger, and G. Carle, "Validated model-based prediction of multi-core software router performance," *Praxis der Informationsverarbeitung und Kommunikation (PIK)*, vol. 37, no. 2, pp. 93–107, 2014.

[28] D. Raumer, F. Wohlfart, D. Scholz, P. Emmerich, and G. Carle, "Performance exploration of software-based packet processing systems," *Leistungs-, Zuverlässigkeits-und Verlässlichkeitsbewertung von Kommunikationsnetzen und verteilten Systemen*, vol. 8, 2015.

[29] A. Beifuß, D. Raumer, P. Emmerich et al., "A study of networking software induced latency," in *2015 International Conference and Workshops on Networked Systems (NetSys)*, pp. 1–8, Cottbus, Germany, 2015.

[30] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle, "MoonGen: a scriptable high-speed packet generator," in *Proceedings of the 2015 Internet Measurement Conference*, pp. 275–287, Nice, France, 2015.

[31] "Intel® Core™ i9-10980XE extreme edition processor," https://www.intel.com/content/www/us/en/homepage.html?ref=https://www.intel.com/content/www/us/en/products/processors/core/x-series/i910980xe.html/.