# The Strategic Role of Inter-Container Communications in RAN Deployment Scenarios

Carlo Vitucci
Ericsson AB
Stockholm, Sweden
Email: carlo.vitucci@ericsson.com

Luca Abeni, Tommaso Cucinotta and Mauro Marinoni
Scuola Superiore Sant'Anna
Pisa, Italy
Email: {name.surname}@santannapisa.com

***Abstract*—This paper elaborates on the importance of having efficient inter-container communications at the edge of the network in Software Defined Network – Network Function Virtualization (SDN-NFV) architectures, when deploying services close to the end-user, due to the broad range of bandwidth and latency requirements as coming from novel scenarios in the 5th telecommunication generation (5G). This results in a proposed service deployment framework that is exploited within the Virtualized Radio Access Network (V-RAN) split architecture, a scenario where the crucial role of efficient communications becomes evident. After a short comparison among common technologies available today for efficient inter-containers communications, this paper identifies primary areas of improvement for future research.**

***Keywords–5G; V-RAN; Edge computing; Server at the edge; Service deployment; Inter-containers communication channel.***

## I. INTRODUCTION

Nowadays, the 5th telecommunication generation (5G) is at its first deliveries, with trials running everywhere in the world. Telecommunication operators and manufacturers have a clear understanding of the 5G architecture and its benefits [1]. Indeed, 5G brings significant innovations, with a new architecture based on a mindset shift from connection-centric to service-centric [2]. There are different 5G solutions offered by telecommunication operators or infrastructure providers, and all of them agree on the vision of 5G enabling a new era where vertical services can be deployed end-to-end in the network. In such a view, the role of the edge of the network, and especially the Radio Access Network (RAN), is strategic. In the 5G architecture, there is a strong need to deploy a wide range of vertical services with low-latency requirements, in order to increase the Quality of Experience (QoE) for the end-user. This means flexible deployment across the physical infrastructure, as achievable by embracing the Software Defined Network – Network Function Virtualization (SDN-NFV) paradigm, and optimized communications at all levels, where traditional Virtual Machines (VMs) have decidedly been superseded by containers thanks to their reduced overheads.

Optimizing end-to-end vertical services in 5G can take advantage of SDN-NFV, so that having a multitude of nodes, e.g., data-center, core network or RAN nodes, managed by a (logically) centralized controller, allows for achieving high flexibility in the network architecture, enabling an optimum deployment of services. However, this also introduces a number of challenges, because the characteristics, requirements and acceptable latencies can be quite heterogeneous for the various nodes, where distance from the RAN plays a crucial role [3] [4] [5]. Novel Virtualized RAN (V-RAN) solutions may also tackle the important foreseen explosion of power consumption, especially in urban environments with high population density [6]. The new architecture for 5G in the RAN domain could be not mature enough to allow seamless deployments. For this reason, the most important operators worldwide established the O-RAN alliance [7] to drive the technology evolution, attracting all the technology providers too.

The rest of this paper is organized as follows: Section II presents the background about service deployment, while Section III proposes an overview on the related 5G issues and Section IV focuses on PoP transparency. In Section V key inter-container communication solutions are compared. The paper is concluded with some final remarks and an overview of future works in Sections VI and VII, respectively.

## II. BACKGROUND ON SERVICE DEPLOYMENT

It the SDN-NFV architecture, as described by European Telecommunications Standards Institute (ETSI) [8], the two concepts work together to provide, manage and control the underlying common end to end infrastructure (topology view) where to deploy, secure and supervise any type of vertical service (service view), as shown in Figure 1.

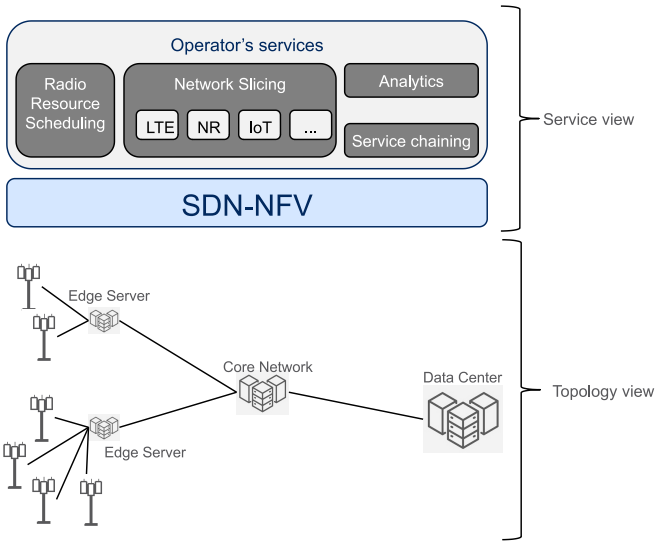


Figure 1. Example of a SDN-NFV architecture views.

The service view shows how the SDN-NFV Management and Orchestration (MANO) framework deploys different ver-

 ISBN: 978-1-61208-695-8 

tical services. These are composed into service-chains, "common bricks" that let new services be deployed quicker. Moreover, establishing and supervising communication channels between two or more services is needed to provide security and protect sensitive data from malicious access. A Virtualized Network Function (VNF) deployment is the responsibility of the MANO framework, where any infrastructure resource is assigned to a service through network slice assignments. Indeed, network slicing is an End-to-End network characterization of a service deployment and is based on resource allocation and control. Any available resource in any node shall be manageable via resource slices. Then, slice definition is actually the assignment of a different network, computing, storage, and radio resources to a vertical service, from access to data center node (see Figure 2).

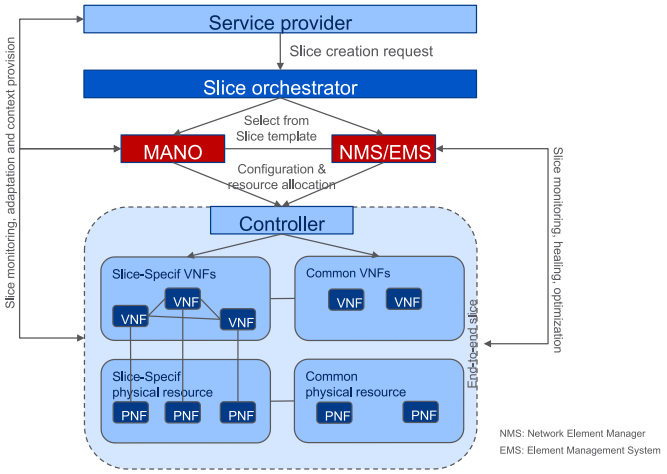


Figure 2. End-to-End slice allocation and control, from [9].

Radio Resources are critical for the RAN, so while bringing the SDN-NFV paradigm into the RAN, constraining placement locations through the Point of Presence (PoP) [10] assignment is mandatory for latency control. This paper is not concerned about how slices are assigned, but it focuses on the need for a *slice definition that includes the function-to-function communication requirements* as a crucial resource to manage. It is critical to consider appropriately the various options that are available to accelerate said communications both in hardware and in software, among services deployed throughout the various servers within the edge infrastructure. For example, the computing allocation in edge servers, and so the definition itself of computing slicing, may be based on radio access service characteristics like bandwidth, latency, and deadline [11]. Network slicing in multiple domains is a foundational building block in the SDN-NFV architecture [12].

Therefore, an effective service framework needs to handle [13] service deployment along with resource allocation and control, coupled with proper network slice definition and management (see Figure 3). To this purpose, it is essential to frame the slice definition in the context of the VNF file descriptors [10]. This makes a service framework usable in the context of service deployment and for the RAN.

## III. 5G, SDN-NFV AND V-RAN

The RAN evolution needed to meet the 5G expectations is entirely driven by the new challenging requirements [5]:

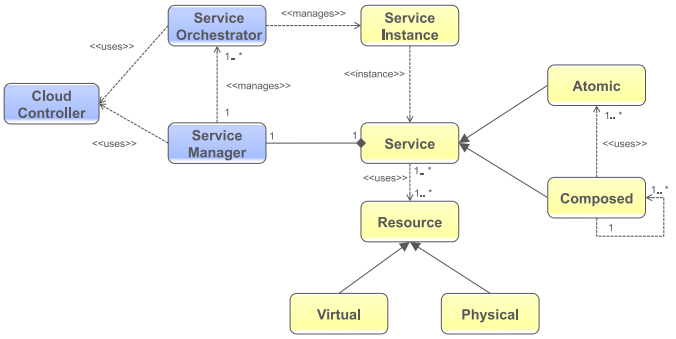


Figure 3. The service deployment framework, from [13].

latency constraints, radio bandwidth and available resources (computing, storage, and connectivity). This is a bare consequence of the primary 5G goal: support a wide range of services, from Internet of Things (IoT) to Machine-Type Communication (MTC) or Machine-to-Machine (M2M), that look promising also in the perspective of expanding operators' opportunities. In such a scenario, RAN architectures need to evolve, embracing more and more reconfigurable radio platforms, flexibility in resource management via a fully compliant SDN-NFV framework (towards V-RAN), and the use of commercial off-the-shelf hardware when possible, to reduce the cost of the infrastructure [14]. In this context, the RAN internal protocol-layer functional decomposition is widely accepted today and brings to have a wide range of deployment options to explore. Possible solutions involve the RAN functions spread throughout the Radio Unit (RU), Distributed Unit (DU) or Centralized Unit (CU) in the 5G case, as exemplified in Figure 4, or throughout the Remote Radio Head (RRH) and Baseband Unit (BBU) in the 4G case. The

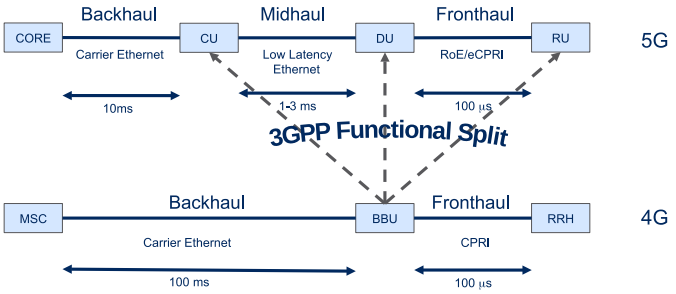


Figure 4. Function view of maximum delays in 5G with respect to 4G.

RAN functional split point can be chosen as a compromise between higher flexibility and higher complexity, in a range that moves from the so-called "Distributed RAN" to the so-called "Centralized RAN" [15]. A specific terminology has been defined [16] to describe the RAN functional split options:

- LLS: Low layer Split; defines the connection and interface between Radio and central units. It is based on CPRI, eCPRI or RoE;
- HLS; High Layer Split; defines the splitting of the internal protocol stack in a typical base-station between distributed and centralized units. Depending on splitting option implemented, interface can be F1, F1-C and F1-U or E1;

 ISBN: 978-1-61208-695-8

- RU: Radio Unit. Contains all RAN functions placed below the LLS interface;
- DU: Distributed Unit. Contains all RAN functions places between LLS and HLS interfaces;
- CU: Centralized Unit. Contains all RAN function above HLS interface and terminates inter-RAN (X2, Xn) interfaces.

Figure 5 shows the 5G splitting architecture concept and the max latency value (based on IEEE 1914.3 [17]), where a proper configuration can be chosen according to the available latency budget for the service [18]. It is worth to note that, in

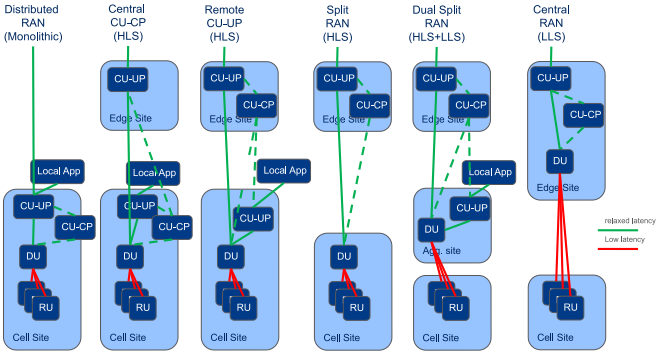


Figure 5. Functional placement scenarios.

the resulting architecture, different functions are not required to be placed at different physical locations. Indeed, where the RU, DU, CU-CP (Control Plane) and CU-UP (User Plane) may be placed depends on:

- the operators' requirements;
- the transport network topology;
- the physical site constraints;
- the latency and capacity infrastructure limitations.

In other words, theoretically, Radio Service Providers should design their split architecture to guarantee all suggested deployment options, unless this needs a too costly orchestration software complexity.

## IV. PoP Transparency

The different placement scenarios can also be seen as a modular migration path from 4G to 5G or, in other words, a smooth method for the introduction of 5G. Technical pros and cons of the RAN functional split are well known and have already been described in literature [5] [19]. Deployments are normally based on OS-level virtualization (i.e., containers) in order to support optimized resource usage [11]: the cost of virtualization is minimized while at the same time the compute slice can be managed at a fine-grain resolution level, allowing a higher degree of flexibility in matching aggressive end-to-end deadline constraints. However, a service deployment approach to 5G architecture implementations implies that, for example, the blue boxes in Figure 5 can be containers managed according to the full flexibility of a service deployment framework. This flexibility in placement throughout the available PoPs in the physical infrastructure needs to be done *transparently* from the applications' viewpoint, and the mapping between the logical topology of containers and their interconnections, on top of the physical infrastructure and PoPs, needs to happen exploiting the descriptive capabilities of MANO VNF descriptors (see Figure 6).

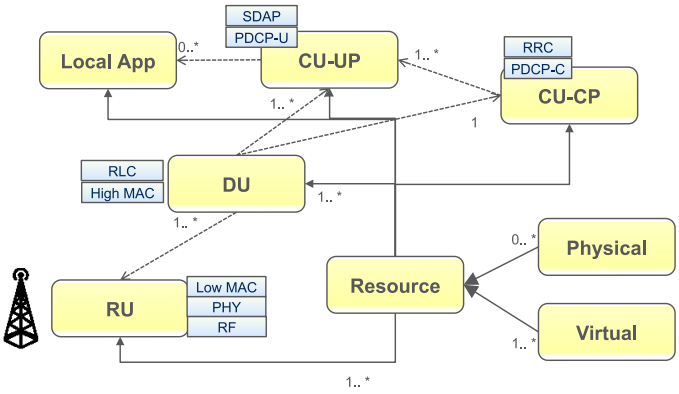


Figure 6. Service chain for functional placement scenarios.

With this new service-oriented mindset, containers need to use communication primitives that:

- are virtualized, so as to be slice definable;
- are always providing the same virtual port to a container, independently of where containers are housed in the infrastructure (see Figure 7);
- are designed in a performance-oriented way, i.e., inter-container communications exhibit the lowest possible latency, fully using special hardware acceleration and software/OS/kernel features to let that be possible.

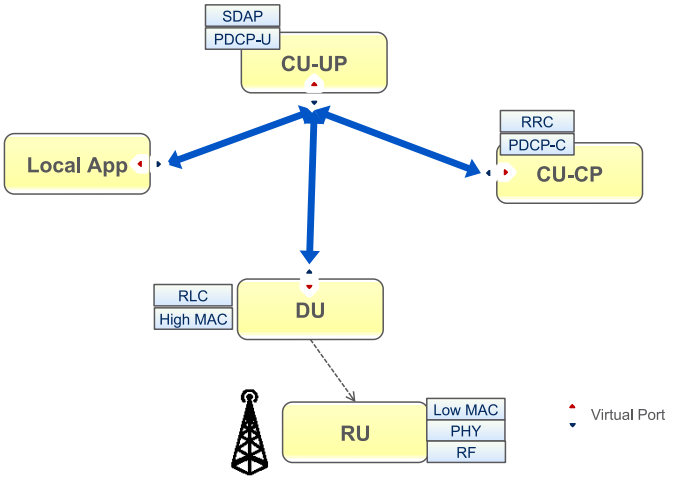


Figure 7. The Virtual Port concept representation.

It is thus crucial to optimize inter-container communications. From a general point of view, this is a key enabler for a RAN solution supporting all possible deployment scenarios.

## V. Inter-Container Communication Performance

Network connections between containers and external nodes have traditionally been implemented by using virtual ethernet pairs and software bridges / virtual switches. When a virtual ethernet pair is created, the kernel creates two software Network Interface Controllers (NICs) (there is no physical NIC

 ISBN: 978-1-61208-695-8

attached to them) connected point-to-point (packets sent to one of the two interfaces are received by the other, and vice-versa). To allow a containerized application to communicate with the external world, one of the two interfaces is inserted in the container namespace, while the other one is attached to a software bridge or a virtual switch (such as openvswitch or similar). This means that, in order to exchange data between applications executing in two containers running on the same physical machine, the following data-path is used:

- The first application sends a network packet using send(), sendto(), write() or similar on the virtual ethernet visible in its namespace.
- The packet is copied from the application address space to the kernel space.
- The kernel networking code moves the packet to the software bridge or virtual switch, that forwards it to the other application.
- The second application receives the packet using recv(), recvfrom(), read() or similar on the virtual ethernet visible in its namespace.
- The packet is then copied from kernel space to the address space of the second application.

As it can be seen, this implies the invocation of at least two system calls, various switches from user-space to kernel space, at least two data copies, different scheduling decisions, and so on. As a result, the networking performance could be penalized. This can be a substantial limitation in supporting the desired 5G functional split, that could be reduced exploiting different communication technologies.

For example, for communications between containers located on the same physical node, it would be possible to map a shared memory region in the address spaces of the two containerized applications and use it for exchanging data. This can be done in a transparent way by using the Intel Data Plane Development Kit (DPDK) framework [20].

DPDK provides a set of libraries originally designed to use a NIC in user space, without passing through the kernel every time a packet is sent or received. Moreover, DPDK allows for sending/receiving packets without relying on hardware interrupts generated by the NIC. This is done by mapping in the application memory the NIC buffer ring and control registers, and directly accessing them at the application level (polling on the NIC registers instead of waiting for interrupts). These techniques allow for a dramatic decrease in the overheads, increasing the achieved throughput and decreasing the latency. DPDK also provides support for virtual NICs, that can be useful for inter-container communications. In particular, it provides drivers for virtio and vhost-user.

Virtio [21] [22] is a para-virtualization standard, also defining virtual NICs based on virtual queues of received and transmitted packets, that can be shared between guest and host. Virtio network devices are generally implemented by hypervisors such as qemu/kvm, that can rely on external services such as vhost [23] to move packets between guest and host, or between different guests on the same host.

The vhost functionalities can be implemented either in kernel space or in user-space. In the former case, the vhost-net kernel module is used, that creates a kernel thread to move packets. In the latter case, a user-space process is responsible [24] for implementing the vhost functionalities, mapping the shared buffers in guest memory. In this approach, known as vhost-user, the user-space process implementing the vhost functionalities uses a UNIX domain socket for low-bandwidth signalling.

DPDK provides a virtio driver that is able to connect to virtio-net virtual interfaces, and a vhost-user driver that can be used by user-space processes (for example, virtual switches) to implement the vhost-user functionalities connecting different VMs. But the vhost-user driver can also be used to implement the virtual interfaces a virtio driver can directly connect to. Hence, a DPDK-based virtual switch running in the host can create virtio-net interfaces which DPDK-based applications running in the containers can connect to. Figure 8 shows the inter-container communication support provided by the presented approaches.

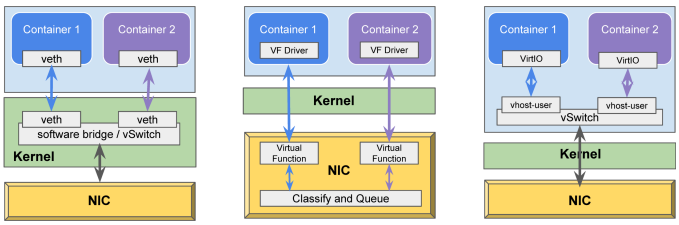


Figure 8. Inter-container virtual switching: (a) software-only solution; (b) using SR-IOV support; (c) using DPDK with vhost in user mode.

In order to evaluate the impact of the overhead introduced by the packet transmission mechanism (and the advantages of using different software architectures), we performed some experiments on an Intel(R) Xeon(R) CPU E5-2640 at 2.40GHz.

The first experiment is designed to measure the overhead caused by the system calls needed to send packets through virtual Ethernet pairs. It is based on two applications sending and receiving small User Datagram Protocol (UDP) packets, located on the same physical machine: the first application (running in an lxc container) sends packets at the maximum possible rate, and the second application (running in a different container) measures the received packet rate. Even without using a software bridge or switch (inserting one of the two virtual Ethernets in the first container and the other one in the second container), the maximum achievable packet rate is about 310000 packets per second (pps). Considering a payload of 64 bytes, this results in a throughput of less than 160 Mbps.

To evaluate the advantages of using the DPDK virtio and vhost-user drivers (running in user space), we performed a second experiment using two lxc-based containers and the "testpmd" DPDK application:

- an instance of testpmd running in the host provides two virtio interfaces (one per container) using vhost-net, and forwards packets between them, acting as a bridge;
- an instance of testpmd in the first container produces packets and sends them on the first virtio interface;
- an instance of testpmd in the second container receives packets from the second virtio interface.

 ISBN: 978-1-61208-695-8 

Using this setup, it has been measured that the applications can transmit about 12800000 pps (considering 64-bytes packets, this is about 6.5Gbps). Note that the "testpmd" application connecting the two containers is not a real switch, but a DPDK application that is used only to test the drivers' performance.

Vector Packet Processing (VPP) [25] is a technology used in the virtual switch provided by the Fast Data Project (FD.io) [26]. In a standard switch, each packet is received, processed, and forwarded before receiving the next packet from the NIC queue, and this way of serving packets can have bad effects on cache locality (and on the forwarding performance). Hence, VPP receives, processes, and forwards packets in batches, resulting more cache friendly and achieving a higher networking performance.

We performed a third experiment using VPP (which can be used as a real bridge, switch, or router) instead of testpmd to connect the two containers. This experiment, performed with the goal of evaluating the performance of a complete switching solution (and not only the performance of the userspace drivers) revealed that the packet rate drops to about 6400000pps (3.27Gbps). Such a lower performance is due to the real switch logic that is present in VPP and not in testpmd.

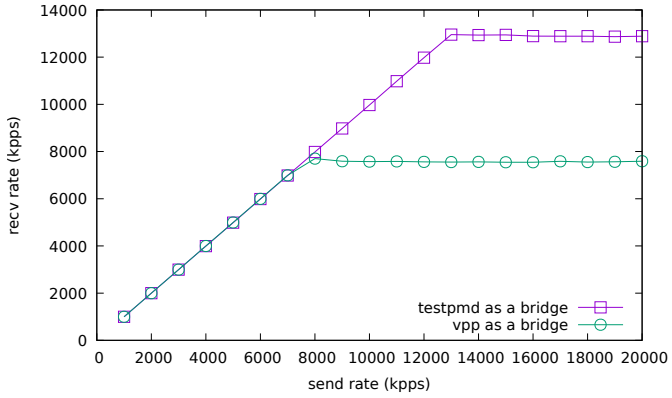


Figure 9. Throughput of testpmd and VPP (in kilo packets per second) as a function of the input packet rate.

To better characterize the performance of the DPDK polling drivers (evaluated using the "testpmd" application as a bridge) and of a DPDK-based switch (we used VPP in this case), we repeated the previous experiments changing the rate at which the 64-byte packets are generated. Figure 9 shows the packet rate (in kpps) that "testpmd" and VPP are able to forward as a function of the input packet rate. While testpmd manages to handle 13 million packets per second, VPP sustains only 8 million pps, due to its higher functional complexity.

## VI. Conclusions

SDN-NFV enables unprecedented flexibility and ease of maintenance for service chains deployments, but the achievable end-to-end performance is greatly affected by what mechanisms are used for the underlying communications among micro services, regardless of these being hosted as traditional Virtual Machines or containers. Solutions based on containers are becoming the de-facto standard for efficient usage of resources, and inter-container communications can bootstrap an effective Radio Access Technology (RAT) software architecture for 5G. This becomes even more important for critical latency-sensitive RAT services, that cannot be hosted anywhere, being constrained to be located not too far from their needed radio elements. In this context, it is also possible to leverage the specification of the PoP through VNF file descriptors, so as to achieve a RAT service chain configuration corresponding to the needed trade-off between distributed and centralized RAN solutions. However, as shown elsewhere [27] and remarked in this paper, the performance of container-to-container communications has a great potential to affect the finally achievable End-to-End performance, also depending on the hardware accelerations and software optimizations that are available in the underlying infrastructure. Therefore, inter-container communication is a key element to realize 5G implementations fully exploiting the potential of SDN-NFV architectures, where the work presented in this paper, including the general overview of the involved technological hardware and software solutions, along with the experimental comparison among a few of them, is just a starting point for a more structured in-depth study, needed to design effective and efficient solutions that become enablement factors for future 5G scenarios.

## VII. Future Work

Concerning directions for future work on the topic, additional experimentation evaluating the impact of different software and hardware architectures on the performance of inter-container communications is needed. For example, the use of SR-IOV capable NICs has the advantage of off-loading CPU packet-processing workload to the NIC [28], but in the case of communications among entities on the same physical node this might be easier to handle in software, avoiding unnecessary bus cycles. This has to be evaluated also in light of the fact that high-performance software-based switching solutions within hypervisors and operating systems is already going towards dropping the support of the full set of features of a switch, in favour of more static (but faster) solutions like macvlan/macvtap [29] or Virtual Ethernet Bridge (VEB) [30] [31]. Indeed, static solutions such as macvtap have been shown [32] to perform better in high-packet rate scenarios than more dynamic and flexible solutions like full-featured virtual switching. Further trade-off points between performance and flexibility might become possible in presence of specific hardware features, like full SR-IOV support. It is also interesting to perform additional experimental results, and compare them with benchmarks already appeared in literature [33] [34]. In this context, the optimality of the solution has to face additional possible constraints, like the ones behind the Virtual Ethernet Port Aggregator (VEPA) [31] and its use with switches supporting the hairpin-mode. This forces packets to reach the external adjacent switch even in case of local communications, due to the need for exposing all traffic, including the internal one, to the networking monitoring and management layer in a uniform way, however the performance is expected to lower in this case. Also, in the context of high-performance packet processing for NFV, approaches that are gaining popularity are the kernel-bypass ones [35] [36], that do not rely on traditional TCP/IP networking support by the operating system or hypervisor, whilst direct access to the hardware NIC is preferred (either its physical or virtual functions if SR-IOV is in place), on top of which custom and optimized user-space networking stacks are built. Having the possibility to control such features from a high level, such as through VNF MANO

 ISBN: 978-1-61208-695-8 

descriptors, is all but straightforward [37], so additional work is needed along such direction.

## REFERENCES


[1] "Open Network Survey Report," NetGate, White Paper, February 2018. [Online]. Available: https://www.netgate.com/resources/whitepapers/open-networking-survey-report.html

[2] "5G White Paper," NGMN Alliance, White Paper, February 2015. [Online]. Available: https://www.ngmn.org/fileadmin/ngmn/content/images/news/ngmn_news/NGMN_5G_White_Paper_V1_0.pdf

[3] "IMT vision – framework and overall objectives of the future of IMT for 2020 and beyond – International Telecommunication Union," ITU-R, Standard Recommendation I.2083-0, September 2015.

[4] Delivering an Integrated, Secure, and Radio-Aware 5G Transport Network. [Online]. Available: https://www.juniper.net/assets/us/en/local/pdf/solutionbriefs/3510647-en.pdf [retrieved: September, 2018]

[5] C. Vitucci and A. Larsson, "Flexible 5G Edge Server for Multi Industry Service Network," International Journal on Advances in Networks and Services, vol. 10, no. 3-4, 2017, pp. 55—65.

[6] "View on 5G Architecture – version 2.0," 5GPPP Architecture Working Group, Standard, December 2017. [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2018/01/5G-PPP-5G-Architecture-White-Paper-Jan-2018-v2.0.pdf

[7] "Building the Next Generation RAN," O-RAN Alliance, White Paper, October 2018. [Online]. Available: https://static1.squarespace.com/static/5ad774cce74940d7115044b0/t/5bc79b371905f4197055e8c6/1539808057078/O-RAN+WP+FInal+181017.pdf

[8] "Network Functions Virtualisation (NFV); Use Cases," ETSI, Standard ETSI GS NFV 001, v.1.1.1, October 2013. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf

[9] A. Kaloxylos, "A Survey and an Analysis of Network Slicing in 5G Networks," IEEE Communications Standards Magazine, vol. 2, no. 1, March 2018, pp. 60–65.

[10] "Network Function Virtualisation (NFV); Management and Orchestration," ETSI, Standard ETSI GS NFV-MAN 001, v.1.1.1, December 2014. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf

[11] M. Marinoni, T. Cucinotta, L. Abeni, and C. Vitucci, "Allocation and Control of Computing Resources for Real-Time Virtual Network Functions," in Proc. of the international Symposium on Advances in Software Defined Networking and Network Function Virtualization (SoftNetworking 2018), April 2018, pp. 52–57.

[12] "5G End-to-End architecture Framework," NGMN Alliance, Standard 180226 NGMN E2EArchFramework V2.0.0, February 2018. [Online]. Available: https://www.ngmn.org/fileadmin/ngmn/content/downloads/Technical/2018/180226_NGMN_RANFSX_D1_V20_Final.pdf

[13] "D2.5 Final Overall Architecture Definition, Release 2," April 2015. [Online]. Available: https://cordis.europa.eu/docs/projects/cnect/9/318109/080/deliverables/001-318109MCND25renditionDownload.pdf

[14] C. Vitucci and A. Larsson, "SEED, A Server Platform for the Edge of the Network," in ICN 2017: The sixteenth Conference on Networks, April 2017, pp. 118–123.

[15] "Study on new radio access technology: Radio access architecture and interfaces," 3GPP, Standard 38.801, TR V14.0.0, March 2017. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3056

[16] "MGMN Overview on 5G RAN Functional Decomposition," NGMN Alliance, Standard 180226 NGMN RANFSX D1 V20 Final, February 2018. [Online]. Available: https://www.ngmn.org/fileadmin/ngmn/content/downloads/Technical/2018/180226_NGMN_RANFSX_D1_V20_Final.pdf

[17] "IEEE Standard for Radio over Ethernet Encapsulations and Mappings," IEEE, Standard IEEE 1914.3-2, October 2018. [Online]. Available: https://standards.ieee.org/standard/1914_3-2018.html

[18] H. Gupta, D. Manicone, F. Giannone, K. Kondepu, A. Franklin, P. Castoldi, and L. Valcarenghi, "How much is fronthaul latency budget impacted by RAN virtualisation ?" in Proc. of the IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN 2017), November 2017, pp. 315–320.

[19] J. Harrison and M. Do. Mobile Network Architecture for 5G Era - New C-RAN Architecture and distributed 5G Core. Netmanias. [Online]. Available: http://www.netmanias.com/en/post/blog/8153/5g-c-ranfronthaul-kt-korea-sdn-nfv-sk-telecom/mobile-networkarchitecture-for-5g-era-new-c-ran-architecture-and-distributed-5g-core (2015)

[20] Intel Corporation. Data Plane Development Kit (DPDK). [Online]. Available: http://www.dpdk.org [retrieved: 14 February, 2019]

[21] R. Russell, "VIRTIO: Towards a De-facto Standard for Virtual I/O Devices," ACM SIGOPS Operating Systems Review, vol. 42, no. 5, 2008, pp. 95–103.

[22] R. Russell, M. Tsirkin, C. Huck, and P. Moll, "Virtual I/O Device (VRTIO) Version 1.0," OASIS Specification Committee, Standard, 2015. [Online]. Available: http://docs.oasis-open.org/virtio/virtio/v1.0/csd01/virtio-v1.0-csd01.html

[23] M. S. Tsirkin, "vhost-net and virtio-net: Need for Speed," in Proc. of the KVM Forum, May 2010.

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

[25] D. Barach, L. Linguaglossa, D. Marion, P. Pfister, S. Pontarelli, and D. Rossi, "High-Speed Software Data Plane via Vectorized Packet Processing," IEEE Communications Magazine, vol. 56, no. 12, 2018, pp. 97–103.

[26] LF Projects, LLC. Fast Data Project (FD.io). [Online]. Available: http://www.fd.io [retrieved: 14 February, 2019]

[27] T. Cucinotta, L. Abeni, M. Marinoni, and C. Vitucci, "The Importance of Being OS-aware - In Performance Aspects of Cloud Computing Research," in Proc. of the 8th International Conference on Cloud Computing and Services Science, CLOSER 2018, March 2018, pp. 626–633.

[28] P. Kutch and B. Johnson, "SR-IOV for NFV Solutions," Technical Brief, February 2017. [Online]. Available: https://www.intel.com/content/dam/www/public/us/en/documents/technology-briefs/sr-iov-nfv-tech-brief.pdf

[29] H. Liu. Introduction to Linux interfaces for virtual networking. [Online]. Available: https://developers.redhat.com/blog/2018/10/22/introduction-to-linux-interfaces-for-virtual-networking/ [retrieved: October, 2018]

[30] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," IEEE Communications Magazine, vol. 51, no. 11, November 2013, pp. 24–31.

[31] "Virtual Networking Management White Paper – Version 1.0.0," Distributed Management Task Force (DMTF), Standard DSP2025, February 2012. [Online]. Available: https://www.dmtf.org/sites/default/files/standards/documents/DSP2025_1.0.0.pdf

[32] L. Abeni, C. Kiraly, N. Li, and A. Bianco, "On the performance of KVM-based virtual routers," Computer Communications, vol. 70, 2015, pp. 40–53.

[33] J. Anderson, H. Hu, U. Agarwal, C. Lowery, H. Li, and A. Apon, "Performance considerations of network functions virtualization using containers," in Proc. of the International Conference on Computing, Networking and Communications (ICNC 2016), February 2016, pp. 1–7.

[34] J. Jose, M. Li, X. Lu, K. Kandalla, M. Arnold, and D. Panda, "SR-IOV Support for Virtualization on InfiniBand Clusters: Early Experience," in Proc. of the 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, CCGrid 2013, May 2013, pp. 385–392.

[35] K. Mahabaleshwarkar, N. Mundada, A. Chavan, and A. Panage, "TCP/IP protocol accelaration," in Proc. of the International Conference on Computer Communication and Informatics, January 2012, pp. 1–4.

[36] J. Tan, C. Liang, H. Xie, Q. Xu, J. Hu, H. Zhu, and Y. Liu, "VIRTIO-USER: A New Versatile Channel for Kernel-Bypass Networks," in Proc. of the Workshop on Kernel-Bypass Networks, ser. KBNets '17. New York, NY, USA: ACM, August 2017, pp. 13–18.

[37] X. Luo, F. Ren, and T. Zhang, "High Performance Userspace Networking for Containerized Microservices," in Proc. of the 16th International Conference on Service-Oriented Computing (ICSOC 2018), November 2018, pp. 57–72.


 ISBN: 978-1-61208-695-8 