Virtual Radios, Real Services: Enabling RANaaS Through Radio Virtualisation

Joao F. Santos, Maicon Kist, Juergen Rochol, and Luiz A. DaSilva, *Fellow, IEEE*

***Abstract*—Network slicing is one of the key enabling techniques for 5G, allowing Mobile Network Operators (MNOs) to support services with diverging requirements on top of their physical network infrastructure. The MNOs should be able to offer Network Slices (NSs) as a Service (NSaaS) and provide customisable and independent virtual networks to tenants. In this paper, we address the functionality and challenges for enabling Radio Access Network (RAN) as a Service (RANaaS) through radio virtualisation. We analyse the requirements for using radio hypervisors to support RANaaS, and we evaluate how the current state-of-the-art on radio virtualisation meets such requirements. We identify, formalise and address the key resource management functionality missing from existing radio hypervisors. Then, we present eXtensible Virtualisation Layer (XVL), a software layer that provides the missing resource management functionality for enabling RANaaS and can be added on top of existing radio hypervisors. We integrated XVL with a radio hypervisor, forming a RANaaS platform that can provision heterogeneous RAN slices as a service. We outline XVL's architecture and design choices, as well as evaluate its performance in terms of the computational overhead, the delay to provision virtual radios, the delay introduced to forward IQ samples, and the signal degradation. Our results show that XVL enables leveraging existing radio hypervisors for supporting the RANaaS paradigm.**

***Index Terms*—Network Slicing, Radio Access Network, Radio Virtualisation, RAN as a Service, Radio Hypervisor**

I. Introduction

In contrast to previous generations of mobile networks, 5G is being envisioned from the very beginning to support a variety of different services, e.g., Enhanced Vehicular-to-Everything (V2X), massive Internet of Things (IoT), and industrial automation [1]. To cope with the requirements of such diverse communication services, 3GPP introduced the concept of network slicing, which proposes the partitioning of the physical network infrastructure of a Mobile Network Operator (MNO) into independent logical networks, known as Network Slices (NSs) [2]. Each NS operates as a separate end-to-end (E2E) network, which can be individually tailored and configured for different purposes [3]. Network slicing creates new business models for the mobile networks market by enabling new ways for MNOs to monetise their infrastructure [4]. Through slicing, the MNOs can offer NS as a Service (NSaaS) to provide independent and isolated virtual networks to tenants, running on top of a shared physical network infrastructure [2].

The 3GPP defines E2E network slicing as the combination of Core Network (CN) slicing and Radio Access Network (RAN) slicing [5]. Both network segments can be independently orchestrated, sliced, and combined for creating E2E NSs [6]. As a consequence, the NSaaS paradigm becomes the combination of CN as a Service (CNaaS) [7] and RAN as a Service (RANaaS) [8]. This separation grants MNOs the flexibility to offer: CN slices, where each tenant can define its own authentication schemes, mobility and session management, whilst sharing a common RAN among all tenants; RAN slices, where each tenant can specify its own coverage area, Radio Access Technologies (RATs), and numerologies, whilst sharing a common CN among all tenants; or full E2E NSs, where tenants can define both their own RAN and CN [4].

The slicing process is performed by entities known as hypervisors, which are tailored for virtualising a particular type of resource: network hypervisors use network resources to create virtual networks, and radio hypervisors use radio resources to create virtual radios. Currently, the virtualisation of the CN is a mature area, with network hypervisors, e.g., FlowVisor [9] and OpenVirteX [10], being adopted in production networks, while RAN virtualisation is still a new research topic.

Prototype implementations of radio hypervisors are essential for analysing the benefits of virtualisation, as well as its drawbacks, e.g., performance impacts due to virtualisation overheads, and signal degradations due the isolation between virtual radios [11] [12]. Open-source radio hypervisors available in the literature can create virtual radios for realising RAN slices using either low-level resources, e.g., time and frequency; or high-level resources, e.g., Physical Resource Blocks (PRBs) and frames [11]. Although paving the way for the realisation of RAN slicing, these prototypes mainly focus on enabling the coexistence of multiple RAN slices on top of a single physical hardware, without considering the additional requirements for RANaaS. For example, the lack of radio resource broker capabilities for tenants to query the available radio resources, request RAN slices, and assess the performance of their RAN deployment, prevents the realisation of RANaaS with current radio hypervisors.

This paper addresses the challenges associated with resource management functionality for RANaaS through a radio virtualisation solution. It extends our earlier work [13] by providing an expanded qualitative evaluation of the features necessary for

Manuscript received August 01, 2019; revised March 01, 2020 and July 13, 2020; accepted July 14, 2020. The research leading to this letter received funding from the European Horizon 2020 Program under grant agreements No. 732174 (ORCA) and No. 732497 (5GINFIRE), and by the Science Foundation Ireland under grant 13/RC/2077.

Joao F. Santos is with CONNECT and Trinity College Dublin, Ireland. Email: facocalj@tcd.ie

Luiz DaSilva is with the Commonwealth Cyber Initiative (CCI) and Virgnia Tech, USA. Email: ldasilva@vt.edu

Maicon Kist and Juergen Rochol are with the Department of Computer Science, Federal University of Rio Grande do Sul, Brazil. Emails: {kistm, juergen}@inf.ufrgs.br

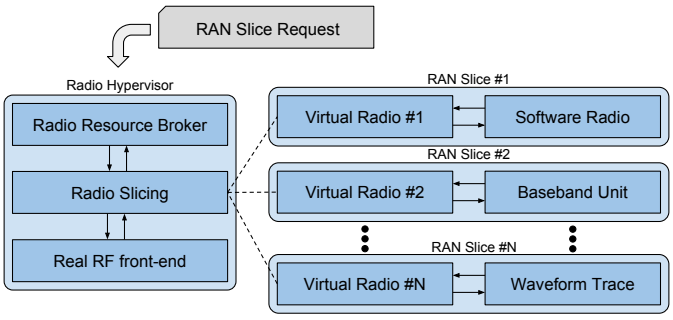


Fig. 1: Functional example of a RANaaS platform leveraging a radio hypervisor and supporting the creation of RAN slices on demand. The radio hypervisor slices the real RF front-end and creates isolated virtual radios. Each RAN slice has a virtual radio and a source/destination for streams of IQ samples, e.g., produced by software radios, baseband units, or waveform traces.

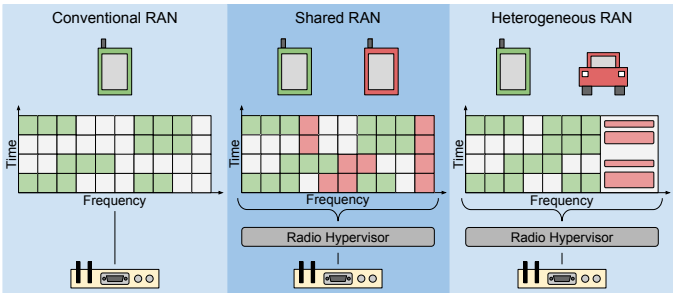


Fig. 2: Radio hypervisors enable sharing of the physical network infrastructure by multiple tenants and/or the use of a single hardware platform to realise heterogeneous RATs. The different colours (green and red) indicate RAN slices belonging to different tenants.

radio hypervisors to support RANaaS. We also formalise the problem of radio processing isolation between virtual radios, as well as propose and implement a solution to achieve this type of isolation. The key contributions of this paper are as follows:

1) We identify the resource management requirements for using radio hypervisors to enable RANaaS.
2) We introduce a software layer that wraps around radio hypervisors and provides them with the missing resource management functionality to be suitable for RANaaS.
3) We formalise and address the isolation of virtual radios at the radio processing level.

The combination of our software layer with a radio hypervisor results in a prototype RANaaS platform, as illustrated in Figure 1. To the best of our knowledge, our RANaaS platform is the first to enable the provisioning of heterogeneous RAN slices as a service, isolated both in radio resource and radio processing levels, including a monitoring interface for tenants to query the performance of their RAN slices.

The remainder of this paper is organised as follows. In Section II, we discuss the role of radio virtualisation in the context of RANaaS and identify the resource management requirements for using radio hypervisors to support RANaaS. In Section III, we introduce a software layer that implements the missing resource management functionality and can be added on top of existing radio hypervisors, allowing them to provision RAN slices as a service. In Section IV, we formalise and address the challenge of isolating virtual radios at the radio processing level. In Section V, we describe the integration of our software layer with a radio hypervisor, evaluate its performance in different scenarios, and validate its capability to serve as an enabler for RANaaS. In Section VI, we categorise the existing radio hypervisors based on their radio virtualisation mechanisms, evaluate how they meet the requirements for RANaaS, and position our work with respect to the current state-of-the-art on radio virtualisation for RANaaS. Finally, in Section VII, we summarise our conclusions and discuss possible directions for future work.

II. Enabling RAN as a Service Through Radio Virtualisation

Radio hypervisors enable shared RAN deployments, where multiple tenants, e.g., Mobile Virtual Network Operators (MVNOs), Service Providers (SPs) and verticals, share a common physical network infrastructure, decreasing the deployment costs and the amount of underutilised radio resources. Some radio hypervisors can support heterogeneous RAN deployments, enabling a single physical radio to realise multiple RATs, reducing the number of physical radios needed for deploying heterogeneous mobile networks. Figure 2 illustrates two scenarios in which a radio hypervisor manages the radio resources for creating different RAN slices. The provision and instantiation of RAN slices can be offered as a service by MNOs and Infrastructure Providers (InPs) (which do not provide services to end users), breaking the monolithic nature of legacy mobile networks into value-chain networks [14].

Virtual Radio [15], one of the initial works on radio virtualisation, argued for the deployment of software-based base stations, with a tailored amount of radio resources and protocol stack to mitigate the slow development cycle in the wireless network industry. Ultimately, the work of [15] proposes the use of radio virtualisation for enabling the deployment of virtual wireless networks on demand, years before the term RANaaS was coined [8]. The majority of the following research efforts on radio virtualisation have focused on creating prototype radio hypervisors to enable radio slicing, and ensuring isolation at the radio resource level [11]. The next natural step is to leverage the capabilities of current radio hypervisors for provisioning RAN slices on demand and as a service, towards enabling the RANaaS paradigm.

Inspired by the initial concepts in Virtual Radio, and further works on wireless network virtualisation [11] [16]–[18], we now examine the use of radio virtualisation for enabling the RANaaS paradigm. In our earlier work [13], we investigated the requirements of a RANaaS platform leveraging radio virtualisation. These requirements include:

- Resource Negotiation: to possess an interface for querying and requesting RAN slices, describing both the radio resources, e.g., in terms of centre-frequency and band-

width; and the virtual radios, e.g., owner, transmitter and/or receiver capabilities.
- Dynamic Allocation: to allow the creation and destruction of virtual radios on demand, without interrupting the operation of the radio hypervisor or other virtual radios.
- Technology Agnostic: to employ radio virtualisation mechanisms that support different RATs, and do not limit the choice of the PHY and MAC layers of the RAN slices.
- Radio Resource Isolation: to allocate non-overlapping radio resources to different virtual radios, and prevent interference between virtual radios, e.g., through filtering, guard bands and/or intervals.
- Radio Processing Isolation: to ensure that virtual radio instances cannot affect the operation and performance of each other, even in case of a malfunctioning or misbehaving virtual radio.
- Service Monitoring: to collect and provide information about the performance of individual virtual radios, e.g., in terms of buffer sizes, the introduced delay, and SINR degradation.

Radio hypervisors that meet these resource management requirements can act as RANaaS platforms, enabling MNOs and InPs to decrease their deployment costs and introduce new sources of revenue [13]. However, not all types of radio hypervisors are suitable or possess the complete set of features for supporting RANaaS (to be further discussed in Section VI). Given its capabilities, reliability, and open-source nature, in this work we adopt HyDRA [12] as the starting point in the development of a prototype RANaaS platform.

III. Extensible Virtualisation Layer

We have developed eXtensible Virtualisation Layer (XVL), a cross-platform software layer that sits on top of existing radio hypervisors. XVL works as a wrapper, leveraging the radio slicing capabilities of an underlying radio hypervisor, and providing it with the missing resource management functionality for supporting RANaaS. This modular design allows the radio hypervisor to focus on the radio virtualisation, offloading to XVL the majority of other tasks, e.g., the communication interface, resource allocation, and service monitoring. In the following subsections, we detail how XVL addresses the limitations and missing features that we identified in Section II. Namely, how XVL: provides a communication interface for resource negotiation and allocation, supports the creation and destruction of virtual radios on demand, ensures their radio processing isolation, and monitors their performance.

A. Resource Negotiation and Dynamic Allocation

XVL operates on a client-server paradigm, with a single server, i.e., one instance of XVL on top of a radio hypervisor, serving multiple clients, i.e., the different tenants. The clients can send messages to the server for querying information about the use of resources and requesting resources at any time. Figure 3 depicts the process of negotiating and using resources through XVL. Upon successful negotiation and reservation of resources, XVL interfaces with the underlying radio hypervisor for creating/destroying the RAN slices. XVL

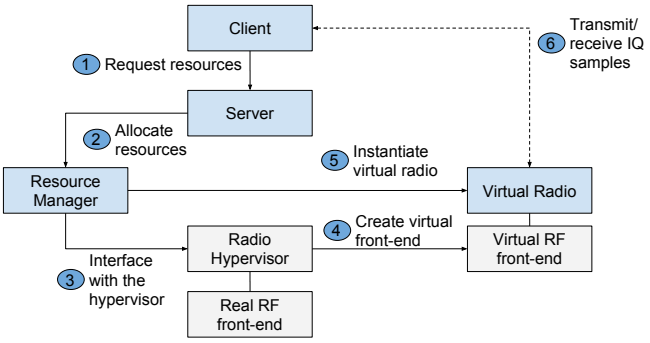


Fig. 3: The steps involved for negotiating the use of resources and allocating new virtual radios. XVL (blue) stands as a wrapper around the radio hypervisor (grey).

has callbacks for informing the radio hypervisor about changes in the resource mapping and/or allocation of virtual radios, so the radio hypervisor can instantiate a new virtual RF front-end. Once the radio hypervisor completes the creation of the virtual RF front-end, XVL performs additional configurations in the virtual radio (to be further discussed in Section III-B). Finally, XVL informs the client of the proper ways to reach its virtual radio, e.g., through CPRI, OBSAI, or a UDP socket, from which the clients can start transmitting/receiving streams of IQ samples.

At present, XVL supports virtualisation in the frequency domain. However, XVL separates the resource management from the underlying techniques employed by the radio hypervisor, which allows the resource manager to be extended and modified with ease. Thus, XVL can easily be extended for supporting time or space instead of frequency resources, or even incorporate time and space as new degrees of freedom in the resource definition. It only depends on the radio virtualisation mechanism of the underlying radio hypervisor. Currently, the radio resources are defined in terms of a centre frequency, a bandwidth, a client ID, and an UDP port.

B. Independence between Virtual Radios

Every virtual radio has a virtual RF front-end, which the virtual radio employs for transmission and/or reception. Each virtual RF front-end uses a fraction of the total radio resources of the real RF front-end. The radio resources can be defined in terms of bandwidth, timeslot, or antennas, depending on the radio hypervisor's approach for virtualising the real RF front-end. Regardless of the type of radio resource, each virtual radio must send an appropriate number of IQ samples to the radio hypervisor at precise timing, as discussed in Section IV. The radio hypervisor consumes the IQ samples of all virtual radios at once, and multiplexes them into a single stream of IQ samples. The number of necessary IQ samples per virtual radio, however, varies depending on the amount of radio resources per virtual RF front-end. In the case of an FFT-based hypervisor, e.g., HyDRA, the bandwidth of each virtual RF front-end is mapped onto a number of FFT bins. The virtual radios must generate a number of samples equal to the number of FFT bins to create an FFT window. Then,

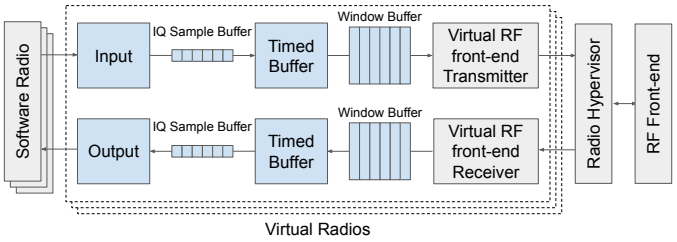


Fig. 4: Timed buffers consume the received IQ samples and generate the windows that the radio hypervisor requires. The same process happens in the opposite direction.

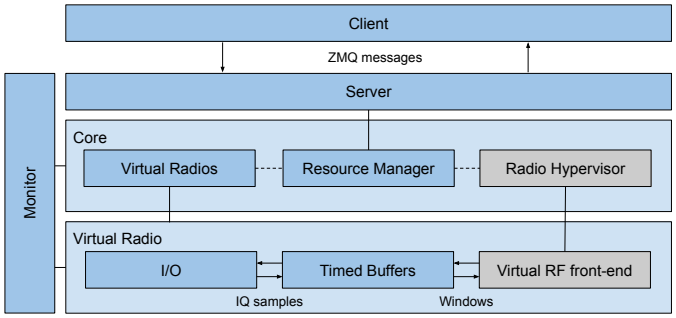


Fig. 5: Block diagram showing the architecture of XVL, the elements that compose it (blue), and the elements it must interface with (grey).

the radio hypervisor multiplexes the windows of the virtual RF front-ends together using an IFFT [12].

XVL employs timed buffers to ensure that the radio hypervisor receives the appropriate number of IQ samples at the right moment. Based on the overall sampling rate of the virtual radio and the hypervisor, the timed buffers output IQ samples at the precise moment for the radio hypervisor's consumption (we will present more details on this in Section IV). Figure 4 illustrates the operation of timed buffers inside a virtual radio, where UDP sockets are used for receiving/sending IQ samples from/to remote tenants. In the case of overflows, XVL's internal timed buffer will start filling, until reaching a cap. After that, incoming samples are dropped to prevent the virtual radio process from overflowing memory. In the case of underflows, XVL pads the streams of IQ samples of empty windows, i.e., fills them with zeroes, while waiting for the missing samples. This way, the radio hypervisor and the remainder of the system will not halt waiting for the delayed or missing IQ samples of a given virtual radio.

The virtual radios can either operate as transmitters, receivers, or transceivers. However, the virtual radios may not be symmetrical in both directions, i.e., they may employ different RATs and require different amounts of radio resources. For that reason, every virtual radio in XVL possesses completely independent transmitter and receiver chains. Figure 4 shows an example of a virtual radio with both types of chains: the transmit chain (top) receives samples from a given software radio, and the receive chain (bottom), sends samples towards a given software radio. The radio hypervisor may realise each type of chain in different manners, i.e., different virtualisation approaches, or may use different radio hardware, i.e., different physical devices for transmission and reception.

C. Architecture and Monitoring Capabilities

Figure 5 illustrates the overall architecture of the XVL implementation. The clients interface with the server for resource discovery and negotiation through ZMQ messages. The server queries the resource manager and tries to fulfil requests. The resource manager has a list of virtual radios and can interact with the radio hypervisor for creating or destroying virtual RF front-ends. Each virtual radio has an Input/Output (I/O), a timed buffer and a virtual RF front-end, which is tied to the radio hypervisor. Aside from the functionality for supporting services, XVL also has a monitor utility for inspecting the Key Performance Indicators (KPIs) of the virtual radios.

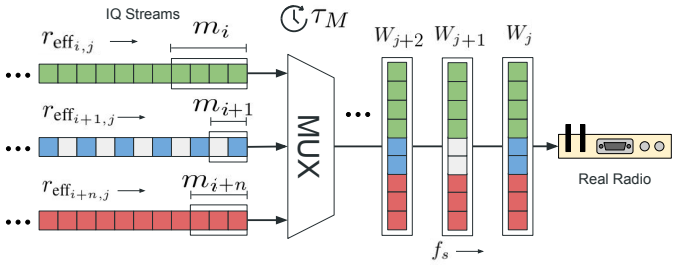
Currently, the monitoring utility is used for assessing the use of computational resources per virtual radio. It measures the buffer sizes, the sampling rates, and the time each virtual radio has to fill multiple windows. Based on the buffer sizes, it is possible to determine whether the software radio is presenting overflows, i.e., buffer sizes increasing steadily, or underflows, i.e., buffer sizes frozen at zero. We plan to extend this mechanism to evaluate the introduced SINR degradation and delay.

IV. Radio Processing Isolation

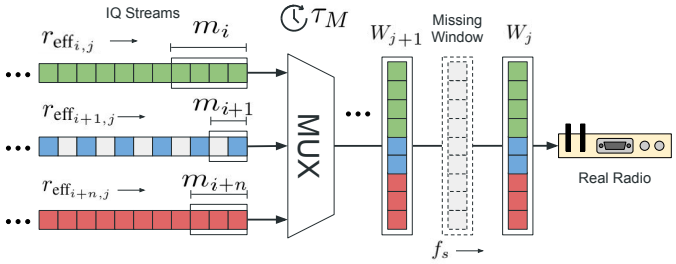
In this section, we focus on formalising and addressing the radio processing isolation functionality. We describe the reasoning behind the operation of the timed buffers, and derive the necessary amount of padding for isolating the processing of the streams of IQ samples of the virtual radios.

The radio virtualisation mechanisms of technology-agnostic radio hypervisors are ultimately multiplexing operations, where each virtual radio has access to a fraction of the overall radio resources of the physical radio. The existing technology-agnostic radio hypervisors (as shown in Table II) employ frequency-domain radio virtualisation mechanisms, which partition the bandwidth of a physical radio's RF front-end (B) into spectrum chunks, abstracted to the virtual radios in the form of the virtual bandwidth (b_i) of their virtual RF front-end, $i = 1, \ldots, N$. The virtual radios interact with their virtual RF front-end as they would with a real RF front-end, sending/receiving streams of IQ samples. The radio virtualisation mechanism ensures isolation between the virtual radios at the radio resources level, using non-overlapping radio resources and including the necessary guard bands and/or intervals.

The multiplexing of the IQ samples generated by multiple virtual radios occurs on a per-window basis, where the radio hypervisor collects a certain number of IQ samples (M) from/to the virtual radios over a time period (τ_M) of duration equal to the ratio between M and the sampling rate of the physical radio (f_s), i.e., $\tau_M = M/f_s$. Only when in possession of M IQ samples, the radio hypervisor is able to multiplex/demultiplex the IQ samples from/to multiple virtual radios. For frequency-domain radio virtualisation mechanisms, each virtual radio is expected to produce/consume a given



(a) With the dynamic padding, the missing IQ samples only affect the performance of the $(i+1)$-th virtual radio, and the radio hypervisor can generate multiplex windows at every τ_M.



(b) Without the dynamic padding, the missing IQ samples affect all virtual radios, as the radio hypervisor cannot generate windows at every τ_M and has to wait until it collected M IQ samples.

Fig. 6: Multiplexing of multiple virtual radios with and without the dynamic padding of IQ samples, when the $(i+1)$-th virtual radio presents an effective rate inferior to its expected nominal rate. The use of dynamic padding ensures isolation between virtual radios at the level of processing their IQ samples, preventing any virtual radio from delaying the production/consumption of multiplex windows.

number of IQ samples (m_i) within τ_M, corresponding to a fraction of M equal to the ratio between its virtual bandwidth b_i and the total bandwidth B:

$$m_i = \left\lceil M \cdot \frac{b_i}{B} \right\rceil . \tag{1}$$

The ceiling operation ($\lceil . \rceil$), required to ensure an integer number of m_i samples, may introduce a mismatch between the allocated and requested virtual bandwidth, leading to signal distortions [19]. For simplicity, we restrict the choice of b_i such that m_i is always an integer, avoiding this mismatch.

On a RANaaS platform, the virtual radios send/receive streams of IQ samples from/to different tenants. It is likely that the virtual radios will not always produce/consume the exact number of m_i IQ samples during every multiplex window, i.e., generate samples at an expected nominal rate $r_i = m_i/\tau_M$. Instead, during the j-th multiplex window (W_j), the i-th virtual radio will present an effective rate ($r_{\text{eff}_{i,j}}$), which may vary for each window and can be inferior to the expected nominal rate ($r_{\text{eff}_{i,j}} \leq r_i$). Such an effective rate may be the result of *(1)* the multiple access scheme of RATs which intermittently transmit frames, e.g., WiFi and Bluetooth; *(2)* the behaviour of software radios and baseband units which cannot process IQ samples at the expected nominal rate; or

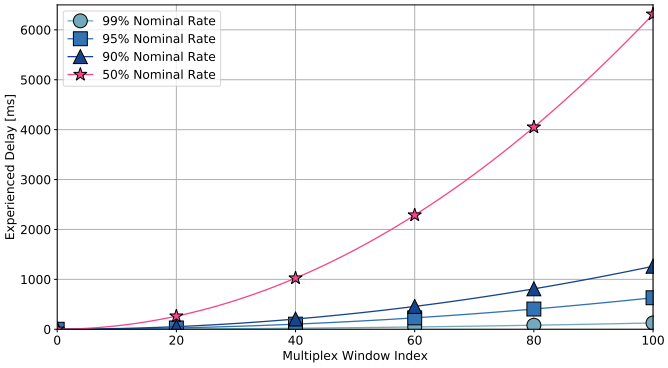


Fig. 7: How different effective rates would affect the experienced delay over time, if dynamic padding were not applied. In this scenario, we have a radio hypervisor at $f_s = 200$ KHz and $M = 1000$ samples, with two virtual radios at $r_i = 100$ KHz and $m_i = 500$ samples, while one of the virtual radios manifests an effective rate shown as a percentage of the expected nominal rate.

(3) RAN slices that stopped operating but were not yet deallocated from the RANaaS platform.

To prevent individual virtual radios from affecting the processing performance of the entire system, we must isolate the processing of their streams of IQ samples and ensure that the appropriate number of M IQ samples is available to the radio hypervisor at every τ_M. We can achieve such isolation through dynamic padding of the streams of IQ samples, i.e., with the inclusion of a number of null samples ($v_{i,j}$) to the stream of the i-th virtual radio during W_j whenever $r_{\text{eff}_{i,j}} < r_i$ so as to provide a total of M samples to fill W_j:

$$\begin{aligned} v_{i,j} &= m_i - \lfloor r_{\text{eff}_{i,j}} \cdot \tau_M \rfloor \\ &= m_i - \left\lfloor \frac{m_i \cdot r_{\text{eff}_{i,j}}}{r_i} \right\rfloor \\ &= \left\lceil m_i \cdot \frac{r_i - r_{\text{eff}_{i,j}}}{r_i} \right\rceil . \end{aligned} \tag{2}$$

The dynamic padding limits the effect of a virtual radio presenting a $r_{\text{eff}_{i,j}} < r_i$ to its own RAN slice, without delaying the production/consumption of multiplex windows, as shown in Figure 6a. If such padding was not added, then the radio hypervisor would need to wait until a full window of M IQ samples was available, as shown in Figure 6b. This implies that the radio hypervisor would not be able to ensure independence between virtual radios in terms of the processing of their streams of IQ samples, i.e., the virtual radios would not be isolated at the radio processing level.

The impact of this delay would accumulate over successive windows, deteriorating the performance experienced by all virtual radios over time. Based on the number of IQ samples collected for generating M_j during a τ_M (C_j), the multiplex window index (j), and f_s, we can calculate the experienced accumulated delay (τ_{exp}) for producing/consuming the given window:

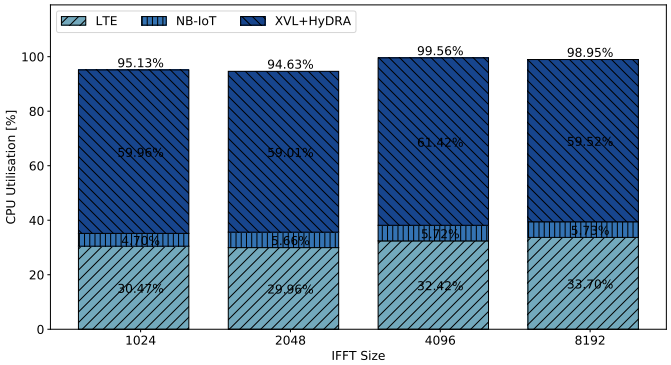


Fig. 8: CPU utilisation when running XVL alongside HyDRA, in addition to an LTE and an NB-IoT virtual radios, under different radio hypervisor configurations.

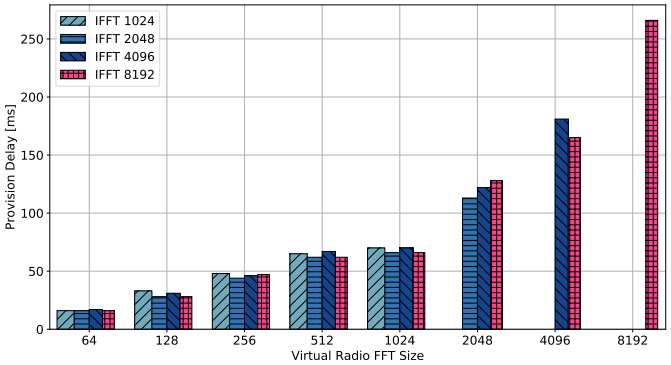


Fig. 9: The provisioning delay to fulfil a virtual radio request. The time taken for creating virtual radios is proportional to the number of allocated FFT bins, but not to the overall IFFT size of the radio hypervisor.

$$\tau_{\text{exp}} = \sum_{k=1}^{j} \frac{k}{f_s} \cdot (M - C_k)\,. \tag{3}$$

The τ_{exp} expresses the total delay experienced by each tenant during the operation of their RAN slices if any of the virtual radios present a $r_{\text{eff}_{i,j}} < r_i$. In Figure 7, we illustrate the impact of τ_{exp}. We show a numerical analysis in which we consider a constant $r_{\text{eff}_{i,j}}$ for every window. It demonstrates how τ_{exp} mounts over successive windows and would degrade the operation of the RANaaS platform over time, if no padding were applied.

V. Experimental Evaluation

In this section, we validate the use of XVL for enabling RANaaS. First, we describe the integration between XVL and HyDRA, forming a RANaaS platform capable of deploying heterogeneous RAN slices as a service. Then, we assess the performance of our RANaaS platform regarding the computational overhead, the delay in provisioning virtual radios, the delay introduced by radio virtualisation, and the signal degradation due to the virtualisation mechanism.

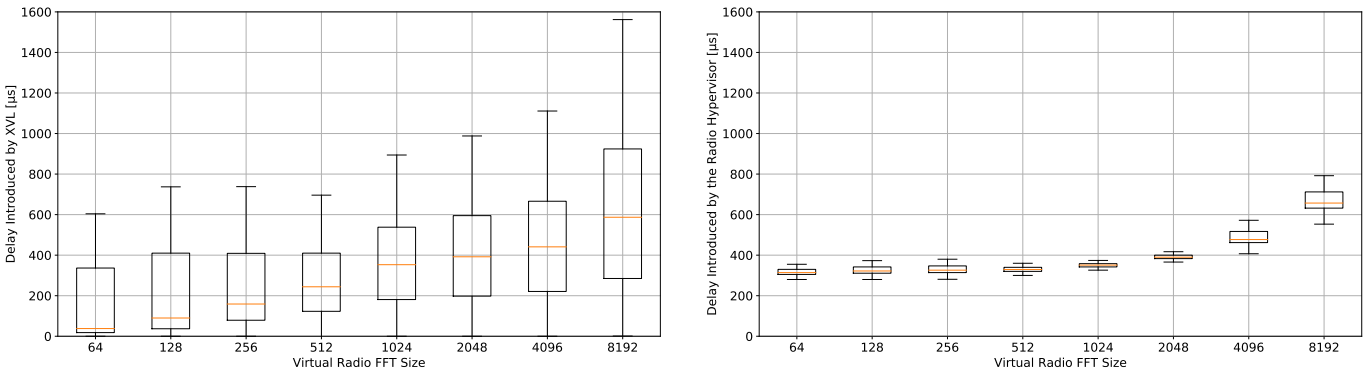
A. Integration with HyDRA and GNU Radio

We integrated XVL with HyDRA [12] for validating XVL's operation and showcasing its features. This integration enables HyDRA to support RANaaS, through the provision of tailored RAN slices on demand. Also, we developed a client-side library for communicating with XVL. This library allows any C++-based client to interface with XVL for querying, requesting, and using XVL's resources. We employ ZMQ, a cross-platform networking library, which facilitates porting our communication library to serve clients implemented in different programming languages, e.g., Python, Java, or Ruby.

In order to facilitate the use of our contribution by the research community, we have also integrated XVL with GNU Radio, a widely known Software Development Kit (SDK) for prototyping and developing on Software-defined Radio (SDR) platforms. GNU Radio realises RATs and signal-processing systems as acyclic directional graphs, known as flowgraphs, which represent continuous streams of IQ samples between signal processing blocks, e.g., modulation, coding, and filtering. Each flowgraph possesses one or more source blocks for inserting IQ samples into the flowgraph, and one or more sink blocks for exporting IQ samples from the flowgraph. We used the ZMQ library for developing GNU Radio blocks that can seamlessly replace USRP source and sink blocks in existing GNU Radio flowgraphs. These new blocks only require the extra information of the IP address and port number of where an XVL instance is running, as well as a client ID. The benefits of this setup are twofold: it allows us to use an existing radio hypervisor and GNU Radio flowgraphs for evaluating XVL, and it enables current GNU Radio flowgraphs to leverage XVL for instantiating virtual radios.

B. Computational Overhead

In this analysis, we consider the computational overhead introduced by running XVL alongside a radio hypervisor to serve an LTE and an NB-IoT virtual radios. XVL and the radio hypervisor introduce a new layer of complexity between the software radios and the RF front-end. We measured the CPU utilisation for an XVL and HyDRA instance for different IFFT sizes and normalised it as a percentage of one core in an Intel Xeon E52620 v2 processor. Figure 8 shows the results of our measurements. Independently of the configuration used in the radio hypervisor, the combination of XVL and HyDRA, the NB-IoT, and the LTE, require roughly 60%, 5% and 30% of the CPU processing power, respectively. The constant CPU utilisation by the virtual radios is expected, as their processing is independent of the radio hypervisor. However, we did not capture any increase in the CPU overhead for the radio hypervisor as reported in [12]. We attribute this to the multithreading that XVL employs for managing the timed buffers and sockets, decreasing the significance of the CPU utilisation of the radio hypervisor. These results indicate that XVL can run alongside a radio hypervisor on commodity hardware for deploying multiple virtual radios.



(a) The service delay introduced by XVL. The service delay increases with the number of resources, due to the longer time required to allocate larger windows in memory.

(b) The service delay introduced by HyDRA. The service delay increases with the number of resources, due to the increasing complexity of the FFT/IFFT operations.

Fig. 10: Delay measurements for different virtual radio configurations. All the measurements were made using a real bandwidth of 2MHz and an overall FFT size of 8192 bins, which corresponds to the worst case scenario in Section V-C.

		LTE			NB-IoT		
	Granularity (Δf)	FFT Size (n_i)	Bandwidth ($\tilde{b_i}$)	MSE	FFT Size (n_i)	Bandwidth ($\tilde{b_i}$)	MSE
Baremetal	-	-	1.400,0 KHz	2.1 dB	-	200,0 KHz	1.1 dB
IFFT 8192	977 Hz	1433	1.400,2 KHz	2.4 dB	205	200,2 KHz	1.1 dB
IFFT 4096	1953 Hz	717	1.400,3 KHz	3.4 dB	103	201,2 KHz	2.2 dB
IFFT 2048	3906 Hz	359	1.402,3 KHz	6.1 dB	52	203,3 KHz	4.1 dB
IFFT 1024	7812 Hz	180	1.406,2 KHz	11.2 dB	27	210,9 KHz	7.9 dB

TABLE I: A comparison between the MSE for over-the-air transmissions using the RATs discussed in Section V-B, running both on baremetal and on top of HyDRA with a fixed B of 8 MHz and different IFFT sizes.

C. Provisioning Delay

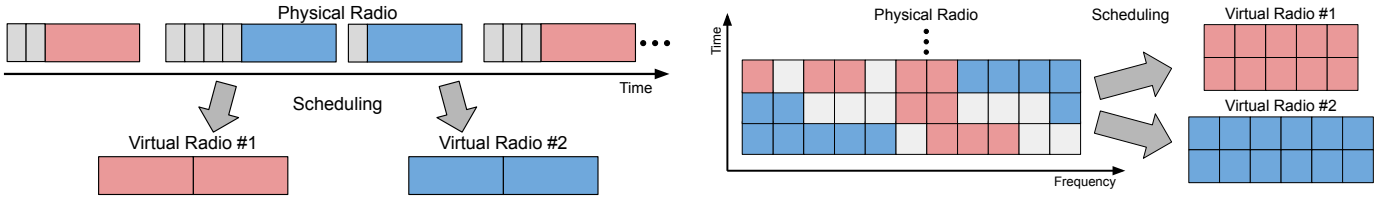
In this analysis, we are interested in the delay for provisioning virtual radios with XVL. The provisioning delay is crucial for planning and evaluating a RAN deployment using the RANaaS paradigm, as it dictates the time interval required to provision the virtual radios that realise the RAN. This interval comprehends the time between the client sending a resource request, and XVL returning the resource allocation confirmation (there are several procedures that occur between these two events, as we have seen in Section III). Figure 9 shows the results of our measurements for different configurations of the virtual radio and radio hypervisor. The exponential growth in the semi-log scale denotes a linear behaviour in relation to the number of resources allocated per virtual radio, regardless of the FFT size of the radio hypervisor. We attribute the linear behaviour to the time for allocating memory for XVL's buffers. The worst case scenario, where a virtual radio required 8192 FFT bins, took less than 300 ms to be fulfilled, hence, indicating that XVL is fast enough to provision virtual radios on the fly.

D. Service Delay

In this analysis, we are interested in the delay that XVL and HyDRA introduce to serve virtual radios and forward their IQ samples. The service delay is crucial to analyse the impact of relying on virtualised infrastructure (instead of real infrastructure), and to evaluate whether it can impair the operation of the virtual radios. First, we measured the time spent by XVL for delivering the IQ samples to the radio hypervisor, i.e., from successful UDP receptions, through the timed buffers, and generation of a multiplex window. Figure 10a shows the results of our measurements, with a median of 343 μs. Then, we measured the time spent by HyDRA to virtualise the RF front-end, i.e., consume the window, multiplex the IQ samples of all virtual RF front-ends, and forward the IQ samples to the real RF front-end. Figure 10b shows the results of our measurements, with a median of 334 μs. These results show that XVL introduces a service delay within the same order of magnitude of the radio hypervisor. However, the overall median introduced delay by using XVL and HyDRA for supporting RANaaS falls under 1ms. A 1ms delay can be impactful in the delay budget for Centralised-RAN (C-RAN) scenarios, and must be taken into account in the network planning.

E. Signal Degradation

In this analysis, we consider how the resource allocation granularity of the radio hypervisor affects on the performance of RAN slices. FFT-based radio hypervisors, e.g., HyDRA, employ IFFTs to partition the bandwidth of a physical radio (B) into a number of FFT bins equal to the IFFT size (n), each with a bin width $\Delta f = B/n$. The ith virtual radio is provided with an integer number of FFT bins (n_i) of total bandwidth ($\tilde{b_i}$) that approximates its requested bandwidth (b_i), according to $n_i = \lceil b_i/\Delta f \rceil$. Therefore, for a fixed B, the IFFT size dictates the resource allocation granularity of the radio hypervisor, i.e., the minimum amount of radio resources which the radio hypervisor can allocate to a virtual radio. We



(a) Frame-based radio hypervisors schedule the frames of a physical radio, e.g., WiFi or Bluetooth, to the different virtual radios.

(b) Grid-based radio hypervisors schedule the PRBs of a physical radio, e.g., LTE or WiMax, to the different virtual radios.

Fig. 11: Different types of technology-dependent radio hypervisors and their virtualisation mechanisms. This class of radio hypervisors operates at the MAC layer, scheduling the available real radio resources to the virtual radios.

measured how different IFFT sizes, and consequently, resource allocation granularities, affect the performance of the virtual radios in terms of the Mean Square Error (MSE), and Table I shows the result of our evaluation. These results show how the performance impact of radio virtualisation on different RATs vary with the resource allocation granularity of the radio hypervisor. Moreover, at an IFFT size of 8192, we were able to achieve a performance comparable to RATs running on baremetal, i.e., without radio virtualisation.

VI. Related Works

There is a vast literature on radio virtualisation, radio slicing and RANaaS [17] [20] [21]. However, the majority of the current research efforts in these areas have been theoretical, while the number of experimental and prototype radio virtualisation platforms pales in comparison [11]. Among the existing experimental research efforts, their proof-of-concept radio hypervisors may differ in terms of their approach to radio virtualisation. We can classify the existing radio hypervisors into two main categories: technology-specific and general-purpose radio hypervisors. In this section, we describe each of these classes, outline their differences, and list some of their notable examples in the literature. Then, we assess how well these radio hypervisors meet the requirements listed in Section II, and position our solution, the XVL, in relation to current radio hypervisors.

A. Technology-Specific Radio Hypervisors

Most of the existing radio hypervisors focus on the virtualisation of a single RAT, leveraging the resource allocation capabilities of the given RAT for creating virtual radios. This class of radio hypervisors typically operates at the MAC layer, virtualising physical radios by scheduling the use of their radio resources to different virtual radios. This scheduling-based virtualisation approach supports a single common PHY, shared among all virtual radios, only allowing the creation of virtual radios that use the same RAT as the underlying physical radio [11]. There are two subclasses of technology-dependent radio hypervisors, based on the type of MAC of the targeted RAT [17]: frame-based and grid-based radio hypervisors, illustrated in Figure 11. The former targets the virtualisation of RATs that transmit intermittent frames with random contention windows and allocate resources as a number of consecutive frames, e.g., WiFi and Bluetooth. These radio hypervisors operate scheduling frames and/or different contention window durations to different virtual radios, queueing the frames from all the virtual radios [22]–[24]. Grid-based radio hypervisors target the virtualisation of RATs that transmit continuously and allocate resources on a well-defined grid of resources blocks, e.g., LTE and WiMax. These radio hypervisors operate scheduling Virtual Resource Blocks (VRBs) to the virtual radios, which are non-overlapping subsets of the underlying PRBs [25]–[27].

B. General-Purpose Radio Hypervisors

There are few examples of radio hypervisors that employ technology-agnostic radio virtualisation mechanisms and support virtual radios with different RATs. This class of radio hypervisors operates at the PHY layer, virtualising the physical radios by multiplexing the use of their RF front-ends. This multiplexing-based virtualisation approach enables each virtual radio to have its own RAT, with independent PHY and MAC. However, general-purpose radio hypervisors require full control over the processing of IQ samples on the physical radio, and hence, can only operate on top of SDR platforms. There are two subclasses of general-purpose radio hypervisors, based on their multiplexing mechanism: FFT-based [12] [28] and filterbank-based [19] radio hypervisors. In both cases, the radio hypervisor partitions the real RF front-end in the frequency domain, creating virtual RF front-ends using non-overlapping spectrum bands. Then, the radio hypervisor provides each virtual radio its own virtual RF front-end, which the virtual radios can use to realise their RAT. However, the ability of general-purpose radio hypervisors to be technology agnostic comes at the price of reduced resource efficiency in comparison to technology-specific radio hypervisors, as guard bands and/or intervals are often necessary for ensuring isolation between the virtual radios [11].

C. State-of-the-Art on Radio Virtualisation for RANaaS

We have evaluated some notable examples of radio hypervisors in the literature with respect to the requirements for supporting RANaaS, as described in Section II, and Table II summarises the results of our analysis. The majority of research efforts on RAN slicing and radio virtualisation either focus only on particular technologies, or do not make their implementation available as open source [17].

One of the few works that are technology agnostic and provide an actual open-source implementation is HyDRA

Radio Hypervisor	Resource Negotiation	Dynamic Allocation	Technology Agnostic	Radio Resource Isolation	Radio Processing Isolation	Service Monitoring
Virtual WiFi [22]	–	✓	–	✓	–	–
CrowdLab [23]	–	–	–	✓	–	–
AMPHIBIA [24]	–	–	–	✓	–	–
FLEXRAN [25]	–	–	–	✓	–	–
NVS [26]	–	–	–	✓	–	–
Orion [27]	✓	✓	–	✓	✓	–
IQ Switch [19]	–	–	✓	✓	–	–
SVL [28]	–	–	✓	✓	–	–
HyDRA [12]	–	–	✓	✓	–	–
XVL [this work] + Radio Hypervisor	✓	✓	✓	✓	✓	✓

TABLE II: Qualitative evaluation of the resource management features necessary for radio hypervisors to support RANaaS.

[12], a radio hypervisor for SDRs developed on top of GNU Radio [29], a SDK that provides signal processing blocks for the realisation of RATs and signal-processing systems (detailed previously in Section V-A). HyDRA is an FFT-based general-purpose radio hypervisor, where each virtual radio can realise any RAT using its own virtual RF front-end. This radio hypervisor allows multiple RAN slices with heterogeneous RATs to share the same underlying physical radio hardware. However, its virtual radios are not isolated in terms of radio processing, given that if any of the virtual radio delays transmitting/receiving or crashes, it will halt the operation of the radio hypervisor and all other virtual radios. Moreover, the number of virtual radios and their resource allocation, i.e., the amount of spectrum for each virtual RF front-end, must be set up before the operation of the radio hypervisor. It is not possible to (de)allocate a virtual radio without interrupting the operation of the radio hypervisor.

Despite the capabilities of existing radio hypervisors for enabling multiple heterogeneous virtual radios to share the same underlying physical radio hardware, both HyDRA [12] and the majority of other radio hypervisors are missing essential features for supporting the RANaaS paradigm. Such limitation motivated us to develop XVL, a software layer that wraps around an existing radio hypervisor to provide the missing resource management functionality needed for RANaaS.

VII. Conclusion

In this paper, we addressed the resource management functionality and challenges for enabling RANaaS through radio virtualisation. We assessed the role of radio virtualisation in the context of RANaaS, and devised a comprehensive list of features necessary for radio hypervisors must incorporate for supporting RANaaS. Then, we evaluated the state-of-the-art on radio virtualisation with respect to these requirements, and identified what we consider as most suitable radio hypervisor for the development of a prototype RANaaS platform. Next, we investigated the challenges for ensuring isolation between virtual radios at the radio processing level and proposed dynamic padding to maintain multiplex synchronicity.

We presented XVL, a software layer that can be added on top of existing hypervisors and provides them with the missing capabilities for supporting the RANaaS paradigm, namely: (*i*) a radio resource broker for the negotiation of radios resources; (*ii*) timed buffers for dynamically padding the streams of IQ samples; (*iii*) and a monitoring interface for tenants to query the performance of their RAN slices. We evaluated XVL regarding its provisioning delay, computational overhead and service delay. Our results show that XVL introduces a delay comparable to a radio hypervisor, can run on commodity hardware and provision RAN slices in real-time. We are continuing to extend XVL and introduce new features, e.g., visualisation of the spectrum of the virtual and real RF front-ends, as well as provisioning of radio functionality, e.g., modulation and coding, on top of the virtual radios. In future works, we are investigating the impacts and trade-offs of using hypervisors on multiple network segments, e.g., RAN and CN, leveraging virtualisation for deploying NSs on physical E2E network infrastructures.

References

[1] 3rd Generation Partnership Project, "3GPP TR 22.891: Feasibility Study on New Services and Markets Technology Enablers," 3rd Generation Partnership Project, Tech. Rep., Sep. 2016.

[2] ——, "3GPP TR 28.801: Study on Management and Orchestration of Network Slicing for Next Generation Network," 3rd Generation Partnership Project, Tech. Rep., May 2017.

[3] 5G America, "5G Americas White Paper – Network Slicing for 5G and Beyond," 5G America, Tech. Rep., 2016.

[4] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network Slicing as a Service: Enabling Enterprises' Own Software-Defined Cellular Networks," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 146–153, 2016.

[5] 3rd Generation Partnership Project, "3GPP TR 38.801: Study on New Radio Access Technology: Radio Access Architecture and Interfaces," 3rd Generation Partnership Project, Tech. Rep., Mar. 2017.

[6] J. F. Santos, J. van de Belt, W. Liu, V. Kotzsch, G. Fettweis, I. Seskar, S. Pollin, I. Moerman, and L. A. DaSilva, "Orchestrating Next-Generation Services Through End-to-End Network Slicing," White Paper, The ORCA Consortium, Oct. 2018. [Online]. Available: https://orca-project.eu/wp-content/uploads/sites/4/2018/10/orchestrating_e2e_network_slices_Final.pdf

[7] T. Taleb, M. Corici, C. Parada, A. Jamakovic, S. Ruffino, G. Karagiannis, and T. Magedanz, "EASE: EPC as a Service to Ease Mobile Core Network Deployment Over Cloud," *IEEE Network*, vol. 29, no. 2, pp. 78–88, 2015.

[8] D. Sabella, P. Rost, Y. Sheng, E. Pateromichelakis, U. Salim, P. Guitton-Ouhamou, M. Di Girolamo, and G. Giuliani, "RAN as a Service: Challenges of Designing a Flexible RAN Architecture in a Cloud-Based Heterogeneous Mobile Network," *IEEE Future Network & Mobile Summit*, 2013.

[9] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "FlowVisor: a Network Virtualization Layer," *OpenFlow Switch Consortium, Tech. Rep*, vol. 1, p. 132, 2009.

[10] A. Al-Shabibi, M. De Leenheer, M. Gerola, A. Koshibe, W. Snow, and G. Parulkar, "OpenVirteX: A Network Hypervisor," in *Open Networking Summit (ONS)*, 2014.

[11] J. van de Belt, L. Doyle *et al.*, "Defining and Surveying Wireless Link and Network Virtualization," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1603–1627, 2017.

[12] M. Kist, J. Rochol, L. A. Dasilva, and C. B. Both, "SDR Virtualization in Future Mobile Networks : Enabling Multi-Programmable Air-Interfaces," in *IEEE Internationl Conference on Communications (ICC)*, May 2018.

[13] J. F. Santos, M. Kist, J. van de Belt, J. Rochol, and L. DaSilva, "Towards Enabling RAN as a Service - The Extensible Virtualisation Layer," in *IEEE International Conference on Communications (ICC)*, May 2019.

[14] T. K. Forde, I. Macaluso, and L. E. Doyle, "Exclusive Sharing & Virtualization of the Cellular Network," in *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2011, pp. 337–348.

[15] J. Sachs and S. Baucke, "Virtual Radio: A Framework for Configurable Radio Networks," in *ACM Conference on Wireless Internet (WICON)*, 2008, pp. 61:1–61:7.

[16] C. Liang and F. R. Yu, "Wireless Network Virtualization: A Survey, Some Research Issues And Challenges," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 358–380, 2015.

[17] M. Richart, J. Baliosian, J. Serrat, and J.-L. Gorricho, "Resource Slicing in Virtual Wireless Networks: A Survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 462–476, 2016.

[18] H. Wen, P. K. Tiwary, and T. Le-Ngoc, "Current trends and Perspectives in Wireless Virtualization," in *IEEE International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, 2013, pp. 62–67.

[19] J. Mendes, X. Jiao, A. Garcia-Saavedra, F. Huici, and I. Moerman, "Cellular Access Multi-tenancy Through Small-cell Virtualization and Common RF Front-end Sharing," *Elsevier Computer Communications*, vol. 133, pp. 59–66, 2019.

[20] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80–87, 2017.

[21] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network Slicing & Softwarization: a Survey on Principles, Enabling Technologies & Solutions," *IEEE Communications Surveys & Tutorials*, 2018.

[22] L. Xia, S. Kumar, X. Yang, P. Gopalakrishnan, Y. Liu, S. Schoenberg, and X. Guo, "Virtual WiFi: Bring Virtualization from Wired to Wireless," in *ACM SIGPLAN Notices*, vol. 46, no. 7, 2011, pp. 181–192.

[23] E. Cuervo, P. Gilbert, B. Wu, and L. P. Cox, "CrowdLab: An Architecture for Volunteer Mobile Testbeds," in *IEEE Internaltional Conference on Communication Systems and Networks (COMSNETS)*, 2011, pp. 1–10.

[24] K. Nakauchi, K. Ishizu, H. Murakami, Y. Kobari, Y. Nishida, A. Nakao, and H. Harada, "Virtual Cognitive Base Station: Enhancing Software-based Virtual Router Architecture With Cognitive Radio," in *IEEE International Conference on Communications (ICC)*, 2012, pp. 2827–2832.

[25] S. S. Kumar, R. Knopp, N. Nikaein, D. Mishra, B. R. Tamma, A. A. Franklin, K. Kuchi, and R. Gupta, "FLEXCRAN: Cloud Radio Access Network Prototype Using OpenAirInterface," in *International Conference on Communication Systems and Networks (COMSNETS)*, 2017, pp. 421–422.

[26] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "NVS: A Substrate for Virtualizing Wireless Resources in Cellular Networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1333–1346, 2012.

[27] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: RAN Slicing for a Flexible and Cost-Effective Multi-Service Mobile Network Architecture," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2017, pp. 127–140.

[28] K. Tan, H. Shen, J. Zhang, and Y. Zhang, "Enable Flexible Spectrum Access with Spectrum Virtualization," in *IEEE Dynamic Spectrum Access Networks (DYSPAN)*, 2012, pp. 47–58.

[29] E. Blossom, "GNU Radio: Tools for Exploring the Radio Frequency Spectrum," *Linux journal*, vol. 2004, no. 122, p. 4, 2004.

Joao F. Santos is pursuing a PhD on wireless networks at the CONNECT Telecommunications Research Centre, headquartered at Trinity College Dublin. He holds a B.Sc. in Telecommunications Engineering from Universidade Federal Fluminense (2016). He worked at Rede Nacional de Ensino e Pesquisa (RNP) as the main developer of the Clearinghouse of the FIBRE testbed federation. His research interests include network slicing, radio virtualisation, and end-to-end network orchestration.

Maicon Kist is a PhD student in Computer Science at the Institute of Informatics (INF) of the Federal University of Rio Grande do Sul (UFRGS), Brazil. He received his M.Sc. title in from INF of UFRGS (2012), Brazil. In addition, he holds a B.Sc. in Computer Science from the University of Santa Cruz do Sul (2008), Brazil. His research involves Wireless Networks, Next Generation Networks, Software Defined Radio, and Cognitive Radio Networks.

Juergen Rochol is an associate professor at the Institute of Informatics of the Federal University of Rio Grande do Sul (UFRGS), Brazil. He received his M.Sc. degree in Physics, and his Ph.D. degree in Computer Science, both from UFRGS in 1972 and 2001, respectively. His research interests include wireless networks, next generation networks, optical networks and traffic control on broadband computer networks.

Luiz A. DaSilva is the Executive Director of the Commonwealth Cyber Initiative (CCI) and the Bradley Professor of Cybersecurity at Virginia Tech. Previously, he held the chair of Telecommunications at Trinity College Dublin, where he served as the Director of CONNECT, the Science Foundation Ireland Research Centre for Future Networks and Communications. His research focuses on distributed and adaptive resource management in wireless networks, and in particular radio resource sharing, dynamic spectrum access, and the application of game theory and machine learning to wireless networks. Prof. DaSilva is an IEEE Fellow and a Fellow of Trinity College Dublin.