Poster: IsoRAN: Isolation and Scaling for 5G RAN via User-Level Data Plane Virtualization

Nishant Budhdev, Mun Choon Chan, Tulika Mitra
School of Computing, National University of Singapore
{nishant,chanmc,tulika}@comp.nus.edu.sg

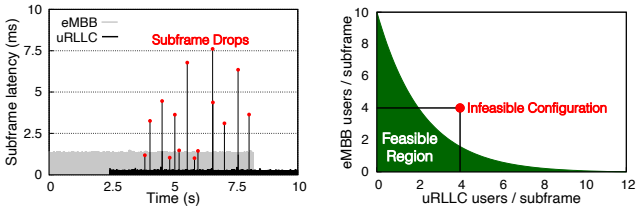
***Abstract*—5G presents a unique set of challenges for cellular network architecture. The architecture needs to be versatile in order to handle a variety of use cases. In this paper we propose IsoRAN, a design for 5G Cloud RAN (CRAN) which provides isolation and scaling along with the flexibility needed for 5G architecture through fine-grained slicing. Our design allows users with diverse service requirements to be executed on the most efficient hardware, to ensure that the Service Level Agreements (SLA) are met while minimizing power consumption. Our experiments show that IsoRAN handles users with different SLA while providing isolation to reduce interference. This increased isolation reduces the drop rate for different users from 42% to nearly 0% in some cases while requiring only half the amount of resources.**

I. Introduction

Beyond the incremental increase in demand for capacity, 5G will also need to cope with a wide range of unique use cases. These range from ultra-Reliable Low Latency Communication (uRLLC) to power-efficient massive Machine-Type Communication (mMTC), alongside traditional high-speed enhanced Mobile Broadband (eMBB) communication [1] [2]. Supporting such a diverse set of use cases requires the 5G Radio Access Network (RAN) to provide flexible processing, efficient scaling, and increased isolation.

In recent years two key technologies have been introduced to achieve these goals of flexibility, scalability and isolation. The first one is Network Function Virtualization, which has allowed the movement of RAN processing, from hardware based implementation to software processes running on general-purpose architectures. Different use cases can now be supported as multiple processes without modifying the hardware; thereby enabling a more flexible RAN. The second key technology is cloud RAN (CRAN), which introduces the ability to scale by multiplexing resources in the cloud [3]. Furthermore, a combination of both virtualization and CRAN is used to enable multiple virtual networks on shared physical infrastructure. These virtual networks known as slices, provide isolation between different use cases running in the CRAN [4].

However, scaling and isolation are heavily dependent on the level of virtualization, which is severely limited due to the monolithic nature of RAN processing. Consequently, users attached to the same base station need to be processed on a single host irrespective of their use cases. To demonstrate the impact of this drawback on isolation, we run a simple experiment on Intel-Xeon server-class machine with two use cases eMBB and uRLLC, with 4 users each. To ensure maximum isolation, we run each slice as a separate application executing on different sets of cores. To simulate L1 processing, we use the LTE PHY benchmark [5]. We plot the processing latency for each of these use cases in Figure 1(a). We see that uRLLC users see a significant amount of deadline misses, i.e. subframe drops, in the presence of eMBB users, even when the eMBB users are running on different set of cores. This implies that critical applications, such as uRLLC, can be significantly impacted even when running on a highly reliable host with sufficient processing capacity. Figure 1(b) shows the significantly reduced feasibility region when two different criticality slices are running on the same host. This highlights the need for better isolation between slices, which has also been documented in the 3GPP standards [6].



(a) Infeasible configuration (b) Feasible Region for for Xeon-2155

Figure 1. Feasibility study for eMBB and uRLLC slices on Xeon-2155

The inability to distribute a base station's processing across multiple hosts also implies the need for constant hardware upgrades to support the ever-increasing data rate and number of devices attached to a single base station. This makes scaling inefficient and expensive using traditional CRAN design. In recent years, a number of works have tried to tackle this problem. As illustrated in Figure 2, there are two types of solutions. Approaches such as Orion [7] and FlexCRAN [8] execute each slice on a different virtual machine. However, users from the same base station are processed on the same host as they have common L1/L2 processing. On the other hand, POSENS [9] introduces independent L1/L2 processing to improve isolation between slices. This approach however limits scaling within a slice as users within a slice are limited to a common host.

To address the above limitations, we present *IsoRAN*, a design for 5G CRAN that provides the desired isolation, and scalability with lower cost. The key idea of IsoRAN is the use of user-level data plane virtualization. In IsoRAN, users attached to the same base station are not restricted to a single

Annex to ISBN 978-3-903176-28-7

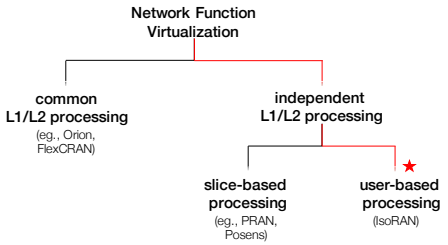


Figure 2. Design space for NFV in RAN architecture

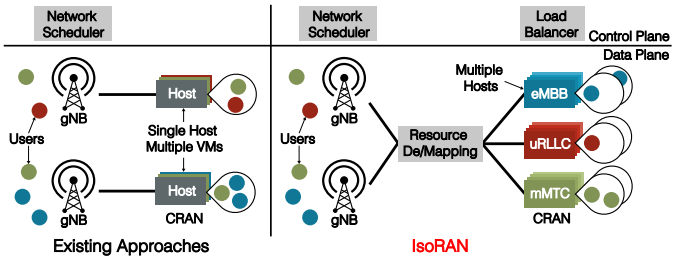


Figure 3. Comparing existing CRAN architecture with IsoRAN

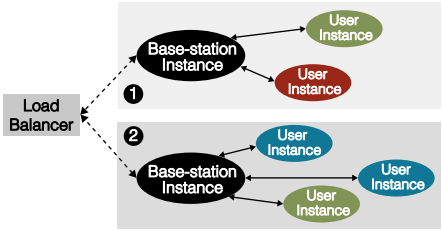


Figure 4. IsoRAN Thread Topology

host and can now instead be placed anywhere in the cloud. Furthermore, users with the same SLA attached to different base stations, can also be grouped together under a large slice, so as to minimize the effect of adverse interference and reduce network orchestration complexity.

II. System

This section provides an overview of the system architecture, followed by a description of the main components and the overall workflow in IsoRAN.

A. Architecture Overview:

The key factors driving the need for IsoRAN are to increase isolation between slices and improve scalability. IsoRAN divides processing into user level and base-station level functions, which are processed in separate threads. New users can be allocated to threads on any host in the CRAN as shown in Figure 3. This ensures fine-grained load balancing to achieve better isolation, and greater savings from higher multiplexing. This design closely resembles the architecture of real-time stream processing engines, such as Apache Storm and Twitter Heron [10] [11]. Additionally, users with similar SLA can also be placed on the same host to minimize interference.

After a user is allocated, the load balancer also informs the corresponding base station thread about the IP address of the allocated host. The base station thread requires this information for both control and data plane communication with user. When a user leaves the network, the base station thread removes it from the list of users, and removes its data from the allocated host by informing the load balancer. For handovers, the IP address of the user's host is removed from the source base station's list, and added to the target base station's list. Next, the target base station thread sends its configuration data to the user thread. This simplified handover procedure is crucial for high-speed mobility scenarios such as high-speed railways. Figure 4 shows IsoRAN's thread architecture and their interaction.

B. Main Components

User-level Instance Each user-level instance consists of two threads responsible for a user's uplink and downlink baseband processing processing, respectively. Uplink threads are responsible for converting analog signals to data packets, and vice-versa for downlink threads. There are a number of user-level instances running on each host. These instance are shared among all users allocated to the same host. To reduce the overhead of creating these instance on demand, user-level instance are run as daemon processes.

Base-station level Instance Base-station level instances perform multiple functions such as Resource De/Mapping, scheduling, paging, handover etc. Each base-station level instance consists of three threads: uplink, downlink, and management. Uplink thread is responsible for managing Random Access calls and receiving ACK/NACK requests. Downlink thread is responsible for paging, broadcast, distributed network scheduling, ACK/NACK transmissions etc. Management thread handles communication with user-level instances and maintains user-to-host mapping for all users attached to the base station. Each base station instance is associated with exactly one base station. This is because base station processing is continuous unlike user processing which is intermittent.

Base station threads are also responsible for Resource De/Mapping of digitized analog data transmitted to/received from the base station. Resource De/Mapping is used to aggregate/separate digitized analog data for different users and channels (see Figure 3). This distribution of processing between the frontend and the CRAN is called the NGFI:IF4 [12] functional split.

Load Balancer Allocating users to hosts is an important component of IsoRAN, as migrating users is risky and can affect end-user QoS. Sub-optimal allocation negatively impacts both the particular user and other users on the same host. The user allocation algorithm is automatically triggered at the end of each Random Access procedure, after the user has been authenticated. For each slice, the load balancer maintains a list of potential hosts, based on their ability to support the slice's SLA. It also maintains metrics such as the host's CPU utilization, network usage, memory consumption etc., which are then used to find an optimal host for a new user. The load balancer in IsoRAN also ensures critical slice users are not put together with non-critical slice users as it can severely affect the performance of the former (see Figure 1(a)). If the load balancer does not find an appropriate host, it requests for

more hosts since we are running in a cloud environment which allows dynamic resource requests.

III. Evaluation

We have evaluated the feasibility of our architecture design through a proof-of-concept implementation using OpenAir-Interface [13]. However, while OpenAirInterface allows us to validate IsoRAN, it does not support large-scale testing. Therefore to evaluate the scalability we use a trace simulation, using network schedules captured from real LTE networks.

To capture these network traces we use IMDEA's Online Watcher for LTE [14], which captures the network schedule broadcast by the base station. The schedule contains user scheduling information such as resource block allocation, modulation scheme and coding rate for each subframe. We capture 10 minute traces from 4 adjacent base stations over an extended period of 15 days. Overall, our traces have over 1.2 million Radio Network Temporary Identifiers with a total duration of 5 hours. We simulate 30 base stations, with each individual trace used to simulate a base station; giving us on average nearly 20,000 users in the CRAN at any time. To simulate the L1 processing of each base station we use the PHY benchmark [5].

In the simulation, for the purpose of slicing, we classify users into 3 different slices based on their traffic patterns. uRLLC slice users are identified as users with medium constant-bit traffic with at most 8 Resource Blocks per frame. Users with Transport Block Size less than 1000 bits for all subframes are classified as mMTC, based on the Transport Block Size table for NB-IoT communication provided in the 3GPP specs [15]. The rest of the users are classified as eMBB users.

We compare against a baseline that executes all users connected to a base station on a single host as existing RAN architectures do not facilitate spilling users from the same base station to different hosts. To meet the worst-case demands for RAN processing, we ensure each base station is assigned to a unique host. For IsoRAN we separate users from different base stations and allocate the hosts using a simple round-robin algorithm based on the arrival of new users. We perform all experiments on Intel Xeon-2155 hosts.

We calculate the percent of subframes whose processing exceeds the subframe latency threshold. Figure 5 shows the drop rates for different slices for the three cases. For eMBB, the drop rate was reduced by 25% for IsoRAN. The reduction in IsoRAN comes largely due to the absence of users from other slices (uRLLC and mMTC) which do not compete for resource with eMBB users. The main advantage is in the reduction of drop rate for uRLLC users from over 40% to nearly 0%. mMTC users also see significantly lower drop rate due to the availability of a dedicated set of hosts.

Note that, the improved performance in IsoRAN is achieved by using nearly 2x lesser hosts as compared to the baseline. This reduction is attributed to the ability to dynamically allocate users in IsoRAN. In the baseline, since each base station is assigned to a different host to meet the worst-case

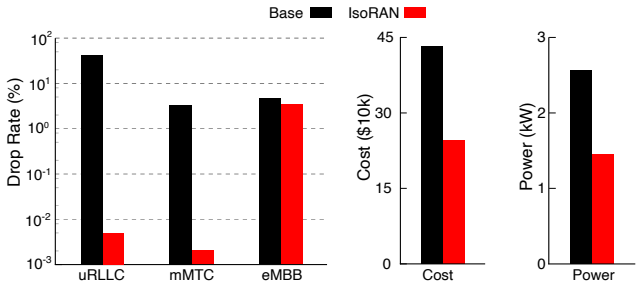


Figure 5. Comparison of drop rate, cost, and power between baseline and IsoRAN for large-scale simulation

demands, it also consumes more power as we cannot multiplex resources efficiently during low traffic. This is also reflected in the 43% lower power consumption for IsoRAN.

IV. Conclusion

In this paper we show how existing slicing proposals do not address isolation and scaling sufficiently, thereby adversely affecting the performance of critical slices. To avoid such interference we propose IsoRAN that virtualizes user-plane RAN processing, to enable per-user orchestration. We propose a distributed processing framework to allow RAN processing to scale across multiple hosts. A large-scale simulation using real-world traces demonstrates how IsoRAN can reduce total costs while meeting user requirements.

Acknowledgment: This work was supported by the Singapore Ministry of Education tier 1 grant R-252-000-B08-114.

References

[1] ITU. Setting the Scene for 5G:Opportunities & Challenges, 2018.
[2] 3GPP TR 38.913. Study on Scenarios and Requirements for Next Generation Access Technologies, 2017.
[3] C-RAN - Road Towards Green Radio Access Network. Centralized baseband, Collaborative radio, and real-time Cloud computing RAN. Presentation, 2010.
[4] NGMN Alliance. 5G: Next generation mobile networks. *White paper*, 2015.
[5] M. Själander et. al. An LTE uplink receiver PHY benchmark and subframe-based power management. In *IEEE ISPASS*, 2012.
[6] 3GPP TR 38.801. Study on new radio access technology: Radio access architecture and interfaces, 2017.
[7] X. Foukas et. al. Orion: Ran slicing for a flexible and cost-effective multi-service mobile network architecture. In *MOBICOM*. ACM, 2017.
[8] C. Chang et. al. FlexCRAN: A flexible functional split framework over ethernet fronthaul in Cloud-RAN. In *IEEE ICC*, 2017.
[9] G. Garcia-Aviles et. al. Posens: A practical open source solution for end-to-end network slicing. *IEEE Wireless Communications*, 2018.
[10] Apache Storm. https://storm.apache.org/.
[11] S. Kulkarni et. al. Twitter heron: Stream processing at scale. In *ACM SIGMOD*, 2015.
[12] Y Zhiling et al. White paper of next generation fronthaul interface v1. 0. *China Mobile Research Institute, Tech. Rep*, 2015.
[13] Navid Nikaein, Mahesh K Marina, Saravana Manickam, Alex Dawson, Raymond Knopp, and Christian Bonnet. Openairinterface: A flexible platform for 5g research. *ACM SIGCOMM Computer Communication Review*, 2014.
[14] N. Bui et. al. OWL: a Reliable Online Watcher for LTE control channel measurements. In *ACM ATC*, 2016.
[15] 3GPP TS 36.213 v13.3.0. E-UTRA Physical Layer Procedures, 2016.