SliceHO: Dynamic Inter-Slice Handover for Intelligent and Programmable 5G and Beyond RANs

Setareh Alagheh Band*, Noura Limam*, and Raouf Boutaba*
*University of Waterloo, Waterloo, Canada
Email: {salagheh, noura.limam, rboutaba}@uwaterloo.ca

***Abstract*—Radio Access Network (RAN) slicing enables multiple services to share the same physical infrastructure while maintaining differentiated performance guarantees. As networks evolve toward 6G, dynamic user equipment (UE)–to–slice mapping becomes crucial to support user-centric and personalized services. Although 3GPP specifications allow slice switching, current mechanisms typically require establishing a new PDU session, introducing substantial delays that render them unsuitable for highly dynamic RAN environments. Leveraging the intelligence and openness of the O-RAN architecture, this paper advocates for performing inter-slice handover directly within the RAN to achieve near-real-time responsiveness. We propose an xApp-based solution deployed in the Near-Real-Time RAN Intelligent Controller (Near-RT RIC) to enable dynamic, RAN-level inter-slice handovers without involving the core network. This design achieves fast and flexible UE-to-slice reassignment through localized control-loop execution. Extensive evaluations across diverse use-case scenarios demonstrate that the proposed solution achieves low-latency inter-slice handovers with a control-loop responsiveness of approximately 650 μs. These results position xApp-based inter-slice handover as a promising enabler for agile and adaptive 5G and beyond RANs.**

***Index Terms*—O-RAN, 5G and Beyond mobile networks, RAN slicing, inter-slice handover, RAN programmability**

I. Introduction

Network slicing enables multiple services to coexist over a shared physical infrastructure while maintaining logical isolation and differentiated resource allocation across the core, transport, and RAN domains. This capability is central to meeting the diverse performance and reliability requirements of 5G and beyond networks [1].

Within this context, *RAN slicing*, realized by assigning distinct scheduling priorities and physical resource blocks (PRBs) to slices (Figure 1), plays a pivotal role in managing radio resources and enforcing slice-specific Quality of Service (QoS) guarantees. It allows heterogeneous service categories, such as enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communication (URLLC), and massive Machine-Type Communication (mMTC), to operate concurrently and efficiently over shared radio resources. In current 3rd Generation Partnership Project (3GPP) procedures, changes to UE–slice association rely on UE- or AMF-triggered core-network signaling and PDU session re-establishment, which makes slice association effectively oblivious to changing RAN conditions. As mobile networks evolve toward user-centric, experience-driven design paradigms, static mapping between UEs and slices remains a major limitation [2]. To preserve an optimal Quality of Experience (QoE), the network must dynamically hand over users to the most appropriate slice. *Inter-slice handover* enables such dynamic reassignment by seamlessly moving UEs between slices, thereby ensuring service continuity and optimal resource utilization.

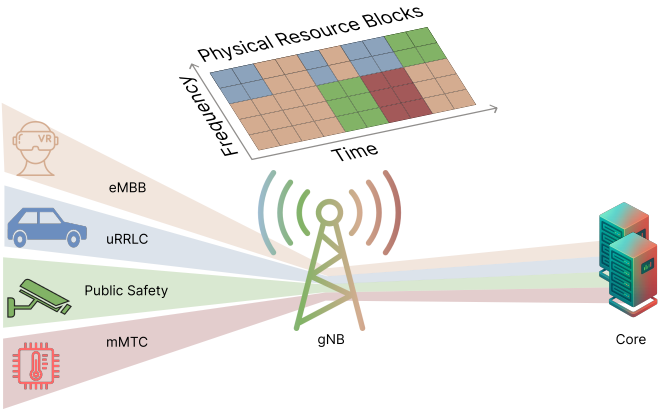
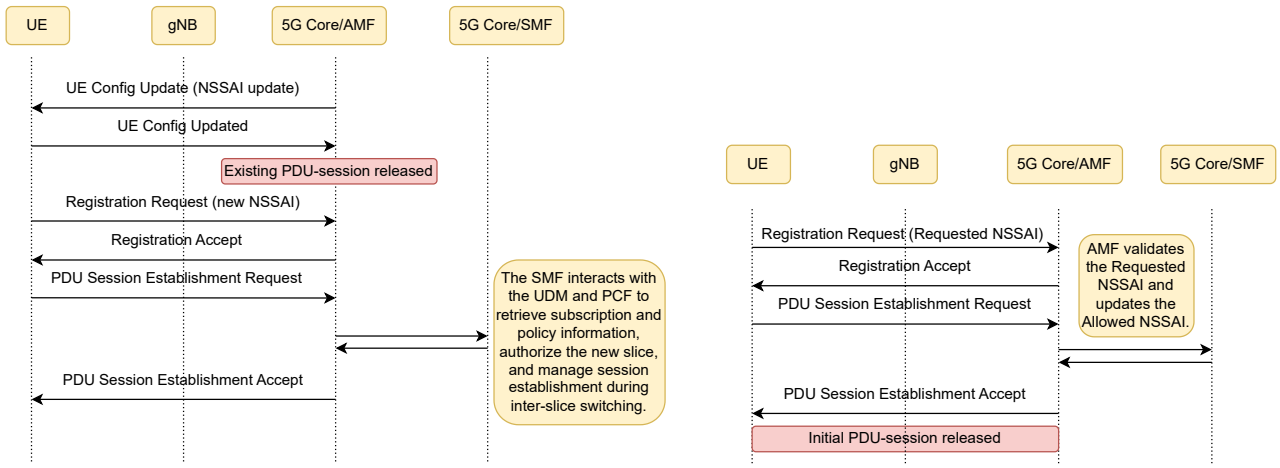


Fig. 1: RAN slicing in 5G and beyond, where services such as eMBB, URLLC, mMTC, and Public Safety share the same physical infrastructure through logical isolation and allocation of PRBs.

While the 3GPP defines *slice switching* through core network procedures, such as Network Slice Selection Assistance Information (NSSAI) updates and Protocol Data Unit (PDU) session re-establishment, the standard framework lacks true inter-slice handover support [1]. Existing mechanisms introduce substantial signaling overhead, depend on UE involvement, and lack real-time awareness of RAN conditions. Moreover, because slice switching requires releasing and re-establishing PDU sessions, it causes a temporary service interruption; hence, the term *slice switching* rather than *handover*. Consequently, current slice mobility mechanisms are slow, reactive, and ill-suited to latency-sensitive or context-dependent RAN environments. These limitations highlight the need for distributed, RAN-aware intelligence capable of making slice decisions locally and autonomously.

The O-RAN architecture [3] offers a promising foundation to address these challenges by introducing multi-layer RAN



(a) AMF-triggered inter-slice switching.

(b) UE-triggered inter-slice switching.

Fig. 2: Signaling workflow for 3GPP inter-slice switching.

intelligence through the Non-Real-Time (Non-RT) RIC and Near-RT RIC. These controllers host external applications, a.k.a rApps and xApps, that enable closed-loop, data-driven control. The Non-RT RIC operates on timescales above 1 second, performing long-term policy management, analytics, and AI model training via rApps, whereas the Near-RT RIC operates at shorter timescales (10 ms–1 s) using xApps for near-real-time optimization based on RAN metrics. Together, they establish a flexible, vendor-neutral framework for intelligent, adaptive, and context-aware RAN management.

Building upon this foundation, this paper presents an xApp-based solution that enables near-real-time, 3GPP- and O-RAN-compliant inter-slice handover directly within the RAN. Unlike conventional 3GPP slice switching, our approach performs the handover locally at the Near-RT RIC using extensions to the RAN Control Service Model (RC SM) [4]. The proposed SliceHO xApp dynamically reassigns UEs between slices without triggering additional core network signaling or requiring UE involvement.

The main contributions of this paper are as follows:

- A fully RAN-based inter-slice handover mechanism implemented as an xApp within the Near-RT RIC.
- An extension of the O-RAN RC SM enabling dynamic UE-to-slice reassignment without PDU re-establishment.
- Implementation and validation on an OAI-based testbed using both over-the-air and emulated environments.
- Public release of the extended RC SM, inter-slice handover logic on MAC layer of OAI RAN and SliceHO xApp code on the FlexRIC platform to foster reproducibility and future research.[1][2]

The remainder of this paper is organized as follows. Section II reviews 3GPP slice management mechanisms. Section III presents the proposed architecture, Section IV discusses the experimental evaluation, and Section V outlines representative use cases. Section VI summarizes related work, Section VII discusses future research directions, and Section VIII concludes the paper.

[1]https://github.com/13781378setar/RAN_inter_slice_handover
[2]https://github.com/13781378setar/flexric_inter_slice_handover

II. Background on 3GPP Slice Management

Standardized by 3GPP, network slicing is a fundamental capability of 5G systems that enables logical partitioning of network resources to support diverse service requirements. Each slice is identified by a Single NSSAI (S-NSSAI), which consists of a Slice/Service Type (SST) and an optional Slice Differentiator (SD). Within a Public Land Mobile Network (PLMN), the collection of all S-NSSAIs supported by the network constitutes the *Configured NSSAI*. During the registration procedure, the Access and Mobility Management Function (AMF) selects a subset of these slices, known as the *Allowed NSSAI*, based on the UE's subscription information, operator policies, and slice availability. The UE is always served by at least one *Default S-NSSAI*, chosen from the Allowed NSSAI, which is used to establish the initial PDU session and ensure baseline connectivity.

Subsequent PDU session establishments may associate the UE with additional S-NSSAIs from the Allowed NSSAI, enabling simultaneous access to multiple slices. Conversely, an existing PDU session, and thus the corresponding slice, can be released upon request by either the UE or the core network. Each PDU session maps to a specific slice and may include multiple QoS flows, which can be carried over one or more Data Radio Bearers (DRBs) inside that PDU session for user-plane data transmission. This hierarchical mapping ensures isolation and QoS differentiation across concurrent slices.

A. 3GPP Inter-slice Switching Mechanisms

After initial registration and default slice assignment, a UE may connect to another slice through *slice switching* procedures defined by 3GPP. Importantly, the current 3GPP framework does not define any mechanism for *inter-slice handover* with session continuity. Instead, slice switching occurs when either the AMF or the UE requests access to a new slice, leading to the establishment of a new PDU session on the target slice and subsequent release of the previous one [1], [5], [6]. Because these two actions are separate, a temporary service interruption occurs between the release and re-establishment phases.

AMF-triggered slice switching: The AMF may modify a UE's slice configuration for load balancing, policy enforcement, or slice unavailability. If the target S-NSSAI already exists within the Allowed NSSAI, the AMF initiates a new PDU session on that slice. Otherwise, it updates the Allowed NSSAI and informs the UE via a *UE Configuration Update* message. When an active slice is removed from the Allowed NSSAI, the corresponding PDU session must be released, interrupting connectivity while re-registration and session setup complete, as illustrated in Figure 2a.

UE-triggered slice switching: Alternatively, the UE may initiate access to a new slice by sending a *Registration Request* message containing a *Requested NSSAI*. The AMF verifies the request using subscription data and, if permitted, updates the Allowed NSSAI and authorizes the establishment of a new PDU session on the target slice. Then, the existing PDU session must be released. The signaling sequence for the UE-triggered procedure is shown in Figure 2b.

In summary, 3GPP procedures support slice switching through the release and re-establishment of PDU sessions but provide no native support for seamless inter-slice handover. This reliance on core-level signaling and AMF coordination introduces latency, signaling overhead, and service disruption. To complement these standard procedures, our proposed approach remains fully 3GPP-compliant while introducing RAN-level intelligence that enables fast, context-aware inter-slice handover directly within the RAN.

III. System Model and Implementation

The proposed inter-slice handover solution is realized via an xApp-based mechanism operating on the Near-RT RIC, leveraging the capabilities of the O-RAN architecture as shown in Figure 3. It enables dynamic, low-latency UE handovers across slices at the RAN level, without relying on core network signaling. Unlike the 3GPP inter-slice switching procedure, which relies on AMF- or UE-triggered PDU session establishment and reconfiguration, SliceHO xApp performs slice handovers entirely within the RAN domain through the Near-RT RIC, significantly reducing control overhead and latency.

A. RAN and RC SM Integration

Since inter-slice handover is not explicitly defined in the standard, we extended the RC SM, responsible for UE-level radio configuration and mobility control, by introducing new

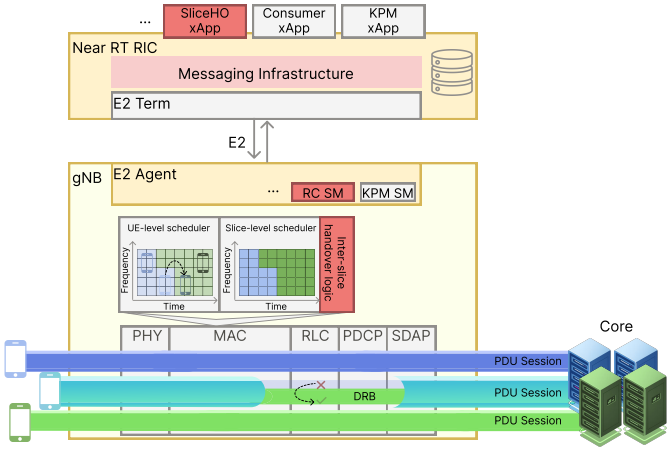


Fig. 3: Proposed system architecture for inter-slice handover in O-RAN.

E2 interface primitives that trigger inter-slice handover logic within the RAN. This extension defines a new control action, `RIC Control Type=Slice Handover`, which, on demand, assigns one or more UEs to a different serving slice, enabling RAN-level management of inter-slice handovers.

Within the RAN, radio resources at the MAC layer are organized in a time–frequency grid, where PRBs are allocated to slices according to QoS and Service-Level Agreement (SLA) requirements. Data from the RLC buffer is transmitted to the MAC layer through logical channels mapped to DRBs associated with PDU sessions within specific slices. Inter-slice handover logic leverages this structure to dynamically remap a UE's logical channels from one slice to another, ensuring uninterrupted data flow and scheduling continuity during the transition.

To minimize signaling overhead, inter-slice handover logic reuses existing DRBs within the active PDU session by handing over the UE's logical channels to the target slice, eliminating the need to establish a new PDU session. This operation occurs prior to MAC-layer scheduling. Within the MAC layer, scheduling occurs in two phases: first, resources are allocated at the slice level; then, UEs are scheduled within each slice during the UE-level scheduling phase. The UE is integrated into the destination slice's scheduling context, and in the next scheduling cycle, it is allocated resources from the target slice. As the DRB is preserved and scheduling remains continuous, the handover is seamless, incurring no observed packet loss or service interruption and achieving low end-to-end latency.

B. SliceHO xApp

The Near-RT RIC serves as the control foundation for the inter-slice handover solution, enabling closed-loop monitoring and control at sub-second timescales (10 ms–1 s). By leveraging near-real-time per-UE RAN metrics, such as PRB utilization, RLC delay, and throughput, the RIC enables data-driven, context-aware decision-making for adaptive net-

work control. The SliceHO xApp extends this capability by performing seamless inter-slice handovers and exposing this functionality as a service to other xApps within the O-RAN ecosystem. Once invoked, the SliceHO xApp executes the handover procedure via the E2 interface through the RC SM. On the RAN side, the RC SM parses this control request and triggers the inter-slice handover logic.

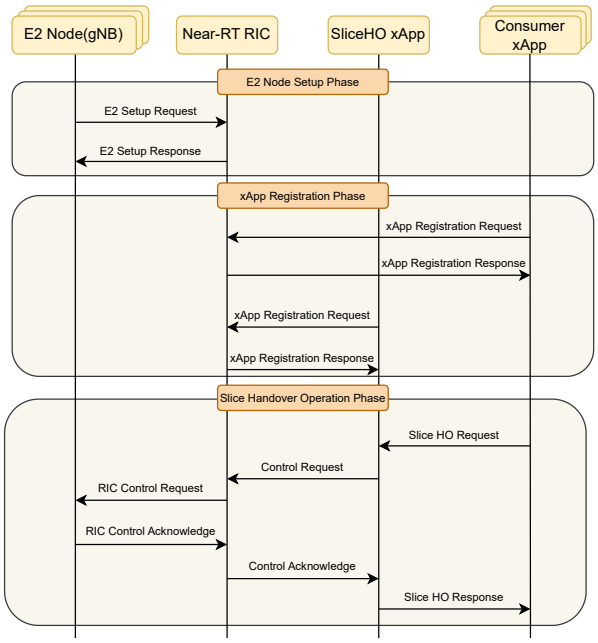


Fig. 4: Messaging procedure of the inter-slice handover in an xApp-based closed-loop control framework.

C. SliceHO xApp in Closed-Loop Control Procedure

Figure 4 illustrates the proposed inter-slice handover solution within an O-RAN–based architecture, achieved through the coordinated interaction of multiple xApps operating on the Near-RT RIC.

After the E2 setup between the E2 nodes (e.g., gNBs) and the Near-RT RIC, the SliceHO xApp registers with the RIC platform following the O-RAN xApp registration procedure. Once registered, the SliceHO xApp exposes its inter-slice handover service to consumer xApps.

The KPM xApp, a native O-RAN component that collects RAN key performance metrics via the standard E2 interface, continuously gathers and stores them in a database.

A *consumer xApp* (i.e., an xApp that utilizes the inter-slice handover service, such as for congestion control, latency assurance, or slice load balancing), processes these metrics to detect SLA violations or performance anomalies and determine the most suitable target slice for a given UE. This xApp then invokes the SliceHO xApp's exposed service through the Near-RT RIC messaging infrastructure to trigger the inter-slice handover. It initiates a `Slice HO Request` containing a list of 64-bit RAN UE ID values for multi-UE inter-slice handover, along with the source and destination S-NSSAI identifiers, each composed of an 8-bit SST and a 24-bit SD.

The SliceHO xApp processes the request and sends a `Control Request` message to the Near-RT RIC. The RIC then forwards a corresponding `RIC Control Request` to the E2 node through the E2 interface, which executes the handover and returns a `RIC Control Acknowledge`. This acknowledgment is relayed back to the SliceHO xApp as a `Control Acknowledge`, which finally responds to the consumer xApp with a `Slice HO Response`.

The inter-slice handover procedure uses only four control messages, making it significantly more efficient and latency-aware than conventional 3GPP slice-switching mechanisms.

IV. Evaluation

A. Testbed Setup

The experimental setup [7] consists of a fully functional 5G standalone (SA) network compliant with 3GPP specifications, encompassing OpenAirInterface (OAI) 5G RAN [8] and OAI 5G Core [9]. All components, including the RAN, core, Near-RT RIC, and xApps, are deployed on a single bare-metal machine. The testbed employs a commercial software-defined radio (SDR) for over-the-air transmission and two COTS UE devices for connectivity testing. To evaluate scalability, additional experiments were conducted using OAI's RF Simulator, which emulates the radio interface and enables multi-UE testing using OAI UEs without physical SDRs.. The hardware and configuration parameters of the testbed are summarized in Table I. We evaluate our proposed inter-slice handover solution, implemented as an xApp operating on top of FlexRIC [10] as the Near-RT RIC.

TABLE I: Configuration of the Experimental Testbed

Component	**Details**
Radio Frontends (RFs)	Ettus USRP X310 with UHD v4.3 (physical) OAI-RFSIMULATOR (software-emulated)
User Equipments (UEs)	Google Pixel 7 Pro (physical) OAI-UE (emulated)
SIM Cards	Sysmocom programmable SIMs
5G Core 5G RAN Near-RT RIC	Intel Xeon E3-1230 @ 3.7 GHz

B. Inter-Slice Handover Testing

To evaluate the impact of the proposed inter-slice handover solution on UE throughput and session continuity, we run a progressive experiment in which: (i) two UEs are initially connected to a non-sliced RAN (baseline), (ii) the RAN is sliced and both UEs are assigned to the same slice with high-capacity, (iii) both UEs are handed over to a limited-capacity slice, and finally (iv) one UE is handed over back to the high-capacity slice.

Figure 5 presents the results of the over-the-air experiment conducted on the SDR-based testbed with COTS UEs running TCP traffic. Initially, both UEs share the total network capacity, each achieving approximately 63 Mbps, implying a

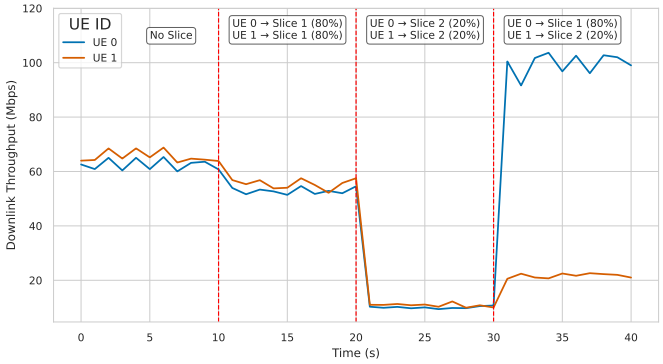


Fig. 5: Throughput variation across four over-the-air scenarios: (i) two UEs sharing bandwidth without slices, (ii) both UEs in a high-capacity slice, (iii) both UEs handed over to a low-capacity slice, and (iv) one UE handed over back to the high-capacity slice

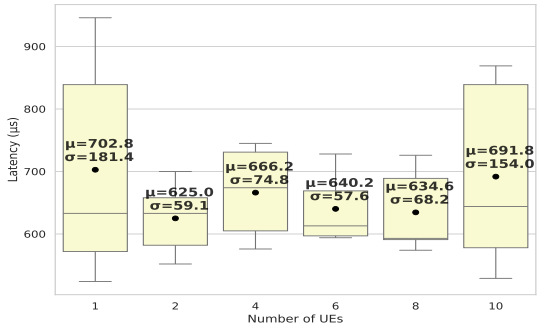
total network capacity of about 120 Mbps. At $t = 10$ s, two slices are instantiated, slice one with a configured capacity of 80%, resulting in a throughput of around 56 Mbps per UE. 10 seconds later, both UEs are handed over to a low-capacity (20%) slice, reducing their throughput to about 10 Mbps each. Finally, at $t = 30$ s, UE_0 is reassigned to the high-capacity (80%) slice, achieving roughly 105 Mbps, as it exclusively utilizes the resources of that slice, while UE_1 remains in the 20%-capacity slice with a throughput of around 20 Mbps.

This experiment clearly demonstrates the effectiveness of the proposed xApp-driven inter-slice handover mechanism in the over-the-air testbed. The results confirm that UEs can be dynamically handed over across slices with seamless connectivity, zero service interruption, and throughput performance closely matching the configured slice capacities. This validates the capacity of RAN-level control to deliver low-latency, resource-efficient, and service-aware inter-slice mobility beyond the conventional 3GPP inter-slice switching procedure.

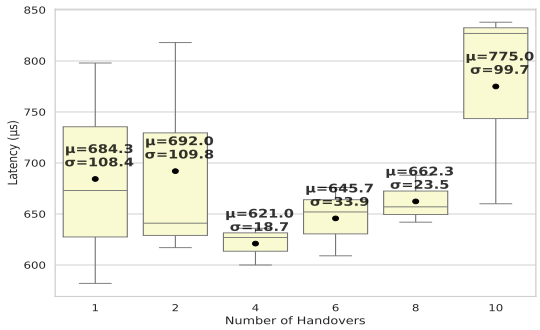
C. Scalability

We evaluate the system's scalability by analyzing the impact of increasing numbers of connected UEs, as shown in Figure 6a. We increase the number of UEs to 10, constrained by the capacity of the emulated testbed. The results show that the latency of the end-to-end control loop between the SliceHO xApp and the RAN for an inter-slice handover of a single UE remains consistently low, averaging around 660 μs, even as the number of connected UEs increases. This is significantly more efficient than establishing a new PDU session through standard 3GPP procedures.

To assess the scalability of our solution from another perspective, we fix the number of connected UEs at 10, again limited by the capacity of the emulated testbed, and vary the number of UEs being simultaneously handed over (1, 2, 4, 6, 8, and 10). As shown in Figure 6b, the control loop delay remains low, around 680 μs, demonstrating that our proposed xApp-based mechanism maintains high responsiveness even under multi-UE handover scenarios.



(a) End-to-End latency variation with the increasing number of connected UEs when handing over a single UE.



(b) End-to-End latency variation with an increasing number of UEs that undergo handover.

Fig. 6: Measured End-to-End latency during inter-slice handover procedure.

V. Use Case

Our proposed inter-slice handover solution within the RAN offers advantages across diverse use cases, including, for example, attack mitigation, monetary slice reassignment, slice load balancing, and subscription policy enforcement. These and more cases are summarized in Table II. By performing handovers entirely within the RAN, without establishing new PDU sessions or requiring UE involvement, the solution enables not only seamless service continuity and low latency, but also policy-driven decision making based on rapidly changing RAN context.

Next, we use two representative scenarios to showcase the effectiveness of SliceHO. In the *Monetary-Based Slice Reassignment* use case (paragraph V-A), UEs with high traffic demand are dynamically reassigned to higher-capacity slices to maximize achievable data rates without disrupting ongoing sessions and improving resource utilization. In the *SLA Degradation Recovery* use case (paragraph V-B), the control loop continuously monitors URLLC flows to preserve their strict latency requirements. Upon detecting anomalous traffic that causes congestion or delays to legitimate flows, the system triggers an inter-slice handover to reassign the offending UE

TABLE II: Representative Use Cases for Inter-Slice Handover via xApps in the RAN

Use Case	Primary KPI(s)	Summary Description
Slice load balancing	Slice throughput, UE satisfaction	Triggered when a slice becomes congested due to high PRB utilization or increased active UEs. The SliceHO xApp redistributes UEs across available slices according to load and QoS priorities, ensuring service continuity and preventing resource starvation.
Attack and overload mitigation	Slice latency and throughput, Network stability	Activated when anomaly detection xApps identify bursty traffic, signaling storms, or slice overload. Affected UEs are handed over to isolated or quarantine slices to safeguard normal services and maintain operational stability across other slices.
Subscription policy enforcement	Slice admission rate, Service Continuity	When a user exceeds its data quota, time quotas, or service-level limits, the xApp coordinates with policy control functions to move the UE to a lower-priority or best-effort slice, preserving fairness and compliance with subscription agreements.
SLA degradation recovery	UE throughput, Slice delay	When the monitored SLA metrics (e.g., throughput, jitter, delay) fall below target thresholds, the SLA xApp triggers SliceHO handover to migrate affected UEs to slices that can satisfy their latency or reliability requirements.
Monetary- or SLA-based slice reassignment	Cost efficiency, SLA compliance	SliceHO xApp dynamically hands over UEs to slices offering the best trade-off between performance and cost, or enforces SLA adjustments based on real-time resource availability.

to an isolated slice, ensuring SLA-compliant performance for benign UEs.

A. Monetary-Based Slice Reassignment

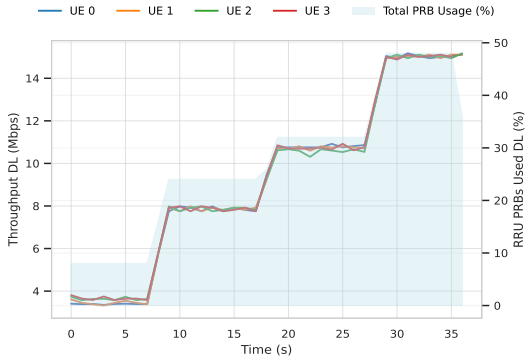
At the UE level, in the MAC layer, standard schedulers, such as the Proportional Fair (PF) algorithm, ensure fairness by evenly distributing a slice's bandwidth among UEs within the slice. However, in scenarios where a UE becomes eligible for higher throughput, such as when upgrading to a "premium subscription" or initiating a legitimate bandwidth-intensive application, this fairness constraint limits the UE's achievable rate. Simply increasing the overall capacity of the slice does not resolve this issue, since the additional resources are proportionally shared among all UEs, including those that are not eligible for enhanced service. This results in inefficient resource utilization and undermines the intended quality differentiation.

In our experiment, four UEs are connected to the network, each generating UDP traffic at a target rate of 50 Mbps to emulate bandwidth-hungry users. Initially, they are all serviced by the same slice, whose capacity (10% of the total number of PRBs) caps their throughput at around 3 Mbps as the capacity of the slice is fairly shared among the 4 UEs.

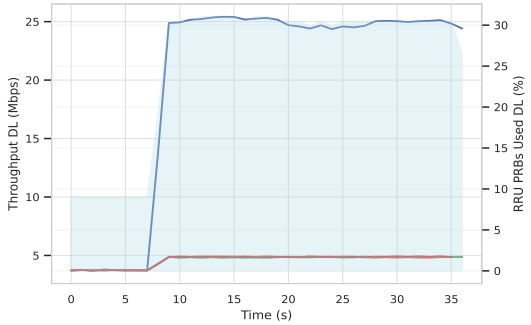
At $t = 7$ s, UE_0 becomes eligible for throughput increase from 3 Mbps to 25 Mbps.

As shown in Figure 7a, at $t = 7$ s, although the resources (i.e., PRBs) dedicated to the slice are scaled up to accommodate the UE_0's throughput upgrade, UE_0 is not able to achieve the desired throughput. We incrementally increase the resources dedicated to the slice, and even with 50% of the total PRBs assigned to it, UE_0 barely achieves 15 Mbps.

In the second experiment, we trigger SliceHO xApp at $t = 7$ s , which hands over UE_0 to a different slice with 20% PRBs. Figure 7b shows that UE_0 indeed achieves the desired throughput (25 Mbps) after it is moved to the second slice, with a 30% PRB utilization in total; 10% dedicated to the first slice shared by UE_1, UE_2, and UE_3, and 20% dedicated to the second slice where UE_0 was handed over. This demonstrates that the proposed solution enables targeted throughput enhancement without compromising fairness or connectivity,



(a) Throughput variation over time without inter-slice handover with total PRB used by UEs.



(b) Throughput variation over time with inter-slice handover with total PRB usage of UEs.

Fig. 7: Comparison of throughput and total PRB utilization over time with and without inter-slice handover, highlighting the impact of dynamic slice handover on resource efficiency and performance.

validating the effectiveness of inter-slice handover for dynamic and efficient PRB utilization within the RAN.

B. SLA Degradation Recovery

In this second series of experiments, we evaluate how the *SliceHO xApp*, operating within a closed-loop control framework, can isolate misbehaving UEs and help recover from

SLA degradation caused by RAN resource contention. Rather than disconnecting suspicious UEs, the proposed mechanism maintains their network connectivity by relocating them to constrained slices, enabling continued monitoring and investigation while protecting high-priority and legitimate services. The closed loop is driven by a custom *SLA xApp*, which monitors slice performance in real time and triggers SliceHO xApp when misbehaving UEs are detected.

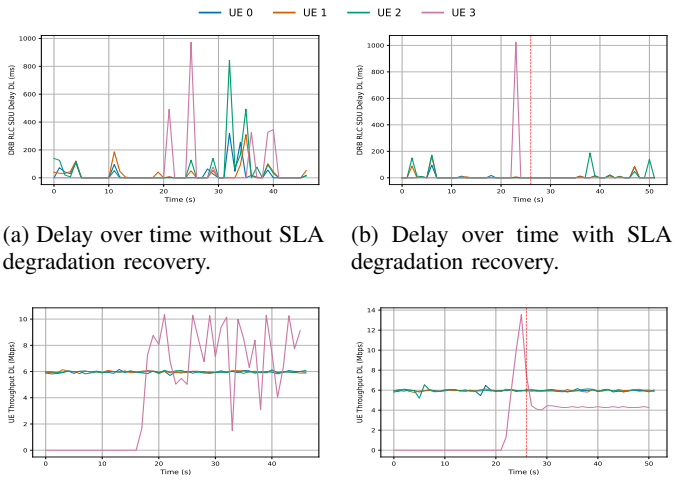
To enable automated detection, we first collected a dataset of URLLC-type traffic generated in our testbed using 64-byte packets at 6 Mbps. From the RAN, we extracted 17 per-UE KPM features, including RLC delay, throughput, PRB utilization, MAC transport block size, packet counts, and RLC/MAC PDU statistics, capturing the expected behavior of URLLC flows. An ML model trained on this dataset enables the SLA xApp to distinguish legitimate URLLC traffic from outlier flows that deviate from the learned profile.

The SLA xApp operates as follows: it continuously retrieves KPM data from the MySQL database populated by the KPM xApp, analyzes the metrics in real time using the trained model, detects abnormal patterns within the URLLC slice, and, upon identifying a misbehaving UE, coordinates with the *SliceHO xApp* to initiate an inter-slice handover via E2 control messages.

For testing, we introduced non-URLLC traffic consisting of 1500-byte packets averaging 8 Mbps with rate fluctuations, generated using Python scripts to emulate realistic bursty behavior. Figures 8a and 8c show the system with three UEs transmitting URLLC traffic. Latency remains stable, i.e, below 200 µs, until $t = 20$ s, when a fourth UE begins generating bursty traffic. This congestion disrupts the scheduler's operation, causing RLC delay for URLLC UEs to spike above 400 µs, violating SLA requirements.

Once the SLA xApp detects this anomaly, it identifies the misbehaving UE and triggers a slice reassignment through the SliceHO xApp. As shown in Figure 8d, the UE is promptly handed over to a lower-capacity slice, isolating the disruptive traffic. The effect of this corrective action is visible in Figure 8b, where RLC delay for URLLC UEs stabilizes again below 200 µs.

This experiment demonstrates that the proposed closed-loop mechanism can preserve URLLC SLA performance under transient disturbances by rapidly isolating misbehaving UEs through RAN-level inter-slice handover.



(a) Delay over time without SLA degradation recovery.

(b) Delay over time with SLA degradation recovery.

(c) Throughput over time without SLA degradation recovery.

(d) Throughput over time with SLA degradation recovery.

Fig. 8: Comparison of per-UE delay and throughput with and without SLA degradation recovery.

VI. Related Work

In the broader context of inter-slice handover research, studies such as [11] and [12] suggest that handover management in network slicing requires new mechanisms beyond traditional cell-level handover procedures. However, only a few efforts, such as [13], have attempted to address this challenge, and none have considered the handover of a UE between slices within the same cell. Sajjad et al. [1] provide a comprehensive overview of 3GPP-defined mechanisms, including UE registration and PDU session re-establishment procedures. They conclude that current standards do not inherently support seamless inter-slice handovers and highlight the potential of data-driven and machine learning approaches to improve decision-making. Nonetheless, 3GPP inter-slice switching procedure remains core-centric, resulting in high latency and temporary service interruption. These limitations motivate the need for RAN context-aware, intelligent, and low-latency inter-slice handover solutions enabled by the O-RAN architecture and xApps, as explored in this work.

Prior studies have suggested incorporating inter-slice handover mechanisms into the 3GPP standards to support objectives such as load balancing, policy enforcement, and delay optimization [1]. Beyond performance improvements, researchers have also explored their potential to enhance network security. For instance, Bousalem et al. in [14] and [15] proposed a deep learning–based framework for DDoS detection in sliced 5G networks, where malicious users are reassigned to isolated sinkhole slices. However, their approach is not aligned with the O-RAN framework, and its slicing implementation does not adhere to 3GPP network slicing specifications.

As discussed in [16], [17], various MAC scheduler designs have been proposed to enable RAN slicing. The two-level scheduler in [16] performs slice-level scheduling followed by UE-level scheduling within each slice, while [17] adopts a recursive approach that first selects a slice and then allocates resources to its UEs. Since PRB allocation occurs at the MAC layer, slicing is inherently enforced there. Our solution operates before MAC scheduling logic to ensure seamless continuity after handover.

In [18], programmable queues controlled by an xApp are used to manage QoS flows within slices, enabling flow-based slicing without explicit UE–slice association. Our solution hands over UEs at the MAC layer, achieving proper slice-level management aligned with SLA and scheduling require-

TABLE III: Comparison between existing literature and the proposed architecture for inter-slice handover. Symbols in the table denote: supported (✓), not supported (×), and partial/limited support (⧫).

Ref.	O-RAN Compliant	Dynamic inter-slice handover	Open Source	3GPP-based RAN Slicing
Flexslice [17]	⧫	⧫	×	×
OranSlice [19]	✓	×	✓	✓
ProSLICE [20]	⧫	×	×	✓
Zipper [21]	×	×	×	✓
SliceHO	✓	✓	✓	✓

ments. Combining UE-level handover with flow-level control through the TC framework could further enhance end-to-end performance and resource efficiency.

As summarized in Table III, all related studies focus primarily on RAN slicing frameworks rather than inter-slice mobility. For example, OranSlice [19] proposes an O-RAN–compliant architecture for network slicing, while ProSLICE [20] develops a customized E2SM and xApp to support slice management. However, ProSLICE does not adhere to the 3GPP RAN slicing model, and its E2SM design is not fully O-RAN–compliant. Similarly, Zipper [21] presents a closed-source slicing framework that allows slice resource adjustment but lacks UE-level handover functionality, as UEs are statically assigned to slices. FlexSlice [17] introduces dynamic UE-to-slice assignment in its RAN slicing framework, which is not 3GPP-compliant, and focuses on UE-to-slice association rather than inter-slice handover.

After reviewing the existing literature, we found no mechanism for inter-slice handover within the RAN. This work fills that gap by introducing SliceHO, a novel solution that supports dynamic, efficient, and seamless handover management across network slices, addressing a limitation not addressed by prior state-of-the-art solutions.

VII. Discussion and Future Work

This work introduced an inter-slice handover mechanism that enables intelligent slice handover directly in the RAN without requiring PDU session (re)establishment. The proposed approach maintains low handover latency and seamless UE connectivity while efficiently scaling to handle simultaneous inter-slice handovers for multiple UEs.

While 3GPP also defines the QoS Flow Identifier (QFI) mechanism for traffic differentiation, it operates only within a single slice. After a UE is handed over to its target slice, QFIs can further differentiate traffic at the flow level, with each representing a specific service requirement such as latency, priority, or bit rate. However, because all DRBs in a PDU session share the same S-NSSAI, QFI-based differentiation remains limited to intra-slice management. In contrast, the proposed inter-slice handover mechanism enables dynamic, cross-slice traffic steering without requiring new PDU sessions [6].

In 6G, where networks aim to deliver personalized, user-centric experiences [2], the proposed approach provides a foundation for self-optimizing RAN slicing. By enabling real-time, context-aware inter-slice handover, it supports dynamic inter-slice mobility to user needs.

Furthermore, integrating rApps into the Non-RT RIC and leveraging the Network Data Analytics Function (NWDAF) [22], [23] in the core can enable coordinated, end-to-end slice management, where NWDAF triggers core-level slicing decisions, and xApps manage RAN-level adaptations, achieving a unified, intelligent cross-domain control architecture.

VIII. Conclusion

The proposed inter-slice handover solution builds on 3GPP-defined network slicing and is fully compliant with the O-RAN framework, introducing flexible control and near-real-time programmability in the RAN user plane to enable seamless, low-latency inter-slice handover. Experimental results demonstrate that performing slice management directly within the RAN is significantly more efficient and context-aware than 3GPP inter-slice switching, reducing control signaling and achieving sub-millisecond responsiveness. By eliminating dependence on core-network procedures, the solution leverages real-time RAN information to enable intelligent, dynamic, and efficient slice management.

IX. Acknowledgment

This work was supported in part by Rogers Communications Canada Inc., NSERC Alliance, and the Ontario Research Fund – Research Excellence program (Project# ORF-RE012-051) from the Province of Ontario. The views expressed herein are those of the authors and do not necessarily reflect those of the Province.

References

[1] M. M. Sajjad, C. J. Bernardos, D. Jayalath, and Y.-C. Tian, "Inter-slice mobility management in 5g: motivations, standard principles, challenges, and research directions," *IEEE Communications Standards Magazine*, vol. 6, no. 1, pp. 93–100, 2022.

[2] X. Li, M. Samaka, H. A. Chan, D. Bhamare, L. Gupta, C. Guo, and R. Jain, "Network slicing for 5g: Challenges and opportunities," *IEEE Internet Computing*, vol. 21, no. 5, pp. 20–27, 2017.

[3] J. F. Santos, A. Huff, D. Campos, K. V. Cardoso, C. B. Both, and L. A. DaSilva, "Managing o-ran networks: xapp development from zero to hero," *IEEE Communications Surveys & Tutorials*, 2025.

[4] O-RAN WG3, "O-RAN E2 Service Model (E2SM) RAN Control 9.0," O-RAN Alliance, Tech. Rep. O-RAN.WG3.TS.E2SM-RC-R004-v09.00, 2024.

[5] 3GPP, "Procedures for the 5G System (5GS)," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 23.502 V17.10.0, 2024.

[6] ——, "System Architecture for the 5G System (5GS)," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 23.501 V17.10.0, 2024.

[7] M. Rouili, N. Saha, M. Golkarifard, M. Zangooei, R. Boutaba, E. Onur, and A. Saleh, "Evaluating open-source 5g sa testbeds: Unveiling performance disparities in ran scenarios," in *NOMS 2024-2024 IEEE Network Operations and Management Symposium*. IEEE, 2024, pp. 1–6.

[8] EURECOM, "OpenAirInterface5G," https://openairinterface.org/, 2024, accessed: 2026-02-12.

[9] ——, "OpenAirInterface 5G Core Network," https://gitlab.eurecom.fr/oai/cn5g, 2024, accessed: 2026-02-12.

[10] R. Schmidt, M. Irazabal, and N. Nikaein, "Flexric: An sdk for next-generation sd-rans," in *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT)*, Dec. 2021, pp. 411–425.

[11] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," *IEEE communications magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[12] X. Li, M. Samaka, H. A. Chan, D. Bhamare, L. Gupta, C. Guo, and R. Jain, "Network slicing for 5g: Challenges and opportunities," *IEEE Internet Computing*, vol. 21, no. 5, pp. 20–27, 2017.

[13] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, and V. C. Leung, "Network slicing based 5g and future mobile networks: Mobility, resource management, and challenges," *IEEE communications magazine*, vol. 55, no. 8, pp. 138–145, 2017.

[14] B. Bousalem, V. F. Silva, R. Langar, and S. Cherrier, "Deep learning-based approach for ddos attacks detection and mitigation in 5g and beyond mobile networks," in *2022 IEEE 8th international conference on network softwarization (NetSoft)*. IEEE, 2022, pp. 228–230.

[15] ——, "Ddos attacks detection and mitigation in 5g and beyond networks: A deep learning-based approach," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 1259–1264.

[16] A. Ksentini and N. Nikaein, "Toward enforcing network slicing on ran: Flexibility and resources abstraction," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 102–108, 2017.

[17] C.-C. Chen, C.-Y. Chang, and N. Nikaein, "Flexslice: Flexible and real-time programmable ran slicing framework," in *GLOBECOM 2023-2023 IEEE Global Communications Conference*. IEEE, 2023, pp. 3807–3812.

[18] M. Irazabal and N. Nikaein, "Tc-ran: A programmable traffic control service model for 5g/6g sd-ran," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 2, pp. 406–419, 2023.

[19] H. Cheng, S. D'Oro, R. Gangula, S. Velumani, D. Villa, L. Bonati, M. Polese, T. Melodia, G. Arrobo, and C. Maciocco, "Oranslice: An open source 5g network slicing platform for o-ran," in *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, 2024, pp. 2297–2302.

[20] A. Kak, V.-Q. Pham, H.-T. Thieu, and N. Choi, "Proslice: an open ran-based approach to programmable ran slicing," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 197–202.

[21] A. Balasingam, M. Kotaru, and P. Bahl, "{Application-Level} service assurance with 5g {RAN} slicing," in *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, 2024, pp. 841–857.

[22] A. Mekrache, K. Boutiba, and A. Ksentini, "Combining network data analytics function and machine learning for abnormal traffic detection in beyond 5g," in *GLOBECOM 2023-2023 IEEE Global Communications Conference*. IEEE, 2023, pp. 1204–1209.

[23] F. Shafiei Ardestani, N. Saha, N. Limam, and R. Boutaba, "Towards nwdaf-enabled analytics and closed-loop automation in 5g networks," *arXiv e-prints*, pp. arXiv–2505, 2025.

[24] 3GPP, "5G Network Resource Model (NRM)," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 28.541 V16.4.0, 2021.

[25] O-RAN WG3, "O-RAN Near-Real-time RAN Intelligent Controller (Near-RT RIC) and E2 Interface," O-RAN Alliance, Tech. Rep. O-RAN.WG3.TS.RICARCH-R004-v07.00, 2024.