Poster: VeriRAN: Explainable and Runtime-Verified Multi-Agent Control for Trustworthy AI-RAN

Osman Tugay Basaran
basaran@ccs-labs.org
EECS, TU Berlin
Berlin, Germany

Falko Dressler
dressler@ccs-labs.org
EECS, TU Berlin
Berlin, Germany

Abstract

Artificial intelligence (AI)-Native sixth-generation (6G) radio access networks (RANs) are expected to support increasingly dynamic control loops, where learning-based agents continuously adapt radio resources, slice priorities, and service behavior. However, this shift from static optimization to autonomous control raises a fundamental trust question: how can AI-driven RAN decisions remain safe when radio conditions change, telemetry becomes unreliable, or latency-critical services experience sudden stress? This paper presents `VeriRAN`, a lightweight runtime-verified AI-RAN control architecture that separates intelligence from authorization. Instead of treating AI agents as trusted actuators, `VeriRAN` uses them as intelligent action proposers that generate radio-aware and service-level-agreement (SLA)-aware decisions. Our verification shield is placed directly in the control path to approve, constrain, or replace unsafe actions before they affect the RAN.

CCS Concepts

• **Computing methodologies** → **Artificial intelligence, Multi-agent systems**.

Keywords

6G, AI-RAN, Agentic Control, Next-G Architecture, Trustworthy AI

ACM Reference Format:
Osman Tugay Basaran and Falko Dressler. 2026. Poster: VeriRAN: Explainable and Runtime-Verified Multi-Agent Control for Trustworthy AI-RAN. In *The 31st Annual International Conference on Mobile Computing and Networking (ACM MOBICOM '26), October 26–30, 2026, Texas, USA.* ACM, New York, NY, USA, 3 pages.

ACM MOBICOM '26, Texas, USA

1 Introduction

Open-RAN is transforming the radio access network (RAN) from; vendor-specific infrastructure into programmable, disaggregated, and artificial intelligence (AI)-enabled system [5]. This creates a natural role for network intelligence: telemetry can be collected from the RAN, processed by AI control functions, and translated into resource-allocation or policy decisions. In future AI-RAN systems, such control loops will be expected to support heterogeneous traffic classes, including high-throughput enhanced mobile broadband (eMBB), mission-critical ultra-reliable low-latency communication (URLLC), and AI/Edge inference traffic.

The challenge is that AI-native control is not automatically trustworthy [1]. A controller that optimizes mean throughput may still behave unsafely under rare but important conditions: channel fading, weak RSRP, increased delay spread, stale key performance measurement (KPM) reports, corrupted telemetry, or sudden URLLC demand[2]. These conditions are especially important for Open-RAN because the same programmability that enables innovation also increases the risk of unsafe control actions. Therefore, AI-RAN controllers should not be treated as trusted actuators but as mere proposals whose decisions must be verified before they affect the RAN.

This paper introduces `VeriRAN`, a runtime-verified multi-agent AI-RAN control architecture. Unlike large multi-agent systems with many loosely defined agents, `VeriRAN` intentionally uses only two operationally meaningful decision agents. The `RAN-State Agent` focuses on radio-side evidence, and the `SLA-Intent Agent` focuses on service-side evidence. A deterministic fusion module combines their proposals, and a runtime verification shield authorizes, clips, or replaces the proposed action. This work makes three contributions. First, we design a compact multi-agent AI-RAN architecture that maps naturally to an O-RAN Near-RT RIC-based [4] control loop. Second, we show how to build the full scenario on top of two NVIDIA Spark DGX [3] nodes: *Spark-1* provides the Sionna/Sionna-RK/Open RAN simulator,[1] and *Spark-2* hosts the RIC-side `VeriRAN` agentic intelligence stack. Third, we evaluate the architecture over

[1] https://nvlabs.github.io/sionna/rk/index.html

Sionna-RT-compatible Open RAN data and show that safety-aware verification eliminates URLLC SLA violations with only a small throughput cost.

2 Preliminaries

AI-RAN Control State: We consider an O-RAN Near-RT RIC-based control loop with a one-second control period. *Spark-1* exports per-UE and per-slice measurements obtained from Sionna-RT/Sionna-RK/OAI-based telemetry. The data contains radio features such as SINR, RSRP, CQI, path gain, mean delay, RMS delay spread, and PRB pressure, together with service features such as offered load, throughput, latency, packet loss, and retransmission rate. These features are aggregated into a control state:

$$s_t = [q_t, r_t, c_t, g_t, \tau_t, p_t, d_t, y_t, \ell_t], \tag{1}$$

where q_t is SINR, r_t is RSRP, c_t is CQI, g_t is path gain, τ_t is delay spread, p_t is PRB pressure, d_t is slice demand, y_t is measured throughput, and ℓ_t is measured latency/loss.

Action and Trust Model: The controller outputs a slice-share action:

$$a_t = [a_t^{eMBB}, a_t^{URLLC}, a_t^{AI}], \quad \sum_i a_t^i = 1. \tag{2}$$

The action can be mapped to an O-RAN RC control message, a scheduler weight update, a slice PRB-ratio command, or a controlled Sionna-RK experiment parameter. The agents are proposal functions, not trusted actuators. The verifier is the final gatekeeper and enforces:

$$a_t^{URLLC} \geq \theta_{URLLC}, \quad a_t^{AI} \geq \theta_{AI}, \quad \|a_t - a_{t-1}\|_\infty \leq \Delta_{max}, \tag{3}$$

when the out-of-distribution (OOD) score exceeds a threshold, the verifier replaces the proposal with a smooth fallback action $a_t^{fb} = \alpha a_{t-1} + (1-\alpha)a^{safe}$.

3 Design and Setup

In Figure 1, *Spark-1* is the real measurement source. In Sionna-RT mode, it loads the radio scene, UE trajectory, and gNB placement, then exports the path and link features for each UE position. In Sionna-RK mode, it runs the OAI-based software-defined 5G stack and exports KPM/application metrics through RF-simulator.

Spark-2 runs `VeriRAN`. A KPM ingestion xApp receives the Spark-1 trace stream and aggregates per-UE measurements into one control state per epoch. The `RAN-State Agent` and `SLA-Intent Agent` independently compute resource-allocation proposals. The fusion module combines them as $\hat{a}_t = \lambda a_t^{RAN} + (1-\lambda)a_t^{SLA}$. The verifier then approves, clips, or replaces $\hat{a}_t$. The final action is recorded with a verdict and explanation.

The `RAN-State Agent` is a radio-aware feedback controller. It improves URLLC protection when SINR is low,

Figure 1: Two-Spark AI-RAN system architecture; connected via Amphenol Cable QSFP-to-QSFP 112G, 32AWG.

RSRP is weak, delay spread is high, or PRB pressure indicates overload. It also relaxes protection under good channel conditions, allowing eMBB and AI/edge traffic to recover throughput. The `SLA-Intent Agent` is a service-aware controller. It reacts to offered load and SLA stress. URLLC receives additional priority when latency approaches 10 ms or packet loss increases. eMBB receives additional weight when delivered throughput falls below its demand target. AI traffic receives weight when its latency budget is threatened.

In an O-RAN deployment, the KPM ingestion xApp subscribes to E2SM-KPM metrics such as downlink / uplink throughput and packet success / loss. When RC control is available, the verified action is translated into an E2SM-RC control message or scheduler / slice policy. In a Sionna-RK/OAI deployment, the same action is applied as a scheduler weight, slice PRB target, or experiment-control parameter.

Fusion Module: The fusion module combines the two proposals:

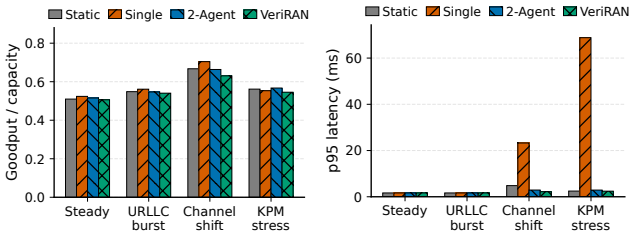
$$\hat{a}_t = \lambda a_t^{RAN} + (1-\lambda)a_t^{SLA}. \tag{4}$$

We use $\lambda = 0.45$ during nominal conditions and $\lambda = 0.52$ during radio stress. Radio stress is triggered when SINR is below 7 dB, RSRP is below -104 dBm, or RMS delay spread exceeds 240 ns. Thus, the `RAN-State Agent` receives more influence when the radio environment becomes unstable, while the `SLA-Intent Agent` receives more influence under normal conditions where service demand is the dominant signal. The fused action $\hat{a}_t$ is not executed directly. It is only a candidate action.

Runtime Verification Shield: The runtime shield is a deterministic safety gate. It enforces base minimum slice shares for eMBB, URLLC, and AI traffic:

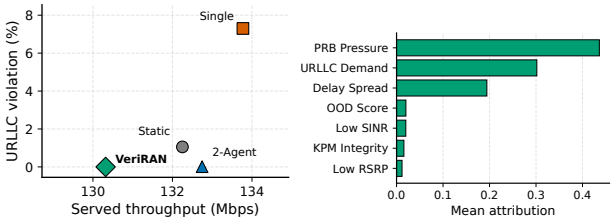
$$\theta = [0.20, 0.25, 0.10]. \tag{5}$$

Under radio resource demand, high URLLC pressure, stale KPM, attack flags, or OOD score above 0.50, the minimum URLLC share is raised to 0.40. The shield also enforces a maximum action step change $\Delta_{max} = 0.20$. If the fused action violates the minimum-share constraints, the shield clips it. If the action changes too abruptly relative to the previous



(a) Goodput-to-capacity ratio (b) URLLC p95 latency

Figure 2: Radio-efficiency and URLLC-safety behavior across the evaluated stress scenarios.



(a) Safety-efficiency tradeoff (b) Verifier explanations

Figure 3: VeriRAN safety-efficiency behavior and runtime verifier actions.

epoch, the shield clips the action step. If the OOD score exceeds 1.00, stale KPM is detected, an attack flag is raised, or a severe low-SINR / high-pressure condition occurs, the shield applies a safe fallback:

$$a^{safe} = [0.28, 0.55, 0.17], \tag{6}$$

$$a_t = \text{norm}(0.45a_{t-1} + 0.55a^{safe}). \tag{7}$$

The fallback is URLLC-prioritized but still leaves resources for eMBB and AI traffic. This is important because an overly conservative fallback would protect URLLC but destroy overall utility.

4 Evaluation

We evaluate four controllers: static slicing, single-agent control, unshielded two-agent control, and VeriRAN. All controllers are evaluated on the same dataset. The dataset contains four scenarios: steady traffic, URLLC burst, channel shift, and KPM-integrity stress. Each controller receives the same sequence of states and produces a slice-share action at every epoch. The evaluation computes the throughput and slice latency from the selected action, the enriched radio capacity, the PRB pressure, the OOD score, and the hidden URLLC demand under KPM-integrity stress. This allows us to evaluate not only average throughput but also whether a controller preserves URLLC under radio resource demand.

Results show that VeriRAN prioritizes verified control. Although its throughput is slightly lower, 130.32 Mbps, due to runtime shielding, it achieves the best URLLC tail-latency

Table 1: Detailed VeriRAN *vs* Baseline Controllers results on the GPU-Accelerated AI-RAN Testbed.

AI-RAN Controllers	Thr. Mbps	p95 ms	p99 ms	Rel. %	Viol. %	Goodput/ Capacity	Fair.	Score
Static	132.25	2.43	5.34	97.95	2.05	0.572	0.710	0.978
Single	**133.77**	22.24	77.69	92.70	7.30	**0.586**	0.695	0.927
2-Agent	132.75	2.52	3.43	98.34	1.66	0.573	0.711	**0.992**
VeriRAN	130.32	**2.20**	**2.43**	**99.60**	**0.40**	0.556	**0.717**	0.974

behavior with 2.20 ms p95 and 2.43 ms p99 latency. The runtime shield blocks 7.53% of unsafe actions and clips 42.60% of risky proposals, demonstrating that it actively shapes the control path rather than passively monitoring it. Figures 2 and 3 illustrate this safety-efficiency tradeoff: VeriRAN sacrifices a small amount of radio utilization but provides the most reliable and auditable behavior under radio and telemetry stress. As shown in Table 1, the single-agent controller achieves the highest throughput (133.77 Mbps) and the highest goodput-to-capacity ratio (0.586), indicating aggressive radio-capacity utilization. However, this comes at a significant safety cost: 7.30% URLLC violation and 22.24 ms p95 latency. This confirms that throughput-oriented AI-RAN control can be efficient on average but unsafe for mission-critical slices under stress.

5 Conclusion

We presented VeriRAN, a runtime-verified multi-agent AI-RAN control architecture for trustworthy Open RAN experimentation. VeriRAN uses a compact design: a RAN-State Agent reasons over radio-link conditions, an SLA-Intent Agent reasons over slice objectives, and a runtime shield authorizes every action before execution. This design keeps the system realistic for intelligent controllers deployment while preserving safety, explainability, and auditability.

References

[1] Osman Tugay Basaran and Falko Dressler. 2025. XAInomaly: Explainable and Interpretable Deep Contractive Autoencoder for O-RAN Traffic Anomaly Detection. *Computer Networks* 261 (April 2025), 111145. doi:10.1016/j.comnet.2025.111145

[2] Osman Tugay Basaran, Martin Maier, and Falko Dressler. 2026. BRAIN: Bayesian Reasoning via Active Inference for Explainable Mobile Networks. In *45th IEEE International Conference on Computer Communications (INFOCOM 2026), 1st IEEE Workshop on AI Native Distributed Intelligence for 6G Networks (6G AI-RAN 2026)*. IEEE, Tokyo, Japan.

[3] NVIDIA. 2025. *NVIDIA Spark DGX*. https://docs.nvidia.com/dgx/dgx-spark/release-notes.html

[4] O-RAN SC. 2024. *OSC Near Realtime RIC*. https://wiki.o-ran-sc.org/display/RICP/2022-05-24+Release+E

[5] Michele Polese, Mischa Dohler, Falko Dressler, Melike Erol-Kantarci, Rittwik Jana, Raymond Knopp, and Tommaso Melodia. 2024. Empowering the 6G Cellular Architecture with Open RAN. *IEEE Journal on Selected Areas in Communications* 42, 2 (2 2024), 245–262. doi:10.1109/JSAC.2023.3334610