IEEE Open Journal of the Communications Society

Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version 11 January, 2024.

Digital Object Identifier 10.1109/OJCOMS.2024.011100

Anomaly Detection for Mitigating xApp and E2 Interface Threats in O-RAN Near-RT RIC

CHENG-FENG HUNG[1] (Graduate Student Member, IEEE), CHI-HENG TSENG[1], AND SHIN-MING CHENG[1] (Member, IEEE)

[1]National Taiwan University of Science and Technology, Taipei, Taiwan

CORRESPONDING AUTHOR: C.-F. Hung (e-mail: D10915002@mail.ntust.edu.tw)

This work was partially supported by the National Science and Technology Council (NSTC), Taiwan, under Grant 111-2221-E-011-067-MY3, 113-2221-E-011-157-MY3, and 113-2923-E-011-005-MY2.

ABSTRACT As 5G networks advance, the Open Radio Access Network (O-RAN) is crucial in enabling openness and fostering collaboration across the telecom industry. O-RAN enhances flexibility, scalability, and interoperability through open interfaces, reducing dependence on a single vendor and promoting interoperability among vendors and solutions. The Near-Real-Time Radio Intelligent Controller (Near-RT RIC) is crucial for optimizing network resources and improving user experience. However, the openness of O-RAN also introduces security challenges, particularly from third-party developed xApps and E2 nodes that may exploit vulnerabilities to launch attacks. This paper proposes an anomaly traffic detector to protect the Near-RT RIC from threats on the E2 interface. The anomaly traffic detector verifies the legality of signaling through an internal state machine analysis module and checks packet fields through a conformance check module while monitoring network traffic in real time to detect and mitigate Denial of Service attacks. Additionally, we designed a fuzzer to simulate random attacks, testing the capability of the anomaly traffic detector. The anomaly traffic detector not only successfully passes the test cases highlighted in the O-RAN Security Test Specifications, effectively detecting unauthorized traffic and signaling, but also identifies real-world vulnerability exploits, including CVE-2023-40997, CVE-2023-40998, CVE-2023-41627, and CVE-2023-41628, thereby significantly enhancing the security of the Near-RT RIC.

INDEX TERMS Anomaly Detection, E2 Node, E2 Interface, O-RAN security, xApp

I. INTRODUCTION

THE Open Radio Access Network (O-RAN) accelerates 5G evolution by enabling open and flexible network solutions. O-RAN achieves intelligent management and dynamic real-time configuration through the RAN Intelligent Controller (RIC) to meet the needs of different scenarios. It enables seamless integration of equipment developed by various vendors into the O-RAN architecture, fostering interoperability and reducing vendor lock-in. The innovative open interfaces, including O1, E2, A1, and R1 interfaces, enable these devices to communicate, accelerating the deployment of 5G [1]. The Near Real-time RAN Intelligent Controller (Near-RT RIC) is an essential component of this architecture [2], [3]. It dynamically allocates and optimizes network resources, enhancing network efficiency and performance.

Furthermore, the Near-RT RIC provides an open platform for deploying third-party eXtended applications (xApps), enabling developers to optimize O-RAN features like defect detection, resource allocation, and traffic management or create customized solutions. This openness fosters innovation, supports multi-vendor integration, and reduces costs by allowing components such as the O-RAN Central Unit (O-CU) and O-RAN Distributed Unit (O-DU) to be developed by different vendors [4], [5]. xApps rely on real-time data from E2 nodes, including the O-CU and O-DU, accessed via the E2 interface. This data exchange is essential for xApps to perform dynamic network optimization and traffic management [2], [3]. For instance, xApps may allocate resources or resolve congestion by analyzing traffic patterns in near real-time. However, the openness of O-RAN's architecture, while

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

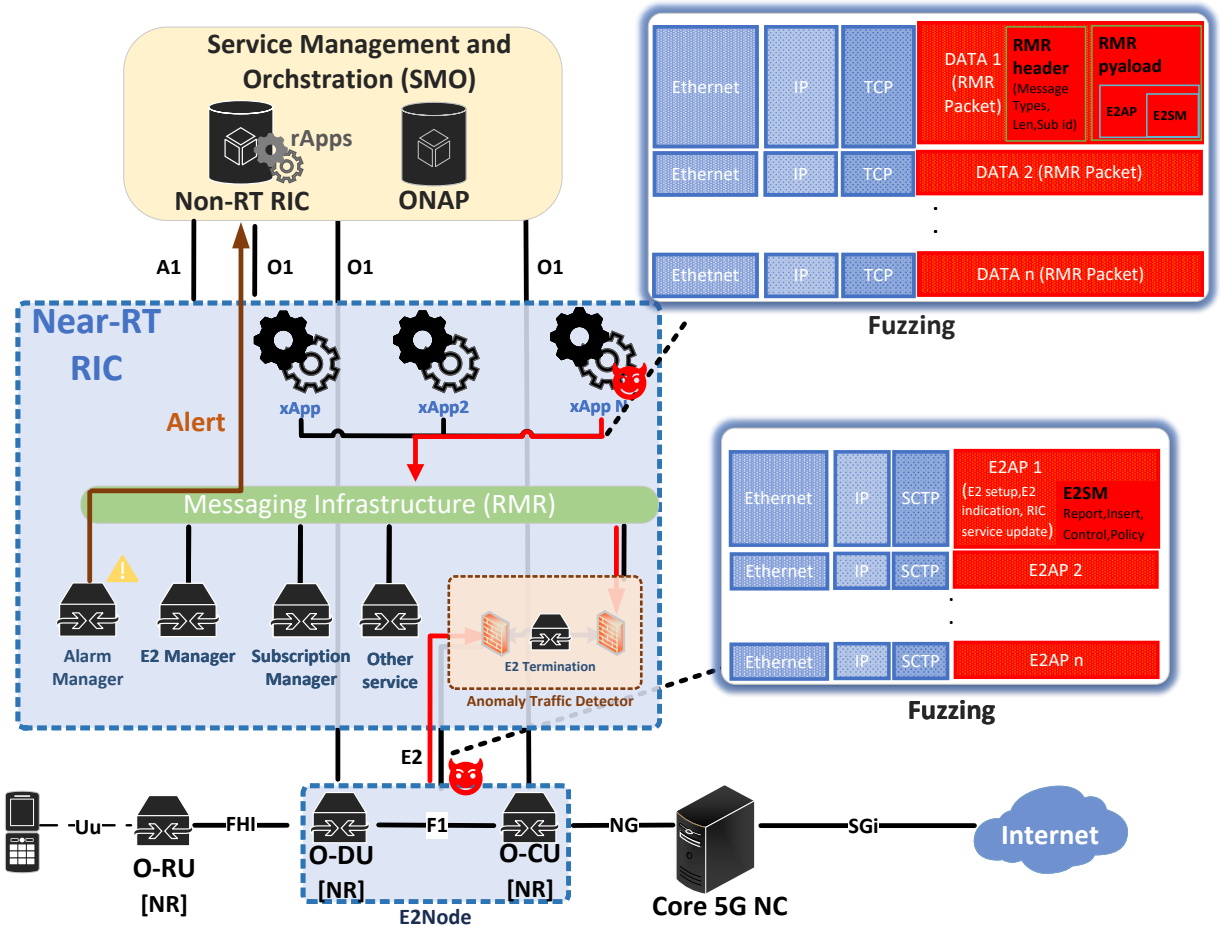

FIGURE 1. **Overview of O-RAN Architecture with Anomaly Traffic Detector Framework**

fostering innovation, also introduces potential security challenges, particularly in areas such as access control and data transmission security [6]–[8]. For instance, vulnerabilities in the E2 interface could be exploited to inject malicious traffic or disrupt critical communication flows, directly impacting the reliability of xApps and the Near-RT RIC.

xApps functionality is integrally related to E2 node data via the E2 interface. This data allows for network optimization, defect detection, and traffic control. Any E2 node vulnerability affects xApps' operational integrity, emphasizing their reliance on the O-RAN ecosystem. From an attacker's perspective, this interdependence broadens the attack surface. Infected E2 nodes may inject false or malicious data, resulting in improper optimization or reduced network performance. Similarly, attackers targeting xApps may use E2 node access to alter network operations or introduce harmful traffic [9]. Malicious or affected xApps may leverage E2 node data to disrupt the Near-RT RIC or degrade network performance [2], [3], [10]–[12]. E2 nodes, which third parties frequently produce, might offer security issues if not correctly implemented or fail to meet specifications. They could send unauthorized traffic to the Near-RT RIC, disrupting its functioning [6], [13]. Attackers may utilize vulnerabilities to inject malicious traffic, make E2 interface disconnections, or interrupt essential services [8], [14], [15]. Several CVEs, including CVE-2023-40997, CVE-2023-40998, CVE-2023-41627, and CVE-2023-41628, are potential points of vulnerability. Monitoring traffic between E2 nodes, the Near-RT RIC, and xApps is essential to ensure robust security. This enables the detection and mitigation of real-time threats, helping to safeguard the integrity and functionality of O-RAN deployments.

In this paper, we present an innovative anomaly traffic detector to safeguard the Near-RT RIC against potential threats from the E2 interface and xApps. This anomaly traffic detector enables real-time identification and alerting of potential security risks, ensuring the stability and security of the system. By integrating the anomaly traffic detector into the Near-RT RIC, we utilize an internal state machine to analyze E2AP protocol communications between xApps/E2 nodes and the Near-RT RIC, identifying non-compliant behaviors. Additionally, the anomaly traffic detector monitors network connectivity changes in real time, providing immediate alerts in cases of abnormally high packet rates, unexpected connection interruptions, or Denial of Service (DoS) attacks, empowering system administrators to take swift action. To

evaluate the effectiveness of our anomaly traffic detector, we introduced fuzz testing [16] and designed a fuzzer that inputs various abnormal or unreasonable data into the system. This includes unreasonable packet format modifications, random packet sequence adjustments, and error injections into packet content. We evaluate our detector's capabilities in handling such scenarios by simulating potential system failures resulting from communication errors or attacks. As demonstrated by experimental results, the anomaly traffic detector can effectively detect and mitigate anomaly attacks and DoS attacks from xApps and E2 nodes. The transmission delay is approximately 105ms, which satisfies the strict requirement of 1s 10ms in the near-RT RIC specification of O-RAN.

The structure of this paper is as follows: Section II introduces the main components of O-RAN; Section III discusses E2 nodes and potential threats on Near-RT RIC; Section IV presents the design of the anomaly traffic detector; Section V explains the comprehensive experimental environment setup and evaluates the feasibility, performance, and experimental results of the framework. Finally, Section VI concludes the paper.

II. BACKGROUND

In this section, we introduce the critical components and interfaces of O-RAN through Fig. 1, including the SMO, Near-RT RIC, E2 node, and the E2 and O1 interfaces.

1) SERVICE MANAGEMENT AND ORCHESTRATION

Service Management and Orchestration (SMO) is the top-level component in the O-RAN architecture. It centrally manages and orchestrates various elements of the O-RAN architecture through the O1 interface, incorporating AI/ML technologies for dynamic resource allocation, network performance optimization, and energy management. Additionally, SMO supports fault management; when a subordinate component fails, it can receive alerts from the component and notify the relevant management personnel [2].

2) NEAR-RT RIC

The Near-RT RIC, a core part of the O-RAN architecture, is adept at managing critical functions such as network slicing, spectrum sharing, radio resource management, and QoS (Quality of Service) control. Its standout feature is its ability to perform these tasks with ultra-low latency, typically ranging from ten milliseconds to one second. The Near-RT RIC supports open, standardized interfaces to simplify network deployment and operations. However, this also introduces corresponding security vulnerabilities. The following introduces the Near-RT RIC's critical components, functionalities, and associated security risks.

- **RIC Message Router (RMR)** The Near-RT RIC, as defined by the O-RAN Alliance, employs the RIC Message Router (RMR) library as the central messaging infrastructure to facilitate low-latency communication between its internal components and xApps [17]. The RMR's routing table facilitates direct communication between components, as shown in Fig. 1. However, current RMR transmissions are not encrypted, leaving the system vulnerable to security threats. Recent research has uncovered multiple RMR-related vulnerabilities, significantly impacting the availability of critical components [8], [14]. Therefore, it is imperative to implement enhanced security mechanisms to safeguard the integrity and confidentiality of these communications.
- **E2 termination** The E2 termination primarily acts as the endpoint for message exchange over the E2 interface. It handles control signaling and policy indications sent from the Near-RT RIC or E2 nodes, as shown in Fig. 1. These messages primarily manage and optimize network performance, such as dynamically allocating network resources and controlling traffic flows. The E2 termination plays a critical role in forwarding these messages. If it is attacked, the entire system's operation may be compromised. Therefore, it is of paramount importance to protect the E2 termination from attacks.
- **E2 Manager** The E2 Manager is a critical component responsible for managing the E2 interface within the O-RAN framework. It efficiently coordinates and manages connections to E2 nodes, monitors communications between nodes, and processes packets from various nodes. These packets may contain multiple types of information, such as configuration settings, status updates, and control commands. The E2 Manager must respond appropriately based on the packet type to ensure the proper operation of E2 processes. In addition to managing communication processes, the E2 Manager provides relevant APIs that allow xApps to call and retrieve necessary E2 node information.
- **xApp** xApps, applications deployed on the Near-RT RIC, offer numerous benefits. They enable third-party vendors to provide specific network functions, services, and real-time monitoring and analysis, thereby enhancing the capabilities of the RIC. However, it's important to note that their development and deployment by third-party vendors also introduce security issues and risks [2], [13]. This emphasizes the need for vigilance in managing xApps. If xApps contain security weaknesses or vulnerabilities, or if malicious xApps are disguised as legitimate ones, the potential impacts on the RIC could be severe [8], [14].
- **Alarm manager** The Alarm Manager is an essential component of the Near-RT RIC, dedicated to efficiently managing alarms and disseminating alarm notifications from other Near-RT RIC components, such as xApps. It plays a crucial role in forwarding these alarms through the O1 interface to the SMO framework, facilitating prompt responses from network administrators [1].

3) E2 NODE

In the O-RAN architecture, an E2 node represents a decoupled O-CU and O-DU of a traditional base station. The E2 node is linked to the Near-RT RIC through the E2 interface, empowering the Near-RT RIC to make real-time decisions for enhancing network performance and resource management [8]. This connection also fosters the creation of solutions by various vendors in adherence to O-RAN Alliance standards, thus integrating into the O-RAN architecture and reducing dependency on specific vendors. Nevertheless, it is critical to note that attackers could exploit incomplete development or vulnerabilities in the O-CU and O-DU. These security risks could significantly disrupt the Near-RT RIC's operations, emphasizing the urgency of addressing these concerns.

- **O-CU** The O-CU is responsible for traditional base station functionalities, including Radio Resource Control (RRC), Service Data Adaptation Protocol (SDAP), and Packet Data Convergence Protocol (PDCP). It manages critical control strategies such as user access, mobility, and session management. Through the E2 interface, the O-CU communicates with the Near-RT RIC, receiving instructions to optimize data flow processing and scheduling. Its connection with the SMO via the O1 interface is crucial, as it reports its operational status to support intelligent resource allocation and management across the network, thereby enhancing network efficiency [2].
- **O-DU** The O-DU is a critical, logical node in wireless communication networks responsible for managing functionalities across various layers, including RLC (Radio Link Control), MAC (Medium Access Control), and PHY (Physical Layer)-High. It handles real-time radio signal processing near the O-RU, such as modulation, demodulation, encoding, and decoding. The O-DU's ability to communicate with the Near-RT RIC via the E2 interface and connect with the SMO through the O1 interface for policy and management signal reception ensures its seamless connectivity and the ability to make real-time adjustments to its operations.

4) E2 INTERFACE

The E2 interface, a critical component of the O-RAN architecture, connects the Near-RT RIC to the RAN elements, specifically the O-CU and O-DU. This interface enables the Near-RT RIC to communicate with O-CU and O-DU from various vendors, simplifying radio resource management and network optimization. The E2 interface generates a low-latency closed-loop required for real-time network status and component information collection. However, the risk of non-compliance with the E2 interface protocol specifications is high. It may pose security risks, highlighting the significance of following standards for network security [8].

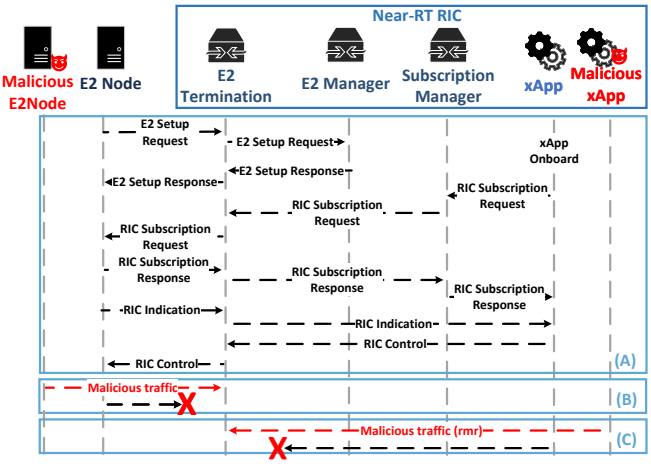

FIGURE 2. The E2 communication process between xApp and E2 node, and the possible attack scenarios involved.

5) O1 INTERFACE

The O1 interface is a communication bridge between the SMO and the O-CU, O-DU, O-RU, and Near-RT RIC. It is responsible for essential functions such as monitoring, configuration, reporting, and fault management to ensure the stable operation of the network. Specifically, the fault management function of the O1 interface is proactive and promptly notifies administrators when network components encounter issues. This proactive approach enables administrators to diagnose and resolve faults promptly, ensuring network continuity and service quality. Additionally, the SMO can dynamically adjust component configurations through the O1 interface to meet the requirements of different scenarios [1].

III. ABNORMAL BEHAVIOR IN NEAR-RT RIC AND E2 NODES

We first explore the threats associated with Near-RT RIC and describe the pivotal role of E2 nodes in establishing the E2 interface through the E2 termination on Near-RT RIC and interacting with xApps. Next, we introduce the threat model and explain how attackers can exploit existing vulnerabilities and weaknesses to affect the operation of the Near-RT RIC.

A. SECURITY THREATS FACED BY NEAR-RT RIC

The deployment method of xApp, which allows third-party development, can lead to the deployment of xApps with vulnerabilities or malicious xApps onto the Near-RT RIC [2], [6], [8]–[14]. This can result in unauthorized access to other services on the Near-RT RIC and the potential for sending malicious signaling, which can disrupt other services [14]. Attackers may exploit vulnerabilities or weaknesses in E2 nodes, or even E2 nodes that have already been compromised, to modify packet field values, sequences, etc., and send these malicious signals to the Near-RT RIC [6], [13], [18]. The consequences can be severe, including

TABLE 1. Analysis Table of Attack Mapping for O-RAN WG11 Threat Models and Related Research. The symbols in the table are represented as follows: Resolved " + ", Mentioned " ◯ ", Partially Mentioned " V " and Unmentioned " * "

Related Research/Threat Type	Deploying Malicious xApp	E2 Node Security Risks	xApp Privilege Threats	E2 Threats	DoS Attack
[13] T-NEAR-RT-01	*	*	◯	*	*
[13] T-NEAR-RT-02	◯	*	◯	*	*
[13] T-NEAR-RT-03	*	*	◯	*	*
[13] T-NEAR-RT-04	*	*	◯	*	*
[13] T-xApp-01	◯	*	*	*	◯
[13] T-xApp-02	◯	*	*	*	◯
[13] T-xApp-03	◯	*	*	*	*
[13] T-E2-01	*	◯	*	◯	*
[13] T-E2-02	*	◯	*	◯	*
[13] T-E2-03	*	◯	*	◯	◯
[13] T-O-RAN-01	◯	◯	◯	◯	*
[13] T-O-RAN-09	*	*	◯	◯	◯
Polese et al. [2]	◯	*	◯	◯	*
Liyanage et al. [6]	◯	◯	◯	◯	*
Atalay et al. [10]	◯	*	◯	◯	*
Groen et al. [11]	◯	*	V	V	*
Sapavath et al. [12]	◯	*	◯	◯	*
El Houda et al. [9]	◯	*	V	*	V
El Houda et al. [19]	*	*	*	*	*
Moudoud et al. [5]	*	*	V	*	*
Tseng et al. [14]	◯	*	◯	*	*
Hung et al. [8]	◯	*	◯	◯	◯
This paper	◯	◯	+	+	+

E2 termination crashes, which can significantly disrupt the operation of the RAN, and the potential for DoS attacks [8].

Many academics have attempted to address these problems. [11] partially solves xApp privilege threats and E2 threats because they implement IPsec on the E2 interface, which can indeed prevent man-in-the-middle attacks. However, if there are weaknesses and vulnerabilities in the development of xApp or E2 nodes, they will still cause threats. [9] introduces TrustORAN, a blockchain-based Zero-Trust Framework that enables xApp verification and secure access control, effectively preventing malicious xApps from unauthorized API access or sending illegitimate signaling. However, as the signaling procedures of xApps have not yet undergone comprehensive validation, attackers may exploit existing permissions to send signaling sequences in an incorrect order, disrupting the operation of the Near-RT RIC and potentially causing DoS attacks. [5] introduces a zero-trust framework with advanced machine learning to enhance O-RAN network security and performance. It prevents unauthorized API use and illegal signaling transmission by malicious xApps or E2 nodes. However, it doesn't address packet order detection, potentially allowing attackers to exploit privileges by sending incorrectly sequenced packets, impacting Near-RT RIC operations. We map the threats and attacks discussed above into O-RAN WG11 [13] threatmodel and collate and analyze related studies through Table 1 for comparison.

B. INTERACTION BETWEEN E2 NODE AND XAPP

The E2 node communicates with the xApp through a subscription mechanism. The xApp sends *subscription requests* to the E2 node to receive real-time information for decision-making analysis. Additionally, the xApp can control the radio bearer, resource allocation, and connected mode mobility of the E2 node [20]. As shown in Fig. 2 (A), the E2 node initiates communication by sending an *E2 Setup Request* to the Near-RT RIC's E2 Termination to establish an E2 connection. The E2 manager processes the *E2 Setup Request* and responds with an *E2 Setup Response*, finalizing the E2 connection setup. Once the xApp is deployed, it sends an *RIC Subscription Request*, handled by the Subscription Manager and relayed to the E2 node through E2 Termination. Subsequently, the E2 node provides an *RIC Subscription Response*, completing the subscription process and allowing communication between the E2 node and the xApp.

It is critical to highlight the importance of E2 Termination in the approach mentioned above. It sends numerous signaling messages and interacts with third-party components. However, this exposes E2 Termination to potential risks. If the E2 node and xApp do not adhere to compliance rules, the availability of Near-RT RIC may be threatened, making E2 Termination the primary cause of concern. As shown in Fig. 2 (B), the E2 node might intentionally send malicious traffic to affect E2 Termination, thereby hindering

the regular connection of other components. Similarly, Fig. 2 (C) illustrates that xApp could send malicious traffic to E2 Termination via RMR, causing it to crash. Therefore, protecting E2 Termination can significantly enhance the security of Near-RT RIC. The threat model is described below:

- **Threat Description:** E2 nodes and xApps can be developed and deployed by third-party vendors in the Near-RT RIC [2]. However, if E2 nodes are not produced in compliance with standards, they may generate abnormal signaling or fail to handle unexpected scenarios correctly, leading to service disruptions [6], [13]. Additionally, there is no apparent security analysis mechanism for xApps before deployment, leaving room for potential vulnerabilities or security threats during development [8]. Furthermore, modifying and deploying open-source xApp projects from the internet could unintentionally introduce malicious xApps, allowing attackers to inject harmful signaling and traffic, disrupt the regular operation of the Near-RT RIC, and even cause a complete collapse of E2 nodes [8], [9], [14].
- **Threat Source:** Internal or external attackers
- **Attackers' Capabilities:** Attackers possess the capability to develop xApps with inherent vulnerabilities and weaknesses, which can be exploited by developers or directly deployed within the network [9]. The lack of a standardized security analysis mechanism for xApps before deployment creates an opportunity for attackers to introduce malicious xApps, unintentionally or otherwise. Additionally, self-developed xApps may contain design flaws or insecure coding practices that attackers can exploit through various means, such as injecting unauthorized signaling, manipulating updates, exploiting insecure interfaces, or injecting backdoors. These actions can lead to unauthorized access, service disruptions, or compromise between E2 nodes and the Near-RT RIC [8]. In the case of E2 nodes, attackers may exploit vulnerabilities arising from improper adherence to standards or security guidelines, resulting in abnormal behavior such as sending unauthorized or abnormal signaling that impacts the operation of the Near-RT RIC. Attackers can gain control of E2 nodes to manipulate signaling, introduce malicious updates, or exploit insecure interfaces, compromising the integrity and availability of the Near-RT RIC. They may leverage reverse engineering, leaked credentials, or unauthorized access to E2 node interfaces, using insecure connections to infiltrate and disrupt operations.
- **Identifies Vulnerability:** Lack of Integrity Protection
- **Threat Asset:** Malicious xApps and improperly designed E2 nodes can send unauthorized or harmful traffic to E2 Termination, impacting the operation of other xApps, the Near-RT RIC, and other connected components.
- **Affected Components:** Near-RT RIC, xApp, O-CU, O-DU, O-RAN Radio Unit (O-RU), UE

IV. ANOMALY TRAFFIC DETECTOR DESIGN

We have developed the anomaly traffic detector to monitor and analyze all signaling traffic and behavior between the E2 node, xApp, and Near-RT RIC's E2 Termination. When the anomaly traffic detector identifies abnormal or potentially malicious activities, it blocks the attack and issues a threat alert.

A. SYSTEM ARCHITECTURE OF DETECTOR

This paper introduces the anomaly traffic detector to identify and mitigate threats on E2 nodes and xApps. The anomaly traffic detector comprises four main components: packet parsing, state machine analysis module, conformance check module, and alarm system. As shown in Fig. 3, the xApp connects to the anomaly traffic detector via an RMR endpoint, while the E2 node connects via an SCTP endpoint. First, packets undergo format parsing, including de-encapsulation, field extraction, and effective decoding of packet data for subsequent analysis. The state machine analysis module extracts the current state value from the parsed packet and secures it using the state machine. Packets that pass the check are sent to the conformance check module for additional processing. While anomalous packets are blocked, an alarm is generated and sent to the SMO. Section IV B provides a comprehensive technique description. The conformance check module verifies packet format compliance and detects malicious activity, such as flood DoS attacks. Conformance checks assure the validity of packet formats while also successfully identifying and preventing malicious attacks. Finally, when the system identifies specific anomalous behavior, the alarm system promptly alerts the SMO, allowing for a quick response to any threats. The alarm system contains systems for recording, generating, and notifying about abnormal events.

B. STATE MACHINE ANALYSIS MODULE

In the state machine analysis module, our state machine is constructed based on the official O-RAN E2AP specifications [21]. It encompasses all possible E2 node state transitions, E2 node subscription behaviors, and xApp transmission actions. As shown in Fig. 4, when an E2 node is initialized, it sends an *E2 Setup Request* message to the Near-RT RIC to establish an initial connection. Upon receiving and successfully processing this message, the Near-RT RIC responds with an *E2 Setup Response*. At this point, the E2 node transitions from Idle (initial state) to *E2 Setup Response*, indicating that the connection has been successfully established and initialization is complete. The E2 node is then ready to handle subsequent operations. During the subscription signaling process, when an xApp sends an *RIC Subscription Request* message to the E2 node to subscribe to specific information or control data,

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2025.3546760

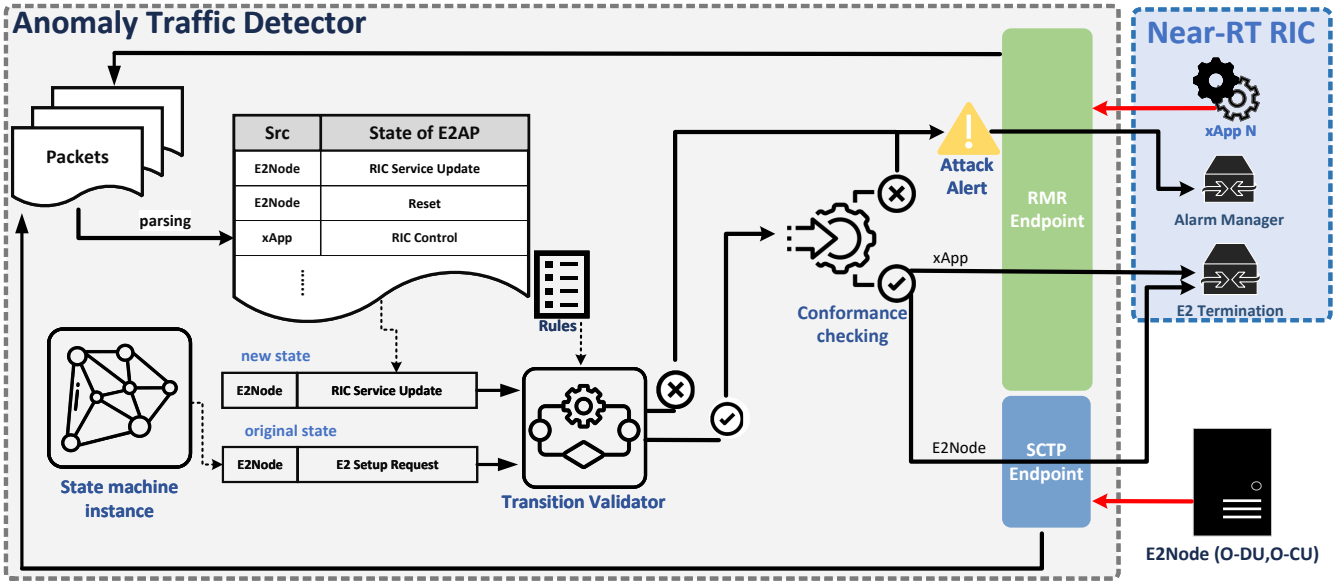

FIGURE 3. **Anomaly traffic detector architecture**

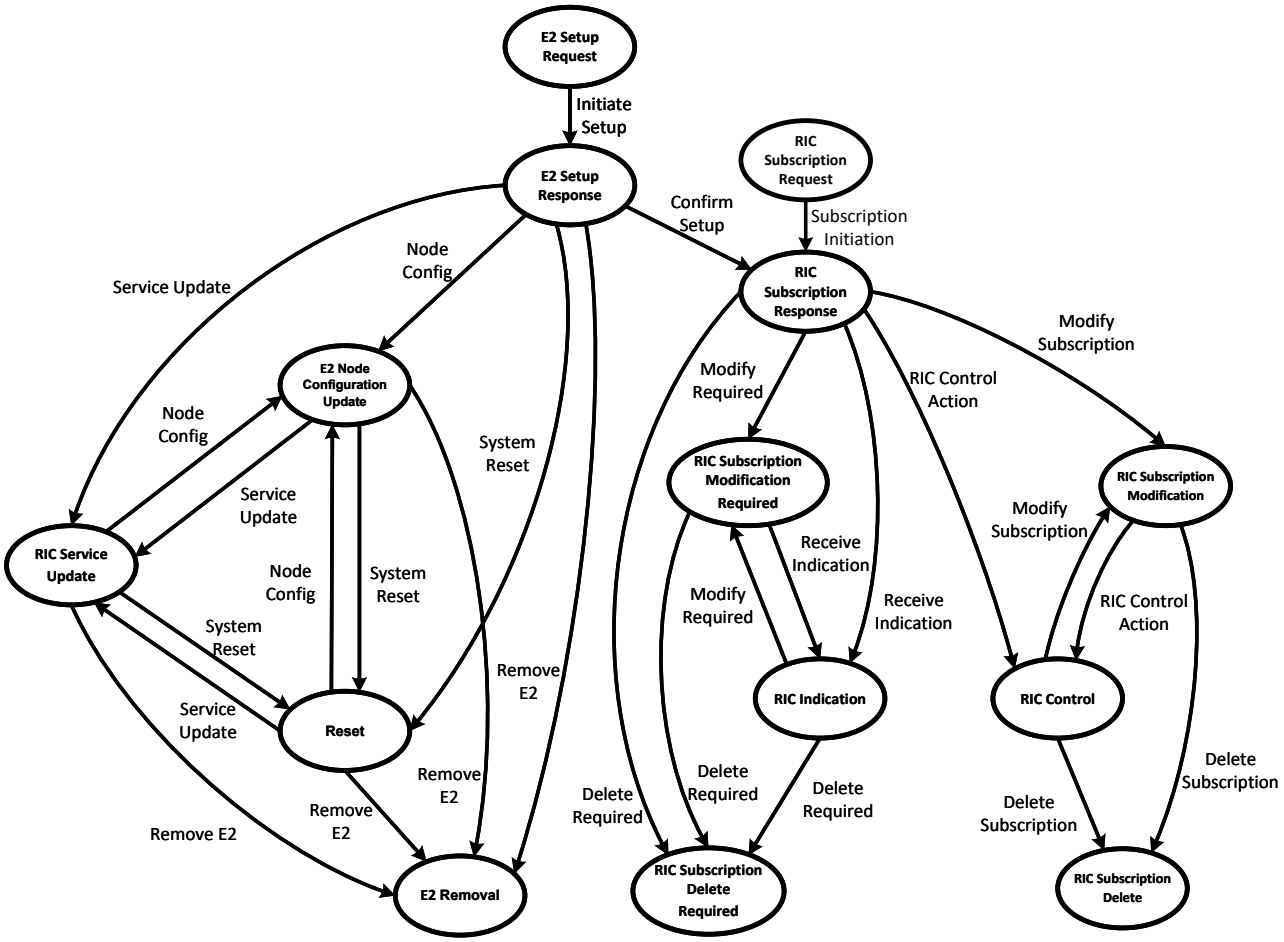

FIGURE 4. **The state machine analysis module**

the E2 node responds with an *RIC Subscription Response*, confirming that the subscription connection has been established. The E2 node's state transitions from *E2 Setup Response* to *RIC Subscription Response*, indicating that the

subscription has been completed. When modifications to an existing subscription are required, the xApp sends an *RIC Subscription Modification* message to the E2 node. Once the E2 node receives and processes this message, its state transitions from *RIC Subscription Response* to *RIC Subscription Modification*, signifying that the subscription has been successfully modified and the system state has been updated accordingly. This unified state machine encompasses all signaling behaviors of E2 nodes and xApps and effectively validates each state transition's legality, ensuring compliance with the O-RAN E2AP specifications [21].

As shown in Fig. 3, the module parses packets from E2 nodes or xApps to track and maintain the current state of each component in the system. Before any state transition occurs, it must pass through a Transition Validator for verification. During this process, the new state is compared with the original state. Our unified state machine covers all possible legal transition paths to ensure the validity of state transitions, as depicted in Fig. 4. These transition rules are derived from the E2AP specifications [21], explicitly defining the legal transition methods for each state. The Transition Validator automatically checks each state transition against these rules to ensure strict compliance with the predefined specifications. Suppose the new state does not align with the descriptions in the unified state machine or follows an illegal transition path. In that case, the system identifies it as an anomaly, triggers an alert, and immediately marks the state invalid. Packets that pass the Transition Validator's inspection will be forwarded to the next stage for conformance checking. Since these state transitions involve a limited set of states and explicit transitions between them, we use Finite State Machines (FSM) to handle this problem rather than machine learning methods, ensuring errors are detected more quickly and accurately.

C. CONFORMANCE CHECK MODULE

Packets inspected by the state machine analysis module are forwarded to the conformance check module for format verification, preventing threats caused by illegal packet formats. The steps for packet inspection are as follows:

1) Parsing Packet Contents: Check the packet header and payload to ensure each part conforms to the specified format.
2) Verifying Data Types: Confirm whether the data types of each field in the packet, such as integers, floats, strings, etc., are correct.
3) Checking Packet Length: Ensure that the packet's total length matches its content to prevent data truncation or exceeding expected limits.
4) Traffic Monitoring: Real-time traffic monitoring among xApps, E2 nodes, and E2 Terminations detects anomalies to prevent DoS and DDoS attacks.
5) Traffic Pattern Analysis: Potential attack behaviors are identified by analyzing historical data and traffic patterns to prevent them from impacting the system.

V. EVALUATION

A. EXPERIMENTAL SETUP

1) Hardware and Software Environment

We utilized a server-grade computer to set up the experimental environment with the following specifications: Intel i7-13700 16-core CPU, 64 GB RAM, NVIDIA RTX 3060 Ti, 1.5TB SSD, and Ubuntu 20.04. For the software, we used the I-Release version provided by the O-RAN Software Community (OSC) to set up components in O-RAN, combined with srsRAN for the 4G core network and OpenAirInterface (OAI) for the 5G core network.

2) Anomaly Traffic Detector

The Anomaly Traffic Detector must be built within a container based on Ubuntu 20.04. A C language program is included inside this container to provide detection functionality. To ensure the container can effectively handle E2 protocol packets, the asn1c library must be installed. This library allows for converting ASN.1 format into C language structures, thereby supporting parsing and encoding operations for E2 protocol packets. Additionally, when deploying the Anomaly Traffic Detector on Near-RT RIC, it is essential to configure and adjust the Kubernetes (K8s) architecture. First, an appropriate Helm Chart or K8s YAML configuration file must be defined for the container, including Deployment, Service, and ConfigMap settings to guarantee smooth deployment and intercommunication with other components. Furthermore, the E2 node must be configured with the correct endpoint, such as defining an appropriate NodePort, to ensure that packets from the E2 node are accurately routed to the Anomaly Traffic Detector.

3) Fuzzer Design

To simulate malicious attacker behavior, we developed a fuzz testing tool (fuzzer) by modifying the E2 Simulator provided by OSC, deployed on malicious xApps and E2 nodes to simulate attacker actions. This tool is designed to automatically generate diverse and randomized attack packets to evaluate the security and stability of the system under various attack conditions. The packet generation process of the fuzzer consists of the following steps:

1) **Protocol Selection:** Selective base protocol, such as RMR or E2AP packets.
2) **Application of Mutation Rules:** Packets are flexibly mutated using predefined or customized rules. For E2AP, mutations may involve modifying critical header parameters (e.g., Procedure Code or message types) or inserting invalid identifiers (IDs) to disrupt protocol parsing. For RMR, mutations may include altering message routing fields to cause routing errors, unlawfully varying payload sizes (e.g., exceeding reasonable limits), or randomly generating invalid routing messages.

3) **Generation of Mutated Packets:** Randomized mutations are applied to produce abnormal packets simulating diverse attack behaviors. The tool can adjust message lengths (0 to 65,535 bytes), insert special characters, or fragment legitimate packets and recombine them to increase attack coverage. Packet lengths and distributions are determined by the attack scenario, categorized as follows:
   • Small packets (64 to 256 bytes): Simulate control signaling attacks.
   • Medium packets (256 to 1,024 bytes): Simulate payload attacks.
   • Large packets (above 1,024 bytes): Simulate resource exhaustion attacks.

4) **Packet Transmission:**The generated packets are transmitted to the target system through selected carriers, such as xApps or E2 nodes.

B. IMPLEMENTATION

When packets are transmitted from xApps or E2 nodes to the system, they are first processed by the Anomaly Traffic Detector. This detector, developed in C language, employs socket programming to achieve high-performance packet reception and processing. During system initialization, the module uses the socket() function to create listening sockets and the bind() function to associate them with specific IP addresses and ports, establishing connections for xApps and E2 nodes, respectively. The system utilizes a non-blocking I/O mechanism based on the select() function to support multi-connection reception. For enhanced concurrency, it implements a multi-threaded architecture using the pthread library, where each thread monitors and processes traffic from a specific set of connections. To prevent race conditions among threads, mutex locks protect shared resources and ensure data processing accuracy. Incoming packets undergo a preliminary integrity check, such as length verification and checksum validation, to filter out corrupted packets. Subsequently, packets are decoded using asn1c and analyzed for headers and payloads. During this parsing process, a multi-layer buffer mechanism is designed to prevent performance degradation caused by memory allocation bottlenecks under high-traffic scenarios.

The parsed structured data is forwarded to the State Machine Analysis Module for processing. In this module, the current state of the packet is extracted from the data and passed to the Transition Validation submodule for verification. This submodule relies on pre-defined state transition rules stored in JSON format and employs a high-performance hash table for rapid lookup. The module implements memory preloading to minimize runtime latency, ensuring that all rules are loaded into memory during system startup. For the alarm system, the implementation leverages O-RAN's official C-language libraries. Upon detecting an illegal state transition, a custom notification mechanism is triggered, logging detailed error records to a log file and sending real-time alert messages to the Alarm Manager via the RMR protocol. The Alarm Manager, in turn, notifies the SMO through the O1 interface, prompting operators to take corrective actions.

The packet state transition proceeds to the Conformance Check Module for detailed inspection if it is valid. A custom C program parses the packet header in this module to extract key information, such as protocol version, packet length, and timestamps. The extracted data is compared against the standard formats defined by O-RAN specifications with an integrated regular expression matching module for efficient conformance validation. The Conformance Check Module also includes a real-time traffic monitoring subsystem. This subsystem uses counters and time-window mechanisms to track the number of packets received per second from xApps and E2 nodes. A dynamic thresholding algorithm adaptively adjusts the anomaly detection criteria for traffic patterns. If abnormal traffic is detected—such as packet counts exceeding predefined thresholds—the system triggers defense mechanisms, including rate limiting or temporarily suspending packet reception from the anomalous source. Detailed logs of all detected anomalies are stored in a database for subsequent analysis and to refine defensive strategies.

To simulate the behavior of a malicious xApp or E2 node in a real-world environment, we set up a malicious xApp in the Near-RT RIC using the official deployment method. We also placed the fuzzer on this xApp to send RMR packets, targeting the E2 Termination. Additionally, the fuzzer was set up on a malicious E2 node to send E2AP protocol packets to the same target. While the malicious xApp and E2 node generated traffic, we captured all malicious E2 node traffic (E2AP/SCTP) and malicious xApp RMR traffic (TCP) every 30 minutes. We analyzed each packet's content from the captured pcap files to understand better the interaction behavior and traffic characteristics of both the malicious E2 node and xApp. Furthermore, we implemented an E2SM performance monitoring xApp that periodically polls the E2 node to obtain the communication time between the xApp and the E2 node. This performance monitoring xApp utilizes timestamps to accurately record the arrival time of each data packet, allowing for better tracking and reconstruction of the event timeline. We observed the impact of the traffic detector on the performance of the Near-RT RIC.

We have designed the following research questions to evaluate the effectiveness of the anomaly traffic detector in the O-RAN environment.

- RQ1: Does the anomaly traffic detector effectively detect and block malicious out-of-sequence signaling?
- RQ2: Does the anomaly traffic detector promptly detect and mitigate attacks involving improperly formatted packets and DoS attacks, and does it cause transmission delays?

TABLE 2. Malformed Packet Fields Detection Results

Source	The name of the packet field	Detected
xApp	*procedureCode*	✓
	ricRequestorID	✓
	ricInstanceID	✓
	RANfunctionID	✓
	ricEventTriggerDefinition	✗
	ricActionID	✗
	ricActionType	✗
	ricSubsequentActionType	✗
	ricTimeToWait	✗
	RICcontrolHeader	✓
	RICcontrolMessage	✓
	RICcontrolackRequest	✓
	RICcallProcessID	✓
E2node	*TransactionID*	✓
	globalGNBID	✓
	ranFunctionID	✓
	ranFunctionDefinition	✓
	ranFunctionRevision	✓
	ranFunctionOID	✓
	ricInstanceID	✓
	RICactionID	✓
	RICindicationSN	✓
	RICindicationType	✓
	RICindicationHeader	✓
	RICindicationMessage	✓
	RICcallProcessID	✓
	e2nodeComponentInterfaceType	✓
	e2nodeComponentID	✓
	e2nodeComponentConfiguration	✓
	transportlayeraddress	✓
	transportlayerport	✓

C. STATE MACHINE ANALYSIS MODULE RESULTS

To address RQ1 and demonstrate that the anomaly traffic detector can effectively detect out-of-sequence packets, we successfully utilized the fuzzer's mutation capabilities to trigger two publicly disclosed CVE vulnerabilities (CVE-2023-41628 and CVE-2023-41627) in our experimental environment. When the fuzzer on the E2 node sent packets with illegal sequences to the Near-RT RIC, these packets were relayed to the Near-RT RIC's E2 Termination. This led to the E2 Termination, which lacked an anomaly traffic detector, crashing and resulting in the disconnection of the E2 interface, thereby enabling a DoS attack on the component (CVE-2023-41628). During the assault, latency surged from approximately 100 ms before the attack to infinity, as shown by the brown line in Fig. 5 (A). In an environment with an anomaly traffic detector, the detector identified and blocked the improperly ordered packets through the State Machine Analysis Module before reaching the E2 Termination. This effectively prevented the E2 Termination from crashing. The blue line in Fig. 5 (A) illustrates the process. Although latency slightly increased during the attack, the detector restored it to around 100 ms within about 7ms, stabilizing with minor oscillations afterward.

Furthermore, when a malicious xApp's fuzzer sent spoofed RMR tables to the E2 termination, the lack of an anomaly traffic detector compromised the xApp. This resulted in packets meant for other xApps being redirected to the malicious xApp, causing packet loss and potential leakage of sensitive information (CVE-2023-41627). However, in an environment where an anomaly traffic detector was deployed, it identified the RMR spoofing attack before the xApp's packets reached the E2 termination. The detector discarded the malicious packets and triggered an alert system, preventing the redirection of packets to the malicious xApp. The experimental results show that the anomaly traffic detector can effectively identify and address these two vulnerabilities, with a CVSS score 7.5. Furthermore, the detector successfully passed the TC_LOG_NEAR_RT_RIC test described in the O-RAN Security Test Specifications [22].

D. CONFORMANCE CHECK MODULE ANALYSIS RESULTS AND DETECTOR PERFORMANCE

To address RQ2 and demonstrate that the anomaly traffic detector can effectively detect improperly formatted packets, we successfully used the fuzzer deployed on a malicious xApp to trigger two publicly disclosed vulnerabilities (CVE-2023-40997 and CVE-2023-40998) in our experimental environment. By randomly mutating the headers and payloads of RMR packets within the xApp, the fuzzer disrupted their intended format, generating malicious packets and causing E2 nodes to send incorrectly formatted packets to the Near-RT RIC. Due to the lack of robust validation mechanisms, the malicious xApp could send forged routing table messages to any RMR service, disrupting communication between components. Exploiting this vulnerability, attackers could deliver improperly formatted RMR packets to the E2 Termination. Upon receiving and attempting to parse such packets, an E2 Termination without a deployed anomaly traffic detector would crash, resulting in an E2 interface disconnection and enabling a DoS attack on the component (CVE-2023-40997). However, in environments with an anomaly traffic detector deployed, the detector plays a crucial role. It utilizes the Conformance Check Module to identify and block malformed packets before they reach the E2 Termination, effectively preventing crashes and ensuring the system's security. This process is illustrated in Fig. 6 (A).

A malicious xApp can also transmit packets containing incorrect information to the E2 Termination by modifying the initial four bytes of the packet size to a negative value. As shown in Fig. 6 (B), the value is changed from 0A 09 00 00 to 00 01 00 82. If the decoded result is interpreted as negative, it can trigger a core dump during subsequent memory allocation operations. As a result, the E2 Termination experiences a crash, especially in the absence of an anomaly traffic detector. This vulnerability allows for a DoS attack

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2025.3546760

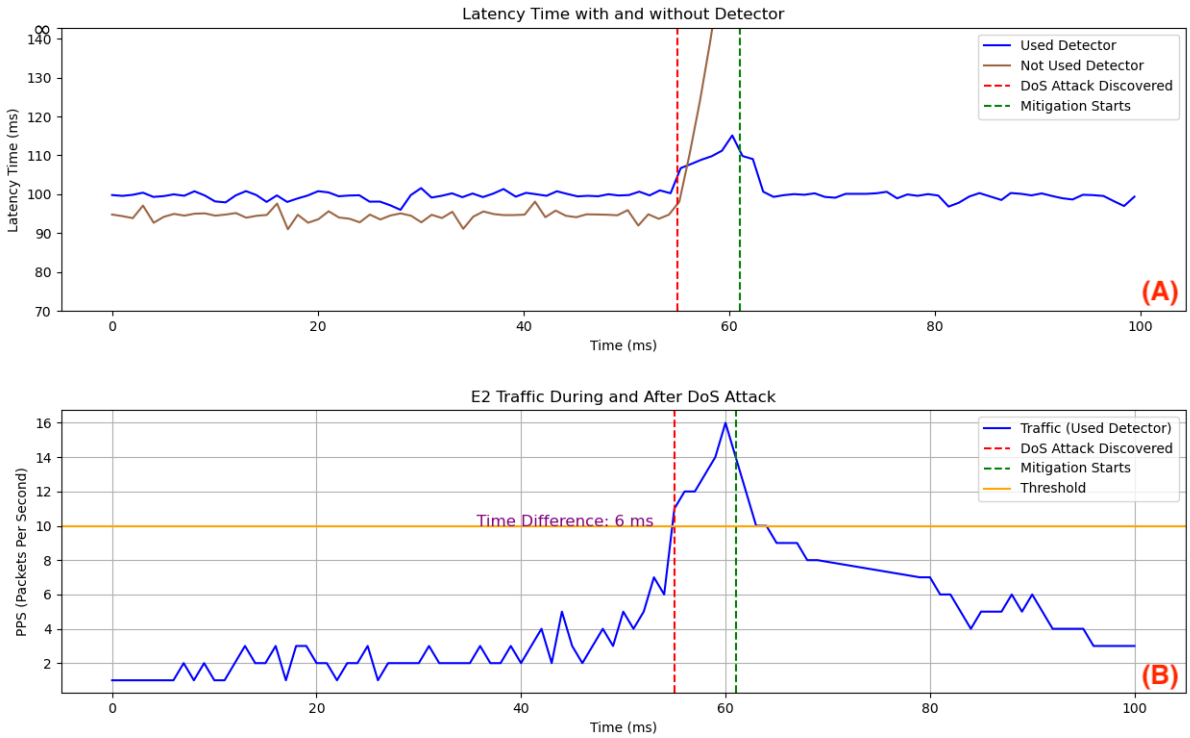

FIGURE 5. **Latency Time with and without Detector & E2 Traffic During and After DoS Attack**

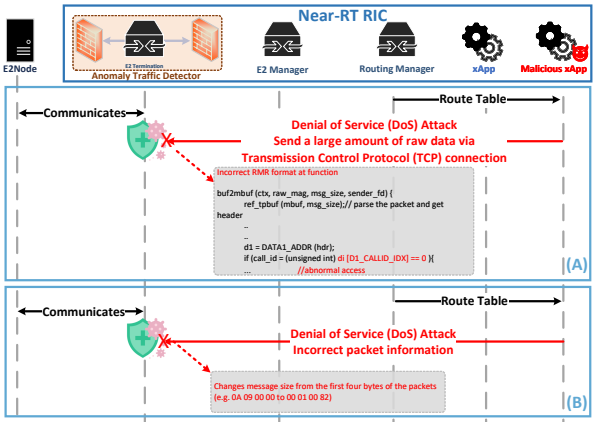

FIGURE 6. **Mitigation of Malicious xApp Non-Compliant Packet Attacks Using Anomaly Traffic Detector**

on the component (CVE-2023-40998). Its function becomes critical in environments where an anomaly traffic detector is present. The detector utilizes the Conformance Check Module to identify and block malformed packets before they reach the E2 Termination, illustrating the system's ability to detect and prevent such attacks.

Table 2 summarizes the results of packet field anomalies detected by our anomaly traffic detector. The detector effectively identifies most field format errors in packets sent from E2 nodes and xApps through automated format checks, allowing timely detection of such issues. Notably, fields such as *ricActionID, ricActionType, RicSubsequentActionType, and ricTimeToWait*, which are included in the *RIC subscription request* packets sent by xApps, are not directly processed by the anomaly traffic detector. Instead, these packets are forwarded to the subscription manager, as illustrated in Fig. 2. Upon receiving these packets, the subscription manager performs detailed parsing to ensure they adhere to predefined specifications before proceeding with subsequent operations. This process prevents the detector from identifying errors in these fields but mitigates the direct threat to the E2 Termination. The experimental results demonstrate that the Anomaly Traffic Detector effectively detects and mitigates two vulnerabilities with CVSS scores up to 7.5. Additionally, our anomaly traffic detector successfully passes TC_INPUT_VALIDATION_ERR_HANDL_NEAR_RT_RIC and TC_LOG_NEAR_RT_RIC tests specified in the O-RAN Security Test Specifications [22].

To defend against malicious DoS attacks, the anomaly traffic detector monitors the current packet flow and activates the mitigation mechanism when a threat is identified. Based on our previous stress tests, the system becomes non-operational when it receives 15 packets per second (PPS). To ensure preventive measures, we have set the threshold to 10 PPS. As shown in Fig. 5 (B), our experiment proves that when the traffic exceeds the preset threshold of 10 PPS, the anomaly traffic detector can start to operate within 7ms to block and relieve the attack, and the overall traffic will gradually return to normal in about 61ms.

Regarding performance, Fig. 5 (A) compares the transmission latency under two scenarios: with and without deploying the anomaly traffic detector. The experimental results demonstrate that, in the absence of an attack, deploying the anomaly traffic detector increases the transmission latency by only 3.29%, with the maximum latency reaching 105 ms, which complies with the official transmission latency requirements (10 ms to 1 s) between the Near-RT RIC and the E2 node. When an attack occurs without deploying the anomaly traffic detector, the latency increases rapidly at approximately 55 ms, eventually becoming infinite, disrupting the regular operation of the Near-RT RIC. In contrast, with the anomaly traffic detector deployed, the system can initiate mitigation measures quickly. The maximum latency reaches 121 ms but begins to recover at approximately 62 ms. During this period, malicious traffic is blocked, triggering the alert system. The transmission latency gradually returns to normal, stabilizing at approximately 105 ms with minor oscillations. The anomaly traffic detector not only effectively mitigates attacks originating from xApps and E2 nodes while meeting the transmission latency requirements between system components but also successfully passes the O-RAN security test specifications emphasized in TC_DoS_RECOV_NEAR_RT_RIC and TC_Robustness_DDoS [22], further demonstrating its security and reliability.

VI. CONCLUSION

This paper proposes the anomaly traffic detector designed to protect the operation of the Near-RT RIC on O-RAN. It detects abnormal signaling and blocks malicious traffic from compromised E2 nodes and xApps. Additionally, it can mitigate and prevent DoS attacks while satisfying O-RAN's official latency standards. Our anomaly traffic detector has successfully passed critical tests outlined in the O-RAN Security Test Specifications, including TC_INPUT_VALIDATION_ERR_HANDL_NEAR_RT_RIC, TC_LOG_NEAR_RT_RIC, TC_Robustness_DDoS, and TC_DoS_RECOV_NEAR_RT_RIC. The anomaly traffic detector is not just a tool but a crucial internal defense component for the Near-RT RIC, significantly enhancing the overall security of O-RAN. Its implementation is paramount in ensuring the the overall transmission security and reliability of O-RAN.

VII. FUTURE DIRECTIONS

To address the evolving cybersecurity challenges in O-RAN, we plan to leverage the advantages of the SMO within the O-RAN architecture by continuously collecting component states and related information to ensure the stable operation of the RAN. As proposed in [19], federated learning (FL) has been shown to enhance O-RAN's ability to mitigate jamming attacks by enabling distributed agents to collaboratively train local models at the Non-RT RIC and aggregate them into a global model. Inspired by [19], we plan to introduce an FL framework to coordinate Near-RT RICs across different domains, enabling them to share information on malicious attacks and events. The Non-RT RIC will train and update a new global model to the Near-RT RICs, significantly improving O-RAN's security defenses. Regarding detection model design, we aim to develop fast and accurate models for detecting malicious traffic and achieving robust capabilities to resist backdoor and adversarial attacks.

REFERENCES

[1] O-RAN ALLIANCE Working Group 1, "O-RAN architecture description 12.0," O-RAN ALLIANCE," Technical Specification, June 2024. [Online]. Available: https://orandownloadsweb.azurewebsites.net/specifications

[2] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges," *IEEE Commun. Surv. Tutor.*, vol. 25, no. 2, pp. 1376–1411, Jan. 2023, doi: https://doi.org/10.1109/COMST.2023.3239220.

[3] A. S. Abdalla, P. S. Upadhyaya, V. K. Shah, and V. Marojevic, "Toward next generation open radio access networks: What O-RAN can and cannot do!" *IEEE Netw.*, vol. 36, no. 6, pp. 206–213, Nov./Dec. 2022, doi: https://doi.org/10.1109/MNET.108.2100659.

[4] K. Alam, M. A. Habibi, M. Tammen, D. Krummacker, W. Saad, M. D. Renzo, T. Melodia, X. Costa-Pérez, M. Debbah, A. Dutta, and H. D. Schotten, "A comprehensive tutorial and survey of O-RAN: Exploring slicing-aware architecture, deployment options, use cases, and challenges," *arXiv preprint arXiv:2405.03555*, Oct. 2024, doi:https://doi.org/10.48550/arXiv.2405.03555.

[5] H. Moudoud and S. Cherkaoui, "Enhancing Open RAN security with zero trust and machine learning," in *Proc. IEEE GLOBECOM 2023*, Kuala Lumpur, Malaysia, Dec. 2023, pp. 2772–2777, doi:https://doi.org/10.1109/GLOBECOM54140.2023.10437043.

[6] M. Liyanage, A. Braeken, S. Shahabuddin, and P. Ranaweera, "Open RAN security: Challenges and opportunities," *J. Netw. Comput. Appl.*, vol. 214, p. 103621, May 2023, doi: https://doi.org/10.1016/j.jnca.2023.103621.

[7] A. S. Abdalla and V. Marojevic, "End-to-end O-RAN security architecture, threat surface, coverage, and the case of the open fronthaul," *IEEE Commun. Mag.*, vol. 8, no. 1, pp. 36–43, Mar. 2024, doi: https://doi.org/10.1109/MCOMSTD.0001.2200047.

[8] C.-F. Hung, Y.-R. Chen, C.-H. Tseng, and S.-M. Cheng, "Security threats to xApps access control and E2 interface in O-RAN," *IEEE Open J. Commun. Society*, vol. 5, pp. 1197–1203, Feb. 2024, doi: https://doi.org/10.1109/OJCOMS.2024.3364840.

[9] Z. A. E. Houda, H. Moudoud, and L. Khoukhi, "Blockchain meets O-RAN: A decentralized zero-trust framework for secure and resilient O-RAN in 6G and beyond," in *Proc. IEEE INFOCOM WKSHPS 2024*, Vancouver, BC, Canada, May 2024, pp. 1–6, doi:https://doi.org/10.1109/INFOCOMWKSHPS61880.2024.10620803.

[10] T. O. Atalay, S. Maitra, D. Stojadinovic, A. Stavrou, and H. Wang, "Securing 5G OpenRAN with a scalable authorization framework for xApps," in *Proc. IEEE INFOCOM 2023*, New York City, NY, USA, May 2023, pp. 1–10, doi: https://doi.org/10.1109/INFOCOM53939.2023.10228961.

[11] J. Groen, S. DOro, U. Demir, L. Bonati, M. Polese, T. Melodia, and K. Chowdhury, "Implementing and evaluating security in O-RAN: Interfaces, intelligence, and platforms," *arXiv preprint arXiv:2304.11125*, Apr. 2023, doi: https://doi.org/10.48550/arXiv.2304.11125.

[12] N. N. Sapavath, B. Kim, K. Chowdhury, and V. K. Shah, "Experimental study of adversarial attacks on ML-based xApps in O-RAN," in *Proc. IEEE GLOBECOM 2023*, Kuala Lumpur, Malaysia, Dec. 2023, pp. 6352–6357, doi: https://doi.org/10.1109/GLOBECOM54140.2023.10437125.

[13] O-RAN ALLIANCE Security Work Group 11, "O-RANsecurity threat modeling and risk assessment 3.0," O-RAN ALLIANCE," Technical Specification, June 2024. [Online]. Available: https://orandownloadsweb.azurewebsites.net/specifications

[14] C.-H. Tseng, C.-F. Hung, B.-K. Hong, and S.-M. Cheng, "On manipulating routing table to realize redirect attacks in O-RAN by malicious xApp," in *Proc. IEEE WPMC 2023*, Nov. 2023, pp. 288–292, doi: https://doi.org/10.1109/WPMC59531.2023.10338835.
[15] D. Mimran, R. Bitton, Y. Kfir, E. Klevansky, O. Brodt, H. Lehmann, Y. Elovici, and A. Shabtai, "Security of open radio access networks," *Comput. Secur.*, vol. 122, p. 102890, Nov. 2022, doi: https://doi.org/10.1016/j.cose.2022.102890.
[16] X. Zhang, C. Zhang, X. Li, Z. Du, Y. Li, Y. Zheng, Y. Li, B. Mao, Y. Liu, and R. H. Deng, "A survey of protocol fuzzing," *arXiv preprint arXiv:2401.01568*, Jan. 2024, doi: https://doi.org/10.48550/arXiv.2401.01568.
[17] O-RAN ALLIANCE Near-real-time RIC and E2 Interface Work Group 3, "O-RAN Near-RT RIC Architecture 6.0," O-RAN ALLIANCE," Technical Specification, June 2024. [Online]. Available: https://orandownloadsweb.azurewebsites.net/specifications
[18] O-RAN ALLIANCE Security Work Group 11, "O-RAN study on security for Near Real Time RIC and xApps 5.0," O-RAN ALLIANCE," Technical Specification, Feb. 2024. [Online]. Available: https://orandownloadsweb.azurewebsites.net/specifications
[19] Z. A. E. Houda, H. Moudoud, and B. Brik, "Federated deep reinforcement learning for efficient jamming attack mitigation in O-RAN," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 7, pp. 9334–9343, July 2024, doi:https://doi.org/10.1109/TVT.2024.3359998.
[20] O-RAN ALLIANCE Near-real-time RIC and E2 Interface Work Group 3, "O-RAN E2 Service Model (E2SM), RAN Control 6.0," O-RAN ALLIANCE," Technical Specification, June 2024. [Online]. Available: https://orandownloadsweb.azurewebsites.net/specifications
[21] O-RAN ALLIANCE Near-real-time RIC and E2 Interface Work Group 3, "O-RAN E2 Application Protocol 5.0," O-RAN ALLIANCE," Technical Specification, Feb. 2024. [Online]. Available: https://orandownloadsweb.azurewebsites.net/specifications
[22] O-RAN ALLIANCE Security Work Group 11, "O-RAN security test specifications 7.0," O-RAN ALLIANCE," Technical Specification, June 2024. [Online]. Available: https://orandownloadsweb.azurewebsites.net/specifications

CHENG-FENG HUNG (Graduate Student Member , IEEE) received his M.E. degree in information technology and applications the college of science and engineering from the National Quemoy University, Kinmen, Taiwan, in 2019. He is currently a Ph.D. candidate in the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei. He visited the Warsaw University of Technology in 2022. His research interests are O-RAN and MEC security in mobile networks. He has received student travel grant in IEEE 2023 WPMC.

CHI-HENG TSENG received his B.S. degree in computer science and information engineering from the National Yunlin University of Science and Technology, Taiwan, in 2022. He is currently studying for a M.S. degree in Computer Science and Information Engineering at the National Taiwan University of Science and Technology, Taiwan, expected to graduate in 2024. His research interest is O-RAN security in mobile networks.

SHIN-MING CHENG (Member, IEEE) received the B.S. and Ph.D. degrees in computer science and information engineering from the National Taiwan University, Taipei, Taiwan, in 2000 and 2007, respectively. Since 2012, he has been on the faculty of the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, where he is currently a Professor. Since 2022, he has served as the Deputy Director-General in Administration for Cyber Security, Ministry of Digital Affairs. His current interests are mobile network security, IoT system security, malware analysis and AI robustness. He has received IEEE Trustcom 2020 Best Paper Awards.