Implementing and Evaluating Security in O-RAN: Interfaces, Intelligence, and Platforms

Joshua Groen*, Salvatore D'Oro, Utku Demir, Leonardo Bonati, Michele Polese, Tommaso Melodia, Kaushik Chowdhury
Institute for the Wireless Internet of Things, Northeastern University, Boston, MA, USA
*groen.j@northeastern.edu, {f.last}@northeastern.edu

***Abstract*—The Open Radio Access Network (RAN) is a new networking paradigm that builds on top of cloud-based, multi-vendor, open and intelligent architectures to shape the next generation of cellular networks for 5G and beyond. While this new paradigm comes with many advantages in terms of observability and reconfigurability of the network, it inevitably expands the threat surface of cellular systems and can potentially expose its components and the Machine Learning (ML) infrastructure to several cyber attacks, thus making securing O-RAN networks a necessity. In this paper, we explore security aspects of O-RAN systems by focusing on the specifications, architectures, and intelligence proposed by the O-RAN Alliance. We address the problem of securing O-RAN systems with a holistic perspective, including considerations on the open interfaces used to interconnect the different O-RAN components, on the overall platform, and on the intelligence used to monitor and control the network. For each focus area we identify threats, discuss relevant solutions to address these issues, and demonstrate experimentally how such solutions can effectively defend O-RAN systems against selected cyber attacks. This article is the first work in approaching the security aspect of O-RAN holistically and with experimental evidence obtained on a state-of-the-art programmable O-RAN platform, providing unique guideline for researchers in the field.**

***Index Terms*—O-RAN, Security, Interfaces, AI/ML, Data poisoning**

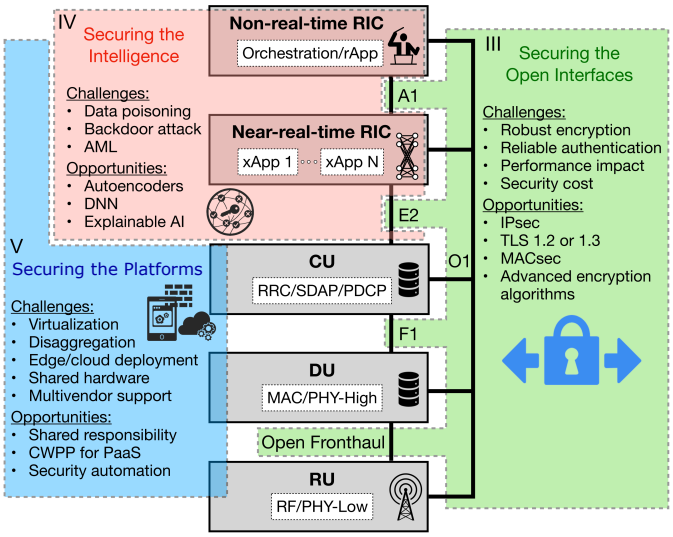

Fig. 1: Challenges and opportunities to secure different groups of components in the O-RAN architecture, which also reflects the paper outline.

I. Introduction

The increasing demand for multi-vendor, horizontally disaggregated systems in cellular Radio Access Network (RAN) has exposed the limitations of traditional, closed, proprietary architectures. These systems are inadequate for meeting the stringent requirements of new services, prompting a shift towards open architectures that offer flexibility, programmability, and observability while reducing CAPEX and OPEX [1]. The Open RAN paradigm addresses these needs by promoting cloud-based, disaggregated, multi-vendor deployments with data-driven control, enabling flexible and customizable networks at lower costs.

However, the *openness* of O-RAN introduces security vulnerabilities, as fine-grained data extraction and network control capabilities can be exploited by attackers. Despite well-defined interface requirements, the impact of securing these interfaces is not fully understood. O-RAN's reliance on cloud-based virtualized functions and third-party Artificial Intelligence (AI)/ML applications for closed-loop control further expands the threat surface. While there is ongoing research on securing AI/ML and cloud-based architectures, a comprehensive security solution for the entire O-RAN architecture remains an open challenge.

Security in O-RAN is still in its early stages. Tab. I summarizes the focus, security topics addressed, and security implementations in the existing literature on O-RAN security. [4] explores zero-trust principles, adding network analytics and anomaly detection to authentication mechanisms. Abdalla *et al.* [2], Mimran *et al.* [3] and Polese *et al.* [1] identify potential threats and vulnerabilities but do not assess the cost of protection. To address these shortcomings, we identify three components within O-RAN—Open Interfaces, AI-native closed loop control, and cloud-based platforms—that pose heightened risks and propose actionable strategies to mitigate these risks. Different from previous works, we empirically evaluate the impact–in terms of cost and effectiveness–of implementing several of these measures. Specifically, the contributions of this article include:

- Reporting the observed effects on latency, throughput, and processing load from integrating encryption into the E2 interface.
- Assessing the efficacy of employing Autoencoders to

Paper	Major Focus	Interfaces	Intelligence	Platforms	Implementation
[1]	Comprehensive overview of O-RAN	✓	✓	✓	
[2]	Survey of current capabilities and limitations of O-RAN	✓	✓	✓	
[3]	Proposed security evaluation ontology for O-RAN	✓	✓	✓	
[4]	Proposed Zero Trust framework for 5G/6G		✓	✓	
[5]	Architectural security considerations for O-RAN	✓		✓	
[6]	Open Fronthaul Security	✓			
[7]	Securing the E2 interface	✓			✓
This paper	Cost and effectiveness of implementing comprehensive security in O-RAN	✓	✓	✓	✓

TABLE I: Survey of prior works showing the general focus of each paper, the security topics addressed, and if any solutions were implemented and evaluated in an O-RAN environment.

mitigate potential attacks targeting the AI.
- Identifying and detailing three pivotal principles of cloud security essential for safeguarding computing platforms in O-RAN deployments.

The organization of this paper, summarized in Fig. 1, is as follows. We first provide a brief background on the O-RAN architecture. Then, we discuss the open interfaces, followed by a baseline implementation evaluating the cost of securing them. Next, we demonstrate how to secure the intelligence and mitigate threats targeting AI/ML applications hosted on the near-RT RAN Intelligent Controller (RIC). After that, we provide an overview on securing platforms and disaggregated functions. Finally, we propose ways to advance security in O-RAN and draw our conclusions.

II. O-RAN Primer

The O-RAN Alliance defines a flexible and softwarized architecture (shown in Fig. 1) for the Open RAN ecosystem. This architecture embraces the 3GPP 7.2x functional split, where RAN functionalities are *disaggregated* into Central Unit (CU), Distributed Unit (DU), and Radio Unit (RU) [8]. These components carry out operations at different layers of the protocol stack. For example, the CU performs RRC, PDCP, SDAP, and PDCP while the DU provides RLC, MAC, and PHY-high functions. Both PHY-low and radio-frequency transmission are carried out at the RU.

Each of these components' functionalities can be controlled in software through exposed Application Programming Interfaces (APIs). Open and standardized interfaces connect these components both among them (e.g., the F1 interface connects CU and DU, and Open Fronthaul interface connects DU and RU), and to the RICs (i.e., through the E2 and O1 interfaces).

These controllers are connected by the A1 interface and oversee the operations on the RAN nodes and enable closed control-loops that operate at different time scales through AI/ML applications. The near-RT RIC acts on time scales between 10 ms and 1 s via applications called xApps, while the non-RT RIC on time scales above 1 s via rApps. Finally, the applications running on the RICs receive live RAN Key Performance Indicators (KPIs) from the RAN nodes through the E2 interface (in the case of xApps) and through the O1 interface (in the case of rApps). The RICs can send control actions, e.g., to prioritize certain User Equipments (UEs) or to modify the scheduling policy of the DU, through the same control interfaces. Interested readers are referred to [1] for a comprehensive overview of the O-RAN architecture and of its specifications.

III. Securing the Open Interfaces

A. New Vulnerabilities

The introduction of open interfaces that carry data between disaggregated components is a pillar of the O-RAN framework. However, the openness of the interface, combined with disaggregated nodes connected over open infrastructure, introduces expanded security vulnerabilities. In fact, some of the primary classes of threats arise from improper or missing ciphering of the data sent across these open interfaces and lack of proper authentication [2]–[4], [7]. For example, these vulnerabilities could be exploited with man-in-the-middle attacks to inject false KPI reports northbound to the near-RT RIC or inject malicious control actions southbound to the gNB.

The O-RAN Alliance recognizes these expanded threat vectors and each interface working group has published guidance to secure their respective interface. While the guidance varies greatly in the level of detail provided, in general each interface should provide confidentiality, integrity, and authentication. While this guidance is reasonable and there is general consensus that security is essential to the deployment of 5G O-RAN infrastructure [2], [4]–[7], [9], [10], there has been little study on the impact of securing the open interface in an O-RAN deployment. It is vital that an informed and risk-based approach is taken to adequately address security risks in O-RAN, while recognizing that any method for enhancing security, such as adding encryption, comes at a cost in terms of performance. This impact could reduce throughput or require additional equipment, affecting scalability and negatively impacting CAPEX.

B. Implementing IPsec on the E2 Interface

To extract actionable insights, we thoroughly test and analyze the effects of adding encryption to the E2 interface as described in [7]. We focus our efforts first on the E2 interface as this is the most mature interface, which makes it possible to add IPsec to this interface and observe the trade-offs between security robustness and performance.

While the E2 interface standards call for the use of IPsec, other open interfaces require the use of TLS 1.2, though use of TLS 1.3 is recommended. There are some differences in the way IPsec and TLS protocols initially establish a connection or Security Association (SA), however, we are

confident we can extend the results generally without loss of accuracy because IPsec and TLS use the same underlying encryption and hashing algorithms once the SA is established. TLS 1.3 has the fastest SA establishment, making it hardest to confidently extrapolate the results directly from IPsec, but the general trends in performance still hold. Regardless of the security protocol used, system designers should only use modern secure encryption algorithms.

After establishing a baseline performance without encryption, we add O-RAN-compliant encryption (as specified in [9]) to the E2 interface. We test multiple combinations of encryption algorithms and implementations, including AES128 (CBC, CCM), AES256 (CBC, CCM, GCM), and ChaCha20-Poly1305. IPsec, as configured in our test, provides confidentiality, integrity, replay protection, and authentication. With this configuration, IPsec adds at least 57 Bytes of overhead to each packet. However, because both AES and SHA2 require fixed input block sizes, padding may be added causing the overhead to further increase. For example, encrypting a 62 Byte selective acknowledgement (SACK) adds 76 Bytes for a total cypher text (CT) SACK length of 138 Bytes.

C. Theoretical Analysis

To measure the impact to performance of adding security we examine delay and throughput as key performance indicators. In packet switched networks there are generally four sources of delay at each node along the path: *queuing* delay, *propagation* delay, *transmission* delay, and *nodal processing* delay [11].

- *Queuing Delay:* In general, the queuing delay is not constant and depends on the packet arrival and departure rates. However, adding encryption to the link does not significantly impact the queuing delay. In other words, the difference in queuing delay with or without encryption is negligible.
- *Propagation Delay:* The propagation delay is strictly a function of the physical length and propagation speed of the link and will remain constant regardless of encryption.
- *Transmission Delay:* The transmission delay is a function of the packet size (in bits), L, and the link transmission rate, R, which is defined as $D_{trans} = L/R$. For any given system, R is fixed but L will increase to some extent with encryption.
- *Processing Delay:* Typically the processing delay is defined as the time required for intermediate nodes to examine the packet header and determine where to direct the packet [11]. For this analysis, we include the encryption delay in the processing delay because it is essential to pass the payload to lower or higher OSI layers.

While queuing and propagation delay are unaffected by encryption, transmission and processing delay are impacted. For this reason our analysis focuses on quantifying the increase in transmission and processing delays. While it is clear that there will be some impact, researchers should fully understand the expected trade-off for their system. Next, we outline a few experiments that system engineers can perform to better characterize the impact of security and ensure they plan for security by design.

D. System Implementation Results

To quantify the effect of encryption for packets of various lengths, we use ping (ICMP echo) of various lengths to capture the network round trip time (RTT), which is twice the one-way delay. We subtract the fixed propagation delay and the known transmission delay and graph the processing delay in Fig. 2. We observe that the specific implementation of the AES algorithm greatly affects the additional processing delay. AES256-CCM causes the processing delay to increase with packet length while AES256-GCM has virtually no impact. Galois/Counter Mode (GCM) offers both confidentiality (via counter mode) and authentication (via arithmetic in the Galois field GF(2^n)), where n represents the key size. Since these operations can be executed in parallel, GCM delivers higher performance compared to modes like CBC, which necessitate sequential operation chaining [12]. In either case, the processing delay difference between CT and Plain Text (PT) is $\Delta D_{proc} \leq 50\mu s$ for all tested packet sizes. We can conclude that encryption has minimal impact on E2 traffic delay.

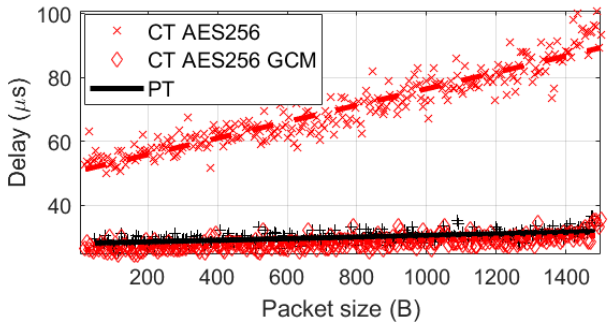

Fig. 2: Processing delay as a function of packet size for PT and CT traffic.

We also design an experiment to quantify the effect of encryption on the total traffic throughput. We use iperf3 to generate traffic at specific transmission rates for 10 s. In Fig. 3 we see that the maximum encryption rate our system is capable of for AES256-CCM with SHA256 is a little under 600 Mbps. We also capture CPU utilization on the gNB for each attempted transmission rate in Fig. 3 for both PT and CT, showing that encryption is very CPU intensive. The CPU utilization grows linearly until it reaches saturation.

We test several other encryption algorithms and find a similar pattern for all. However, some algorithms are more CPU intensive than others. To illustrate the different impact a given encryption algorithm has on maximum throughput, we used iperf3 to send the maximum amount of traffic possible for 30 s. AES-CBC achieves 505 and 512 Mbps for key lengths 128 and 256 bits, respectively. AES-CCM achieves 573 Mbps for both key lengths of 128 and 256 bits. Cha-Cha20-Poly1305 achieves 989 Mbps, while AES-GCM achieves 1370 Mbps for all key lengths, 64, 128, and 256 bits. The specific algorithm implementation is the most significant factor and AES-GCM vastly outperforms the other algorithm implementations. One key observation is that there is virtually no difference in performance based on key size (AES128 verse AES256). We encourage all system designers to use 256 bit key length

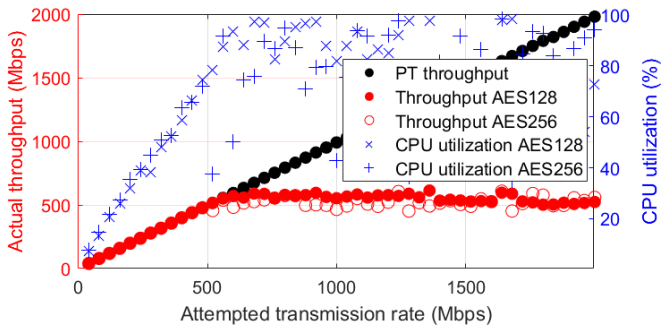

Fig. 3: Measured throughput as a function of attempted transmission rate for PT and CT traffic. CPU utilization is the limiting factor for throughput with encryption.

as that provides higher security with virtually no impact to performance.

E. Key to Securing Open Interfaces

The key trade-off for securing O-RAN open interfaces is processing power. Any disaggregated gNB component must have enough CPU resources to manage all of its explicit functions. However, as the total traffic over the open interfaces increases, CPU resources needed for encryption alone will also increase. One potential solution is to increase the total compute resources in the distributed gNB. For example, we compare OpenSSL benchmark tests on our Colosseum emulation environment built on Intel Xeon e5-2650 CPU to a desktop running an Intel Core i7-11700KF CPU and found that the desktop offers approximately a $3.5\times$ performance improvement. System designers could also add dedicated encryption hardware to the network stack to remove the burden from the CPU. A third option is to set strict limits on the amount of traffic that can be sent over the interfaces. This option requires careful consideration, as limiting interface throughput can significantly impact system performance. However, there may be safe reduction opportunities. For instance, in our system with 10 UEs, the E2 interface averages 3.5 Mbps with updates every 250 ms. However, this rate of updates may be higher than what is actually needed. By reducing the update frequency to every 333 ms the average throughput is reduced to 2.6 Mbps, decreasing link utilization by 25%. In any case, system engineers must understand the amount of traffic expected across given interfaces and include the overhead of encryption in their system design.

IV. Securing the Intelligence

O-RAN comes with the promise of AI-native cellular networks where a fabric of RICs, rApps and xApps monitor the network and take autonomous decisions to maximize performance and meet operator goals and intents [1]. By leveraging AI/ML techniques, rApps and xApps will enable applications ranging from control of network slicing and scheduling policies, traffic steering, mobility management, beamforming, and energy saving functionalities. However, these data-driven techniques are also known to have pitfalls and vulnerabilities that might expose rApps, xApps, RICs and the entire network to a variety of security threats.

A. Vulnerabilities of AI/ML O-RAN Applications

As rApps and xApps are the decision-making engines of O-RAN systems, it is fundamental that their decisions are unbiased, made with confidence, resilient against attacks or anomalies, and do not deviate from the goal of the network operator. Next, we highlight some vulnerabilities that affect AI/ML routines embedded in O-RAN applications.

- *Before Deployment:* AI/ML solutions heavily rely upon the quality and quantity of data used to train them, which makes them vulnerable against *data poisoning* and *backdoor attacks*. Data poisoning aims at injecting misleading data into the data lakes to negatively impact the decisions made by data-driven xApps and rApps. One example is an adversarial UE falsely reporting poor KPI measurements (e.g., a low throughput level while experiencing high values). Any xApp or rApp trained using such biased and misleading data will result in sub-optimal decisions. Backdoor attacks, instead, aim at making AI/ML models sensitive against specific inputs, events, or KPI patterns. For example, an attacker can develop an xApp controlling slicing policies with a backdoor such that the xApp makes decisions that are unfair toward certain subset of UEs whenever a certain input triggers the backdoor, or takes actions that are in conflict with those of legitimate applications.
- *After Deployment:* Attacks might also affect real-time execution and inference of xApps and rApps. A well-known attack is Adversarial Machine Learning (AML), where the attacker aims at either generating completely synthetic inputs, or at modifying an existing legitimate input via small and hard to detect perturbations. In O-RAN, these attacks can be extremely effective and can completely disrupt networking operations. This includes misleading a traffic classifier used to determine traffic type and corresponding Quality of Service (QoS), bypassing anomaly detection mechanisms, steering control decisions away from their optimal, or leading the non-RT RIC into producing an inaccurate representation of network state. Similarly, an attacker can potentially generate undetectable adversarial inputs that mislead an rApp into triggering wrong handover procedures, while at the same time causing an xApp to allocate fewer resources to the hand-off UEs, resulting in unfairness and poor performance.

B. Toward Reliable AI/ML O-RAN Applications

Although the spectrum of attacks against data-driven xApps and rApps is broad, practitioners can rely upon a portfolio of well-established tools and solutions to protect the intelligence of the RICs.

- *Preventing Attacks:* The first step starts from designing and training xApps and rApps that are secure and robust against attacks by design. In this area, xApp/rApp developers can leverage several tools designed to minimize the

impact of data variance on the output of AI/ML solutions embedded into O-RAN applications. A few examples are: projecting data into a latent space (e.g., Autoencoders (AEs)), embedding synthetic adversarial and poisoned data into the training process (e.g., contrastive and adversarial learning, contaminated best arm identification, defensive distillation), and mitigating attack effectiveness via attention networks that aim at maintaining attention level on relevant features only and neglect adversarial influence.

- *Detecting Attacks:* Although preventing attacks is always desirable, learning how to detect novel and increasingly sophisticated attacks becomes a necessity. Anomaly detection is the most relevant area of research that focuses on this issue and can help in designing secure O-RAN systems. Among others, we mention techniques that detect anomalies by monitoring the distance between input data and the expected distribution (e.g., clustering- and distance-based algorithms, statistical methods), as well as deep neural networks that can identify unexpected data features that characterize anomalies. The former techniques could be used, for instance, to compare the decisions made by xApps/rApps at run-time with their statistical behavior, raising an anomaly warning in case these differ significantly. The latter, instead, could be leveraged to flag data with unexpected formats or patterns caused by anomalies in the network.
- *Reacting to Attacks:* In the case where an attack is successful and it has been detected, the best way to react is to understand what caused the attack, how it was performed and eventually learn to recover from it. Explainable AI tools can be used to analyze the behavior of O-RAN applications, and to provide the non-RT RIC with information on which xApps and rApps were affected, what type of input produced the attack and why certain applications produced unexpected outputs. This information can then be used by the non-RT RIC to retrain model to eliminate vulnerabilities.

C. Autoencoders for Threat Mitigation

In the following, we provide experimental results that illustrate how embedding AI-based solutions like Autoencoders (AEs) into O-RAN applications can effectively help in mitigating adversarial attacks targeting the data-driven logic running therein. We experimentally demonstrate that conventional autoencoders can deliver a good degree of protection against adversarial attacks, but more complex variations (e.g., variational, denoising and convolutional autoencoders) could be used for improved performance).

We consider the case of an xApp embedding a Deep Reinforcement Learning (DRL) agent trained to jointly control RAN slicing and scheduling policies of Enhanced Mobile Broadband (eMBB), Ultra Reliable and Low Latency Communication (URLLC) and Massive Machine-Type Communications (mMTC) slices. These agents are trained using a Proximal Policy Optimization (PPO) architecture with a reward that weights slice-specific KPIs. Due to space limitations, we refrain from reporting details on the DRL design and training process, which can be found in [13]. We consider two state configurations: direct feeding of KPIs to the DRL agent, and preprocessing KPIs through an AE for dimensionality reduction and outlier suppression before feeding them to the DRL agent. We consider the case of black-box adversarial attacks. The attacker uses an attack vector consisting of corrupted KPI measurements generated by taking legitimate KPI data and perturbing them with additive random noise following a normal distribution with mean equal to the KPI value being perturbed and variance σ.

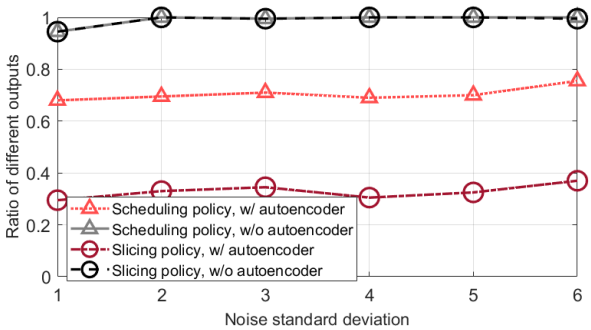

Fig. 4: Percentage of actions deviating from the expected outcome after the attack with and without the autoencoder.

To evaluate the effectiveness of the attack, we vary the standard deviation of such a random variable and run 200 independent runs. For each run, the DRL agent is fed first with KPIs reported by a cellular base station, and then by those affected by the attack. In Fig. 4, we show the percentage of actions that deviate from the intended original actions taken by the DRL agent after attack. We perform this analysis by considering two cases. In the first case, we feed the perturbed KPIs directly to the DRL agent, while in the second case the KPIs are first processed by an AE. Our results show that the AE always results in fewer deviations from the intended actions even in the case of high noise variance. On the contrary, the case where no AE is used results in more than 95% deviation from the intended action. Moreover, we notice that the most affected action is the scheduling policy, which deviates 70% of times, while the slicing policy is affected only 30% of times.

In Fig. 5, we investigate how much an attack can steer the xApp/rApp agent away from the intended behavior, i.e., how distant from the intended outcome the actions taken by the DRL agent are, by reporting the Euclidean distance between the intended action and the one taken by the DRL agent. Scheduling policies are enumerated as $\{0, 1, 2\}$, while slicing actions consist of three integer numbers (one per slice) whose sum must be equal to 50 (i.e., the number of Physical Resource Blocks (PRBs) to assign to each slice). We measure the Euclidean distance between the intended action and the one taken by the DRL agent, and in Fig. 5 we report the normalized Euclidean distance. Results show that the AE reduces the distance (and therefore the effectiveness of the attack) between scheduling decisions made after the attack by approximately $2\times$, and by more than $13\times$ with respect to the slicing policies.

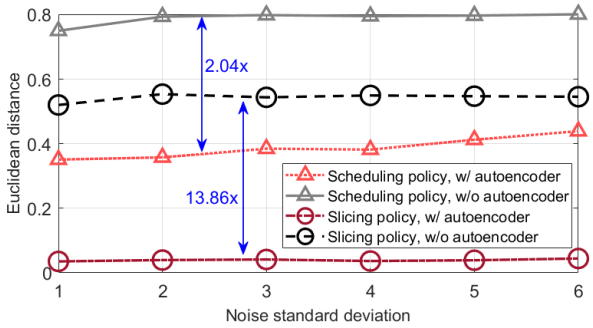

Fig. 5: Action distance comparison after attack with and without autoencoder. Scheduling policies are enumerated as $\{0, 1, 2\}$, while slicing actions consist of three integer numbers (one per slice) whose sum must be equal to 50 (i.e., the number of Physical Resource Blocks (PRBs) to assign to each slice).

V. Securing the Platforms

One of the defining characteristic of O-RAN is the shift toward virtualization of network functions [1]–[3], [5]. This shift supports several goals including multi-vendor support, disaggregation of nodes, scalability, and open source development. In practice, this is accomplished in a way that follows cloud service deployments. Specifically, network functions are hosted and executed in virtual machines (or containers) managed by a hypervisor or host operating system (OS) to provide resource management, performance optimization, and access to common interfaces, among other services. While virtualization and cloud-based deployments result in a much more agile and flexible network infrastructure, they inevitably result in a much larger threat surface area.

A. Cloud Security Risks

One of the key components of any cloud infrastructure is common hardware components shared by multiple software instances. In the context of cellular communications, this introduces a new risk as isolation between applications is logical only, without physical isolation across hardware [5]. Additionally, the increased number of layers controlled by different parties in a virtual system provides additional threat vectors. An application must rely on multiple lower layers to perform specific security functions. For example, the host operating system has access to all RAM memory, disk volumes mounted on virtual machines, and containers [5]. As another example the hypervisor must access hardware security functions and pass those functions to the application. This means that the application must be able to trust both the hardware itself and the virtualization layer. To fully trust an application, one needs to trust all the layers in the stack.

There are several existing cloud security architectures built on the concept of shared responsibility [14]. In general, the idea of shared responsibility means the cloud service provider is responsible for the security of all the components necessary to operate the cloud service, while the customer is responsible for protecting their data. In cloud security, the specific breakdown in responsibility is often a function of the type of cloud service (i.e., Infrastructure as a Service versus Platform as a Service (PaaS)). An O-RAN deployment most closely resembles a PaaS deployment. One of the security services typically used for a PaaS deployment is a Cloud Workload Protection Platform (CWPP). The objective of the CWPP is to keep the application, whether a VM or container, secure. System engineers should review PaaS security services for valuable lessons when designing O-RAN security and applications.

B. O-RAN Specific Security Recommendations

Unlike typical cloud deployments, one of the features of O-RAN is enabling multiple vendors to provide different functions. This creates additional challenges to principles such as supply chain security, secure service administration, and personnel security. This also means traditional appliances such as external security gateways are infeasible in terms of both CAPEX and OPEX to deploy between each and every component of a distributed O-RAN system. Some of these challenges could be mitigated through a strong governance framework and appropriate visibility and auditing. On the other hand, O-RAN's stated goal of increasing automation and using AI and ML naturally supports principles such as security automation and flexibility.

Regardless of the cloud service architecture, there are many existing cloud security frameworks that can be applied. For example, the UK National Cyber Security Centre published 14 Cloud Security Principles [15], while Amazon AWS uses 7 Design Principles. A full discussion of all of these principles is beyond the scope of this paper, but we will address a few of the key principles that should be applied to O-RAN systems.

- **Separation Between Customers.** O-RAN deployments must ensure separation between customers so that a malicious or compromised component cannot affect the service or data provided by another component. This responsibility belongs to both cloud service providers to secure the hypervisor or host OS and on developers providing new functionality, such as ensuring xApps do not leak customer information to other entities.
- **Visibility and Auditing.** O-RAN deployments should also embrace visibility and auditing so that it is clear to all parties who is responsible for what function and the actions taken by each component can be traced. While visibility is inherently present to a degree in open source deployments, auditing is often overlooked. Further, as more complicated intelligence is introduced to closed loop control, understanding which component directed a particular action becomes both more difficult and more necessary to secure each component. One way to achieve this is to ensure the RICs maintains logs of the control messages sent by each xApp and rApp.
- **Security by Design.** O-RAN must embrace security by design at all layers of the O-RAN stack. Each component in Fig. 1 should be built with security in mind from the beginning. This can be especially challenging at the lower levels such as the RU and DU where performance is at a premium. There is always a trade off between security and performance. However, starting with security as one of

the necessary objectives for each block allows researchers to optimize the solutions before deployment and before an attack.

VI. Advancing the Security Framework

Our study serves as a foundational reference for future O-RAN security research, highlighting vulnerabilities in its components and demonstrating the efficacy of proposed methods through practical implementations. However, achieving a completely secure O-RAN deployment requires further research to build integral, in-depth, comprehensive security. Future efforts should focus on securing the Open Fronthaul and implementing a Zero Trust Architecture.

• **Open Fronthaul.** The Open Fronthaul includes the User plane, Control plane, Sycnhronization plane, and Management plane. While each of these planes has unique security requirements, all of them require low latency. For example, the synchronization plane does not require confidentiality, but authentication and integrity are crucial for timing packets crossing an insecure switched network. A man-in-the-middle attack could insert, delete, modify, or replay these messages causing a significant degradation or outage of service. Ensuring each plane provides the appropriate security function without impacting the quality of services is an open problem.

• **Zero Trust Architecture.** Existing security models often assume a high level of trust among entities, which doesn't align well with the O-RAN framework [4]. In contrast, Zero-Trust Architectures (ZTA) align well with emerging 5G network infrastructure [10]. ZTAs offer security assurances by adopting a data-centric model rather than traditional perimeter-based methods. This paradigm shift redefines networks as platforms for distributed data management, emphasizing data protection at every stage of the data cycle.

O-RAN presents an opportunity for implementing ZTA. Key ZTA elements include dynamic risk assessment and continuous trust evaluation. Researchers can leverage the AI/ML integration inherent in O-RAN for these tasks. Further development and implementation of ZTA xApps and rApps should focus on real-time monitoring, risk assessment, and ensure the availability, integrity, and confidentiality of a dynamic, multi-vendor, virtual O-RAN platform.

VII. Conclusions

In this article, we provide a holistic guideline for securing an O-RAN system by classifying the potential targets as open interfaces, data-driven decision making components, i.e. intelligence, and cloud-based platforms. We indicate possible threats and suggest how these can be addressed using various methods. We then provide baseline implementation results for open interfaces and intelligence in order to showcase the impact of the few suggested solutions.

Specifically, we observed that the IPsec protocol, which is the recommended security protocol for E2 interface by O-RAN Alliance, has minimal affect on delay for encrypting packets in the open interfaces, as the encryption adds a delay of less than $50\mu s$. However, encryption comes with a computation cost, requiring higher CPU for increasing throughput. In our environment, CPU utilization is the key cost to encryption.

Threats to the intelligence (i.e., AI/ML enabled RICs) can come before and after deploying ML models. Security measures for the intelligence include preventing, detecting, and reacting to attacks. In this paper, we showcased an implementation to mitigate data poisoning attacks against an xApp that runs a DRL model using an autoencoder layer before the input to the DRL. Our experiments revealed that autoencoders can decrease the unexpected ML outputs for scheduling and slicing by 30% and 70%, respectively.

Lastly, we provide security measure directions on the overall platform, where we suggest researchers to focus on shared cloud responsibility. In this model, the service providers and users are responsible for securing the service and data, respectively, which can be realized through a CWPP, for example. We also highlight three key principles for O-RAN specific cloud deployment, including separation between customers, visibility and auditing, and security by design.

Acknowledgment

This article is based upon work partially supported by Qualcomm Inc. and by the U.S. National Science Foundation under grants CNS-1925601, CNS-2112471, CNS-1923789 and CNS-2120447.

References

[1] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 1376–1411, January 2023.

[2] 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 Network*, 2022.

[3] D. Mimran, R. Bitton, Y. Kfir, E. Klevansky, O. Brodt, H. Lehmann, Y. Elovici, and A. Shabtai, "Evaluating the security of open radio access networks," *arXiv preprint arXiv:2201.06080*, 2022.

[4] K. Ramezanpour and J. Jagannath, "Intelligent zero trust architecture for 5G/6G networks: Principles, challenges, and the role of machine learning in the context of O-RAN," *Computer Networks*, p. 109358, 2022.

[5] J. Boswell and S. Poretsky, "Security considerations of open ran," *Stockholm: Ericsson*, 2020.

[6] D. Dik and M. S. Berger, "Open-RAN Fronthaul Transport Security Architecture and Implementation," *IEEE Access*, vol. 11, pp. 46 185–46 203, 2023.

[7] J. Groen, B. Kim, and K. Chowdhury, "The Cost of Securing O-RAN," in *ICC 2023-IEEE International Conference on Communications*. IEEE, 2023, pp. 5444–5449.

[8] O-RAN WG1, "O-RAN-Architecture-Description-v07.00," Technical Specification, 2022.

[9] O-RAN Working Group 3, "Near-Real-time RAN Intelligent Controller Architecture & E2 General Aspects and Principles," ORAN.WG3.E2GAP-v02.02, Tech. Rep., July 2022.

[10] "5G Strategy Implementation Plan," Department of Defense, Tech. Rep., December 2020.

[11] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach, Seventh Eddition*, 2017.

[12] J. Groen, S. D'Oro, U. Demir, L. Bonati, D. Villa, M. Polese, T. Melodia, and K. Chowdhury, "Securing O-RAN Open Interfaces," *IEEE Transactions on Mobile Computing*, 2024.

[13] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "ColO-RAN: Developing machine learning-based xApps for open RAN closed-loop control on programmable experimental platforms," *IEEE Transactions on Mobile Computing*, 2022.

[14] M. Lane, A. Shrestha, and O. Ali, "Managing the risks of data security and privacy in the cloud: a shared responsibility between the cloud service provider and the client organisation," *Bright Internet Global Summit 2017*, 2017.

[15] "Cloud security guidance," https://www.ncsc.gov.uk/collection/cloud/the-cloud-security-principles, accessed: 2022-02-21.

Joshua Groen is a Ph.D. student at Northeastern University. Previously he worked in the US Army Regional Cyber Center – Korea as the Senior Network Engineer. He received his MS ('17) and BSE ('07) in Electrical Engineering respectively from the University of Wisconsin and Arizona State University. His research interests include wireless communications, security, and machine learning.

Salvatore D'Oro is a Research Assistant Professor at Northeastern University. He received his Ph.D. from the University of Catania in 2015. He serves on the Technical Program Committee of IEEE INFOCOM. His research focuses on optimization and learning for NextG systems.

Utku Demir is a Postdoctoral Research Fellow at Northeastern University, where he is supported by the Roux Institute's Experiential AI program. He received his PhD from the University of Rochester in 2020. His research interests lie in the areas of wireless communications, mobile networks, signal processing, and machine learning.

Leonardo Bonati is an Associate Research Scientist at Northeastern University. He received his Ph.D. from Northeastern University in 2022. His research focuses on softwarized NextG systems.

Michele Polese is a Research Assistant Professor at Northeastern University. He obtained his Ph.D. from the University of Padova in 2020. His research focuses on architectures for wireless networks.

Tommaso Melodia is the William Lincoln Smith Professor at Northeastern University, the director of the Institute for the Wireless Internet of Things, and the director of research for the PAWR Project Office. He received his PhD from the Georgia Institute of Technology in 2007. His research focuses on wireless networked systems.

Kaushik Chowdhury is a Professor at Northeastern University, Boston, MA. He received his PhD from Georgia Institute of Technology in 2009. His current research interests involve systems aspects of machine learning for agile spectrum sensing/access, unmanned autonomous systems, programmable and open cellular networks, and large-scale experimental deployment of emerging wireless technologies.