This paper has been accepted for publication at the 10th IEEE International Conference on Network Softwarization (NetSoft2024)

Securing the Open RAN Infrastructure: Exploring Vulnerabilities in Kubernetes Deployments

Felix Klement*, Alessandro Brighente†, Michele Polese‡, Mauro Conti†, Stefan Katzenbeisser*
*Computer Engineering, University of Passau, Passau, Germany
†Department of Mathematics, University of Padova, Padova, Italy
‡Institute for the Wireless Internet of Things, Northeastern University, Boston, MA, USA
Email: {felix.klement, stefan.katzenbeisser}@uni-passau.de,
{alessandro.brighente, mauro.conti}@unipd.it, m.polese@northeastern.edu

***Abstract*—In this paper, we investigate the security implications of virtualized and software-based Open Radio Access Network (RAN) systems, specifically focusing on the architecture proposed by the O-RAN ALLIANCE and O-Cloud deployments based on the O-RAN Software Community (OSC) stack and infrastructure. Our key findings are based on a thorough security assessment and static scanning of the OSC Near Real-Time RAN Intelligent Controller (RIC) cluster. We highlight the presence of potential vulnerabilities and misconfigurations in the Kubernetes infrastructure supporting the RIC, also due to the usage of outdated versions of software packages, and provide an estimation of their criticality using various deployment auditing frameworks (e.g., MITRE ATT&CK and the NSA CISA). In addition, we propose methodologies to minimize these issues and harden the Open RAN virtualization infrastructure. These encompass the integration of security evaluation methods into the deployment process, implementing deployment hardening measures, and employing policy-based control for RAN components. We emphasize the need to address the problems found in order to improve the overall security of virtualized Open RAN systems.**

***Index Terms*—Open RAN, security, virtualization, RIC**

I. INTRODUCTION

The Open Radio Access Network (RAN) paradigm is moving next-generation wireless networks toward systems which are more flexible, programmable, and can be customized and optimized to support new use cases through data-driven intelligent control. Open RAN solutions, and their implementation as part of the O-RAN ALLIANCE specifications, transition the RAN toward softwarized solutions, extending the programmability in domains which have usually been associated with custom silicon and dedicated circuits [1], [2].

The adoption of software in the RAN increases the level of programmability of the full protocol stack, making it easier to interact programmatically with Open RAN base stations. This, in turn, can be leveraged to expose telemetry and performance metrics from the RAN, collect them at a large scale in controllers at the edge of the network via the RAN Intelligent Controllers (RICs), and apply data-driven techniques for the intelligent optimization of the stack [3]. At the same time, leveraging software in the RAN allows for a faster innovation cycle, driven by the possibility of running continuous integration, deployment, and testing processes. Finally, it also increases the diversity of the supply chain as it lowers that barrier to entry in the cellular market. A software-first RAN requires proper support by the RAN infrastructure, i.e., a set of virtualization solutions, automation pipelines, and hardware accelerators. This is generally referred to as the *O-Cloud* in the O-RAN ALLIANCE architecture. Virtualization and software-first infrastructure can enable dynamic scaling of compute resources to accommodate and tailor the network deployment to users and traffic requirements, toward a more energy efficient RAN. It also allows for multi-tenant RAN data centers, e.g., as in a neutral host environment with multiple operators sharing the same physical infrastructure to reduce costs [4]. At the same time, a virtualized environment introduces more parameters to configure and tune, additional software components (e.g., multiple software layers between the application and the hardware appliance), and a set of more heterogeneous workloads. This translates into a larger threat surface that can exploited by malicious attackers, either internal or external, or can impact the network performance because of misconfigurations [5].

This paper takes a first step toward understanding threats and quantitatively profiling the vulnerabilities that virtualization introduces in the O-Cloud, focusing specifically on a micro-services-based architecture for the O-RAN RIC implemented by the O-RAN Software Community (OSC) [6]. Compared to prior literature on security in Open RAN systems [2], [7]–[10], we focus on an assessment of how the software vulnerabilities in underlying virtualization solutions (specifically, the Kubernetes platform [11]) impact the services that support the O-RAN RIC. Our preliminary analyses unveil a substantial quantity of insecure components, configurations, and software dependencies within the Near Real-Time RIC (Near-RT RIC) cluster presently employed by the OSC. We leverage static scanning and combine it with a quantitative assessment

---

The authors acknowledge the financial support by the German Federal Ministry of Education and Research – Bundesministerium für Bildung und Forschung (BMBF), as part of the Project “6G-RIC: The 6G Research and Innovation Cluster” (project number 825026). This paper was also partially supported by the U.S. National Telecommunications and Information Administration (NTIA)’s Public Wireless Supply Chain Innovation Fund (PWSCIF) under Award No. 25-60-IF054.

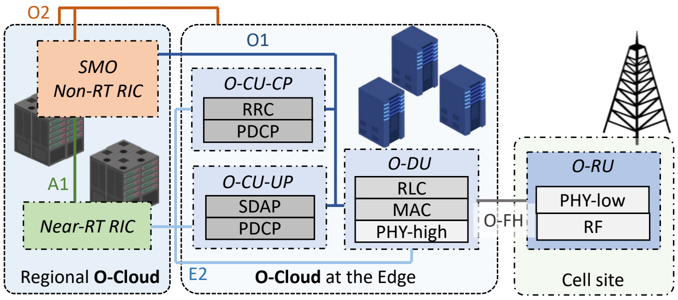

Fig. 1. Disaggregated O-RAN architecture with open interfaces and a typical deployment across different data centers implementing the O-RAN O-Cloud and a proprietary cell site.

methodology based on multiple deployment auditing frameworks, including MITRE ATT&CK, National Security Agency (NSA) and Cybersecurity and Infrastructure Security Agency (CISA) report [12], and the Center for Internet Security (CIS) CIS-v1.23-t1.0.1 [13]. Our analysis shows 792 vulnerabilities within the Near-RT RIC and 70 Common Vulnerabilities and Exposures (CVEs) in the currently used platform versions for the virtualized deployment.

This work testifies to how openness allows for clarifying the attack surface of RAN systems, a key advantage compared to security-by-obscurity adopted in previous RAN deployments. We commend the efforts of the OSC in providing an open-source reference framework for the RIC, and provide suggestions on how to integrate security assessment methodologies and the general hardening of deployments in the software development lifecycle. We also intend to share the discoveries from this paper with the OSC and address certain issues through the submission of pull requests in the relevant code repositories. We believe that this analysis can spark further research and focus on securing virtualized and software-based Open RAN systems, a key step toward deploying open, programmable, and intelligent networks that are reliable, resilient, and leverage the best practices of cloud security.

The remainder of the paper is organized as follows. In Sec. II, we review the O-RAN architecture, O-Cloud, and virtualization solutions. In Sec. III, we discuss the threat model considered in the paper, combining O-RAN notions and Kubernetes systems. In Sec. IV, we present security assessment methods, and we present results on their application to the OSC RIC software in Sec. V. Finally, we discuss best practices in Sec. VI and conclude the paper in Sec. VII.

II. O-RAN ARCHITECTURE AND DEPLOYMENT METHODOLOGIES

In the following section, we briefly introduce the Open Radio Access Network (O-RAN) architecture and explore the various deployment options available.

A. O-RAN architecture

The O-RAN architecture embodies the principles of disaggregation, intelligence, and programmability of the Open RAN paradigm. This translates into cellular networks which are based on software and are highly automated, and rely on cloud-based or edge computing platforms as essential components of the overall infrastructure. Figure 1 shows a high-level logical diagram of an O-RAN deployment with softwarized components deployed in multiple data centers implementing the O-RAN ALLIANCE O-Cloud, and a proprietary cell site with radio components and front-end, i.e., the O-RAN Radio Unit (O-RU) [14]. The O-RAN O-Cloud is a collection of physical infrastructure and software that provides the necessary abstractions and computing power to execute softwarized RAN workloads. These include the Service Management and Orchestration (SMO) of the overall network, which also hosts the Non Real-Time RIC (Non-RT RIC), the first of two RICs that can host custom applications (i.e., rApps) for network management and optimization. The second RIC is the Near-RT RIC, which has a direct interface to the RAN for near-real-time, data-driven resource management and hosts xApps. Finally, the RAN base stations are disaggregated into the O-RAN Central Unit (O-CU-CP), itself split into a user plane and a control plane function, and the O-RAN Distributed Unit (O-DU). The SMO orchestrates and manages the deployment of services and solutions in the O-Cloud through the O2 interface, as shown in the top part of Fig. 1. The specifications on the O-Cloud [15] focus on high-level abstractions rather than mandating a specific technology that needs to be adopted to implement the O-Cloud itself. Nonetheless, the industry has widely adopted microservices implemented through containers as the technical solution to deploy most O-RAN softwarized workloads. In [16], an empirical threat analysis method was used to illustrate that almost none of the threat classes predefined by the ALLIANCE exist in the O-Cloud without a critical vulnerability. In the upcoming paragraphs, we therefore examine and discuss the contributions of Docker and Kubernetes to enhance RAN deployments and their associated security considerations.

B. Docker as a container, Kubernetes for orchestration

The complexity of RAN systems has led to ongoing efforts to transition from challenging-to-manage monolithic approaches to more adaptable, service-oriented solutions based on atomic Network Functions (NFs). As explained earlier, the virtualized RAN has become essential, particularly since the inception of the O-RAN initiative. Docker containers play a key role in implementing a software-defined mobile network. To ensure the overall security of the containers, their configurations must be secure and consistently kept up to date. Achieving this can often be a substantial task, given that complex systems are constructed from a multitude of such configurations.

In a virtualized RAN NFs are executed in Docker containers within a server cluster using Kubernetes as an orchestrator. The use of Kubernetes facilitates the monitoring and management of NFs and thus the dynamic scaling of resources and services to meet changing O-RAN requirements. As a result, a more flexible and more efficient radio network enables the operator to offer its users a better experience. The results are overall cost savings, improved scalability, and increased agility of the network. Although the advantages of the Kubernetes ecosystem are clear, there are other challenges related to general system security in addition to high performance and latency, e.g., in edge data centers. The use of default configurations can prioritize flexibility over security, which can lead to vulnerabilities [12], [13]. Inadequate pod security policies and network vulnerabilities can lead to unauthorized access, service disruption, and data disclosure [17]. Human error is another important factor to consider, as the complicated nature of Kubernetes increases the likelihood of misconfigurations or oversights. To address these concerns, best practices must be implemented within an O-RAN deployment, components must be regularly updated and security checks must be performed. Using security tools and applying a defense-in-depth strategy are essential towards improving the overall Kubernetes cluster security.

III. THREAT MODEL

In this section, we present an attacker model and explore potential threat vectors that could be exploited by a malicious entity.

A. *Attacker Model*

We consider an attacker exploiting vulnerabilities in the cloud-based components of O-RAN. The attacker can be an insider attacker (i.e., an authorized user of the system) or an external attacker (i.e., a non-legitimate user of the system). The attacker may have different motivations to launch the attack, for instance, gaining access to sensitive data for commercial purposes, disrupting the network due to hacktivism or being a competitor operator, or gaining access to privileged functions they do not own and hence control the network. Insider attackers may also be employees bribed by external actors or enraged with the current employer, hence attacking the system from the inside.

B. *Threat Vectors*

In this paper, we focus on the software component's current security posture of OSC and refer the reader to the O-RAN WG 11 technical specification for details on hardware-related attacks [5]. As OSC leverages Kubernetes for orchestration, we refer to Kubernetes security guidelines, i.e., OWASP Kubernetes Top 10 [17], the NSA and CISA Kubernetes hardening guide [12], the MITRE ATT&CK, and the CIS Kubernetes benchmark [13]. Due to the lack of space, we can not provide the full details on the possible attacks and their implementations. However, based on the aforementioned guidelines, we summarize the most relevant threat vectors as weak Authentication and Access Control, lack of Network Segmentation and Isolation, Supply Chain vulnerabilities, and use of Outdated Components.

In case of authentication and authorization misconfigurations, the attacker can gain access to restricted resources with capabilities they should not have (e.g., writing permissions), configuration secrets, resource configuration, or impersonate a legitimate user. In O-RAN, attackers might gain control over virtual network functions, configurations dealing with network neutrality, confidential data of other operators/infrastructure, and launch resource exhaustion attacks. Broken network segmentation and isolation imply that the attacker can move inside the Kubernetes deployment. For example, an attacker may access pods belonging to other users. In O-RAN, this might lead to, among others, data theft, incomplete termination of network functions, attacks on internal network services, and false resource advertisements.

Pods can be configured to run containers, which should be conveyed from a dedicated repository via a supply chain. Uploading vulnerable or malicious containers represents a severe threat in a Kubernetes deployment. In O-RAN, supply chain vulnerabilities might lead to attackers exploiting misconfigurations in the container to gain privileged access to network functions and sensitive user data, or might be an entry point for attackers exploiting lateral movements in the infrastructure. Finally, old vulnerable Kubernetes versions might still be available and used. An attacker can hence leverage known vulnerabilities in such configurations.

IV. ASSESSMENT METHODS

In the following, we discuss our security evaluation approaches within the context of RANs and the specific security attributes they assess.

A. *Static Scanning*

Static application security testing (SAST) is a methodology employed to analyze static code to uncover potential weaknesses and existing vulnerabilities within the current codebase. It plays a crucial role in code tests associated with the actual business logic. Additionally, it is commonly utilized for validating infrastructure code, such as Docker files. Previous studies [18] even show the importance of establishing a connection to runtime security through the use of SAST.

B. *Deployment Auditing*

Regular deployment audits are an important part of securing the cluster against new attacks that may not have existed at the time of initial deployment. The audits can be divided into different benchmarks: i) The benchmark as a single entity, such as that of CIS, contains a series of globally recognized and consensus-based best practices, and ii) the compliance score as a benchmark. This complements individual risk scores, which are often an illusory concept and inconsistent between different frameworks. The compliance score provides a quantifiable measure of overall security about a set of specific frameworks. The compliance status percentage is calculated by averaging the control compliance scores of all controls within a single framework. Frequently used methodologies for mapping this

TABLE I

ENUMERATION OF NEAR-REALTIME RIC CONTAINERS AND THEIR COUNTED VULNERABILITIES AND MISCONFIGURATIONS

Generic Infos			Vulnerabilities					Misconfigurations				
Container Name	Registry	Image Tag	C	H	M	L	N	C	H	M	L	N
ricplt-dbass-redis	nexus3.o-ran-sc.org:10002	ric-plt-dbaas:0.6.2	6	14	26	2	0	0	1	3	9	0
influxdb2	Docker.io	influxdb:2.2.0-alpine	10	44	28	2	0	0	1	3	9	0
ricplt-e2term	nexus3.o-ran-sc.org:10002	ric-plt-e2:6.0.3	0	0	30	31	13	0	1	3	9	0
ricplt-rtmgr	nexus3.o-ran-sc.org:10002	ric-plt-rtmgr:0.9.4	0	10	119	43	19	0	1	3	9	0
ricplt-e2mgr	nexus3.o-ran-sc.org:10002	ric-plt-e2mgr:6.0.1	0	4	115	43	19	0	1	3	9	0
ricplt-submgr	nexus3.o-ran-sc.org:10002	ric-plt-submgr:0.9.5	0	10	119	43	19	0	1	3	9	0
ricplt-appmgr	nexus3.o-ran-sc.org:10002	ric-plt-appmgr:0.5.7	0	8	36	24	19	0	1	3	9	0
ricplt-a1mediator	nexus3.o-ran-sc.org:10002	ric-plt-a1:3.1.1	0	9	8	8	7	0	1	3	9	0

Vulnerabilities & Misconfiguration Scores: C ≡ Critical, H ≡ High, M ≡ Medium, L ≡ Low, N ≡ Negligible

value are, for example, the CIS as mentioned above, MITRE ATT&CK, SOC2, DevOpsBest and the NSA CISA [12].

C. Penetration Testing

With penetration testing, ethical hackers mimic the tactics of malicious actors to uncover potential vulnerabilities that might otherwise go undetected by traditional security measures. This proactive assessment is important to understand a system's vulnerability to various cyber threats. This type of security testing goes beyond the mere identification of vulnerabilities and provides a comprehensive assessment of the existing security protocols and mechanisms within, for example, an O-RAN deployment. It is important to emphasize that penetration testing is not a one-time event, but an iterative process that evolves as the threat landscape changes. This makes it an important pillar of proactive defense against security threats.

D. Runtime Security

The last of the four pillars for the secure operation of O-RAN networks is active runtime security. This is about continuously detecting unexpected behavior, configuration changes, intrusions, and data theft in real-time. Intrusion Detection Systems (IDS) are used in the conventional sense to detect anomalies during runtime. However, these must be complemented through additional tools to validate system-critical configurations and role and rights distributions at iterative intervals, for instance.

V. SECURITY CONCERNS

In the forthcoming subsections, we discuss security issues present in the existing open-source implementation of the OSC. The concerns outlined in Section V-A pertain not to a specific component but to the overall implementation status. In Section V-B, we analyze the Near-RT RIC, concentrating on a particular virtualized component. This component is deemed crucial from a system-critical perspective and, as such, serves as a potential initial target for attacks. Moreover, other components can be analyzed and evaluated using the same methodologies explained in Section IV.

It is crucial to note that the current focus of the OSC is on implementing a functional version of the O-RAN specification. Consequently, the issues we highlight may not be the community's immediate priority, as their main emphasis lies in ensuring the proper functionality of their implementation.

A. Outdated Versions

One of the main concerns is the fact that the official documentation and scripts used to install the dependencies contain very old versions that are often no longer supported. Notably, within the ric-plt-ric-dep repository, the installation script installs Kubernetes 1.16.0 from 2019, currently associated with 23 publicly available CVEs. These vulnerabilities span a Common Vulnerability Scoring System (CVSS) rating range between 3.0 and 8.8, encompassing potential threats like directory traversal, Server-Side Request Forgery (SSRF), Open Redirect, Improper Input Validation, and Denial of Service. Additionally, the referenced Kubernetes Container Network Interface (CNI) version 0.7.5 is susceptible to 9 CVEs with a CVSS rating range between 7.5 and 8.2, incorporating vulnerabilities like SSRF, Infinite Loop, and Resource Exhaustion. The Docker version specified as 20.10.21 is currently exposed to 31 CVEs with CVSS ratings ranging between 3.3 and 9.8. These issues include concerns such as Improper Certificate Validation, Integer Overflows, and Resource Exhaustion. Lastly, Helm 3.5.4, set to be installed, carries 7 released CVEs with a CVSS rating range between 4.3 and 8.6, featuring vulnerabilities like Denial of Service, Information Leakage, or Memory Corruption. While acknowledging the likelihood of telecommunications operators resolving and updating such outdated versions in an O-RAN deployment, it remains a security risk for the OSC to advocate the use of these versions in its documentation and tutorials. Entities unaware of the security risks associated with outdated versions, and consequently neglecting version checks, may unintentionally deploy insecure networks vulnerable to various malicious entry points.

B. Scanning Results

For the following assessment, we deploy the Near-RT RIC in a cluster using the latest version of Kubernetes and employ the methodologies we outline in Section IV-A and IV-B. In our analysis, we observe a cumulative total of 792 vulnerabilities, covering a range from critical to low impact ratings, summarized in Table I. The Kubernetes cluster in the ”ricplt” namespace demonstrates an average compliance score of 78% for NSA CISA, 76% for MITRE ATT&CK, and 71% for CIS-v1.23-t1.0.1. This overall assessment underscores a significant imperative for security measures. Notably, addressing the 16 critical vulnerabilities is paramount, with 10 of them enabling remote code execution. Presently, 13 of these critical vulnerabilities are actionable.

Certainly, it is important to acknowledge that these vulnerabilities need to be actively exploited by an adversary to pose a threat, and their mere existence does not imply danger. Nevertheless, given that a Kubernetes cluster is a favored target for attacks, it becomes crucial to prevent any known vulnerabilities from being exposed. Additionally, we have identified certain misconfigurations within the cluster, which are issues easily rectified through proper configuration of containers and clusters. The primary challenges encountered include: No resources memory and CPU limits, List Kubernetes secrets, Allow privilege escalation, Anonymous access enabled, Applications credentials in configuration files. Preventing such issues is a straightforward process, as tools typically offer predefined solutions.

VI. Best Practices

A. Integration of evaluation methods into the deployment

Ensuring comprehensive security for the RAN involves the integration of security assessments throughout all deployment phases. This is crucial for validating security measures and ensuring compliance with specified criteria, such as those for newly uploaded Docker images. Established tools can be utilized to examine configurations, particularly for the potential exposure of sensitive information, including patterns related to Personally Identifiable Information (PII). Additionally, it is fundamental to conduct thorough examinations on the container registry to identify and address potential open CVEs. Maintaining encrypted transmission for individual artifacts is a key necessity, ensuring basic security across all provisioning phases. Furthermore, it is crucial to actively incorporate the aspects outlined in Section IV into the deployment process. This integration can be achieved through the implementation of Continuous Integration (CI) and Continuous Deployment (CD).

B. Deployment hardening

This includes securing the Kubernetes API server, through which a malicious actor could cause a lot of damage to the environment. Furthermore, security context hardening is essential, as root users are used by default when pods are started and nothing else is stored in the configuration; this also includes the use of pod security policies. Configuring Kubernetes network policies to control communication between pods and services within the cluster helps to prevent unauthorized access and secure the communication channels.

VII. Conclusions

This paper leverages some of the most promising methodologies for assessing security in the virtualized configuration, deployment, and operation of O-Cloud instances within O-RAN deployments. We identified several issues, yet they can be resolved through appropriate and tailored configurations or by updating affected dependencies. A significant challenge in assessing the security status of an O-RAN deployment lies in the absence of a singular deployment model. Currently, numerous initiatives reference and employ varying levels of open-source implementations. Consequently, at this developmental stage, establishing a definitive status for O-RAN, crucial for making relative comparisons, proves impractical. Nevertheless, we remain optimistic about the feasibility of formulating standardized approaches to evaluate these challenges accurately.

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, Second quarter 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] M. A. Habib, H. Zhou, P. E. Iturria-Rivera, M. Elsayed, M. Bavand, R. Gaigalas, Y. Ozcan, and M. Erol-Kantarci, "Intent-driven Intelligent Control and Orchestration in O-RAN Via Hierarchical Reinforcement Learning," in *IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, 2023, pp. 55–61.

[4] L. Bonati, M. Polese, S. D'Oro, S. Basagni, and T. Melodia, "NeutRAN: An Open RAN Neutral Host Architecture for Zero-Touch RAN and Spectrum Sharing," *IEEE Transactions on Mobile Computing*, pp. 1–14, 2023.

[5] *WG11: Study on Security for O-Cloud*, O-RAN Alliance, Oct. 2023.

[6] O-RAN Software Community. I release. https://wiki.o-ran-sc.org/display/REL/I+Release. Accessed Jan 2024.

[7] 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:2304.11125*, 2023.

[8] M. Liyanage, A. Braeken, S. Shahabuddin, and P. Ranaweera, "Open RAN Security: Challenges and Opportunities," *arXiv preprint:2212.01510*, 2022.

[9] 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.

[10] 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:2201.06080*, 2022.

[11] O-RAN Software Community. Near Realtime RAN Intelligent Controller (RIC) Project. https://wiki.o-ran-sc.org/pages/viewpage.action?pageId=1179659. Accessed Jan 2024.

[12] National Security Agency and Cybersecurity and Infrastructure Security Agency, "Kubernetes Hardening Guide," Cybersecurity Technical Report, Version 1.2, August 2022.

[13] Center for Internet Security, "Kubernetes Benchmark," https://www.cisecurity.org/benchmark/kubernetes, accessed: 30/01/2024.

[14] O-RAN Alliance White Paper, "O-RAN Use Cases and Deployment Scenarios," http://tinyurl.com/58f8zbyu, February 2020.

[15] O-RAN Working Group 6, "O-RAN Cloud Architecture and Deployment Scenarios for O-RAN Virtualized RAN 2.02," O-RAN.WG6.CAD-v02.02 Technical Specification, July 2021.

[16] F. Klement, W. Liu, and S. Katzenbeisser, "Toward securing the 6g transition: A comprehensive empirical method to analyze threats in o-ran environments," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 2, pp. 420–431, 2024.

[17] "OWASP Kubernetes Top Ten," https://owasp.org/www-project-kubernetes-top-ten/, accessed: 2024-01-24.

[18] C. Banse, I. Kunz, A. Schneider, and K. Weiss, "Cloud property graph: Connecting cloud security assessments with static code analysis," in *2021 IEEE 14th International Conference on Cloud Computing (CLOUD)*, 2021, pp. 13–19.