Secure Container Orchestration in Cloud-Native Architectures

Tritons:
Marlon Brenes (1314316,mbrenesr@nyit.edu)
NYIT - Cybersecurity in Data Center Course
Spring 2025

***Abstract*—This research paper analyzes the security challenges and solutions for container orchestration within cloud-native environments. It also contrasts these approaches with traditional methodologies used in monolithic architecture applications, highlighting their security, scalability, and flexibility limitations. Emphasis is placed on Kubernetes and Docker security practices, highlighting IAM policies, network segmentation, runtime security, and compliance automation. The proposed methodology leverages policy-as-code (PAC) and runtime monitoring to strengthen container security and orchestration. It also highlights the advantages of adopting a modern approach to continuous integration and continuous development (CI/CD); furthermore, incorporating the recommendations of highly mature institutions such as NIST and CISA adds a layer of strategic complexity that significantly strengthens organizations' security posture. These organizations provide guidelines for the secure development of services and application creation and offer a comprehensive view that encompasses ongoing operations, cloud migration processes, and proactive management of cybersecurity posture. By adopting these frameworks, organizations can remain up-to-date, resilient, and aligned with industry standards, which is crucial to addressing the current challenges of the digital environment. It streamlines development times and improves coordination between operations, security, and development teams, enabling a faster and more adaptive response to changes. The results will demonstrate a measurable reduction in the attack surface within multi-tenant cloud deployments and real-time monitoring and analysis of code, containers, libraries, and orchestration components. The proposed solution also enables a highly automated environment for continuous testing and real-time metrics, facilitating data-driven discussions and stakeholder decision-making.**

I. Introduction

In IT, taking a pause and analyzing what secure container orchestration is and how it impacts cloud-native architectures is essential. As outlined in [20] ”Cloud Security Technical Reference Architecture.” CISA's recommendations, the constant threats in today's digital environment and the increasing complexity of cloud environments demand stronger strategic and technical leadership than ever. This is no longer a secondary aspect: a lack of control and security in orchestration can allow the entry of malicious actors capable of destabilizing critical operations, disrupting essential services, and even affecting entire economies. Maintaining a solid posture based on the Cloud Security Technical Reference Architecture aims to provide a comprehensive view and structured guidance for all components involved in supporting security needs throughout the various phases of the cloud lifecycle. This includes everything from cloud deployment, adopting flexible solutions based on business requirements, continuous assurance of the architecture and its secure operation, and the ability to comply with industry standards, such as the Zero Trust model.

As indicated by [20] ”Cloud Security Technical Reference Architecture.” CISA and the corresponding architectural references, security architecture in cloud environments can be divided into three key sections:

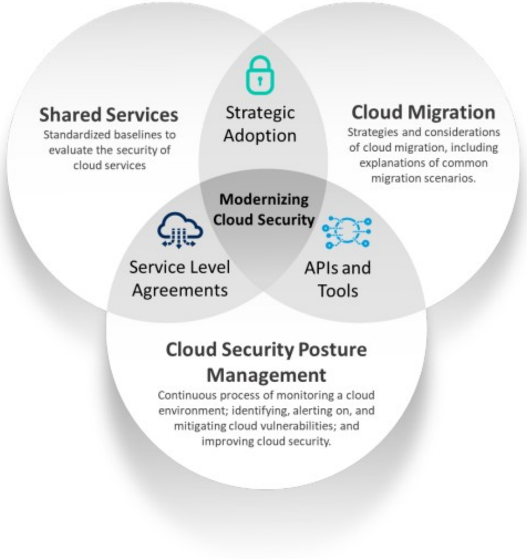
Shared Services
Standardized baselines to evaluate the security of cloud services
Strategic Adoption
Cloud Migration
Strategies and considerations of cloud migration, including explanations of common migration scenarios.
Modernizing Cloud Security
Service Level Agreements
APIs and Tools
Cloud Security Posture Management
Continuous process of monitoring a cloud environment; identifying, alerting on, and mitigating cloud vulnerabilities; and improving cloud security.

Fig. 1: Cloud Security Technical Reference Architecture Composition and Synergies, in [20].

Shared Services: This section promotes the use of industry standards and the adoption of pre-assessed baseline configurations. The goal is to ensure consistency, interoperability, and security from the most fundamental layers of the infrastructure.

Cloud Migration: This phase proposes various strategies for migrating from traditional models, such as monolithic architectures (where the entire application resides on a single server), to distributed and scalable cloud environments. Different migration scenarios are considered depending on the application type, dependency level, and business needs.

Cloud Security Posture Management (CSPM): This section defines the concept of CSPM and details the components related to cloud security posture management. It includes security, monitoring, secure development, continuous integration, risk assessment, incident management, and threat response tools in cloud environments.

In the specific case of applications that have been migrated or developed from scratch using a Platform as a Service (PaaS) architecture and technology approach, such as the use of containers with Docker and their orchestration through Kubernetes, it is important to consider that these technologies can be provided by various market providers, such as Microsoft Azure, Amazon Web Services (AWS), or Google Cloud Platform (GCP). The provider and deployment model choice will directly impact security configuration, resource governance, and integration with other enterprise solutions.

However, what is a ”dockerized” application or one inside a container? A Docker container is a standardized, executable unit of software that packages code along with all its dependencies, libraries, and configurations necessary for the application to run quickly, consistently, and reliably in any environment, whether development, testing, or production.

It is important to note that while Docker is one of the most popular technologies for containerization, it is not the only one. Other container tools and engines widely used in the industry, such as Podman, contained CRI-O, LXC, and runs, each with its features, use cases, and levels of integration with orchestration platforms like Kubernetes.

According to [18], ”Kubernetes, also known as K8s, is an open-source system for automating containerized applications' deployment, scaling, and management.” This platform allows for efficient orchestration of container management, facilitating control over the availability, scalability, and distribution of applications. In addition, it offers capabilities for handling container versioning and updates, provided it is appropriately configured, thus ensuring operational continuity and consistency in dynamic and distributed environments.

II. Background and Related Work

Numerous studies discuss container vulnerabilities and orchestration frameworks:

- Kubernetes network policies [**?**]
- Container runtime isolation mechanisms [**?**]
- RBAC models for cluster security [**?**]

III. Problem Statement

Despite the efficiency of orchestrators, they remain vulnerable to privilege escalation, pod escape, and misconfigured access controls. These challenges demand a more integrated approach to security.

IV. Proposed Approach / Methodology

A. Architecture Overview

B. Security Measures

- RBAC enforcement and IAM mapping
- Network segmentation using Calico policies
- Runtime scanning with Falco
- Policy-as-Code using Open Policy Agent (OPA)

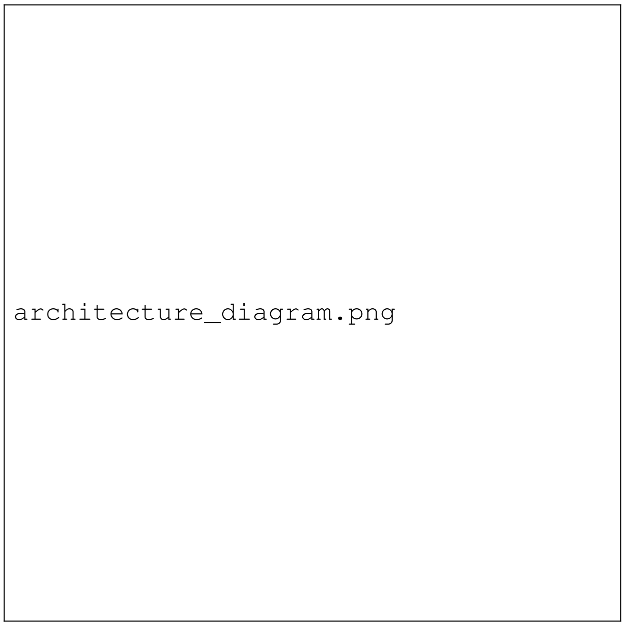

Fig. 2: Proposed Secure Orchestration Architecture

V. Analysis and Discussion

The approach was tested in a simulated AWS EKS cluster. Table I shows a reduction in detected CVEs post-policy integration.

TABLE I: Security Scan Results Before and After Hardening

Component	Before CVEs	After CVEs
Kubelet	12	3
Containerd	9	1
Ingress Controller	7	2

A. Code Example: OPA Policy

Listing 1: OPA Policy to Deny Privileged Pods

```
package kubernetes.admission

deny[msg] {
  input.request.kind.kind == ”Pod”
  input.request.object.spec.containers[_].securityC
  msg := ”Privileged pods are not allowed”
}
```

VI. Conclusion

The proposed secure orchestration framework enhances container security by enforcing least privilege, runtime visibility, and continuous compliance checks. Future work will address AI-driven anomaly detection and cross-cloud policy standardization.

VII. Aditional Points

Using kubectl.Ai to complete and retrive AI yaml autogenerated:

- Kubernetes network policies [**?**]
- Container runtime isolation mechanisms [**?**]
- RBAC models for cluster security [**?**]

References

[1] OWASP, *OWASP DevSecOps Guideline*, [Online]. Available: https://owasp.org/www-project-devsecops-guideline/.

[2] C. Feio, N. Santos, N. Escravana, and B. Pacheco, *An Empirical Study of DevSecOps Focused on Continuous Security Testing*, International Journal of DevSecOps Research, 2024.

[3] M. K. Kushwaha, P. David, and G. Suseela, *Automation and DevSecOps: Streamlining Security Measures in Financial System*, Department of Networking and Communications, SRM Institute of Science and Technology, Chennai, India, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10502917. [Accessed: Feb. 28, 2025].

[4] S. Yulianto and G. N. C. Ngo, *Enhancing DevSecOps Pipelines with AI-Driven Threat Detection and Response*, Doctor in Information Technology, School of Graduate Studies, AMA University, Quezon City 1106, Philippines, 2024. [Online]. Available: semi.yulianto2009@gmail.com, gncngo@amaes.edu.ph.

[5] National Institute of Standards and Technology (NIST), *NIST SP 800-204D: Security and Privacy Controls for Information Systems and Organizations*, NIST, 2020. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-204D.pdf. [Accessed: Feb. 28, 2025].

[6] *Implementing CSMA – the Cloud Mindset*, SecurityInfoWatch.com, Endeavor Business Media LLC, May 2024. [Online]. Available: https://www.securityinfowatch.com/cybersecurity/article/55040906/implementing-csma-the-cloud-mindset.

[7] R. Chandramouli, F. Kautz, and S. Torres-Arias, *Strategies for the Integration of Software Supply Chain Security in DevSecOps CI/CD Pipelines*, NIST Special Publication 800-204D, National Institute of Standards and Technology (NIST), 2024. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-204D.

[8] M. Fu, J. Pasuksmit, and C. Tantithamthavorn, *AI for DevSecOps: A Landscape and Future Opportunities*, ACM, vol. 1, no. 1, Art. , Sep. 2024, pp. 1-58. [Online]. Available: https://doi.org/10.1145/3712190. [Accessed: Mar. 8, 2025].

[9] Snyk Ltd., ”Snyk: Developer-first security,” Snyk Documentation, 2024. [Online]. Available: https://snyk.io. [Accessed: Mar. 8, 2025].

[10] Marlon Brenes, ”DevSecOps Project Repository.” [Online]. Available: https://github.com/BrenesRM/DevSecOps-NYIT-VAN

[11] OWASP, *OWASP Software Assurance Maturity Model (SAMM)*, [Online]. Available: https://owaspsamm.org/. Accessed: [Insert Date Accessed].

[12] GitHub, *CodeQL*, [Online]. Available: https://codeql.github.com/.

[13] Aqua Security, *Trivy*, [Online]. Available: https://github.com/aquasecurity/trivy.

[14] OWASP, *OWASP ZAP*, [Online]. Available: https://github.com/zaproxy/zaproxy.

[15] GitHub, *GitHub: Where the world builds software*, [Online]. Available: https://github.com/.

[16] PyPI, *Detect Secrets*, [Online]. Available: https://pypi.org/project/detect-secrets/.

[17] ControlPlane, *Kubesec*, [Online]. Available: https://github.com/controlplaneio/kubesec.

[18] Kubernetes, *Kubernetes Official Website*, [Online]. Available: https://kubernetes.io/.

[19] Kubernetes, *Docker Hub Official Website*, [Online]. Available: https://hub.docker.com/.

[20] CISA, *Cloud Security Technical Reference Architecture*, 2nd ed., Feb. 2023. [Online]. Available: https://www.cisa.gov/sites/default/files/2023-02/cloud_security_technical_reference_architecture_2.pdf

Appendix

- Kick-off: 26-5-2025 - Define project scope
- Week 8: 18-6-2025 - Define responsibilities in the paper: Marlon (Abstrack, Introduction and Background and Related Work)
- Week 8: 18-6-2025 - Define responsibilities in the paper: Jin (Problem Statement, Architecture Overview and Security Measures)
- Week 8: 18-6-2025 - Define responsibilities in the paper: Jason (Analysis and Discussion, example and Conclusion)
- Week 11: Review of the paper