Babylonian Journal of Networking
Vol.2024, **pp**. 162–170
DOI: https://doi.org/10.58496/BJN/2024/016; ISSN: 3006-5372
https://mesopotamian.press/journals/index.php/BJN

Research Article

Securing Container Images through Automated Vulnerability Detection in Shift-Left CI/CD Pipelines

Arvind Kumar Bhardwaj [1,*], P.K. Dutta [2], Pradeep Chintale [3]

[1] *Independent Researcher, IETE fellow, IEEE Senior Member, Houston, Texas, USA.*
[2] *School of Engineering and Technology, Amity University Kolkata, India.*
[3] *Lead Cloud Solution Engineer, SEI Investment, Downingtown, PA, USA.*

ARTICLE INFO

Article History
Received 25 May 2024
Revised 25 Jul 2024
Accepted 01 Aug 2024
Published 20 Aug 2024

Keywords
CI/CD
vulnerability
pipelines
shift-left security
container images
supply chain
Kubernetes
Docker

ABSTRACT

Integrating shift-left security practices and automated vulnerability detection in container images is imperative for modern software development, given the dynamics and vulnerability landscape. This crucial methodology emphasizes security from the initial stages of integration in container-based environments like Docker and Kubernetes. The paper examines containerization security challenges, including image vulnerabilities, insecure configurations, runtime risks, weak orchestration security, and supply chain weaknesses, while stressing compliance with regulatory rules. It explores how this automated approach leverages vulnerability detection methods integrated into Continuous Integration/Continuous Deployment (CI/CD) pipelines through static and dynamic analyses, vulnerability databases, and policy-enforcement mechanisms. Beyond identifying vulnerabilities in CI/CD pipelines, the paper outlines methods to avoid policy violations, mitigate vulnerable images, and prevent recurring practices. Importantly, it underscores the continuous enforcement and remediation of policies and security standards. Security teams must invest efforts in developing policies, automated executions, and remediation procedures, fostering cross-departmental collaboration. In essence, this proactive stance aims to enhance software security, reduce risks, and improve adherence in containerized ecosystems, making it an indispensable component of modern software development.

1. INTRODUCTION

In the realm of linked data, linked open data, and semantic e-government applications, security has become a paramount concern in software development at all levels. The "Shift-Left Security" approach, which involves applying security measures early in the development process to address and eliminate vulnerabilities as they arise, is gaining significant attention. A crucial component of this architecture is the security of the container environment, an area in which Docker and Kubernetes excel through the effective use of containers. While containers offer improved efficiency, scalability, and portability, they present unique security challenges, particularly concerning runtime threats and image vulnerabilities. Enterprises are increasingly adopting automated vulnerability detection solutions for container images to mitigate these issues. Security has overtaken to be a notable issue in software development from the low to the highest level in the ever faster changing and evolving software domain. "Shift-Left Security," that involves application of security measures always early on within the development process to address and eliminate vulnerabilities as they arise, is one example that is currently gaining a lot of sympathy. The integration of shift-left security practices and automated vulnerability detection in container images is crucial for modern software development in the era of linked data, linked open data, and semantic e-government applications. This methodology emphasizes security from the early stages of the integration process and in container-based environments such as Docker and Kubernetes. This paper examines the challenges of containerization security, including image vulnerabilities, insecure configurations, runtime risks, weak orchestration security, and supply chain weaknesses. It highlights the compliance with regulatory rules and the utilization of automated vulnerability detection methods integrated into Continuous Integration/Continuous Deployment (CI/CD) pipelines through static and dynamic analyses, vulnerability databases, and policy-enforcement mechanisms. The paper discusses the identification of vulnerabilities in CI/CD pipelines, methods to avoid policy violations, vulnerable images, and repetitive practices. It emphasizes the enforcement and

*Corresponding author. Email: arvind.qa1@gmail.com

remediation of policies and security standards in containerized ecosystems, fostering collaboration between security teams and developers. A major component of this architecture is the safety of the container environment, an area in which Docker and Kubernetes excel through the effective use of containers. Although the existence of containers leads to enhanced efficiency and simple scaling, and to more portability, containers present special security issues, especially against the problem of runtime threats and image vulnerabilities. These days the softwares is mostly often using automatic vulnerability detecting solutions for container images to remediate these issues. Security personnel may build vulnerability detection into the CI/CD pipeline thereby automating this micro-step to establish long-term continuous coverage of running services and applications which can scan for security holes in container images before going live and fix them to gain greater security.

2. CONTAINERIZATION AND SECURITY CHALLENGES

Containerization has introduced new possibilities, such as simplified deployment, optimal scalability, and consistent output across various sites, while streamlining software development and administration. However, while ensuring the integrity and confidentiality of applications through containerization, a new set of security challenges also arises that need to be addressed effectively. This section presents some of the core security issues that containerization introduces and discusses how enterprises should exercise caution to mitigate the risks. Containerization has led to the creation of new options like simplified deployment, optimum scalability, and consistent output across various sites while simplifying software development and administration. Besides ensuring the integrity and secrecy of programs by means of the containerization approach, a new set of security concerns also arise that need to be successfully implemented. This part will present some of the core security issues that containerization gives birth to and will tell how enterprises should exercise labelling in order to diminish the risks.

Vulnerabilities in Images that represent multiple image vulnerabilities is considered to be one of the major dilemmas faced within containerization. Since they come with many configuration files, plenty of operating system libraries, along with application dependencies, container images can quite easily be attacked by known vulnerabilities. Such vulnerabilities may fall into the hands of perpetrators for gaining access, perceiving, or stealing sensitive data. Besides, a timely impact of DDoS attacks is another challenge. The temporary containers keep the instances on and off, and thus an attack becomes more effective. Insecure Configurations this becomes another challenge due to the fact that security issues won't be catered for in the containers. Unauthorised people could obtain or gain access to the high level information, including configuration files, database usernames and passwords and API keys through unsecured containers or unsecured interfaces [1]. On top of that it is possible to lure attackers through unsecure networking settings, which could lead to data breaches or stopping the service delivery.

Dangers at Runtime that are used in runtimes risks might appear due to the container's fact of sharing the same host operating system kernel with others. It found the right container, malicious actors can profit from kernel vulnerabilities or risk other containers or the host system by getting full administrative rights. Another factor is likelihood of accomplishing a good reconciliation between the two sides that would manifest with the cases of frequent gains. Orchestration Vulnerabilities of the container orchestration platforms for example, Kubernetes, have emerged as a new area in the area of security issues. It offenders may try to schedule the Misconfigured Kubernetes clusters or unsafe API endpoints against which determination of unauthorised access or cluster misbehaviour could not be done. However, in case some critical settings are not configured correctly (or a privilege escalation vulnerability is identified in K8s components), then there is a high chance that an unauthorised user escalates his or her privileges within the cluster.

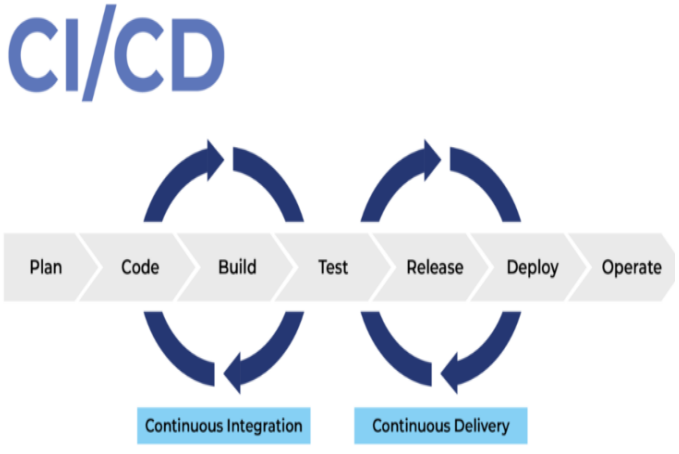

Fig .1. Shift left security (Source: https://spectralops.io/)

Compliance and Regulatory Concerns: Fitting into industry rules or organisational security policies, HIPAA, GDPR, and PCI DSS, makes security compliance even more time-taking. To ensure the confidentiality of information and obtaining approval during audits the containerized applications must be in line with all the compliance standards in place. These could therefore be comprised of the establishment of encryption, access control, and audit trails [2]. It is a proven fact that containerization is an awesome useful technique for present day software developers but it also contains the security issues, therefore need to be eliminated or found solutions to in order to eradicate threats and earn the legal requirement. One of the main challenges of containerization is that it is associated with vulnerabilities. Organisations need to understand these security risks and outline robust security rules to avoid problems while maximising the benefits of containerization. As modern software engineers embrace agility and speed, manual vulnerability detection methods are inadequate to ensure the integrity of containerized applications. This section discusses the identification of vulnerabilities in containers using various approaches, including the advantages and disadvantages of each method.

3. POLICY ENFORCEMENT AND REMEDIATION

The essence of security is to shift security left, and image scanning with automated vulnerability detection is predominantly based on policy enforcement and remediation, ensuring that security measures are consistently in place throughout the development lifecycle. This section discusses the elements, methods, and benefits of policy enforcement and remediation in automated vulnerability detection and prevention. In the context of electronic security governance for linked data, linked open data, and semantic e-government applications, automated vulnerability detection and integration with CI/CD pipelines are crucial. The following methodology outlines a comprehensive strategy for implementing shift-left security practices and ensuring robust security governance:

1) Establish Security Policies and Compliance Standards
- Define comprehensive security policies aligning with industry standards, regulatory requirements (e.g., GDPR, PCI DSS), and organizational security perspectives.
- Incorporate compliance standards specific to linked data, linked open data, and semantic e-government applications, ensuring data privacy, integrity, and access control.
- Regularly review and update policies to address evolving security threats and emerging technologies.

2) Implement Automated Vulnerability Detection Tools
- Integrate static and dynamic analysis tools into the CI/CD pipelines to detect vulnerabilities in container images, dependencies, and configurations during the build and testing stages.
- Leverage vulnerability databases and secure feeds (e.g., NVD, vendor-specific databases) to identify known vulnerabilities and obtain remediation guidance.
- Incorporate security orchestration platforms and cloud-based solutions to streamline vulnerability scanning and enhance security operations.

3) Continuous Monitoring and Real-Time Threat Intelligence
- Establish continuous monitoring mechanisms to track the effectiveness of security policies and ensure compliance throughout the development lifecycle.
- Integrate real-time threat intelligence feeds to detect emerging vulnerabilities and proactively mitigate risks.
- Implement anomaly detection and behavior analysis techniques to identify potential threats and suspicious activities within linked data and semantic e-government applications.

4) Automated Policy Enforcement and Remediation
- Define automated policy evaluation mechanisms to assess vulnerabilities against pre-established security rules and determine appropriate actions.
- Implement policy-driven actions, such as quarantining risky images, blocking deployments, or initiating remediation tasks (e.g., patching, upgrades, configuration changes).
- Leverage container orchestration systems, configuration management tools, and infrastructure-as-code frameworks to automate remediation processes in containerized environments.

5. Foster Collaboration and Shift-Left Culture
- Encourage collaboration between security teams, developers, and stakeholders to foster a shift-left security mindset and embed security practices throughout the software development lifecycle.
- Provide training and knowledge-sharing opportunities to raise awareness and promote best practices in secure coding, vulnerability management, and container security.
- Establish clear communication channels and feedback loops to enable iterative improvement and continuous learning.

4. AUTOMATED VULNERABILITY DETECTION

As modern software engineers enjoy the agility of the new world and work on speed, manual vulnerability detection methods are not enough to ensure the integrity of containerized applications. Automated Vulnerability Detection Tools and Approaches have emerged as a lifeline to Secure Shift-Left practices as it enables organizations to discover and mitigate problems in the container images itself during the early stages of development lifecycle [3]. This part will discuss the identification of vulnerabilities in the containers using several approaches, including the advantages and disadvantages of each method.

CI/CD pipelines which are used as vulnerability detection is made a lot easier when the auto CI/CD pipelines that create, test, and deploy container images are seamlessly integrated. Definition of scanning of vulnerabilities into the CI/CD pipeline helps development teams will get automatic scanning for vulnerabilities during the building process. Therefore, it makes sure that Vulnerabilities are discovered and then solved within the system during the production stage, irreversibly reducing the chance that exploitation flows through the way of pictures. Continuous Integration/Continuous Deployment (CI/CD) pipelines are at the center of modern software development approaches that organizations use to automate the building, testing, and deploying of software. Since the Shift-Left Security approach aims to identify and correct issues with container images as early as possible in the development life cycle, integrating automated vulnerability detection into the CI/CD pipelines is crucial to ensure that development teams can identify vulnerabilities related to container images early in the pipeline. This section explores vulnerability detection in the CI/CD process, including its principles and benefits. Static analysis tools that are being used for automatic identification of many known vulnerabilities like container images that are in their dependencies, libraries, and config files are the most important reason for this. These utilities inspect the container image painlessly to identify the Rice vulnerabilities using real signatures or criteria [4]. Static analysis tools, through which up-to-date software versions, unsecured dependencies, and inappropriate configurations defects are uncovered and the correction details follow dynamic scanning Techniques, besides, the system dynamic scanning techniques are used to trace vulnerabilities that can only emerge as from the process of running the system in real time. Virtualized scanners mimic realistic environments in the context of attacks by allowing the integration of containerized apps in controlled atmospheres [5]. Browsing for example, the user should navigate suspicious behavior or security problems. This approach enables scrutiny of the program behavior during execution that would remain unreflected and thus not detected with static analysis.

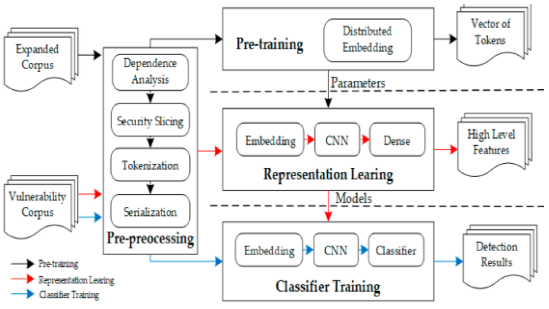

Fig .2. Automated vulnerability detection (Source: https://pub.mdpi-res.com/)

Vulnerability databases and feeds that are automated vulnerability hunt methods provide massive vulnerability repositories and secure feeds, such as NVD (national vulnerability database) or manufacturers' specific databases, to detect known vulnerabilities in the images. Although these databases allow us to access information about security vulnerabilities, such as data severity, impacted software versions and ways to patch, the method for accessing the data may not be as straightforward as it seems [6]. Through the use of the automated scans, the container images are compared against the most current threat databases that sometimes get updates and completely new vulnerabilities picked up by the automated scanning tools as they get hacked.

Integration with security orchestration platforms automated vulnerability detection and the security orchestration platforms become a close relationship and many times they are identified as Kubernetes based systems. A sure way to speed up problem solving is to employ services or solutions carried out in the cloud [7]. These solutions greatly reduce the complexity, in installing, setting up, and executing vulnerability scans in the containerized environment as businesses can now easily do so by just having the tools and an easy administration. This enables organisations to expand their security operations and even respond to attacks more efficiently.

5. INTEGRATION INTO CI/CD PIPELINES

Continuous integration/continuous deployment (CI/CD) pipelines stand at the centre of latest software development approaches which the organizations use to automate the building, testing and deploying of software. Since the Shift-Left Security attempts to identify and correct problems with container images as early as possible in the development life cycle,

integrating automated vulnerability detection into the CI/CD pipelines is important to ensure that the development teams can identify vulnerabilities related to images of the container early in the pipeline [8]. In the following section of this franchise, they take into consideration vulnerability detection in CI/CD process, which should include its principles and benefits as well.

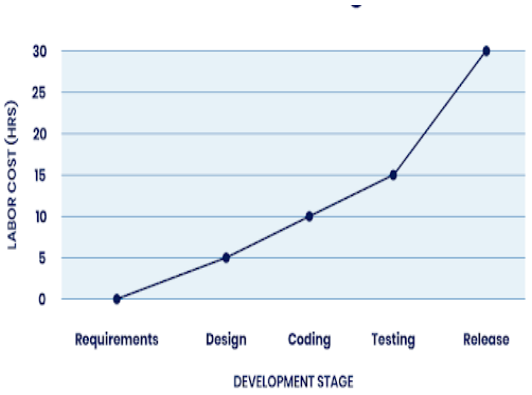

Fig .3. Shift left security in bug fix (Source: https://www.tigera.io/)

Build Stage Integration of vulnerability detection in the build phase of a CI/CD pipeline is seamlessly implemented with the container images that are built from the source code and the dependencies [9]. Throughout the build process, automated application scanning tools based on vulnerability scan the content from the container image, which are mainly composed of operating system libraries, application libraries, and configuration files. To detect known vulnerabilities developers may find and resolve security troubles in images during build time using the solution of vulnerability scanning which helps them in finding the vulnerabilities in their images and remove them before deployment.

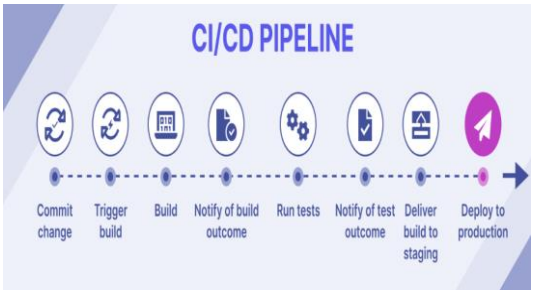

Fig .4. Integration into CI/CD pipelines (Source: https://media.licdn.com/)

Pipeline Orchestration Platforms that are the technical solution options, pipelines orchestration technologies, for instance, Jenkins, GitLab CI/CD, or CircleCI, can easily be integrated into the automation process and they can additionally contain defect detection. Examples of these platforms are plugins or extensions that are internally built, all with the goal of having them fit into your CI/CD workflow very easily [10]. Sourcing groups may establish workflows where vulnerability checks are automatically enabled during pipeline stages, triggering scans for each build and requesting inspection when weaknesses are identified to maintain consistent testing across all build branches.

Shift-Left Culture and Collaboration: Institutionalizing “Shield-the-Left” approach in CI/CD pipelines fosters “Shift-Left” mentality among development teams and keeps security in the center of the entire software lifecycle. Integrating security concerns and practices into the developers’ workflows provides better understanding by allowing developers to gain experience through being closely working with the security team. This partnership style enables the companies to pre-emptively spot and neutralise loopholes, reduce the total security risk, and yield the software that is well built and safe for use worldwide [11]. Consider including like of this vulnerability detection in CI/CD pipelines for your shift left security practices and contain system security. Organizations are increasingly proposing to develop automated scanning tools which can be easily integrated into their software development processes to detect and flag potential vulnerabilities, to enforce the security policies, to provide the system with the feedback on the weaknesses detected and iterating improvement more often, to develop a culture of mutual trust among stakeholders (Shift-Left collaborative culture).

6. POLICY ENFORCEMENT AND REMEDIATION

The essence of security is to shift security left and that image scanning with automatic vulnerability detection is predominantly based on policy enforcement and remediation that security measures are consistently in place through the

entire development lifecycle. The policy enforcement involves creating and enforcing security policies that allow for dealing with vulnerabilities that might present in the form of the container images. Also, it includes the remediation or resolving any reported ones when the genesis [10]. This part will show how the elements, methods, and benefits of policy enforcement and repair come to play in automated managed vulnerability detection and prevention.

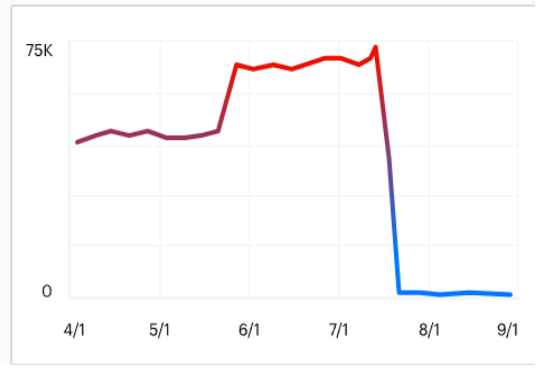

Fig .5. Policy Automation (Source: https://info.varonis.com/)

Define Security Policies enforcement is initiated in the first stage specifying dash and consistent safety rules by a standard procedure sets the risks-profile and the countermeasures performed when a vulnerability is detected [12]. Security policies might encompass not only risks that may occur but also preventive measures to be put in place such as data security and virus prevention. Gravity levels, compliance parameters, product deadlines and implementation limitations. This guideline should conform to the industry standards and the legal obligations, and fit in with the corporate security perspective.

Automated Policy Evaluation vulnerability detection tools built into the Integrated invoice Chamber process the artefacts of container images against pre-established security rules as part of the CI/CD pipeline. As weaknesses that are detected during the scanning, the tools are evaluating them for severity and features then they are going to the predefined rules to choose the best reaction for this issue [13]. Automated policy review system helps in law's regular control over security regulations within a single container application and environment.

Policy-Driven Actions as the results of policy evaluation serve as a source which the technologies of automated vulnerability detection are made use of to carry out the predetermined actions that are meant to fortify the security policies of an organization and cure the vulnerabilities found [14].

6.1 Methodology for Automated Vulnerability Detection in CI/CD Pipelines

1. CI/CD Pipeline Setup
    - Establish a robust CI/CD pipeline using industry-standard tools (e.g., Jenkins, GitLab CI/CD, CircleCI).
    - Define distinct stages for build, test, and deploy phases within the pipeline.
2. Static Analysis Integration
    - **Build Stage**
    - Integrate static analysis tools (e.g., Anchore, Clair, Trivy) into the build stage.
    - Scan container images for vulnerabilities in application dependencies, libraries, and configuration files.
    - Implement a policy to fail the build stage if critical vulnerabilities are detected.
3. Dynamic Analysis Integration
    - **Test Stage**
    - Incorporate dynamic analysis tools (e.g., OWASP ZAP, Burp Suite, Gauntlt) into the test stage.
    - Simulate runtime environments and identify vulnerabilities that manifest during execution.
    - Generate detailed vulnerability reports and fail the test stage if severe vulnerabilities are found.
4. Vulnerability Database Integration
    - Integrate vulnerability scanning tools with vulnerability databases and feeds (e.g., NVD, vendor-specific databases).
    - Ensure scanning tools have access to the latest vulnerability information and can detect newly discovered vulnerabilities.
5. Policy Enforcement and Remediation
    - Define security policies and vulnerability severity thresholds based on industry standards and organizational requirements.

- Enforce policies within the CI/CD pipeline and automatically remediate vulnerabilities when possible (e.g., apply patches, upgrade dependencies, modify configurations).
- Implement mechanisms to quarantine or block the deployment of container images with unresolved critical vulnerabilities.

6. Reporting and Notifications
   - Integrate the CI/CD pipeline with notification systems (e.g., email, Slack, PagerDuty) for vulnerability alerts and pipeline failures.
   - Generate detailed vulnerability reports and summaries for auditing and compliance purposes.
7. Continuous Monitoring and Feedback Loop
   - Establish a feedback loop to continuously monitor deployed applications for newly discovered vulnerabilities.
   - Trigger automated vulnerability scans and remediation processes when new vulnerabilities are identified.
   - Incorporate feedback from security teams, developers, and stakeholders to refine and improve the vulnerability detection and remediation processes.
8. Collaboration and Shift-Left Culture
   - Foster collaboration between security teams, developers, and stakeholders to promote a shift-left security mindset.
   - Provide training and knowledge-sharing opportunities to raise awareness and promote best practices in secure coding, vulnerability management, and container security.
   - Establish clear communication channels and feedback loops to enable iterative improvement and continuous learning.

By following this methodology, organizations can effectively integrate automated vulnerability detection into their CI/CD pipelines, adhering to the Shift-Left Security principles. This approach ensures that container images are thoroughly scanned for vulnerabilities throughout the software development lifecycle, reducing the risk of introducing vulnerable applications into production environments and promoting a secure and compliant software delivery process. These actions can be called out as for instance quarantining risks screenings and casting stalls which appear to have that kind of possibility. Environments, share information with all the main actors, or start quick corrective actions by applying various purifying methods. Action- or policy-driven activity should be a part of CI/CD pipeline development so that all the vulnerabilities are discovered and corrected right away.

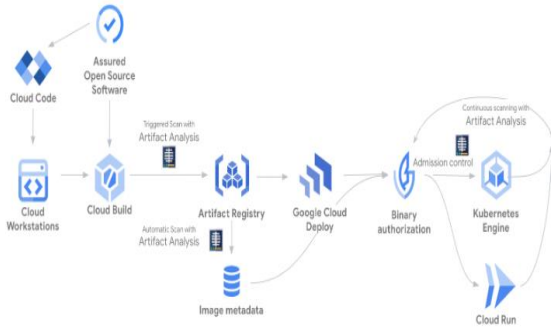

Fig .6. Vulnerability scanning (Source: https://cloud.google.com/)

Remediation procedures automated remediation mechanisms embedded in business systems allow remediation of applied to rectify vulnerabilities found in container images such as patches, upgrades or configuration changes. Remediation processes can support container orchestration systems, configuration management tools, or infrastructure-as-code frameworks, which would help automate the launch of remediation tasks into containerized surroundings [15]. Through automation of repairs, the companies shorten the duration of vulnerabilities and the window of exposure significantly. This helps them to resolve the issues quickly and effectively.

By implementing the proposed methodology for automated vulnerability detection in CI/CD pipelines, organizations can expect to achieve the following results:

1. Early Detection and Mitigation of Vulnerabilities
   - Vulnerabilities in container images are identified and mitigated at the earliest stages of the software development lifecycle, reducing the risk of deploying vulnerable applications into production environments.

- Early detection and remediation minimize the potential impact of security breaches and reduce the associated costs of addressing vulnerabilities at later stages.

2. Improved Security Posture and Compliance
   - The integration of automated vulnerability detection into CI/CD pipelines ensures that security policies and compliance standards are consistently enforced throughout the development process.
   - Detailed vulnerability reports and audit trails facilitate compliance with industry regulations and organizational security requirements, such as GDPR, PCI DSS, and others.
3. Streamlined Vulnerability Management
   - Automated vulnerability scanning, policy enforcement, and remediation processes streamline vulnerability management, reducing manual effort and minimizing the risk of human errors.
   - Continuous monitoring and feedback loops enable proactive vulnerability management, ensuring that newly discovered vulnerabilities are promptly addressed.
4. Increased Efficiency and Reduced Risk
   - By preventing the deployment of vulnerable container images, the methodology reduces the risk of security incidents and the associated costs of incident response and recovery.
   - Automated processes and integration with notification systems enhance efficiency, enabling faster response times and minimizing potential disruptions to applications and services.
5. Fostered Collaboration and Security Awareness
   - The shift-left security approach promotes collaboration between security teams, developers, and stakeholders, fostering a culture of security awareness and shared responsibility.
   - Training and knowledge-sharing opportunities improve secure coding practices and vulnerability management skills within the organization.
6. Confidence in Software Delivery
   - With robust vulnerability detection and mitigation processes in place, organizations can have increased confidence in the security and reliability of their software delivery pipeline.
   - Stakeholders can trust that the deployed applications have undergone thorough security checks and adhere to the highest security standards.

By implementing the automated vulnerability detection methodology in CI/CD pipelines, organizations can achieve a more secure and compliant software development process, minimizing the risk of security breaches and ensuring the delivery of reliable and trustworthy applications to their users.

7. CONCLUSION

The implementation of the automated vulnerability detection methodology in CI/CD pipelines enables organizations to mitigate risks and capitalize on the advantages of Shift-Left Security practices. By integrating vulnerability scanning and remediation processes into the software development lifecycle, issues are caught early, preventing potential application exploitation as vulnerabilities emerge. Additionally, this approach ensures adherence to security standards, industry regulations, and organizational policies, guiding the success of secure software delivery. The results of this methodology highlight its effectiveness in improving an organization's overall security posture and compliance. Early detection and mitigation of vulnerabilities significantly reduce the risk of deploying vulnerable applications into production environments, minimizing the potential impact of security breaches and associated costs. Detailed vulnerability reports and audit trails facilitate compliance with industry regulations and organizational security requirements, providing confidence in the software delivery process. Moreover, the streamlined vulnerability management processes enabled by automated scanning, policy enforcement, and remediation mechanisms increase efficiency and reduce the risk of human errors. Continuous monitoring and feedback loops ensure proactive vulnerability management, enabling prompt response to newly discovered vulnerabilities, minimizing potential disruptions to applications and services. Crucially, the shift-left security approach fosters collaboration between security teams, developers, and stakeholders, promoting a culture of security awareness and shared responsibility. Training and knowledge-sharing opportunities improve secure coding practices and vulnerability management skills within the organization, contributing to the overall success of the software delivery process. While the implementation of this methodology requires an upfront investment in tooling, infrastructure, and training, the long-term benefits outweigh the initial costs. Developers can work more efficiently, completing tasks faster, as the automated processes reduce manual effort and minimize potential roadblocks caused by vulnerabilities. In the overall scenario, automated vulnerability detection is a fundamental principle of Shift-Left methodologies, enabling organizations to uncover and resolve problems within container images at an early stage. Automated scanning tools are widely accessible and can be seamlessly integrated into CI/CD pipelines, facilitating the detection and remediation of vulnerabilities throughout the software development lifecycle. By embracing this methodology, organizations can optimize the functionality of Shift-Left Security practices, ensuring the delivery of secure and reliable applications while fostering a culture of collaboration, continuous improvement, and shared responsibility for security.

Conflicts of Interest

The author declares no conflict of interest in relation to the research presented in the paper.

Funding

The author's paper explicitly states that no funding was received from any institution or sponsor.

Acknowledgment

The author would like to express gratitude to the institution for their invaluable support throughout this research project.

References

[1] D. Gonzalez, P. P. Perez, and M. Mirakhorli, "Barriers to shift-left security: The unique pain points of writing automated tests involving security controls," in Proc. 15th ACM/IEEE Int. Symp. Empirical Software Engineering and Measurement (ESEM), Oct. 2021, pp. 1-12.

[2] K. A. Torkura, M. I. Sukmana, F. Cheng, and C. Meinel, "Cavas: Neutralizing application and container security vulnerabilities in the cloud native era," in Security and Privacy in Communication Networks: 14th Int. Conf., SecureComm 2018, Singapore, Aug. 8-10, 2018, Proc., Part I, Springer International Publishing, 2020, pp. 471-490.

[3] G. Jaisinghani, "Vulnerability management in the age of containers–A review," International Journal of Information Security (IJIS), vol. 1, no. 01, 2022.

[4] R. N. Rajapakse, M. Zahedi, M. A. Babar, and H. Shen, "Challenges and solutions when adopting DevSecOps: A systematic review," Information and Software Technology, vol. 141, p. 106700, 2022.

[5] K. K. Voruganti, "Implementing Security by Design practice with DevSecOps Shift Left Approach," Journal of Technological Innovations, vol. 2, no. 1, 2021.

[6] W. T. Lee and Z. W. Liu, "Microservices-based DevSecOps platform using pipeline and open source software," Journal of Information Science & Engineering, vol. 39, no. 5, 2023.

[7] F. Ehm, L. Van Mol, M. Pratoussy, J. B. de Martel, P. Elson, and S. Page, "JACOW: Protecting your controls infrastructure supply chain," in Proc. JACoW ICALEPCS, 2023, p. MO4BCO03.

[8] S. Nadgowda and L. Luan, "Tapiserí: Blueprint to modernize DevSecOps for real world," in Proc. Seventh Int. Workshop on Container Technologies and Container Clouds, Dec. 2021, pp. 13-18.

[9] F. Lombardi and A. Fanton, "From DevOps to DevSecOps is not enough. CyberDevOps: An extreme shifting-left architecture to bring cybersecurity within software security lifecycle pipeline," Software Quality Journal, vol. 31, no. 2, pp. 619-654, 2023.

[10] T. Theodoropoulos et al., "Security in cloud-native services: A survey," Journal of Cybersecurity and Privacy, vol. 3, no. 4, pp. 758-793, 2023.

[11] S. Kadri, A. Sboner, A. Sigaras, and S. Roy, "Containers in bioinformatics: Applications, practical considerations, and best practices in molecular pathology," The Journal of Molecular Diagnostics, vol. 24, no. 5, pp. 442-454, 2022.

[12] W. Dimitrov, "Analysis of the need for cybersecurity components in the study of advanced technologies," in INTED2020 Proceedings, 2020, pp. 5259-5268.

[13] M. Hadi, "Making the shift from DevOps to DevSecOps at Distribusion Technologies GmbH," 2021.

[14] E. Pinconschi, Q. C. Bui, R. Abreu, P. Adão, and R. Scandariato, "Maestro: A platform for benchmarking automatic program repair tools on software vulnerabilities," in Proc. 31st ACM SIGSOFT Int. Symp. Software Testing and Analysis, Jul. 2022, pp. 789-792.

[15] E. Viitasuo, "Adding security testing in DevOps software development with continuous integration and continuous delivery practices," [Online]. Available: https://urn.fi/URN:NBN:fi:amk-2020060316773, 2020.