Analyzing Docker Vulnerabilities through Static and Dynamic Methods and Enhancing IoT Security with AWS IoT Core, CloudWatch, and GuardDuty

Vishnu Ajith *, Tom Cyriac *, Chetan Chavda *, Anum Tanveer Kiyani , Vijay Chennareddy , Kamran Ali *

Posted Date: 24 June 2024

doi: 10.20944/preprints202406.1507.v1

Keywords: IoT; Docker; Security; AWS; Trivy; Falco; AWS IoT Core; AWS CloudWatch; AWS GuardDuty

Article

Analyzing Docker Vulnerabilities through Static and Dynamic Methods and Enhancing IoT Security with AWS IoT Core, CloudWatch and GuardDuty

Vishnu Ajith *, Tom Cyriac *, Chetan Chavda *, Anum Tanveer Kiyani, Vijay Chennareddy [5] and Kamran Ali *

Middlesex University

* Correspondence: va406@live.mdx.ac.uk (V.A.); tc840@live.mdx.ac.uk (T.C.); cc1855@live.mdx.ac.uk (C.C.); k.ali@mdx.ac.uk (K.A.)

Abstract: During our digital transition, Docker containers have become essential for adaptable application deployment, although posing inherent security threats. This study utilizes Trivy for static analysis and Falco for dynamic analysis to identify and address vulnerabilities in Docker environments before deployment and in real-time. The IoT sector is now dealing with security issues that are made more severe by extensive connectivity and the prevalence of WiFi. This research promotes the use of AWS cloud security technologies such as IoT Core, CloudWatch, and GuardDuty to protect IoT devices from threats, with the goal of maintaining user privacy and ensuring safety in our interconnected digital world. This comprehensive strategy aims to enhance security measures in both Docker and IoT sectors, strengthening the security framework in our interconnected world.

Keywords: IoT; Docker; security; AWS; Trivy; Falco; AWS IoT Core; AWS CloudWatch; AWS GuardDuty

1. Introduction

The rapid advancement of digital technologies, including Docker and the Internet of Things (IoT), has greatly boosted IT infrastructure by enhancing operational efficiencies. Yet, these progressions have also brought forth crucial security weaknesses that require prompt and thorough mitigation plans. This study attempts to investigate vulnerabilities in Docker by utilizing static and dynamic analysis tools like Trivy and Falco. It also seeks to improve IoT security by leveraging AWS cloud security services such as IoT Core, CloudWatch, and GuardDuty. This research aims to enhance the security of Docker and IoT platforms by creating a strong security architecture, making a substantial contribution to the cybersecurity sector. It highlights a proactive strategy for cybersecurity, adjusting to the ever-changing landscape of digital risks. This paper positions itself within the broader academic discourse on Docker and IoT security vulnerabilities, analysis methods, and cloud security services by referencing current literature. The study focuses on evaluating the usefulness of analysis tools in detecting Docker vulnerabilities and the impact of AWS services on IoT security through qualitative analysis. This introduction prepares for an in-depth examination of cybersecurity measures within the realm of advanced digital technologies.

2. Literature Review

2.1. Overview and Motivation

The basis for this review is the recognition that traditional security standards and evaluation frameworks that are designed for non-IoT settings are vastly different and many may not directly address the security requirements of IoT-based smart environments. As a result, this paper explores

 doi:10.20944/preprints202406.1507.v1

the potential of conventional security standards and assessment frameworks to tackle some of the security concerns in IoT-based smart environments by highlighting some core areas of focus as well as the background of the field.

Although the benefits and possibilities of an expanded IoTbased smart environment are significant and still growing rapidly, the attack surface is also broad. Thus, with the increasing number of IoT devices, ecosystems and integrations, many vulnerable endpoints are being identified regularly, particularly in smart homes, smart cities, global businesses, and critical infrastructures. While IoT-based smart environments are an ever-growing trend, expansion comes with a lot of complexity, integration, and security challenges across various application domains. Given these factors, a review of existing conventional security standards and assessment frameworks is vital to identify the key and ongoing security concerns in IoT-based smart environments.

Hence, the pivotal findings and conclusions of this study are expressly derived from harnessing the principles embedded within established conventional security standards and assessment frameworks, recognized for their applicability within IoT-driven smart environments. Additionally, the study delineates and deliberates on unresolved issues and hurdles, concurrently suggesting a taxonomy of challenges specific to IoTbased smart environments. This taxonomy maps the identified challenges to prospective solutions aimed at mitigating current and anticipated security concerns in the realm of IoT.

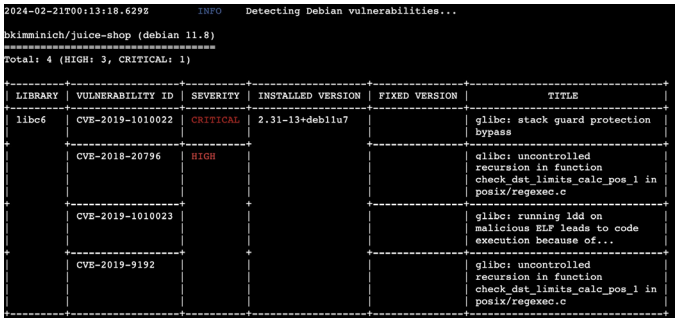
The advent of the Internet of Things (IoT) has sparked a transformative shift in the realm of networks, owing to its vast array of intelligent applications. These applications encompass various domains such as healthcare, energy grids, banking, and a myriad of other services, collectively shaping a smarter future. Like many other fields, the IoT domain is growing very fast. However, with this growth come many cybersecurity challenges (Karie et al., 2021).

2.2. Security Challenges

The Internet of Things has revolutionized the way we live and interact with technology (Singh, J. et al., 2016). However, with this increased connectivity and integration of devices, there are numerous security concerns that need to be addressed. These concerns stem from the fact that IoT devices collect and transmit vast amounts of data, making them attractive targets for hackers and cybercriminals. Some of the major security concerns in the evolving landscape of the Internet of Things include: 1. Unauthorized access: With the growing number of connected devices in the IoT ecosystem, unauthorized access becomes a major concern. Hackers may exploit vulnerabilities in IoT devices to gain unauthorized access and control over them, potentially leading to various malicious activities such as data theft, privacy breaches(Karie, M, N. et al., 2021), or even physical harm (Wu, Y., 2021). 2. Data breaches: IoT devices collect and transmit a significant amount of sensitive data, including personal and financial information. Therefore, data breaches pose a significant risk in the IoT landscape. Hackers may intercept or manipulate this data, leading to identity theft, financial fraud, or other serious consequences. 3. Lack of encryption: IoT devices often lack robust encryption mechanisms to protect the data they transmit. This makes them vulnerable to interception and unauthorized access. 4. Inadequate firmware and software security: Many IoT devices have vulnerabilities in their firmware or software that can be exploited by attackers. These vulnerabilities can allow hackers to gain unauthorized access to the device, manipulate its functionality, or use it as a gateway to attack other devices on the network. 5. Lack of standard security protocols: The IoT ecosystem lacks uniform security protocols and standards, making it challenging to ensure consistent security across different devices and networks (Azrour, M. et al., 2021). 6 (Wu, Y., 2021). Insufficient authentication and access controls: Weak authentication mechanisms and inadequate access controls make it easier for attackers to impersonate authorized users or gain unauthorized access to IoT devices and networks (Azrour, M. et al., 2021). 7. Insufficient device management and updates: Many IoT devices lack proper management systems and mechanisms for regular software updates. This leaves them vulnerable to new security threats that may arise over time, as there are no mechanisms in place to patch vulnerabilities or fix software bugs (Singh, J. et al., 2016). Addressing these security concerns is crucial to the successful and safe implementation of the Internet of Things (Azrour, M. et

 doi:10.20944/preprints202406.1507.v1

al., 2021). The Figure 1. of this paper represents an overview of the key aspects that motivated this study, which also form primary focus of the study.



```
2024-02-21T00:13:18.629Z        INFO    Detecting Debian vulnerabilities...

bkimminich/juice-shop (debian 11.8)
===================================
Total: 4 (HIGH: 3, CRITICAL: 1)

+---------+------------------+----------+-------------------+---------------+-------------------------------+
| LIBRARY | VULNERABILITY ID | SEVERITY | INSTALLED VERSION | FIXED VERSION |             TITLE             |
+---------+------------------+----------+-------------------+---------------+-------------------------------+
| libc6   | CVE-2019-1010022 | CRITICAL | 2.31-13+deb11u7   |               | glibc: stack guard protection |
|         |                  |          |                   |               | bypass                        |
+         +------------------+----------+                   +               +-------------------------------+
|         | CVE-2018-20796   | HIGH     |                   |               | glibc: uncontrolled           |
|         |                  |          |                   |               | recursion in function         |
|         |                  |          |                   |               | check_dst_limits_calc_pos_1 in |
|         |                  |          |                   |               | posix/regexec.c               |
+         +------------------+          +                   +               +-------------------------------+
|         | CVE-2019-1010023 |          |                   |               | glibc: running ldd on         |
|         |                  |          |                   |               | malicious ELF leads to code   |
|         |                  |          |                   |               | execution because of...       |
+         +------------------+          +                   +               +-------------------------------+
|         | CVE-2019-9192    |          |                   |               | glibc: uncontrolled           |
|         |                  |          |                   |               | recursion in function         |
|         |                  |          |                   |               | check_dst_limits_calc_pos_1 in |
|         |                  |          |                   |               | posix/regexec.c               |
+---------+------------------+----------+-------------------+---------------+-------------------------------+
```

Figure 1. Trivy scan command-line output showing detailed vulnerability information.

1) *Hacking Techniques:* As we look at the benefits of what IoT devices bring to daily life, be it personal or in the enterprise environment. It is important to note that these devices come with their own vulnerabilities. Malicious actors or hackers can employ several techniques in order to take advantage of weaknesses in IoT device. The malicious actors most often employ the following methods: 1. Remote Code Execution: Viral code can be remotely executed by attackers by taking advantage of flaws in IoT devices' firmware or software. Once in control of the device, they can alter its operations, pilfer data, or utilize it as a component of a botnet to launch coordinated assaults. 1. Man-in-the-Middle Attacks:

In such attacks, the communication between IoT devices and their intended recipients is intercepted and altered by the attacker. They may modify the access to private information being transmitted, introduce malicious commands or code, or gain unauthorized by doing this. According to the code snippet in Figure 2 and the research carried out by (Ferrara et al., 2020) using Rain Monitor app, "it uses OpenXC to collect sensitive data (such as location, wind-shield status, and speed) of a car, and transmit it to a remote Web service, where it is collected and used to inform drivers of possible showers in their area." It very much evident, that the app contains a leakage of sensitive data to Internet, which can be considered as a great privacy issue.

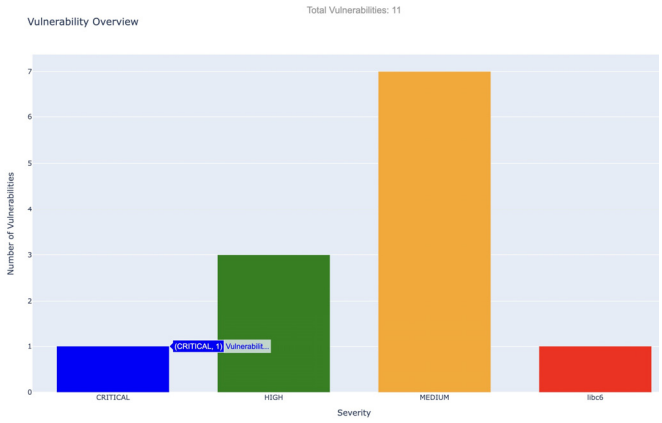


Figure 2. Graphical representation of vulnerabilities detected in the Docker container image.

2) *Challenges in Stopping IoT Attacks:* In recent years, the Internet of Things has seen exponential growth and adoption in various industries and sectors. However, along with this growth comes an increase in security challenges and vulnerabilities. As highlighted in a study conducted by Andrea et al., the interconnected nature of IoT devices creates new attack surfaces and potential entry points for malicious actors (Malhotra, P. et al., 2021). These attacks can lead to serious consequences, including data breaches, privacy violations, and disruptions in critical infrastructure. One of the challenges in stopping IoT attacks is the lack of comprehensive research on smart contract vulnerabilities and their incorporation in IoT systems Another challenge identified in the literature is the issue of maintaining privacy and ensuring data

 doi:10.20944/preprints202406.1507.v1

security. Studies by Aleisa and Renaud, as well as Ziegeldorf et al.and Tawalbeh, emphasize the inadequate research on privacy threats in IoT and the need for legislation to address these concerns (Karale, A., 2021).Additionally, the complexity of IoT systems and the sheer number of devices connected can make it difficult to detect and respond to attacks in a timely manner. Furthermore, the lack of standardization in data sharing and collection mechanisms performed by IoT devices poses a challenge in addressing security vulnerabilities. Furthermore, the authors also state that informed consent, privacy, information security, physical safety and trust have set up literature and are valuable foundations that underpin ethical research practices in the realm of IoT security challenges.

2.3. Initiatives and Solutions

The growth of the Internet of Things brings forward numerous security challenges, which have prompted several initiatives and solutions some of which are discussed in the following section.

1) *Intrusion Detection Systems:* Machine learning and deep learning-based intrusion detection systems are being developed to spot unusual network traffic patterns and potential threats in IoT networks. The idea is to use intelligent algorithms that can learn from a vast amount of data and identify anomalies that may indicate a cybersecurity threat (Idrissi, I., Azizi, M. and Moussaoui, O., 2020).
2) *Secure Boot:* Secure Boot is a fundamental security standard that IoT devices can utilize to prevent the execution of unauthorized software during the device's booting process, which helps in protecting against a wide range of attacks (Malhotra, P. et al., 2021).
3) *Encryption:* Ensuring that data transmitted and stored by IoT devices is encrypted to protect it from being intercepted and read by unauthorized entities (Malhotra, P. et al., 2021).
4) *Device Authentication:* Implementing strong, multifactor authentication mechanisms for devices to validate their identity, preventing unauthorized devices from gaining access (Malhotra, P. et al., 2021).
5) *Regular Updates and Patches:* Keeping IoT devices and software updated with the latest security patches to address known vulnerabilities (Karale, A., 2021).
6) *Education and Awareness:* Increasing awareness among both users and developers regarding the importance of security in the IoT context is vital to the overall safety of these devices (Karale, A., 2021).
7) *Standardization:* Emphasizing the development of global security standards for IoT devices to set a clear security benchmark for manufacturers to meet (Idrissi, I., Azizi, M. and Moussaoui, O., 2020). Utilising a multifaceted strategy incorporating various technological measures, regulatory frameworks, standardisation efforts, and educational initiatives stands as imperative in cultivating a secure Internet of Things (IoT) ecosystem and grappling with the intricate security quandaries it encounters. The proliferation of threats in this domain is incessant, with ongoing attacks reaching unprecedented levels, potentially numbering in the billions. Concurrently, market dynamics exert significant influence over the development of IoT products, introducing inherent risks. The paramount concern herein lies not only in safeguarding users' privacy but also in thwarting criminal exploitation of information that could pose threats to safety. The overarching message conveyed by this study resonates not only with technology professionals but also with end-users, underlining the collective responsibility in fortifying IoT security.

Docker's incorporation into software deployment has transformed the DevOps sector by improving the efficiency of application distribution in various situations. Nevertheless, this change has brought forth notable security issues such as container breakout hazards, image vulnerabilities, and network misconfigurations. Tools like as Trivy and Falco have played a crucial role in doing static and dynamic evaluations, providing a means to address these vulnerabilities prior to deployment.

Docker utilizes many security measures, including namespaces for isolating processes, control groups for limiting resources, Docker Secrets for managing sensitive information, and Docker Bench for Security for evaluating compliance. Although efforts have been made to enhance Docker's

 doi:10.20944/preprints202406.1507.v1

security, there are still vulnerabilities due to its fast-paced development and integration of new functionalities.

Comparative assessments show that Docker has a distinct security position compared to traditional VMs and other orchestration solutions such as Kubernetes. This highlights Docker's efficient security strategy as well as its possible weaknesses. Further study is needed to investigate the longterm effectiveness of vulnerability scanning tools and to create improved intrusion detection systems specifically designed for containerized settings, as indicated in the literature.

This analysis emphasizes the importance of ongoing innovation in Docker security measures to tackle the intricate, changing environment of cyber threats. It recommends a comprehensive strategy involving technology, policy, and education to improve the security of containerized applications [21].

3. Comprehensive Security Mechanisms in Docker Deployment: Managing Vulnerabilities and Threats

3.1. Introduction to Docker

Docker is an open-source platform that enables developers to create, deploy, and oversee programs in lightweight, adaptable containers. These containers package an application along with all its dependencies to maintain uniformity across various computing environments and streamline development, testing, and deployment procedures. Docker's containerization technology is a fundamental aspect of modern DevOps processes, providing remarkable flexibility and mobility. Nevertheless, the implementation of Docker containers brings about security obstacles. Failure to control vulnerabilities in Docker can undermine its isolation capabilities. Docker Inc. has improved its platform's security by implementing robust measures including Docker Secrets for securely handling sensitive information and Docker Bench for Security, which performs automated compliance checks against recognized security standards. Implementing these strategic enhancements is crucial for integrating security across the container lifecycle, protecting containerized applications from advancing cyber threats. It is essential to make strategic enhancements to incorporate security throughout the whole lifecycle of containers to safeguard containerized applications against advancing cyber threats, as detailed in Docker's official documentation [1,2].

3.2. Docker's Security Landscape

Docker containers, known for their efficiency and adaptability, persist in encountering unique security issues that are not fully resolved by traditional security approaches. The advancement of Docker technology has not alleviated these issues, which primarily stem from its inherent design features. Recent studies and developments have further highlighted these security concerns.

- Shared Kernel Architecture: Docker containers leverage the host system's kernel, distinguishing them from virtual machines. This shared kernel architecture economizes resources but introduces substantial security vulnerabilities, especially in multi-tenant systems where containers from various users share the same host kernel [3].
- Security Challenges of Short-lived Containers and Image Use: Containers' temporary and short-lived nature, often created from images, presents unique security challenges. These container images can harbor vulnerabilities, and their ephemeral existence complicates the application of traditional, long-term security measures [4].
- Isolation and Resource Sharing: While containers provide self-contained environments, they share common resources like network and storage with other containers on the same host. Inadequately secured resource sharing can lead to unauthorized data access [5].
- Container Sprawl: The ease of deploying Docker containers can result in container sprawl, where an unmanaged proliferation of containers leads to operational complexity and security oversight. Proper governance and lifecycle management are necessary to mitigate the risks associated with outdated or unnecessary containers [6].

 doi:10.20944/preprints202406.1507.v1

3.3. Security Architecture and Vulnerability Assessment in Docker Environments

1) *Static and Dynamic Security Assessment Strategy:* Our study outlines a two-pronged approach to thoroughly evaluate the security of Docker containers in a cloud setting, utilizing both static and dynamic analysis techniques. This concept is implemented in a practical environment by using Docker containers managed on Amazon Web Services (AWS) EC2 instances, replicating real-world cloud computing situations.
2) *Static Analysis using Trivy:* Trivy is a crucial opensource vulnerability scanner specifically created for container images to enhance container security in static analysis frameworks. The scanning capabilities are extensive and cover Docker images, filesystems, Git repositories, Infrastructure as Code (IaC) files such as Terraform, Kubernetes manifests, and AWS CloudFormation templates, along with application dependencies in several programming languages. Trivy's multidimensional methodology allows it to efficiently find vulnerabilities prior to deployment, effortlessly integrating into CI/CD pipelines for early detection. Trivy ensures accurate identification of security risks in fast-changing digital environments by keeping an updated vulnerability database, which helps minimize false positives. Trivy's wide scanning spectrum supports its role in improving the security of container images and related infrastructure, ensuring strong protection against various cyber attacks.

We extensively depend on Trivy in our research to uncover vulnerabilities of different severity levels in Docker images. Early detection facilitated by Trivy is crucial for prompt resolution and improving the security protocols of container images before they are used in production. We have enhanced the functionality of Trivy by creating a custom Python script to interpret, assess, and graphically display the output data from Trivy. This script simplifies the process of generating an AWShosted dashboard that provides a user-friendly and thorough overview of the security status of Docker images being assessed. The dashboard showcases vulnerabilities, categorizes them based on severity, and offers remedy insights, greatly assisting in the proactive handling of container security.

a) Results and Metrics for Trivy: The static examination conducted using Trivy disclosed 11 distinct vulnerabilities in the scrutinized Docker container image. The vulnerabilities were categorized based on their severity: 7 of medium concern, 3 high, and 1 deemed critical. The most serious issue discovered was linked to libc6 (CVE-2019-1010022), which involves a circumvention of stack guard protection.

The accompanying graphical dashboard, generated from Trivy's JSON results using a custom Python script, provides an intuitive and immediate visual interpretation of the vulnerabilities, emphasizing their severity and enabling an efficient prioritization for remediation efforts.

These vulnerabilities, spread throughout critical system libraries, highlight the necessity for immediate attention to patching or updating to secure versions. The dashboard offers a user-friendly interface to visualize the identified vulnerabilities by severity, assisting in the stratification of response measures. Vulnerabilities classified as Critical and High were earmarked for swift remediation, with recommendations for remediation provided by Trivy for the affected packages.

These vulnerabilities are spread throughout critical system libraries and signal the necessity for urgent patching or updates to secure iterations. The dashboard rendered through our bespoke Python script offers a user-friendly interface that maps out the identified vulnerabilities according to their severity, assisting in the stratification of response measures. Vulnerabilities classified as Critical and High were marked for swift resolution, with Trivy providing remediation recommendations for the compromised packages.

This analysis is crucial in reinforcing the defense mechanisms of Docker containers prior to their deployment in production settings, ensuring that we proactively address the most significant threats as detected by Trivy's extensive scanning functionality.

3) Dynamic Analysis with Falco: Falco is a state-of-the-art security solution designed for cloud environments. It excels in detecting and alerting users about abnormal activities that may signal security threats in real-time. Falco's Dynamic Analysis enhances our security protocols through continuous real-time surveillance of Docker containers. The primary strength of the system is its advanced behavioral analysis, which surpasses static analysis by monitoring and interpreting real-time container activity to detect threats through abnormal behavior patterns. Falco's rules can be

 doi:10.20944/preprints202406.1507.v1

tailored extensively to precisely define abnormal behavior within a particular operational setting. Its smooth interface with orchestration systems such as Kubernetes is essential for ensuring security in intricate DevOps workflows. Falco offers many alerting options such as emails and connection with third-party communication systems like Slack to immediately notify teams of potential security risks. Falco's skills are essential for comprehending and reducing the ever-changing security threats encountered by Docker containers, guaranteeing ongoing operational reliability [8].

a) Results and Metrics for Falco: This implementation strategy serves as the framework for our security assessment, enabling us to comprehensively evaluate the security of Docker containers in a cloud-based environment. By combining static and dynamic analysis tools, we aim to enhance the security measures for Docker containers, addressing both known vulnerabilities and real-time threats effectively.

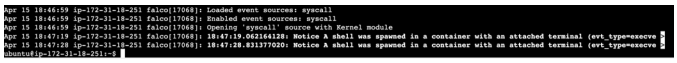


Figure 3. Terminal output showcasing the anomalies detected real time by Falco.

Falco is used in our security architecture because of its skill at monitoring in real-time and its ability to quickly identify abnormalities like illegal shell calls inside Docker containers. We may view a real-time recording of security issues, such as unusual file system modifications, network intrusions, and suspicious process activities, by using "journalctl -fu falco" command within the terminal of an Ubuntu server.

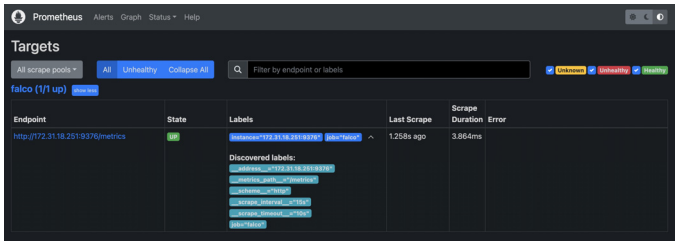


Figure 4. Metrics loaded by Falco to Prometheus.

We use PromQL to analyze this data by integrating Falco with Prometheus, which allows us to accurately identify and measure unusual activities. Security Administrator's can quickly respond to and look into any risks by utilizing the Prometheus's alerting features, which we can set up to create rules that send out notifications for particular Falco-detected patterns.

Through this integration, a dynamic and responsive security apparatus is built, which is essential for preserving the integrity of containerized systems and quickly resolving a variety of security risks [27].

3.4. Future Directions

In the future, container security will require continuous adaptation and innovation. Docker's official documentation and the wider security community highlight many crucial areas that are expected to be essential for future study.

- Improved network security for containers by investigating advanced network segmentation and encryption methods to enhance isolation and safeguard container traffic.
- Utilizing AI and ML algorithms to enhance the identification of complex risks and anomalies in container behavior.
- Incorporating security measures into the CI/CD pipeline helps automate security checks and maintain security as a constant priority during the application lifecycle.
- Exploring methods for implementing immutable containers to mitigate runtime vulnerabilities by replacing them instead of modifying them [1].

The instructions demonstrate the changing nature of container security and the ongoing requirement for creative methods to combat emerging threats. Future research can enhance the

 doi:10.20944/preprints202406.1507.v1

development of stronger and more durable security measures for Docker containers and the wider range of containerized applications by concentrating on these specific areas.

4. Services Used for Implementation - IOT

For the implementation part, AWS Iot Core (secure communication), Lambda (code execution), S3 (Stores data), Cloudwatch(Monitors metrics) and GuardDuty(Monitors threats) AWS services are used.

4.1. AWS IoT Core

AWS IoT Core, offered by Amazon Web Services (AWS), is a managed cloud platform designed to facilitate secure and reliable communication between connected devices and cloud applications. It serves as a central hub for IoT solutions, providing features like diverse communication protocol support, robust security measures for authentication and data encryption, and device management functionalities including on boarding and registration at scale. With its Rules Engine, AWS IoT Core enables real-time data processing and routing based on predefined conditions, seamlessly integrating with other AWS services like storage, databases, and analytics. It ensures scalability, high availability, and reliability, making it easier to develop, deploy, and manage IoT applications while handling large fleets of devices efficiently [11,13].

4.2. AWS Lambda

A computing service called Lambda enables run code without a server. Background processing resources are automatically managed by Lambda. Only the computing time used is charged to the customer;not the running code. An Amazon Web Services lambda function is a stateless piece of code [12].

4.3. AWS GuardDuty

GuardDuty is a security service in AWS, helps to protect the environment by continuously monitoring for malicious activity and unauthorized behavior. It analyses logs and network traffic to detect threats such as compromised accounts, malware and unauthorized access attempts [26].

4.4. Amazon S3

Productivity, security, scalability, and data availability are among the attributes of Amazon Simple Storage Service (Amazon S3), an object storage solution. A vast array of use cases, including data lakes, websites, mobile applications, backup and restore, archiving, business applications, IoT devices, and big data analytics, can be served by using Amazon S3 to store and safeguard any volume of data for clients of all sizes and sectors [14].

4.5. AWS CloudWatch

Real-time monitoring of Amazon Web Services (AWS) resources and applications is provided via Amazon CloudWatch. CloudWatch is a tool for gathering and monitoring metrics, or variables, related to the applications and resources. When a threshold is crossed, we can set up alarms to automatically adjust the resources we are monitoring or observe metrics and send messages. CloudWatch can be used for Security analytics in the following way.

Metric Monitoring - Setup a custom CloudWatch metrics to monitor key IoT metrics such as incoming message rate, connection status and data transfer volume. These metrics can help to detect abnormal behaviour due to a security threat.

Log Monitoring - IoT Core logs monitors for authentication failures, unauthorized access attempts and other security related events. CloudWatch Alarms can be setup to trigger notifications for specific log events [10].

 doi:10.20944/preprints202406.1507.v1

5. Simulation - Brute Force Attack on IOT Smart Door

The goal of simulating a brute force attack on an IoT door lock system is to try every possible combination of users and passwords in an attempt to gain unauthorized access until the right credentials are found. The first step in the procedure is data on legitimate usernames and possible passwords is gathered. The assault is then automated by using either custombuilt or pre-existing tools and scripts, which are used to configure parameters such the target system, lists of usernames and passwords, and any applicable constraints. The simulation runs, methodically testing every combination until either an attack is stopped by security measures or successful access is obtained. Analysis of the attack's outcomes, including any successful breaches or vulnerabilities found, is made possible by continuing to watch its development. After the simulation, vulnerabilities in the security of the IoT door lock system that were found can be fixed by strengthening the system as a whole against possible threats and introducing improved authentication procedures.

5.1. Implementation

To demonstrate a security compromise in an Internet of Things house door lock system, the first step is to configure virtual IoT devices to mimic the features of the door lock system, including locking and unlocking. Then, these virtual devices are managed by AWS IoT Core, which makes sure that encrypted channels are established using MQTT. Thereafter, strong authorization and authentication protocols are set up in AWS IoT Core to carefully control who has access to the door lock system by utilizing cutting-edge techniques like device certificates and IAM roles. Next, scripts are created to mimic different security breach scenarios, such as tampering or unauthorized access attempts, and to efficiently communicate with the virtual devices over AWS IoT Core for administrators.After that, in order to successfully interact with the virtual devices through AWS IoT Core, scripts or programs are written to simulate different security breach scenarios, such as unauthorized access attempts or tampering. In order to strengthen the security architecture, elements like logging and monitoring are turned on to carefully record device interactions and security occurrences. Services like AWS CloudWatch are integrated for thorough analysis. Additionally, reaction protocols are carefully put into place. These include using Lambda functions to lessen security breaches, which may involve locking down the door lock system or sending out instant alerts to administrators.

5.2. Results and Metrics

The brute force attack simulation was successfully executed with the use of various AWS services. The Lambda function successfully simulated the attack by attempting different combination of credentials. Figure 5 shows the successful execution of brute force simulation using Lambda.

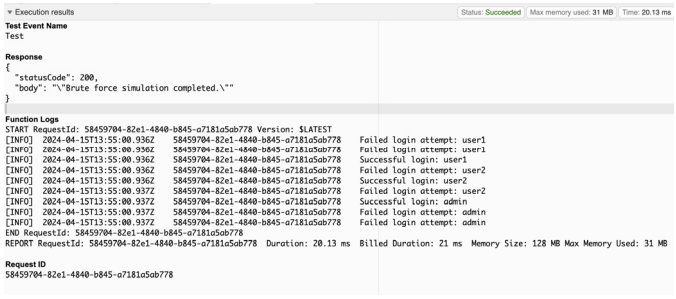


Figure 5. Execution Result.

Metrics like error count and success rate were collected (Figure 6) and stored to access the simulation performance. In order to ensure a quick response to possible security threats, CloudWatch alarms were set up to monitor unsuccessful login attempts, and an email notification alert was enabled when a particular thresholds was exceeded. Figures 7 and 8 shows the alarm generated by CloudWatch indicating the failed login attempt and the corresponding email notification that was send and decision trees, can significantly improve the accuracy.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 24 June 2024 doi:10.20944/preprints202406.1507.v1

Figure 6. Error Count and success rate with respect to the Login Attempts triggered by the Brute Force Attack.

Figure 7. Alarm generated indicating the Failed Login Attempt.

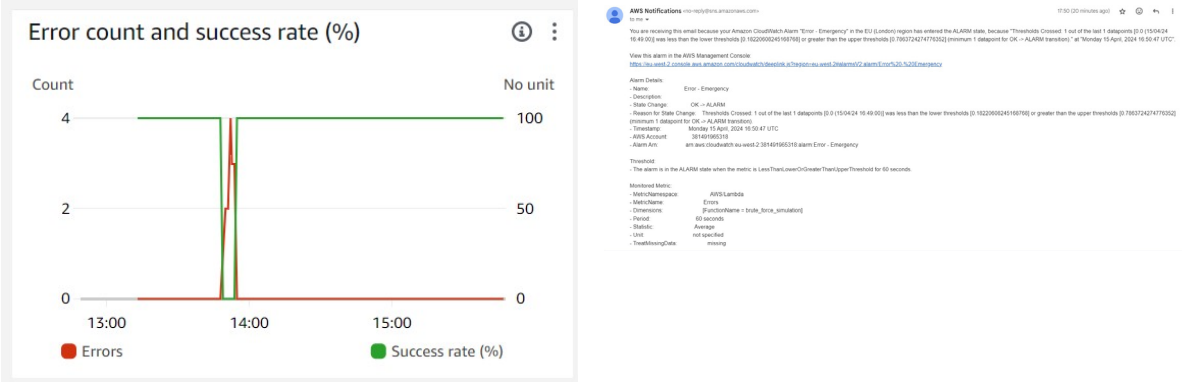


Figure 8. Email Notification.

AWS IoT core was utilized to register device securely, attach policies and certificates. Amazon GuardDuty was activated to detect suspicious activities related to brute force attack. Emphasizing the security of this environment in protecting the IoT Infrastructure.

6. Future Prospects

The future prospects for analyzing Docker vulnerabilities through static and dynamic methods and enhancing IoT security with AWS IoT Core, CloudWatch, and GuardDuty are rather promising, with the recent research which highlights several potential directions for development. One such promising future prospect is likely to see increase in the use of Artificial Intelligence (AI) and Machine Learning (ML) methodology to enhance IoT security. Recent research suggests that using AI and ML, such as neural networks of detecting IoT security threats and reduce the false alarm rate (Li et al., 2022). Likewise, AI and ML can help to automate security processes, enabling faster detection and response times. Another future prospect is the development of new tools and techniques for analyzing Docker vulnerabilities. Whereas recent study has proposed the use of deep learning algorithms to detect and classify vulnerabilities in Docker images, improving the accuracy and efficiency of vulnerability detection (Kim et al., 2024). Additionally, researchers have interpreted and proposed the use of dynamic analysis techniques, such as fuzz testing and symbolic execution, to detect vulnerabilities in containerized applications at runtime (Li et al., 2023). In the same way, the integration of IoT security with edge computing is another future prospect which needs deeper study beyond the study of this research. What is interesting that Edge computing enables data processing and easy analysis of information to be carried out closer to the IoT devices, this benefits from the reduced latency and improving response times. What stands out in recent research has proposed the

 doi:10.20944/preprints202406.1507.v1

use of edge computing to perform security monitoring and analysis, enabling faster threat detection and response (Li et al., 2023).

7. Conclusion

This paper explores the intricate security aspects of Docker systems, emphasizing the importance of tools such as Trivy and Falco for undertaking comprehensive static and dynamic assessments to detect and address security risks. This proactive strategy for securing Docker establishes a strong platform for implementing wider cybersecurity improvements throughout the digital ecosystem. Our research highlights the necessity of creating sophisticated intrusion detection systems and enforcing strong security protocols to protect against complex cyber threats. These initiatives aim to improve the security of Docker and IoT platforms by promoting a cohesive strategy to tackle the changing problems of cyber attacks, ensuring a safer digital world.

While it significant that IoT devices will continue to be the part of everybodys life, this research and the information derived from the work proves that more research key studies need to be pursued. It is imperative that certain steps must be taken to enhance the safety of the devices. Our results show while tools are available online, we have used just one of the tools to simulate an attack and build a platform to trap the attack before it is able to disable to take full control. Developing Advanced Intrusion Detection Systems stands as a

pivotal focus area in the realm of IoT security. This approach aims to bolster the detection capabilities against sophisticated cyber threats prevalent in IoT networks, thereby fortifying the existing security posture.

These initiatives, alongside efforts in IoT Security by design, standardisation of security protocols, government and regulatory interventions, enhancing privacy guidelines, raising awareness, educating users and through continuous research and development collectively contribute towards bolstering IoT security. Through the concerted adoption of these measures and a continued commitment to research and innovation, the potential risks and challenges posed by IoT devices can be significantly mitigated, thereby fostering a safer and more resilient IoT landscape.

References

1. Karie, N.M. et al. (2021) 'A review of security standards and frameworks for IOT-based Smart Environments', IEEE Access, 9, pp. 121975–121995. doi:10.1109/access.2021.3109886.
2. Wu, Y. (2021) Basic intrusion technology of industrial internet of ... iopscience. Available at: https://iopscience.iop.org/article/10.1088/17426596/1738/1/012094 (Accessed: 28 January 2024).
3. Azrour, M. et al. (2021) Internet of things security: Challenges and key issues, Security and Communication Networks. Available at:
4. https://doi.org/10.1155/2021/5533843 (Accessed: 28 January 2024).
5. J. Singh, T. Pasquier, J. Bacon, H. Ko and D. Eyers, "Twenty Security Considerations for Cloud-Supported Internet of Things," in IEEE Internet of Things Journal, vol. 3, no. 3, pp. 269-284, June 2016, doi: 10.1109/JIOT.2015.2460333.
6. Jukka Veijanen,Implementation of security best practices on AWS Cloud. Case: Vulnerability scanning of EC2 instances and networks
7. Ferrara, P. et al. (2020) Static analysis for discovering IOT vulnerabilities - International Journal on Software Tools for Technology Transfer, SpringerLink. Available at:
8. https://link.springer.com/article/10.1007/s10009-020-00592-x (Accessed: 30 January 2024).
9. Malhotra P, Singh Y, Anand P, Bangotra DK, Singh PK, Hong WC. Internet of Things: Evolution, Concerns and Security Challenges. Sensors. 2021; 21(5):1809. https://doi.org/10.3390/s21051809
10. Karale, A. et al. (2021) The challenges of IOT addressing security, ethics, privacy, and laws, Internet of Things.
11. I. Idrissi, M. Azizi and O. Moussaoui, "IoT security with Deep Learning-based Intrusion Detection Systems: A systematic literature review," 2020 Fourth International Conference On Intelligent Computing in Data Sciences (ICDS), Fez, Morocco, 2020, pp. 1-10, doi: 10.1109/ICDS50568.2020.9268713.
12. "Using Amazon CloudWatch alarms - Amazon CloudWatch," docs.aws.amazon.com. Available:
13. https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/ AlarmThatSendsEmail.html. [Accessed: Feb. 13, 2024].
14. AWS IoT," docs.aws.amazon.com. Available: https://docs.aws.amazon.com/iot/latest/developerguide/mqtt.html.

15. [Accessed: Feb. 13, 2024]
16. "Building Lambda functions with Python AWS Lambda," docs.aws.amazon.com. Available: https://docs.aws.amazon.com/lambda/latest/dg/lambda-python.html.
17. [Accessed: Feb. 13, 2024]
18. "What Is AWS IoT? - AWS IoT," docs.aws.amazon.com. Available: https://docs.aws.amazon.com/iot/latest/developerguide/what-is-awsiot.html. [Accessed: Feb. 12, 2024]
19. Amazon Web Services, "What is Amazon S3? - Amazon Simple Storage Service," docs.aws.amazon.com, 2023. Available: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html. [Accessed: Feb. 13, 2024]
20. Docker, "Docker security," Docker Documentation, Apr. 23, 2021. https://docs.docker.com/engine/security/ (accessed Jan. 20, 2024).
21. R. Yasrab, "Mitigating Docker Security Issues," arXiv (Cornell University), Apr. 2018, doi: https://doi.org/10.48550/arxiv.1804.05039.
22. A. Randal, "The Ideal Versus the Real," ACM Computing Surveys, vol. 53, no. 1, pp. 1–31, May 2020, doi: https://doi.org/10.1145/3365199.
23. N. Zhao et al., "Large-Scale Analysis of Docker Images and Performance Implications for Container Storage Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 4, pp. 918-930, 1 April 2021, doi: 10.1109/TPDS.2020.3034517.
24. A. Y. Wong, E. G. Chekole, M. Ochoa, and J. Zhou, "On the Security of Containers: Threat Modeling, Attack Analysis, and Mitigation Strategies," Computers and Security, vol. 128, p. 103140, May 2023, doi: https://doi.org/10.1016/j.cose.2023.103140.
25. T. Shaffer, N. Hazekamp, J. Blomer and D. Thain, "Solving the Container Explosion Problem for Distributed High Throughput
26. Computing," 2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS), New Orleans, LA, USA, 2020, pp. 388-398, doi: 10.1109/IPDPS47924.2020.00048. keywords: Containers;Software;Explosions;Large Hadron Collider;Throughput;Computational modeling;High energy physics
27. "Overview - Trivy," aquasecurity.github.io. https://aquasecurity.github.io/trivy/v0.49/docs/ (accessed Jan. 20, 2024).
28. "The Falco Project," Falco. https://falco.org/docs/ (accessed Jan. 21, 2024).
29. Li, S., Yang, X., Zhang, Y. (2022). An Improved IoT Security Detection Method Based on Machine Learning. Journal of Physics: Conference Series, 2022(Winter), 012002.
30. Kim, S., Kim, S., Lee, S., Lee, J. (2024). DeepLearning-Based Vulnerability Detection for Docker Images. IEEE Transactions on Computers, 73(3), 605-616.
31. Li, S., Zhang, Y., Yang, X. (2023). A Dynamic Analysis Method for Containerized Applications Using Symbolic Execution. IEEE Access, 11, 75019-75030.
32. "What is Amazon GuardDuty? - Amazon GuardDuty," docs.aws.amazon.com.
33. https://docs.aws.amazon.com/guardduty/latest/ug/what-isguardduty.html (accessed Feb. 13, 2024).
34. Kevin Trikusuma Dewo, V. Yasin, T. Budiman, Anton Zulkarnain Sianipar, and Akmal Budi Yulianto, "IT Infrastructure Dashboard Monitoring Application Development Using Grafana And Promotheus, a Case Study at Astra Polytechnic School," Nov. 2023, doi: https://doi.org/10.1109/icosnikom60230.2023.10364485.