Monitoring the Security Vulnerabilities in CI/CD Pipeline Using DevSecOps Security Testing Tools

MSc Research Project
Cloud Computing

Aniket Ashok Shetty
Student ID: x23217529

School of Computing
National College of Ireland

Supervisor: Prof. Vikas Sahni

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Aniket Ashok Shetty
Student ID:	x23217529
Programme:	Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Prof. Vikas Sahni
Submission Due Date:	12/12/2024
Project Title:	Monitoring the Security Vulnerabilities in CI/CD Pipeline Using DevSecOps Security Testing Tools
Word Count:	6363
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Aniket Ashok Shetty
Date:	10th December 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Monitoring the Security Vulnerabilities in CI/CD Pipeline Using DevSecOps Security Testing Tools

Aniket Ashok Shetty
x23217529

Abstract

With the increasing use of Development & Operations (DevOps) in the Software Development Life Cycle, the security is an important concern which needs to be taken into consideration. So, “Static Application Security Testing” (SAST) and “Dynamic Application Security Testing” (DAST) have been integrated in CI/CD Pipelines to check the vulnerabilities of a software at the build and deployment stage. However, the current tools were not efficient enough to detect the vulnerabilities in real-time, so in this research the main motive is to overcome those gaps by advancing the security tool to check vulnerabilities and attacks in a software. By integrating new methods like “Interactive Application Security Testing” (IAST) it provides a real-time vulnerability detection by monitoring the internal behaviour of applications. In this research, the demonstration of improved results of security testing tools are showcased. Where “Snyk” (SAST) tool identified 82 vulnerabilities in more effective and efficient way than the previous work, on the other hand “Stackhawk” (DAST) tool delivered decent results, lastly the integration of “Datadog” (IAST) was successfully done, as it does the work of both SAST & DAST in one tool.

Key Words: Software Development Life Cycle (SDLC), DevOps, DevSecOps, Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), Interactive Application Security Testing (IAST).

1 Introduction

The advancement of software development practices nowadays has increased the standard of traditional method like DevOps and DevSecOps, which helps in highlighting the performance of the software with automation, speed and security during the software delivery process. DevOps is basically a combination of both Development and Operation in a Software Development Life Cycle, which enables the integration and deployment process that streamlines together. In DevOps, Continuous Integration and Continuous Deployment (CI/CD Pipeline) life cycle play’s a vital role, which allows integration of source code and its continuous changes that are made and deployed automatically within testing and production environment. However, during this entire process, security needs to be considered and taken into consideration, particularly with the data which needs to be secured without any vulnerabilities being detected in that environment.

DevSecOps, which is built upon DevOps by enhancing the security parameters in CI/CD pipeline, which ensures security to be embedded at every stage of the pipeline in the software development life cycle. DevSecOps which basically means adding an extra layer in the Development and Operation process which detects and reduces those vulnerabilities in the software. So, the extra layer contains security methods like Static Application Security Testing & Dynamic Application Security Testing which are been integrated in the pipeline Diaz et al. (2019). SAST which mainly focuses on analyzing the source code of the software to detect vulnerabilities within the build stage, Masood and Java (2015)Gauthier et al. (2018), and on the other hand DAST detects the vulnerabilities of the running application in real time within the deployed environment Putra and Kabetta (2022). Even, after following both this approach in the software development life cycle it lacks effectively in detecting vulnerabilities in real-time which impacts the software.

The increasing difficulty level of the software system and growing culture of the cyber-attacks requires more robust security tools which can detect and mitigate risk. From the research it has shown that the traditional security tools like SAST and DAST are effective but lacks in certain stages in real-time which increases the threat levels. "Implementing and Automating Security Scanning to a DevSecOps CI/CD Pipeline" Marandi et al. (2023) demonstrated the application after integrating the security tools SAST & DAST, and found the vulnerability gaps during the real-time. Correspondingly, for the other research, "Continuous Security Testing: A Case Study on Integrating Dynamic Security Testing Tools in CI/CD Pipelines"Rangnau et al. (2020) proposed DAST integration but noticed problems in detecting vulnerabilities in the CI/CD pipeline dynamically.

The above study problem motivates to come up with the following Research Question:

"How security can be enhanced in CI/CD pipeline using SAST, DAST & IAST Security testing tools, in terms of efficiency, performance, and effectiveness?"

This research paper mainly focuses on answering the above research questions by focusing on implementing and developing a standardized method for ideally integrating Interactive Application Security Testing (IAST) in the CI/CD Pipeline. IAST basically does the security testing in real-time during the built and deployment stage in the CI/CD Pipeline which fills up the gaps that were not fulfilled by SAST & DAST. It does the work of both SAST and DAST at the same time in the CI/CD pipeline which makes it more efficient and effective.

Integration of SAST (Synk) and DAST (StakHawk) security tools on Damn Vulnerable Web Application has been done and which was later deployed on cloud to check the vulnerabilities. So, now the objective of this research was to propose the next step of implementation by integrating IAST security tool to that application and get more efficient and effective results, which might help in enhancing the security method of CI/CD pipelines. By following this process and addressing the gaps which are related to SAST & DAST, this research will evaluate the performance of IAST in real time, specifically focusing on identifying the vulnerabilities in the CI/CD pipeline and analyzing its efficiency and ensuring more accurate vulnerability detection count.

Going forward the research paper is in the following format: Section 2 helps us to understand the Related Work, that is been done till now and with achieved results and detected gaps. Section 3 explains the detailed Methodology, that will be implemented for this research. Section 4 showcases the Design Architecture & the Flow diagram which makes the better understanding of the research. Additionally, Section 5 tells the Implementation part, where it highlights the integration of tools and the entire process. Later, the Evaluation has been discussed in the Section 6, by providing the comparison and test cases that were performed during this research. Lastly, Section 7 tells the Conclusion and Results that are achieved during this research by answering the research question, and ending up with Future work by suggesting something that can be done with more time period.

2 Related Work

DevSecOps is a process that integrates security between the Development and Operation framework in the CI/CD pipeline to ensure security embedded into the Software Developemt Life Cycle (SDLC). As many organizations are moving towards the rapid and continuous software development, the need to address security concerns at every stage has become crucial. This literature review mainly focuses on the various aspects of DevSecOps implementation, tools, challenges, and benefits, based on the papers that are taken into consideration which mainly focuses on the adoption of DevSecOps in various field like finance, micro-services, continuous security testing, and CI/CD pipelines.

2.1 Integrating Security into DevOps

The main motive of integrating the security in the SDLC is to reduce the vulnerabilities in the CI/CD pipeline and to mitigate risks.

Grigorieva et al. (2024) helped us to understand the use of the Continuous Integration and Continuous Deployment in a pipeline, by securing the code and the automating the process in SDLC, whereas on the other hand Rajapakse et al. (2021) evaluated the standard security techniques like Static & Dyanamic Application Security Testing. The author also advised some new security tools such as Interactive Application Security Testing & and Runtime Application Security Protection, to better match with DevOps process. By examining the two research it was summarized that the integration of tools and cultural methods were challenging.

So, Ahmed et al. (2019) showcased a real-time case study of integrating Synk, Github & NPM in one of his research paper, were the solutions showcased massive drop in vulnerabilities like "Denial of Service (DOS)" & "Insecure Encryption". Apart from this, Aljohani and Alqahtani (2023) presented a unique framework using micro-services methods and tools like Shuffle and Docker for automating the security checks. By doing this they were able to analyze efficient vulnerabilities, but were still facing issues like false positive.

From analyzing all the above papers it was summarized that the common challenges were related to security and development speed.

Feio et al. (2024), helped in focusing the integration of the security tools and process in the Software Development Life Cycle, by showcasing the practical DevSecOps architecture with an effective CI/CD pipeline, also by using the security tools like OWASP ZAP & SonarQube. This architecture was really helpful as it helped in identifying the relevant vulnerabilities, which included 1795 security issues which were been detected with the help of the dependency check.

Additionally, by looking into this research there was another research where DevSecOps was integrated into the Financial Web Application using Microsoft Azure platform which identified 25 possible risks, and all this risks were related to the Data Breach & Injection attack David et al. (2024). They also offered a mitigating methods which included encryption and RBAC.

Throner et al. (2021)used a template based DevOps CI/CD pipeline using several tools like Kubernetes, Docker & GitLab which mainly focuses on the operating efficiency and security in micro-services deployment. By doing this it helped in reducing the maintenance efforts and vector attacks.

However, in contrast to this research Seth et al. (2023) focused on evaluating the effectiveness of Interactive Application Security Testing & and Runtime Application Security Protection security tools by identifying 91 issues within 2.14 issues per-hour. This outcomes were better than DAST but were not that effective than SAST and EMPT.

So, by looking to all the research it was analyzed that the automation and security has the main focus, but they also mainly focused on the importance of development and testing in the real world.

2.2 Improving Security in DevOps and CI/CD Pipelines

After completing the research on integrating security in DevOps, this section discusses how the security can be enhanced in the Software Development Life Cycle by addressing the critical constraints, efficiency and automation. Yadati (n.d.), highlighted the use of integrating dynamic testing tools like JMeter, Selenium Base and OWASP ZAP to identify the vulnerabilities during the execution process in the CI/CD pipeline, by enhancing the efficiency and getting a runtime overhead of 15% which is less than the actual time. However, there were certain limitations in the dynamic testing, so Masood and Java (2015), addressed those issues in his research by integrating HP Fortify and IBM AppScan as a static testing tool to mitigate SOAP and RESTful API based vulnerabilities. During this period the downtime of integrating the security has been hard, Kushwaha et al. (2024) stepped in by featuring his research where the automatic security work flowed in the financial application by using several tools like OWASP ZAP, GitHub Advanced Security Tool and Codacy which helped in achieving some good results and uncovering 26 vulnerabilities during threat modelling within 71 seconds. By following this process, it connected the gaps in terms of speed and consistency but failed in the architectural creation. So, Chen and Suo (2022), proposed the DevSecOps security architecture in his research, which not only reduced the release life cycle by 14 to 12 days but also examined the mitigation for those issues by increasing the security and risk and detecting the security vulnerabilities from 16 to 3. Later, Zunnurhain and Duclervil (2019) appreciated the work, which was done by using the DevSecOps framework in the project management, by integrating continuous security to the customer tools with the helps of the agile method, which helped in addressing the gaps in an organization by managing the development process along with security objectives.

In the DevOps CI/CD pipeline field, Singh and Singh (2016) acknowledged the traditional importance of containerizing an application with the help of Docker and Kubernetes tools, and by showcasing their capabilities to improve the efficiency of the resource and to increase the speed of deployments process. In addition to this Şengül et al. (2021), found some basic vulnerabilities in the containerized environment with the help of Anchor and Snort3 which are basically the Static & Dynamic security testing tool which helped in identifying 48 critical vulnerabilities within a single docker image which was been containerized. By doing this process it highlighted the gaps which were left by

Singh and Singh (2016), which mainly focused on the performance and speed of the deployment. Later, Sun et al. (2021) further, explored the security of the pipeline by automating it and integrating several tools like Fortify, Gitlab, Jenkins and SonarQube which allowed to manage the security and helped in reducing the cost and increased the efficiency of identifying vulnerabilities. Lastly, Ji et al. (2022) addressed the security during the runtime by integrating Runtime Application Self Protection with static taint analysis, which provided a traditional approach for classifying the threats and responses effectively by reducing the false positives rate and improving the quality of production.

2.3 Integration of DevSecOps Security Tools in CI/CD Pipeline

After analyzing the standard research for the integration of DevSecOps in SDLC, this section makes us understand the importance of integrating security in the CI/CD pipeline using DevSecOps security tools.

As per Marandi et al. (2023) research, they showcased how with the help of StackHawk & Snyk security tools they automated the image scanning security testing process which helped in enhancing the Static & Dynamic Security Testing for the applications those are containerized. Furthermore, the research by Putra and Kabetta (2022) extended the future of SDLC by creating a ultimate DevSecOps pattern which contains the process that starts from the development stage and ends at monitoring stage, also there are tools that are been used during this process which are Docker and Gitlab for generating the pipeline, OWASP ZAP tool for Dynamic Testing & NJSSCAN for Static tetsing.

Rangnau et al. (2020) mainly found the standard issues which were been faced while integrating Dynamic Application Security Testing in CI/CD pipelines, which helped in detecting and mitigating the configuration and aligning issues of the containers, also there were several tools like SeleniumBase, OWASP ZAP & JMeter which were been used during the process.

By looking at the above studies its mainly focuses on some key objective of enhancing the security and efficiency of the CI/CD pipeline in the SDLC. The researchers showcased how they were able to achieve the results by detecting the vulnerabilities in the software like SQL Injection, Cross-site scripting (XSS) and many more. But they also mentioned the gaps and limitations that they faced during the process, like real-time security testing and integrating some additional tools which can enhance the life cycle of the software. So, this study will try to solve those gaps by automating the process of detecting vulnerabilities in the DevSecOps CI/CD pipeline for a Software Development Life Cycle.

2.4 Research Niche

In Table 1 Previous Work Comparison and Limitations.

Table 1: Comparison of Work Process

Research Work	**Proposed Work**	**Constraints**
Grigorieva et al. (2024)	Introduced DevOps CI/CD Pipeline in SDLC.	Security tools were not integrated into the CI/CD Pipeline.

Table 2: Comparison of Work Process

Research Work	**Proposed Work**	**Constraints**
Singh and Singh (2016)	Introduced Containerized Application with Docker and Kubernetes.	Helped increase deployment efficiency but missed security concerns.
Rajapakse et al. (2021)	Focused on Standard Security tools like SAST & DAST.	Suggested integration of IAST & RASP tools.
Ahmed et al. (2019)	Real-time integration with Synk, GitHub & NPM.	Achieved vulnerability detection but lacked IAST.
Aljohani and Alqahtani (2023)	Unique Framework using Shuffle and Docker.	High false positive rates affecting software.
Feio et al. (2024)	Introduced DevSecOps Architecture with SAST & DAST.	Lacked IAST and RASP due to open-source limitations.
Putra and Kabetta (2022)	Enhanced DevSecOps pattern by integrating SAST & DAST from development to monitoring stage.	Real-time testing and simultaneous mitigation were lacking.
Marandi et al. (2023)	Scanned Docker images with SAST and DAST tools.	Lacked IAST tool integration and efficiency enhancements.

3 Methodology

An in-depth research methodology is built to achieve the required results for this project by following certain implementation steps. This research starts with an through investigation of how the security vulnerabilities can be monitored & reduced in the CI/CD pipeline with the help of DevSecOps security tools. Also, this methodology is motivated by Marandi et al. (2023) where the security tools like SAST and DAST were been integrated in the CI/CD pipeline to detect the vulnerabilities of Damn Vulnerable Web Application which is Dockerized. DVWA is a application which already contains vulnerabilities in it, also it is a open source application which can be studied and accessed directly from Github Repository.[1] However, this study helped in providing security to the dockerized application but lacked in efficiency and performance of detecting accurate vulnerabilities.

So, a new security method called Interactive Application Security Testing (IAST) was been integrated to make the process more efficient and to increase the performance of the application as it detects the vulnerabilities in real time and provide more accurate results than SAST & DAST. While following this process certain tools were taken into consideration like GitHub Private/ Public Repository, Google Cloud Platform, Docker, Visual Studio Code, Security testing tools like Synk, StackHwak & Datadog which were integrated with the help of several documentation, tools configuration manual, blogs and YouTube videos.

Additionally, during the implementation process, each steps are documented, which are mentioned in the configuration manual like the errors, results, version updates, API keys for the security tools and many more which might help others to recreate the same scenario at their local machine environment.

[1]https://github.com/opsxcq/docker-vulnerable-dvwa

3.1 Interactive Application Security Testing

IAST is a security tool which is a combination of both Static Application Security Testing and Dynamic Application Security Testing, which allows identification of real time vulnerabilities and monitors the source code while the application has been executed and deployed on production. Basically, SAST helps in detecting the vulnerabilities from the source code and on the other hand DAST, which detects vulnerabilities during the application run time. But both the process works independently which takes time and that makes the application more unsecure by time being. So, IAST helps in providing the accurate awareness of the vulnerabilities by monitoring it in real time once it is pushed on any environment. By following this process, it helps the developers to identify the data risks earlier in the CI/CD pipeline, so that they can take the appropriate action to mitigate those risks and get the application running without any harm in the software development life cycle.

In this project, the IAST security tools has been integrated into the CI/CD pipeline which will be showcased in the architecture diagram section.1 Once the application has been built and deployed on cloud using Google Cloud Platform, then IAST tools gets triggered at the same time and it tries to identify the vulnerabilities in the application and monitors it. The integration of SAST and DAST is also done at the same time so in the evaluation section the comparison can be showcases on the bases of efficiency and performance.

After performing this step this not only tries to answer the research question but also tries to fulfil the gaps that were left out from the literature review by other papers.

3.2 IAST Security Tools Selection

Selecting the appropriate tool for the IAST was the crucial job in this research. While studying the previous research there were several tools which came up like Contrast Security [2], HCL AppScan [3], OWASP ZAP Rangnau et al. (2020) and BurpSuit [4], that allows IAST service enabled in their tool. But those tools were not open source and easily accessible as they provide service to the organization or enterprise which was not relevant to this research.

So, after doing some more research for the tool selection, Datadog was the only tool which was open source and can be accessed and integrated in the test environment. [5] Datadog provides IAST service, and it offers a complete package in one tool also it prioritize the critical vulnerabilities in the application which needs to be taken care of by accelerating the mitigation process and managing the software development life cycle. However, from the previous research it shows that SAST and DAST tools have been integrated but IAST was lacking. The main objective for this research was to do that integration, later when the implementation part started it came to know that Datadog doesn't support PHP language and it cannot detect the vulnerabilities for that application which was built previously, as IAST code security support Java, Python and some other language.

To take the implementation part forward the new application was been chosen and integrated into CI/CD pipeline. This application was "Altro Mutual" which was very

[2]https://www.contrastsecurity.com/contrast-assess
[3]https://www.hcl-software.com/appscan
[4]https://portswigger.net/burp
[5]https://www.datadoghq.com/product/code-security/

much similar like DVWA, and it was a already vulnerable J2EE Banking Application where the testing of security can be performed as it contains pre-defined vulnerabilities in it. This application was in Java language, so Datadog can be integrated for this application and perform the IAST Code Security process.

Lastly, in the next section the detailed design will be mentioned which will clear the idea that how this process will take place and how the integration of all the tools are made for this research.

4 Design Specification

This section elaborates the architecture design for the proposed solution for this research:

4.1 Architectural Design for the DevSecOps CI/CD Pipeline

The architecture diagram explains the DevSecOps CI/CD pipeline as shown in the Figure 1 where it explains multiple steps, stages and security tools which are been integrated to establish a secure development and deployment of a application. This architecture is built upon a prior work by Marandi et al. (2023) where the integration of Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) security tools was been done for testing the efficiency of Damn Vulnerable Web Application and to detect vulnerabilities. So, in this architecture diagram the new advancement lies by adding a Interactive Application Security Testing (IAST) security tool into the CI/CD pipeline, where it operates in real time to identify vulnerabilities.

The figure showcases that this architecture is a combination of two main features, one is Google Cloud Platform and the other one is the Security Testing Tools. Also, it starts from the source code which is developed on Visual Studio following several stages in the project which contains repository handling, building, deploying and at last security testing.

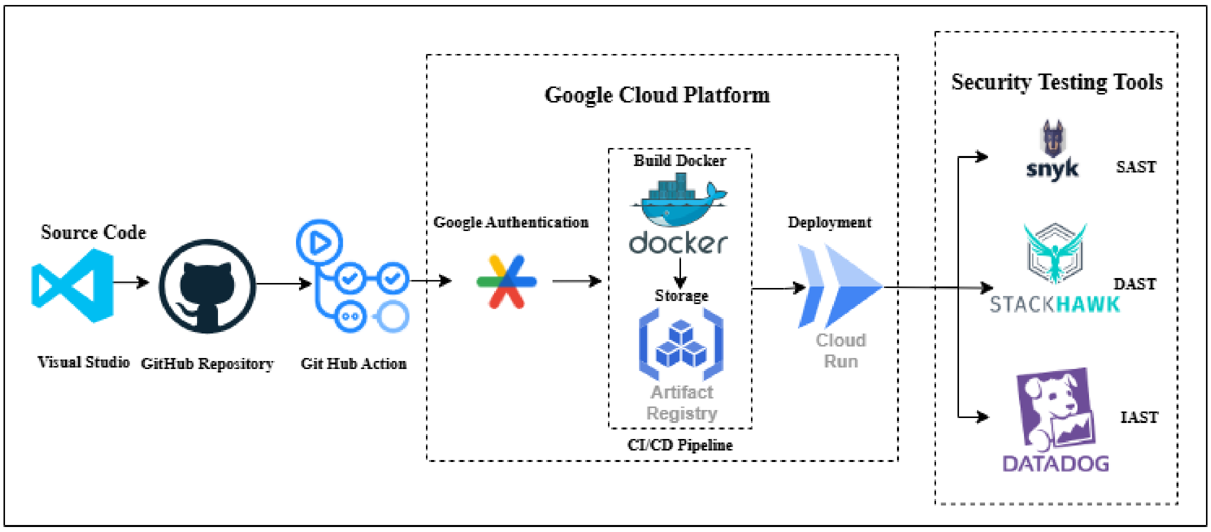


Figure 1: Architectural Design for the DevSecOps CI/CD Pipeline

The following are the steps, which are explained in more detail:

- **Source Code and Repository Management**:

  The application is built with the help of the source code using Visual Studio, as it's an Integrated Development Environment (IDE). So, in this case the application which was cloned was "Altro Mutual" banking J2EE web application as it already contains security vulnerabilities in it and once the source code has been developed then it is pushed to the GitHub Private Repository, which works as a version control and collaborator among the team.

- **Continuous Integration and Continuous Deployment (CI/CD) Pipeline**:

  To carry the process of CI/CD (Continuous Integration and Continuous Deployment) pipeline, GitHub Action has been used and integrated in this research as it works as an automation tool and orchestrates the pipeline. The pipeline starts when the source code is been pushed from the local machine to the private github repository which ensures the automated deployment of the application which takes place simultaneously.

- **Google Authentication and Build Process**:

  The Google Authentication has been integrated with CI/CD pipeline as it provides a secure access to the Google Cloud Platform where the build and development of the application takes place. In GCP the authentication takes place by validating the Google ID and generating a build ID. Once this process is completed then the Docker Images are built for an application. Docker helps in packaging the images of an application in a container which contains all its dependencies and requirements as per the application's need which makes it more consistent and efficient. Later, this built image has been stored in the Google Cloud Artifact Registry, as it is a more scalable and containerized storage for the artifacts.

- **Deploying the Docker Image**:

  This is the second stage in the CI/CD pipeline, where the deployment of the built Docker images takes place. The Docker images are deployed on Google Cloud Run, as it is a serverless platform which provider access for deploying the containerized application. By doing this process it makes sure that a secure and efficient deployment can be performed with no operational issues.

- **Integration of Security Testing Tools**:

  Once the application has been deployed on cloud then the security testing tools gets triggered where it checks all the security vulnerabilities in an application. So, the integration of three security tools has been done in the CI/CD pipeline which helps in enhancing the security of the application that is deployed.

  - **Static Application Security Testing- Snyk**:
    SAST is method which takes place at the build stage of the CI/CD pipeline to identify the vulnerability in an application. For this research, "Synk" security testing tool has been taken into consideration as it is an open-source tool, which works as a SAST in the CI/CD pipeline. SAST basically means to detect the vulnerabilities in the static code before the application is deployed on production like over here its cloud, and if there are any critical vulnerabilities

that are found during this process, then the CI/CD pipeline fails, and the source code won't be pushed to the next stage which is deployment.

– **Dynamic Application Security Testing- StackHawk**:
DAST is the method that takes place after the deployment stage where it detects the vulnerabilities of the application in real time. For this research DAST is executed with the help of "StackHawk" Security testing tool as it an open-source tool. DAST is a dynamic security testing tool so once the application has been deployed after the static security testing the DAST tools gets triggered, and it check vulnerabilities dynamically which makes the testing process faster and more efficient.

– **Interactive Application Security Testing- Datadog**:
IAST is a security tool which is been developed in such a way that it does the work of both SAST and DAST at the same time in the CI/CD pipeline. This addition of tools in this research is the novel contribution. "Datadog" works as a IAST security testing tool which is an open source. So, after integrating this tool the application gets more secure in real time, and the performance also get increased with more efficiency. Additionally, these tools detect the vulnerability at the build and deployment stages with less false positive and negatives.

5 Implementation

This section elaborates how the integration of SAST, DAST and IAST tools are done in the CI/CD pipeline, for the "Altro Mutual" banking application. Additionally, in this section a detailed steps are elaborated where the configuration of security tools with the CI/CD pipeline will be showcased and the deployment of the application on cloud.

5.1 Application Setup

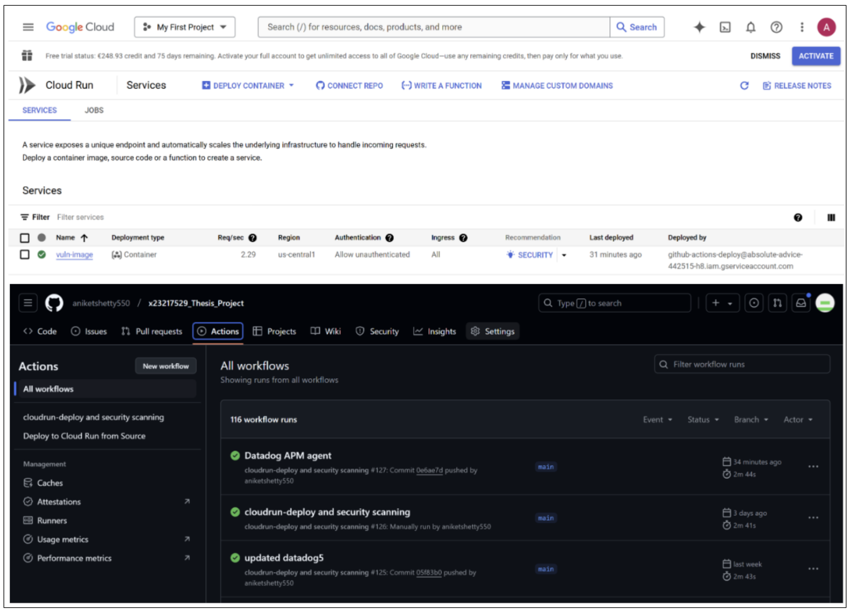
The "Altro Mutual" application has been accessed from the official GitHub Repository, [6] as it's an open-source resource which contains predefined vulnerabilities in it and was configured on Visual Studio (VS code). This application contains vulnerabilities like SQL injection, Penetrating Testing, Authentication Flaws, Server-Side vulnerabilities, IDOR and Cross-site scripting (XSS) which were used for testing the security tools like SAST, DAST, and IAST to check the effectiveness, performance and efficiency. Inside VS Code, a new directory was created named as ". github/workflow" to describe the CI/CD pipeline workflow in the YAML format, which describes the project configuration. Furthermore, one more directory was generated named "Dockerfile" to containerize the application, and allowing the application to deploy and integrate in the testing and production environment.

5.2 CI/CD Pipeline Creation & GCP Integration

The CI/CD pipeline basically automates the process by running the application smoothly in testing and production environment. So, in this case once the application was completely setup then it was deployed on Google Cloud Run Platform by using GitHub Action

[6]https://github.com/HCL-TECH-SOFTWARE/AltoroJ

CI/CD pipeline. The workflow of the GitHub action- CI/CD pipeline is showcased in the figure 2, where it shows the performed testing and the failed/passed pipelines, as per the changes that were made in the source code.



Free trial status: €248.93 credit and 75 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

A service exposes a unique endpoint and automatically scales the underlying infrastructure to handle incoming requests.
Deploy a container image, source code or a function to create a service.

Services

Name	Deployment type	Req/sec	Region	Authentication	Ingress	Recommendation	Last deployed	Deployed by
vuln-image	Container	2.29	us-central1	Allow unauthenticated	All	SECURITY	31 minutes ago	github-actions-deploy@absolute-advice-442515-h8.iam.gserviceaccount.com

aniketshetty550 / x23217529_Thesis_Project

Actions

All workflows

cloudrun-deploy and security scanning

Deploy to Cloud Run from Source

All workflows
Showing runs from all workflows

116 workflow runs

Datadog APM agent
cloudrun-deploy and security scanning #127: Commit 0e6ae7d pushed by aniketshetty550
main
34 minutes ago
2m 44s

cloudrun-deploy and security scanning
cloudrun-deploy and security scanning #126: Manually run by aniketshetty550
main
3 days ago
2m 41s

updated datadog5
cloudrun-deploy and security scanning #125: Commit 05f83b0 pushed by aniketshetty550
main
last week
2m 43s

Figure 2: Github Action Integration with CI/CD Pipeline & Google Cloud Platform

For automating the process in the CI/CD pipeline, it used cicd.yml file, which was created within Github action, which organized several stages like setting up the environment, defining jobs, building Docker images and deploying it to Google Cloud Run Platform. Additionally, security tools like Synk, StackHawk and Datadog was also integrated in this workflow to make sure the application is secure and free from vulnerabilities and is monitored correctly.

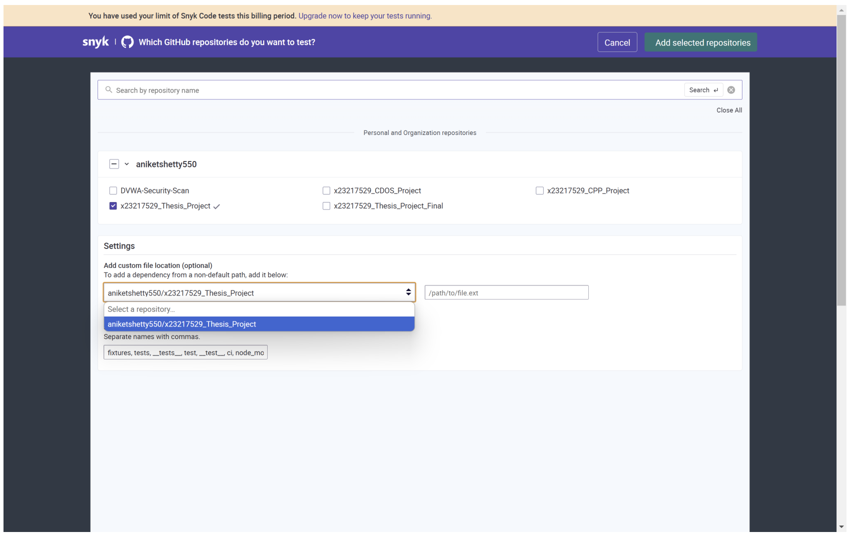
Also, after containerizing the application using Dockerfile it was deployed on cloud without any disturbance and Google Cloud Platform services were accessed easily like: "Cloud Run" and "Artifact Registry". The deployment of the dockeralized application was done by cloud run and once it was deployed the docker images were stored securely in the artifact registry. By following all this steps, it assured that CI/CD process worked smoothly and which increased the performance and efficiency of the project.

5.3 Security Testing Tools

Security testing tools have been integrated after the deployment process on GCP Cloud Run to detect the vulnerabilities of the application in real time during the testing and production environment:

1: **Snyk**: It is an open-source security testing tool which enables detection of vulnerabilities within the application with the help of Static Code Analysis. Integration was done by cloning the Synk tool with the Github Repository as shown in the figure 3, and by using the official YAML configuration file of Synk which is available [7] ,and adding it to the actual cicd.yml file of this project. Also, the API Key of the Synk tool was mentioned in the cicd.yml file so it get's integrated correctly and once the CI/CD pipeline runs the tool get's triggered and the static code analysis of the application is done. On the Synk

[7]https://github.com/snyk-labs/snyk-cicd-integration-examples


You have used your limit of Snyk Code tests this billing period. Upgrade now to keep your tests running.

Which GitHub repositories do you want to test?

Personal and Organization repositories

aniketshetty550

- [ ] DVWA-Security-Scan
- [ ] x23217529_CDOS_Project
- [ ] x23217529_CPP_Project
- [x] x23217529_Thesis_Project ✓
- [ ] x23217529_Thesis_Project_Final

Settings

Add custom file location (optional)
To add a dependency from a non-default path, add it below:

aniketshetty550/x23217529_Thesis_Project

/path/to/file.ext

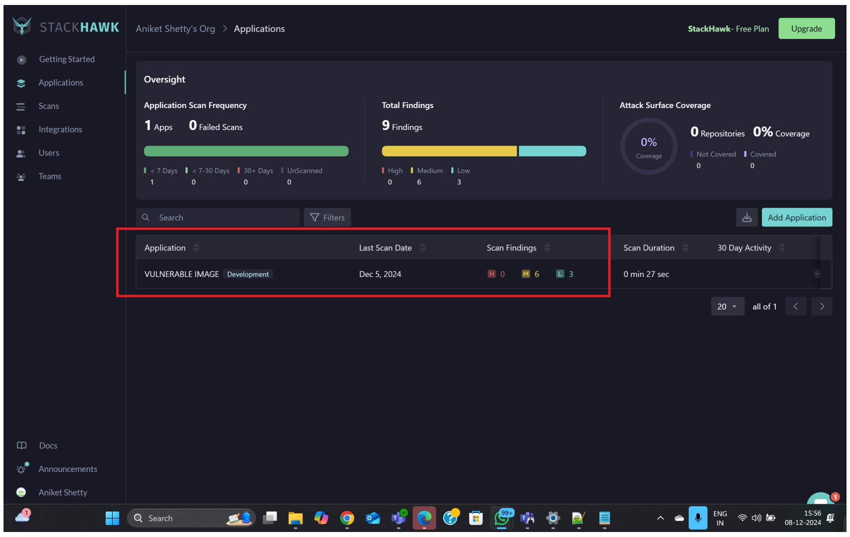
Separate names with commas.

fixtures, tests, __tests__, test, __test__, ci, node_mo

Figure 3: Synk Tool Integration with GitHub Repository

dashboard the logs of vulnerabilities are mentioned which can be accessed, for mitigating it going forward.

2: **StackHawk**: It is a Dynamic security testing tool which detects vulnerabilities in real-time application. Like Synk, StackHawk's integration start's with cloning the tool with the GitHub repositories and creating a user/testing account for this research. An application was created at the stackhawk dashboard as shown in the figure 4, which needs the environment (production, pre-production, development) section to be selected, also attach the URL of the deployed application which was generated by GCP Cloud Run. The created application generates an Application ID and YAML configuration file which was added in the project by creating a new directory and in cicd.yml file.


Aniket Shetty's Org > Applications

Oversight

Application Scan Frequency: 1 Apps, 0 Failed Scans

< 7 Days	< 7-30 Days	30+ Days	UnScanned
1	0	0	0

Total Findings: 9 Findings

High	Medium	Low
0	6	3

Attack Surface Coverage: 0% Coverage, 0 Repositories, 0% Coverage

Not Covered	Covered
0	0

Application	Last Scan Date	Scan Findings	Scan Duration	30 Day Activity
VULNERABLE IMAGE Development	Dec 5, 2024	H 0 M 6 L 3	0 min 27 sec	

20 all of 1

Figure 4: Stackhawk tool Integration with Github by creating an Application ID

3: **Datadog**: This is an Interactive Application Security Testing tool which was integrated by adding the Datadog serverless init and Java tracer in Dockerfile of the source code for installing a "Agent". Integration involves steps like enabling IAST service in datadog as shown in the figure 5 and by following the documentation that is provided which was bit challenging.[8] During integration it generated API and Application Key which is was added in the cicd.yml file. Also, a synthetic test was generated for the checking how well the tools is working with the application that is deployed on cloud and checks the traffic that's passing during the process. The logs were showcased at the dashboard where it showcases the real-time vulnerabilities and monitors it critically on priority.

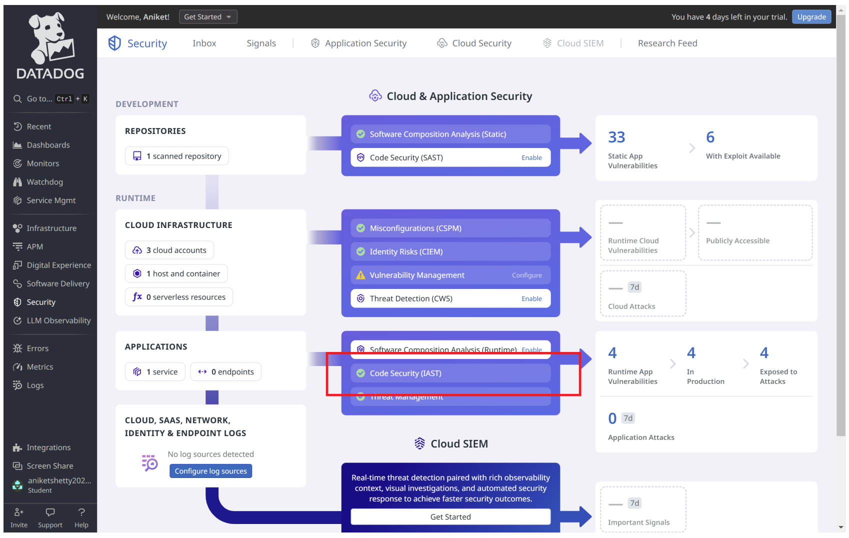


Figure 5: Datadog Integration with Github Action

6 Evaluation

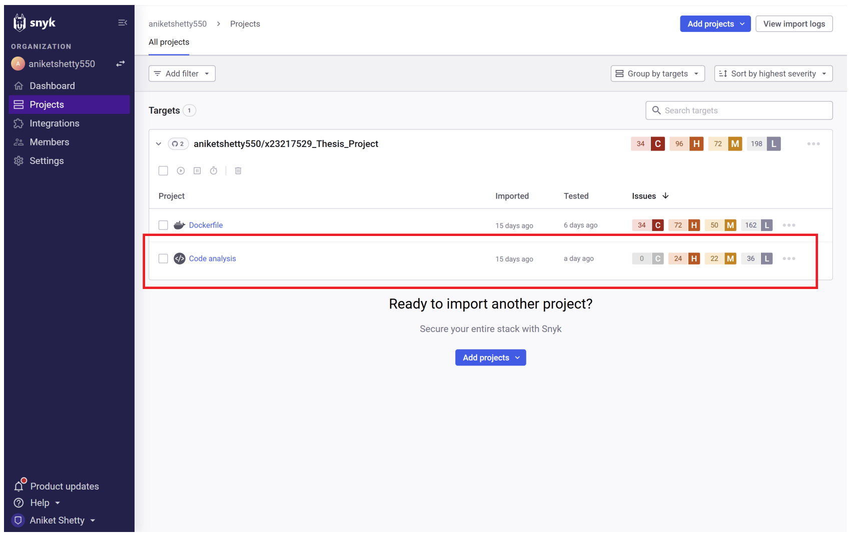
After the implementation part the results are showcased in this section with the help of comparison chart which is divided into three sections where efficiency, effectiveness and performance of the security tools are mentioned. This comparison is between the previous work which was performed on "Damn Vulnerable Web Application" Marandi et al. (2023)where "Static and Dynamic Application Security Testing" tools were used and the work which was proposed in this research where "Altro Mutual Online Banking Application" was used with one additional tool that is "Interactive Application Security Testing". Both the application works in a same way as they contain the similar vulnerabilities in it like SQL injection, XSS and Authentication Flaws. So, with the help of the security tools the number(performance) of vulnerabilities can be detected and the time(efficiency) taken to do that work, and the severity(effective) of the vulnerability.

6.1 Synk Security Tool

SAST, detected the static code vulnerability for both the applications. Synk has been integrated for DVWA application and it successfully detected vulnerabilities which are showcased in the table.3 But, when the tool was integrated with Altro Mutual, the results showcase 6 better advancement in terms of detecting number of vulnerabilities, time consumed and its severity level. It particularly identified the critical dependencies in the

[8]https://app.datadoghq.eu/security/appsec/vm/code?detection=runtime

application and the authentication flaws that were missed out during DVWA application. By doing this it showcases that the tool performs better for Altro Mutual application.



aniketshetty550/x23217529_Thesis_Project — 34 C 96 H 72 M 198 L

Project	Imported	Tested	Issues
Dockerfile	15 days ago	6 days ago	34 C 72 H 50 M 162 L
Code analysis	15 days ago	a day ago	0 C 24 H 22 M 36 L

Figure 6: Synk Output with- 29:High 15: Medium 1: Low Vulnerabilities

Additionally, with this improved performance and efficiency, it enhanced the performance of the CI/CD pipeline and allowed to work more smoothly for both the testing and production environment. Lastly, by integrating Snyk for Altro Mutual application showcased more effective, efficient & good-performing results.

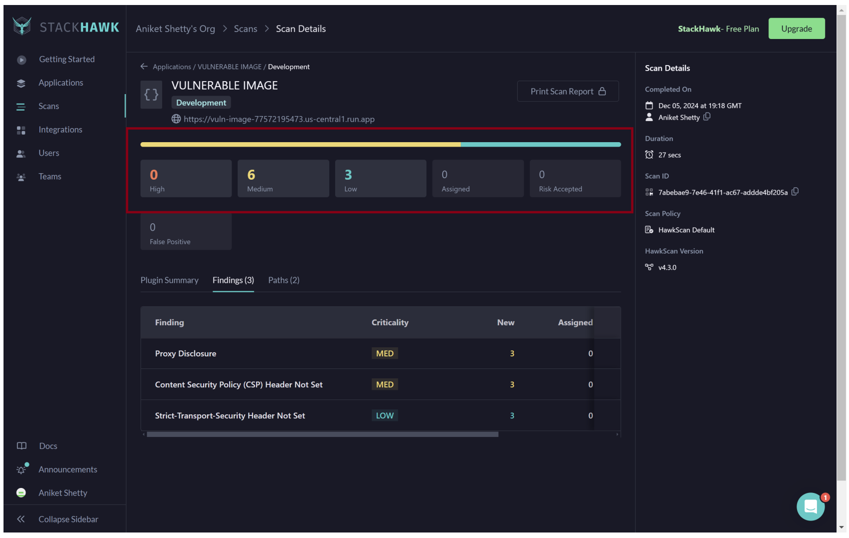
Table 3: Comparison of Synk Tool for both the Applications

Applications	**Effectiveness**	**Performance**	**Execution Time**
DVWA Marandi et al. (2023)	29: High 15: Medium 1: Low	45 Total Vulnerabilities	19sec
Altro Mutual	24: High 22: Medium 36: Low	82 Total Vulnerabilities	13sec

6.2 StackHawk Security Tool

This tool works dynamically for the application to check the vulnerability in real-time. From the previous workMarandi et al. (2023) which was performed it showcases that this tool shows some good results while detecting number of vulnerabilities and took less time and was effective. But, when this tool was integrated with Altro Mutual application, it didn't showed the expected results as shown in the figure.7 Even though stackhawk tool was powerful enough to identify the vulnerability, but the number of vulnerabilities detected, and its count was less than expected as shown in the chart.4 Also, it showed some good results during the vulnerability scanning after the tool was triggered by CI/CD pipeline, but lacked in checking the severity of it.

The main reason for this kind of behaviour by this tool for the Altro Mutual application was because of the less vulnerabilities this application had during the testing phase, and also stackhawk tool was not able to detect more vulnerabilities during this environment. From this integration it can be justified that stackhawk is efficient enough to



Applications / VULNERABLE IMAGE / Development

VULNERABLE IMAGE

Development

https://vuln-image-77572195473.us-central1.run.app

0 High | 6 Medium | 3 Low | 0 Assigned | 0 Risk Accepted

0 False Positive

Finding	Criticality	New	Assigned
Proxy Disclosure	MED	3	0
Content Security Policy (CSP) Header Not Set	MED	3	0
Strict-Transport-Security Header Not Set	LOW	3	0

Scan Details

Completed On: Dec 05, 2024 at 19:18 GMT, Aniket Shetty

Duration: 27 secs

Scan ID: 7abebae9-7e46-41f1-ac67-addde4bf205a

Scan Policy: HawkScan Default

HawkScan Version: v4.3.0

Figure 7: Stackhawk Output with: 0: High 6: Medium 3: Low: 9 Vulnerabilities

provide good performance during the run-time environment, but it will not always provide good results in identifying vulnerabilities, like in this case for Altro Mutual application.

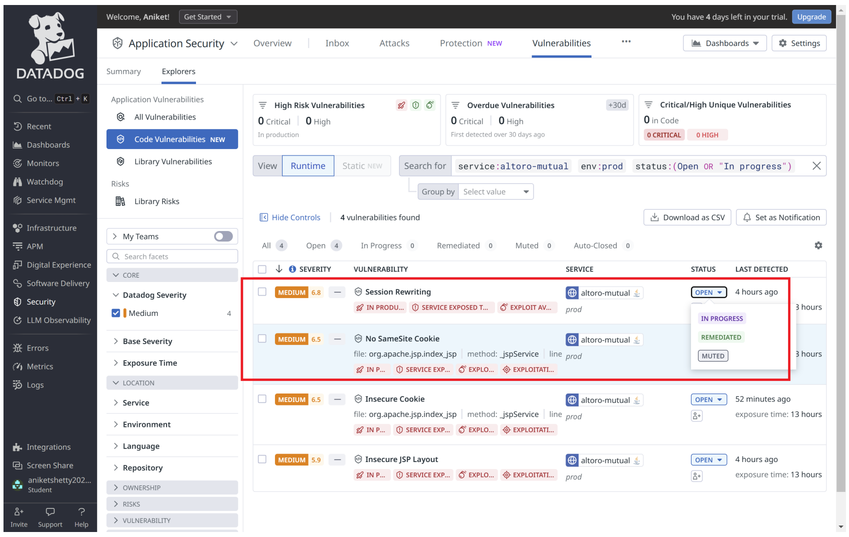
Table 4: Comparison of StackHawk Tool for both the Applications

Applications	**Effectiveness**	**Performance**	**Execution Time**
DVWA Marandi et al. (2023)	0: High 39: Medium 43: Low	82 Total Vulnerabilities	2min 1sec
Altro Mutual	0: High 6: Medium 3: Low	9 Total Vulnerabilities	43sec

6.3 Datadog Security Tool

Integrating Datadog Security testing tool in CI/CD pipeline was bit challenging, as the provided documentation was not enough to get the results. But after doing lots of trail and error testing, the proposed results were achieved by enhancing the security of the Altro Mutual Application, and by detecting vulnerabilities and remediating it in real-time. From the results as shown in the figure 8 it showcases 4 medium level vulnerabilities for the application during runtime, by visualizing the effectiveness of detected vulnerabilities and monitoring the security threat that were established. By following this process, it filled up the gap of static & dyanamic testing, which was not satisfied by the previous work.Marandi et al. (2023)

Furthermore, with the help of datadog lots of services were enabled, like Logs for the application, monitoring Errors, analysing Traffic, tracking Latency, CPU utilization, Application Performance Monitoring, checking Code Vulnerability, and lastly Threat Detection. By using this service, it helped us to know the performance and effectiveness of the application. Lastly, after implementing Datadog IAST for this project it was worth doing it, even it was not easy enough but was effective after implementing it into the CI/CD pipeline for securing the data from threats.



Welcome, Aniket! You have 4 days left in your trial.

Application Security — Overview | Inbox | Attacks | Protection NEW | Vulnerabilities

High Risk Vulnerabilities: 0 Critical | 0 High — In production
Overdue Vulnerabilities (+30d): 0 Critical | 0 High — First detected over 30 days ago
Critical/High Unique Vulnerabilities: 0 in Code — 0 CRITICAL, 0 HIGH

View: Runtime | Static NEW — Search for `service:altoro-mutual env:prod status:(Open OR "In progress")`

4 vulnerabilities found

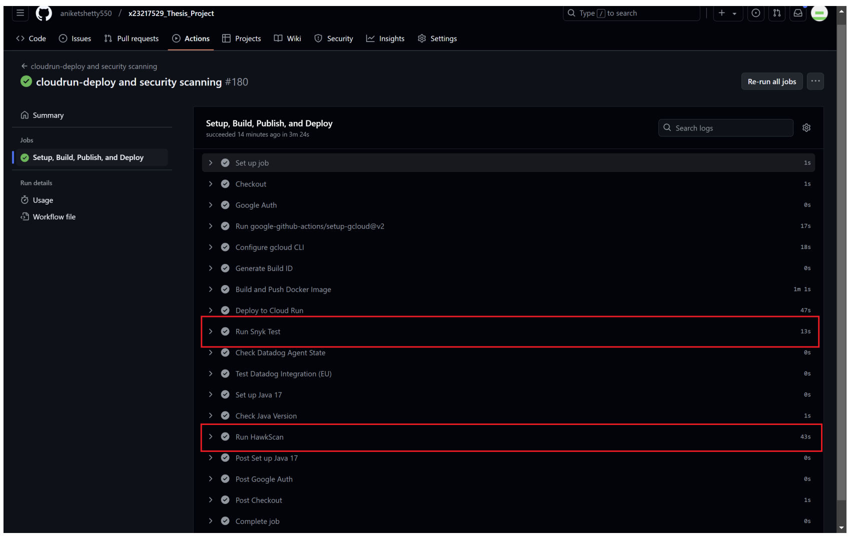
All 4 | Open 4 | In Progress 0 | Remediated 0 | Muted 0 | Auto-Closed 0

SEVERITY	VULNERABILITY	SERVICE	STATUS	LAST DETECTED
MEDIUM 6.8	Session Rewriting	altoro-mutual prod	OPEN	4 hours ago
MEDIUM 6.5	No SameSite Cookie (file: org.apache.jsp.index_jsp, method: _jspService)	altoro-mutual prod		
MEDIUM 6.5	Insecure Cookie (file: org.apache.jsp.index_jsp, method: _jspService)	altoro-mutual prod	OPEN	52 minutes ago; exposure time: 13 hours
MEDIUM 5.9	Insecure JSP Layout	altoro-mutual prod	OPEN	4 hours ago; exposure time: 13 hours

Figure 8: Datadog Output with- 4 Medium Vulnerabilities

6.4 GitHub Action CI/CD Pipeline Workflow

The implementation of CI/CD pipeline for this project is helpful as it automated the process, also the integration of security tools (Synk, Stackhawk, and Datadog) was successful as it performed good without any error or loss during the entire process. CI/CD pipeline helped in automating the "Build-Test-Deploy" process which ensured an ideal working environment. All the tools which were integrated were triggered in a correct manner at the planned stages in the workflow.9



aniketshetty550 / x23217529_Thesis_Project

cloudrun-deploy and security scanning #180

Setup, Build, Publish, and Deploy
succeeded 14 minutes ago in 3m 24s

```
Set up job                                   1s
Checkout                                     1s
Google Auth                                  0s
Run google-github-actions/setup-gcloud@v2   17s
Configure gcloud CLI                        18s
Generate Build ID                            0s
Build and Push Docker Image              1m 1s
Deploy to Cloud Run                         47s
Run Snyk Test                               13s
Check Datadog Agent State                    0s
Test Datadog Integration (EU)                0s
Set up Java 17                               0s
Check Java Version                           1s
Run HawkScan                                43s
Post Set up Java 17                          0s
Post Google Auth                             0s
Post Checkout                                1s
Complete job                                 0s
```

Figure 9: GitHub ActionCI/CD Pipeline Output for Altros Mutual Application

Later, using this pipeline, the altro mutual application was deployed in GCP cloud run `https://vuln-image-77572195473.us-central1.run.app/altoromutual/` for scalability and better interaction. By successfully achieving the CI/CD pipeline running on production and making it secure by using tools is the great design which is implemented in this project.

6.5 Discussion

The main objective of this research was to enhance the security vulnerabilities in CI/CD pipeline with the help of security testing tools like Synk, Stackhawk & Datadog which monitors the security vulnerabilities in an application. From the previous work that was been proposed Marandi et al. (2023), SAST and DAST was integrated to check the vulnerability in an DVWA application, which was based on PHP language, however in this research a minor changes was done by choosing Altro Mutual application that's based on Java based because, Datadog's IAST Code Security service doesn't supports PHP language.

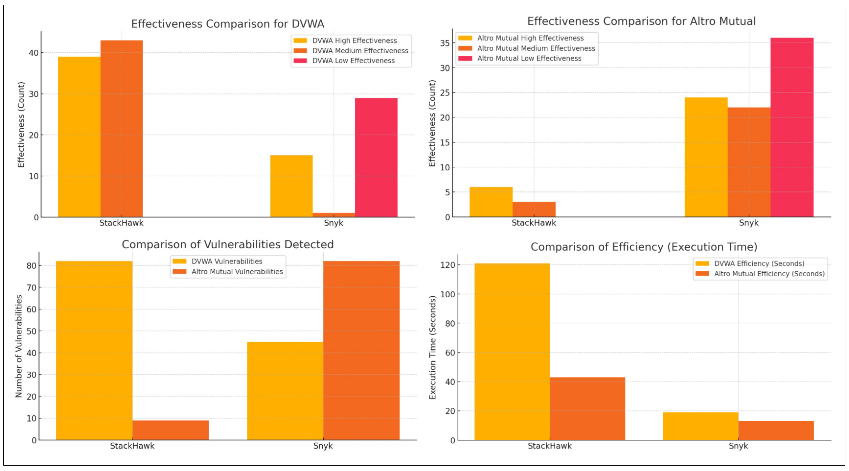
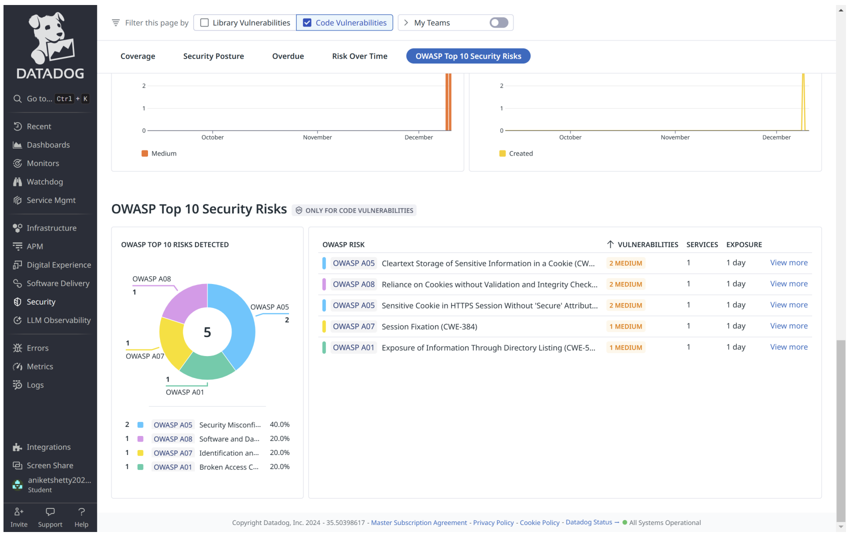


Figure 10: Discussion Chart for both the Application

The discussion chart 10 showcases the comparison for the achieved results and the previous work Marandi et al. (2023) in terms of Execution time, Performance and Severity levels. Additionally, Datadog IAST showcased the Top 10 OWASP results in the figure 11 where it discusses the vulnerabilities like SQL injection, XSS, which were monitored and analysed critically. Also, with the help of Datadog service, logs, traffic, detecting vulnerabilities, Application Performance Monitoring features are provided.



OWASP Top 10 Security Risks ONLY FOR CODE VULNERABILITIES

OWASP TOP 10 RISKS DETECTED

2	🟦	OWASP A05	Security Misconfi...	40.0%
1	🟪	OWASP A08	Software and Da...	20.0%
1	🟨	OWASP A07	Identification an...	20.0%
1	🟩	OWASP A01	Broken Access C...	20.0%

OWASP RISK		VULNERABILITIES	SERVICES	EXPOSURE	
OWASP A05	Cleartext Storage of Sensitive Information in a Cookie (CW...	2 MEDIUM	1	1 day	View more
OWASP A08	Reliance on Cookies without Validation and Integrity Check...	2 MEDIUM	1	1 day	View more
OWASP A05	Sensitive Cookie in HTTPS Session Without 'Secure' Attribut...	2 MEDIUM	1	1 day	View more
OWASP A07	Session Fixation (CWE-384)	1 MEDIUM	1	1 day	View more
OWASP A01	Exposure of Information Through Directory Listing (CWE-5...	1 MEDIUM	1	1 day	View more

Copyright Datadog, Inc. 2024 - 35.50398617 - Master Subscription Agreement - Privacy Policy - Cookie Policy - Datadog Status → All Systems Operational

Figure 11: IAST TOP 10 OWASP Vulnerability Detection

During this entire process there were several challenges that were faced while implementing Datadog IAST tool, as lack of documentation was there on internet and by the service provider it was bit challenging, so lots of trial and error were performed to get the proposed results. Additionally, by increasing the computation power of IAST, it may help many industries and organizations to keep their data secure, but this may be bit costly. Regardless of these challenges, IAST have provided a secure service for CI/CD pipeline by detecting vulnerabilities in real time environments.

7 Conclusion and Future Work

The reason of this research project was to answer the research questions that was been proposed, and to check how well the security testing tools (Synk, Stackhawk, Datadog) were functioning in terms of efficiency, effectiveness and performance while detecting vulnerabilities in an application. From the previous work the results were compared and to showcase the improvement of work that was done in this research. The results, that were achieved by "Synk" (SAST) tool for "Altro Mutual" application were better in performance (82 vulnerabilities detected), efficiency (13 sec) & effectiveness (complete severity) than the "DVWA" application that was worked on previously. On the other hand, the results for "Stackhawk" (DAST) tool when integrated with "DVWA" application showcased better results in terms of performance as it detected 82 vulnerabilities for "DVWA" applications (as compared to 9 vulnerabilities for Altro Mutual).

Lastly, the novelty of this research was the integration of "IAST" using "Datadog" security testing and monitoring tool, which was not performed in the previous research, and was integrated for "Altro Mutual" application in this research. By taking crucial steps and making this integration possible was a big task. As Datadog, detected the vulnerability with more precision and performed well, as it contains both the features of SAST & DAST in a single tool.

Finally, after doing this integration of SAST, DAST & IAST security tools, the performance of CI/CD pipeline was scalable, and the application security was enhanced, by that means this answered the research question.

7.1 Future Work

For future, this research can be taken to another level of security enhancement in CI/CD pipeline by integrating "Runtime Application Security Protection" (RASP) in an application. As RASP provides runtime mitigation for the threats and vulnerabilities that are detected during the production environment. Additionally, IAST tool needs to be introduced to more vulnerable applications so that it helps us to understand how effective and efficient it works in the worst-case scenario for all stages in CI/CD pipeline. By, doing this the tools can be used by various organizations and industries to keep their data safe and secure from any threats or causes in the software development life cycle.

References

Ahmed, Z., Ahmed, Z., Francis, S. C. and Francis, S. C. (2019). Integrating security with devsecops: Techniques and challenges, *IEEE International Conference on Dielectrics* .

Aljohani, M. A. and Alqahtani, S. S. (2023). A unified framework for automating software security analysis in devsecops, *International Conference on Software and Computer Applications* .

Chen, T. and Suo, H. (2022). Design and practice of security architecture via devsecops technology, *2022 IEEE 13th International Conference on Software Engineering and Service Science (ICSESS)*, IEEE, pp. 310–313.

David, P., Kushwaha, M. K. and Suseela, G. (2024). Devsecops in finance: Strengthening the security model of applications, *International Conference Design, Engineering and Computer Sciences* .

Diaz, J., Pérez, J. E., Lopez-Peña, M. A., Mena, G. A. and Yagüe, A. (2019). Self-service cybersecurity monitoring as enabler for devsecops, *Ieee Access* **7**: 100283–100295.

Feio, C., Santos, N., Escravana, N. and Pacheco, B. (2024). An empirical study of devsecops focused on continuous security testing, *IEEE European Symposium on Security and Privacy Workshops (EuroSPW)* .

Gauthier, F., Keynes, N., Allen, N., Corney, D. and Krishnan, P. (2018). Scalable static analysis to detect security vulnerabilities: Challenges and solutions, *2018 IEEE Cybersecurity Development (SecDev)*, IEEE, pp. 134–134.

Grigorieva, N. M., Petrenko, A. S. and Petrenko, S. A. (2024). Development of secure software based on the new devsecops technology, *2024 Conference of Young Researchers in Electrical and Electronic Engineering (ElCon)* .

Ji, M., Yin, M. and Zhou, Y. H. (2022). Application of static taint analysis in rasp protection strategy, *Proceedings of the 2022 International Conference on Cyber Security*, pp. 40–45.

Kushwaha, M. K., David, P. and Suseela, G. (2024). Automation and devsecops: Streamlining security measures in financial system, *2024 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, IEEE, pp. 1–6.

Marandi, M., Bertia, A. and Silas, S. (2023). Implementing and automating security scanning to a devsecops ci/cd pipeline, *2023 World Conference on Communication Computing (WCONF)* .

Masood, A. and Java, J. (2015). Static analysis for web service security-tools & techniques for a secure development life cycle, *2015 IEEE International Symposium on Technologies for Homeland Security (HST)*, IEEE, pp. 1–6.

Putra, A. M. and Kabetta, H. (2022). Implementation of devsecops by integrating static and dynamic security testing in ci/cd pipelines, *2022 IEEE International Conference of Computer Science and Information Technology (ICOSNIKOM)* .

Rajapakse, R. N., Rajapakse, R. N., Zahedi, M., Zahedi, M., Babar, M. A. and Babar, M. A. (2021). An empirical analysis of practitioners' perspectives on security tool integration into devops, *International Symposium on Empirical Software Engineering and Measurement* .

Rangnau, T., Rangnau, T., v. Buijtenen, R., v. Buijtenen, R., Fransen, F., Fransen, F., Fransen, F., Türkmen, F., Turkmen, F. and Turkmen, F. (2020). Continuous security testing: A case study on integrating dynamic security testing tools in ci/cd pipelines, *IEEE International Enterprise Distributed Object Computing Conference* .

Şengül, Ö., Özkılıçaslan, H., Arda, E., Yavanoğlu, U., Doğru, I. A. and Selçuk, A. A. (2021). Implementing a method for docker image security, *2021 International Conference on Information Security and Cryptology (ISCTURKEY)*, IEEE, pp. 34–39.

Seth, A., Bhattacharya, S., Elder, S., Zahan, N. and Williams, L. A. (2023). Comparing effectiveness and efficiency of interactive application security testing (iast) and runtime application self-protection (rasp) tools in a large java-based system, *arXiv.org* .

Singh, S. and Singh, N. (2016). Containers & docker: Emerging roles & future of cloud technology, *2016 2nd international conference on applied and theoretical computing and communication technology (iCATccT)*, IEEE, pp. 804–807.

Sun, X., Cheng, Y., Qu, X. and Li, H. (2021). Design and implementation of security test pipeline based on devsecops, *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, Vol. 4, IEEE, pp. 532–535.

Throner, S., Throner, S., Hütter, H., Hutter, H., Sänger, N., Sanger, N., Schneider, M., Schneider, M., Hanselmann, S., Hanselmann, S., Petrovic, P., Petrovic, P., Abeck, S. and Abeck, S. (2021). An advanced devops environment for microservice-based applications, *International Symposium on Service Oriented Software Engineering* .

Yadati, N. S. P. K. (n.d.). Integrating dynamic security testing tools into ci/cd pipelines: A continuous security testing case study.

Zunnurhain, K. and Duclervil, S. R. (2019). A new project management tool based on devsecops, *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, IEEE, pp. 239–243.