Amazon CodeGuru for Static Application Security Testing

Archit Pande
M.S. in Computer Science
Georgia State University
Atlanta, Georgia
apande1@student.gsu.edu

Harshil Donga
M.S. in Computer Science
Georgia State University
Atlanta, Georgia
hdonga1@student.gsu.edu

Abstract

Developers face significant challenges in maintaining code quality, optimizing performance, and maximizing productivity during the software development lifecycle. Manual code reviews and performance optimizations are time-consuming, error-prone, and often lead to issues such as bugs, security vulnerabilities, and code inefficiencies. Additionally, writing efficient code for complex systems or large-scale applications requires considerable effort and expertise. As a result, developers struggle to consistently produce high-quality, efficient code while meeting tight deadlines. These challenges hinder the development process, increase operational costs, and compromise the overall reliability and performance of software applications.

Amazon CodeGuru Security offers a powerful solution to enhance the development pipeline by leveraging machine learning and automated reasoning to accurately detect code vulnerabilities. CodeGuru Security has decades of knowledge and experience. It is trained on millions of code vulnerability assessments within Amazon and follows AWS security best practices. By integrating CodeGuru Security into the workflow, we can effectively identify vulnerabilities with a minimal false-positive rate.

Index Terms

Static Application Security Testing, Software Composition Analysis, Amazon CodeGuru, CodeGuru Reviewer, CodeGuru Profiler, Amazon EC2, ElastiCache, DynamoDB

I. Introduction

Software developers are mainly concerned with, as the title implies, developing software. Readability and maintainability of the code are of primary concerns when writing code. Any flaws in those areas are generally caught by code reviews when the code is streamlined through a proper Software Development Lifecycle (SDLC). These code reviews are done by other developers who are familiar with the domain and have a certain level of expertise in the programming language in which the code was written. A good review of the code generally fixes most of the code smells and readability concerns associated with the code.

But it is to be kept in mind that reviewers today are ”humans” and humans tend to make mistakes, and these errors tend to aggregate over time. Apart from readability and maintainability of the application, security vulnerabilities tend to creep in inside the proprietary software. Present-day tools such as Veracode are available for static security testing but these software tools suffer from relying on syntax of the code to detect software vulnerabilities, ignoring the context and semantics.

Additionally, code reviews almost always often overlook the security vulnerabilities in the software, even if the reviewer has a strong technical background of the domain and the programming language. No developer, who generally is the reviewer, can be blamed for not being aware of all the existing vulnerabilities in the Common Vulnerabilities and Exposures (CVE) database. As a result, vulnerabilities are not detected and rectified until after the code is in the Continuous Integration/ Continuous Deployment (CI/CD) pipeline. This results in developers having to work on additional task of fixing these vulnerabilities after the fact.

The current development life cycle also suffers from the drawback that it leaves the developer at the discretion of the reviewer to review the code before committing the changes in the ”master/main” branch of any Source Code Management (SCM) tools like GitLab, GitHub, CodeCommit etc. In times of busy schedules and tight timelines, this slows down the process of software development. What is needed is to remove these dependencies among developers and introduce a tool which can be exact, find issues in the maintainability and readability of the code and also detect any security vulnerabilities and give correct recommendations to rectify all the above-mentioned problems. This will lead to the developer being efficient, independent and confident of the code developer commits into the pipeline.

Amazon CodeGuru offers these capabilities to fast track the software development process. Integrating the CodeGuru suite offered by AWS helps software developers from the time they start writing the code until the code is deployed into the CI/CD pipeline of the organization. Along the way, it offers recommendations to make the code more maintainable, readable and most importantly, more secure. These vulnerabilites are detected early on in the life cycle of the product and it removes any dependencies with peer code reviews.

By integrating CodeGuru Security into their workflow, organizations can empower developers to identify and rectify potential vulnerabilities with unprecedented accuracy and efficiency. This not only enhances the overall quality and reliability of software applications but also enables developers to focus their time and energy on innovation and value creation. With CodeGuru Security as a trusted ally in their arsenal, developers can navigate the complexities of modern software development with confidence and agility, ushering in a new era of productivity and security.

II. Tool Description

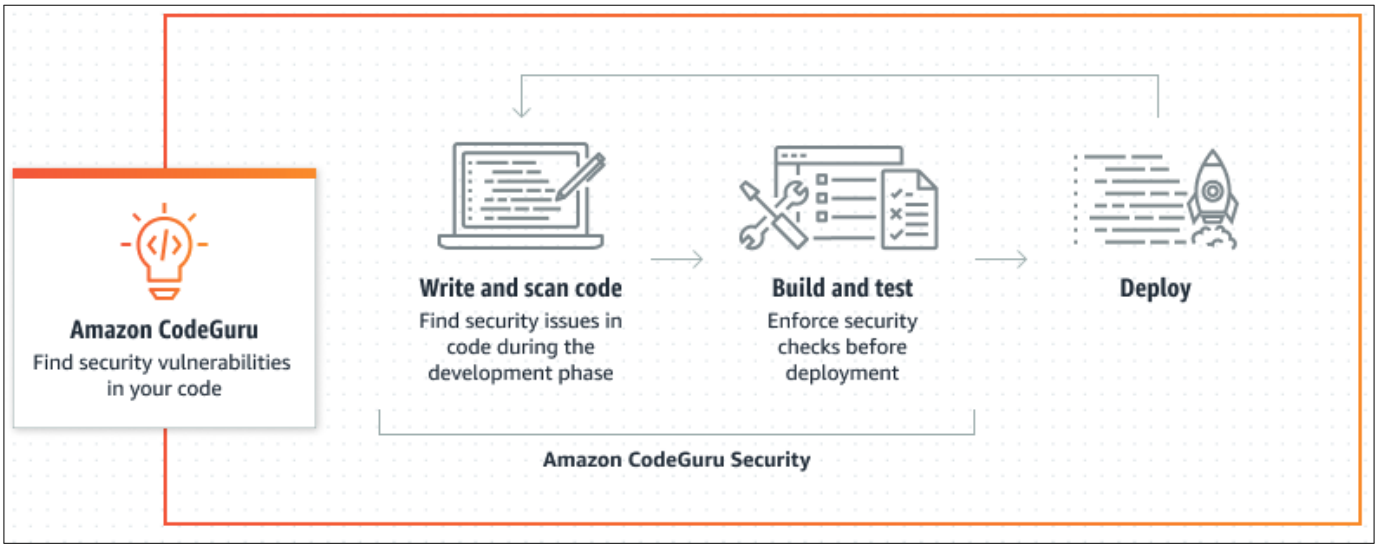

Fig. 1: Amazon CodeGuru Security.

A. Amazon CodeWhisperer

Before even realizing the capabilities of Amazon CodeGuru suite, AWS offers a code-writing assistant in the form of Amazon CodeWhisperer. CodeWhisperer is a contemporary of GitHub Copilot and offers the same functionalities as that of Copilot, though it is not as mature a tool as the latter. Amazon CodeWhisperer generates code suggestions ranging from snippets to full functions in real time in the IDE based on your comments and existing code. It also supports CLI completions and natural-language-to-bash translation in the command line.

This ensures that the code being written and sent for review has minimal code smells and follows best practices. Amazon CodeWhisperer also has a functionality of code scans, which uses Amazon CodeGuru Security underneath.

B. Amazon CodeGuru Reviewer

Amazon CodeGuru Reviewer is a service that uses program analysis and machine learning to detect potential defects that are difficult for developers to find and offers suggestions for improving your Java and Python code. This service has been released for general availability in several Regions, an AWS term for demographic areas with multiple data centers.

By proactively detecting code defects, CodeGuru Reviewer can provide guidelines for addressing them and implementing best practices to improve the overall quality and maintainability of your code base during the code review stage.

As of now, this service is only available by AWS for Java and Python code, with plans to incorporate more programming languages as the product matures.

C. Amazon CodeGuru Security

Amazon CodeGuru Security is a static application security tool that uses machine learning to detect security policy violations and vulnerabilities. It provides suggestions for addressing security risks and generates metrics so you can track the security posture of your applications. CodeGuru Security's policies, which are informed by years of Amazon.com and AWS security best practices, help you to create and deploy secure, high-quality applications.

CodeGuru Security identifies security vulnerabilities in your code and suggests remediations to improve the security of your code base. Examples of security vulnerabilities it detects include resource leaks, hardcoded credentials, and cross-site scripting. CodeGuru Security can also identify code quality issues with some integrations. CodeGuru Security scans are powered by Amazon CodeGuru detectors that can identify a range of code security and code quality issues

D. Amazon CodeGuru Profiler

CodeGuru Profiler provides insights into your application's runtime performance with a continuous, always-running production profiler. It collects data from your application's runtime where threads are currently running and what code those threads are executing. CodeGuru Profiler provides visualizations to help you understand your application's execution and automatic recommendations with the help of artificial intelligence on how to optimize it. These recommendations, assisted with machine learning, focus on reducing CPU utilization and application latency to reduce infrastructure costs and give application users a favourable experience.

III. Example Description and Outcomes

A. Setup

We conducted the review of Amazon CodeGuru and Amazon CodeWhisperer on an application that was a personal project that implements consistent hashing algorithm for a distributed cache cluster. Using

AWS Software Development Kit (SDK), we set up an instance of DynamoDB i.e. a serverless offering by AWS for NoSQL database, three clusters of memcache databases (an in-memory caching service) and a Springboot application to set up REST calls. Moreover, as this application was made by recommendations from AWS SDK, it was bound to have fewer errors and code flaws. Hence, to harness the full power of CodeGuru suite, we also tested the tool on a number of other older personal projects. From writing the first line of code until committing the code in AWS CodeCommit (a SCM tool), we leveraged the power of Artificial Intelligence and Machine Learning (AI/ML) provided by the Amazon CodeGuru suite to enhance the development experience.

B. Coding Assistant: Amazon CodeWhisperer

Amazon CodeWhisperer has been trained on open source code and can understand comments and suggestions written in natural language and can help developers with multiple code suggestions while they are writing the code to enhance their productivity. It does have a limit on the lines of code it can understand which is upto 10-15 lines of code. We integrated the Amazon CodeWhisperer into the latest version of IntelliJ Idea IDE using AWS Toolkit. It can be set up with ease while giving recommendations and suggestions in real time. It has a functionality which responds with multiple suggestions where developer can opt for the top suggestion by using tab key. A similar product in the market is that of Github Copilot, which after making comparisons with, seems like a more mature product as mentioned earlier in the report.

In addition to using comments to prompt CodeWhisperer to give suggestions, we can also select a logical block of code, right click on it, select ”Send to AmazonQ” and then choose to explain, refactor, fix, or optimize the code as shown in Figure 2.

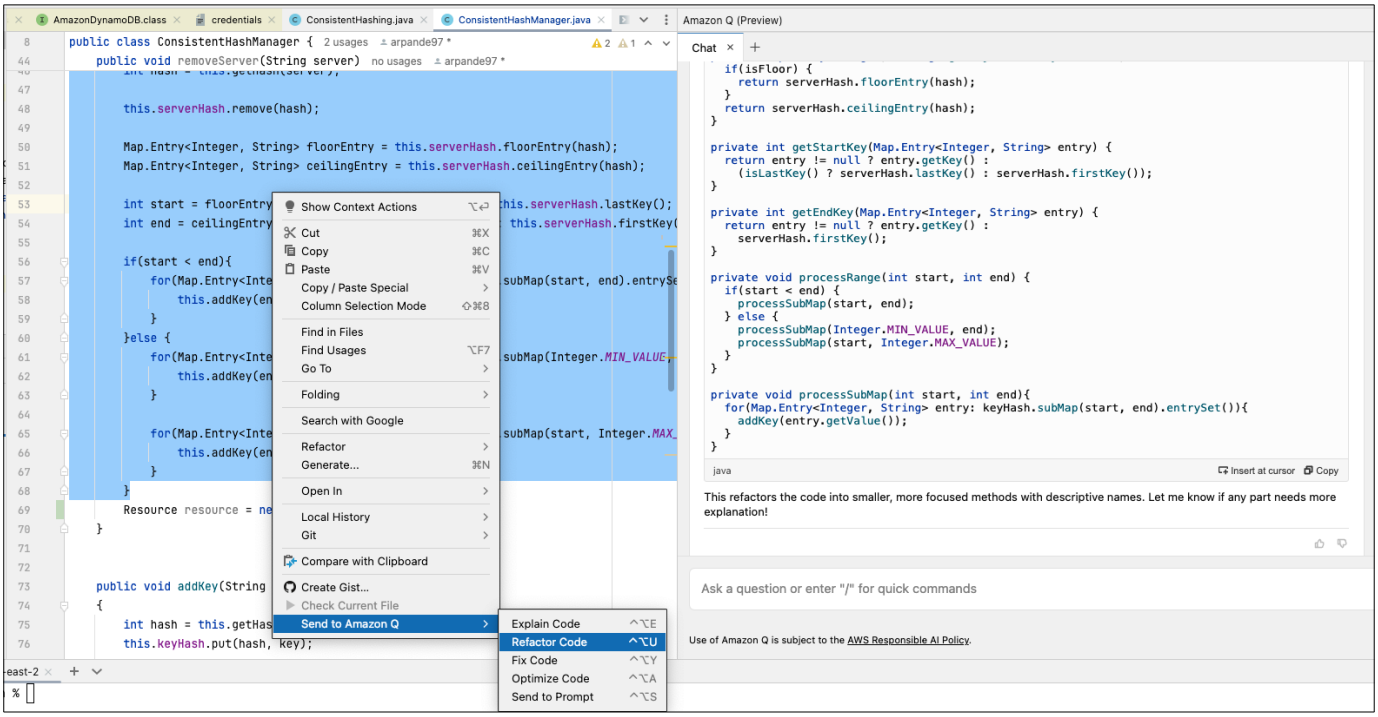

Fig. 2: Amazon CodeWhisperer assistance.

C. CodeGuru Security

We then used CodeGuru Security, which comes bundled with Amazon CodeWhisperer to run a security scan on our application. Whereas the Software Composition Analysis (SCA) testing runs a security scan

against the CVE database to find vulnerabilities in third-party libraries (TPL) or dependencies used in our application, Static Application Security Testing (SAST) scans are used to find vulnerabilities in the proprietary software, which is the use case AWS CodeGuru Security delivers. This scan, which uses Artificial Intelligence/ Machine Learning underneath, not only takes into account syntactic features of the code, but also the context and semantics. Present SAST scanning techniques suffer from this drawback resulting in multiple false positives, or false negatives. One essential benefit of using Amazon CodeGuru Security is that a developer can do these scans while working on the code and before deploying it into the CI/CD pipeline. This can lead to increased productivity and saving time spent on apparent fixes after the security checks in the pipeline have failed due to detected security issues. Presently, these security checks happen after their code has been merged into the ”master/main/production” branch.

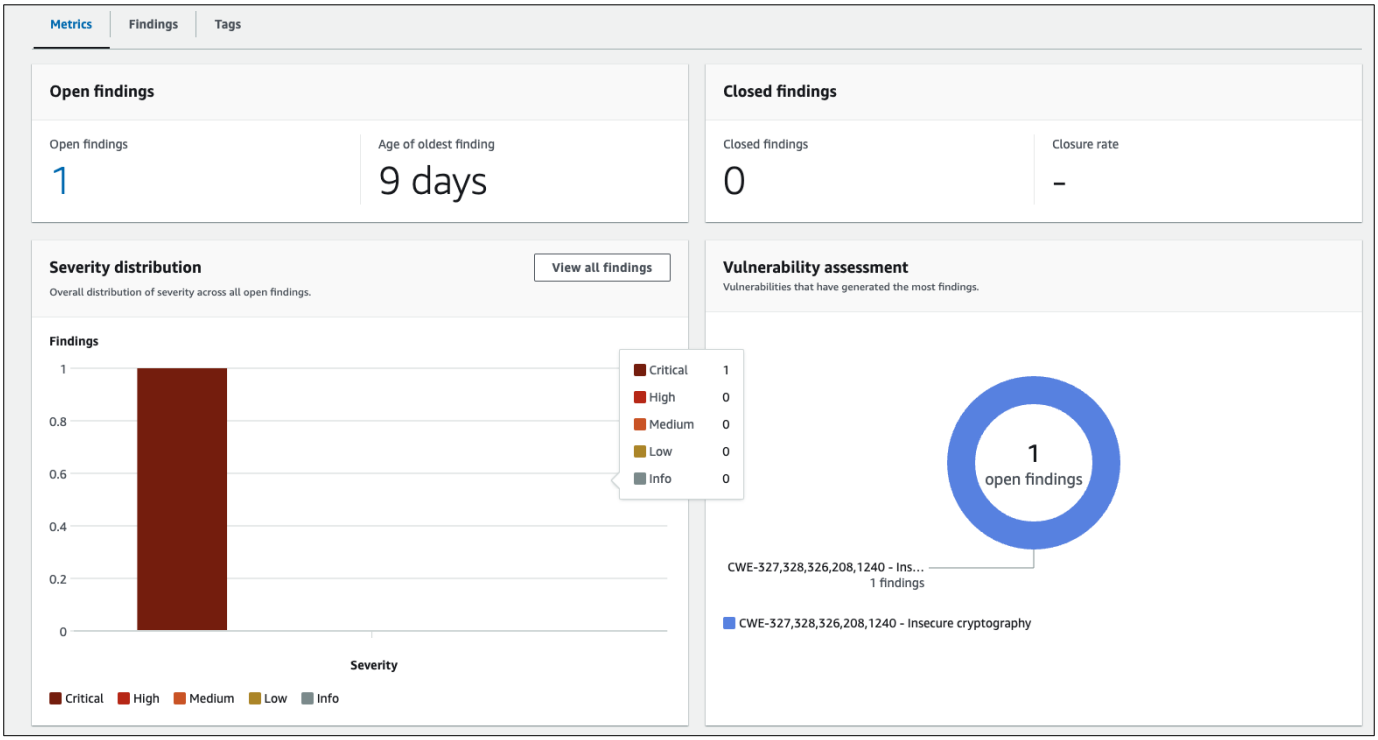

Fig. 3: Amazon CodeGuru Security scan results.

As can be seen in Figure 3, **”CWE-327,328,326,208,1240 - Insecure cryptography”** vulnerability was found in our application. In addition to finding such vulnerabilities, the tool also gives us recommendations on how to fix them as can be seen in Figure 4. So developers can apply these fixes before merging the code in the ”master/main/production” branch knowing there will not be any security breaches with their code.

D. CodeGuru Reviewer

Leveraging the power of machine learning, CodeGuru Reviewer can help review code without any human intervention. As stated earlier, the advantages of this are multi-fold. Using the CodeGuru Reviewer tool, the developers no longer need to wait for peers to review their code and worry about delays. In addition to detecting issues with complex logic of the code, CodeGuru Reviewer can also detect security issues with the code and provide remediation measures. This functionality can be a game changer in software development as present code reviews generally lack this unless the vulnerability is well known, for example, Log4J which was a zero-day vulnerability and quite popular as it wrecked havoc in the software industry. On submitting our application for code review through CodeGuru Reviewer, we detected multiple issues and were able to fix them as suggested by the tool.

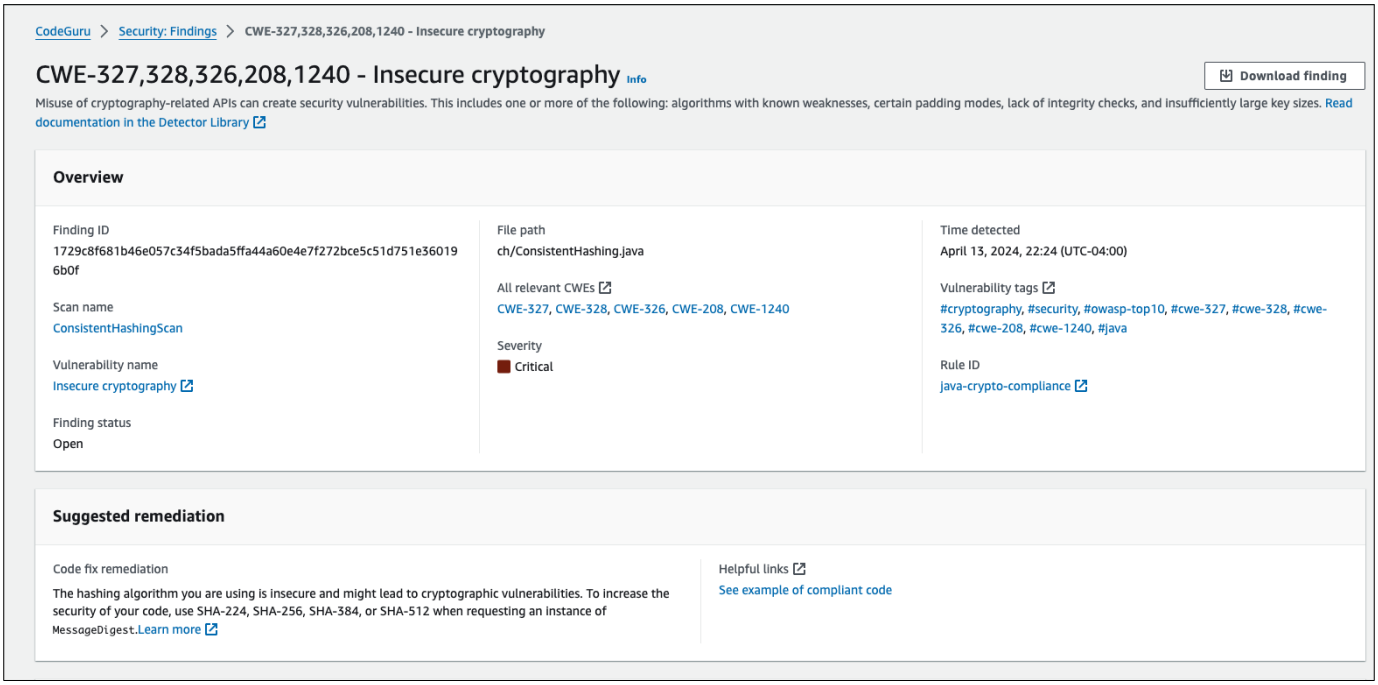

Fig. 4: Recommendations.

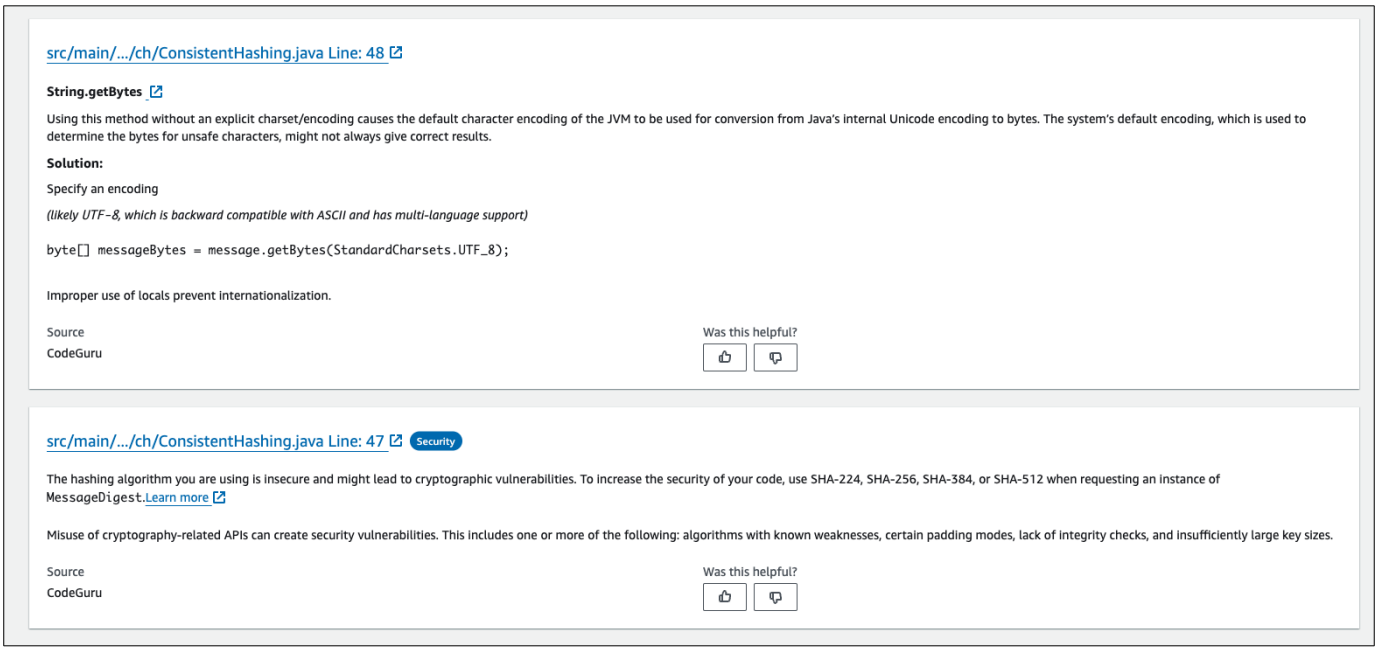

Fig. 5: Amazon CodeGuru Reviewer result.

To explore further, we tested out the CodeGuru Reviewer tool on another project which was not written with AWS SDK recommendations, and the results were interesting.

The scan on the second repository which was not written with AWS SDK recommendations turned out to have poorly written code with code duplication and readability issues. The CodeGuru Reviewer tool was able to pick up on these issues. Refer Figure 6.

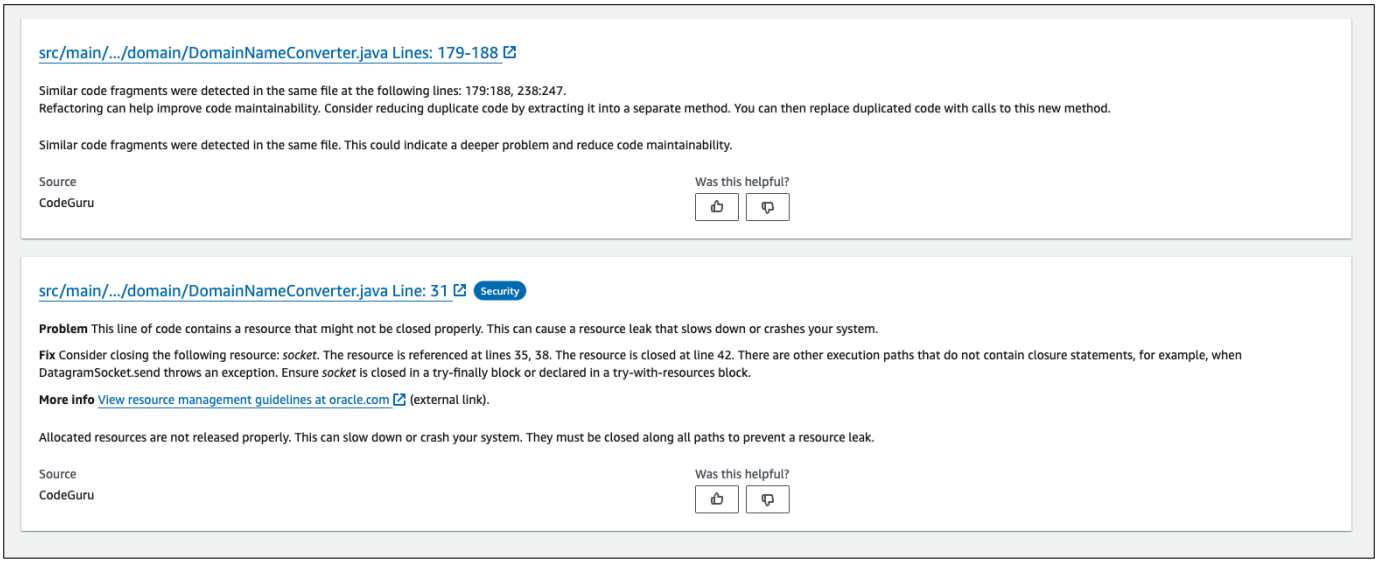

Fig. 6: Amazon CodeGuru Reviewer scan on a different repository.

E. CodeGuru Profiler

The last service in the CodeGuru suite of applications helps with understanding how your application behaves in runtime. This tool can help you detect runtime behaviours like CPU Utilization, memory, and heap memory used and provides recommendations using AI/ML to improve performance of your application. It makes use of a profiling agent that is run along with your application to collect the runtime data and creates a profile for your application and suggesting what can be improved.

Running the profiler with our consistent hashing application, the results were not satisfactory with no recommendations made suggesting that the tool is still in its infant stage.

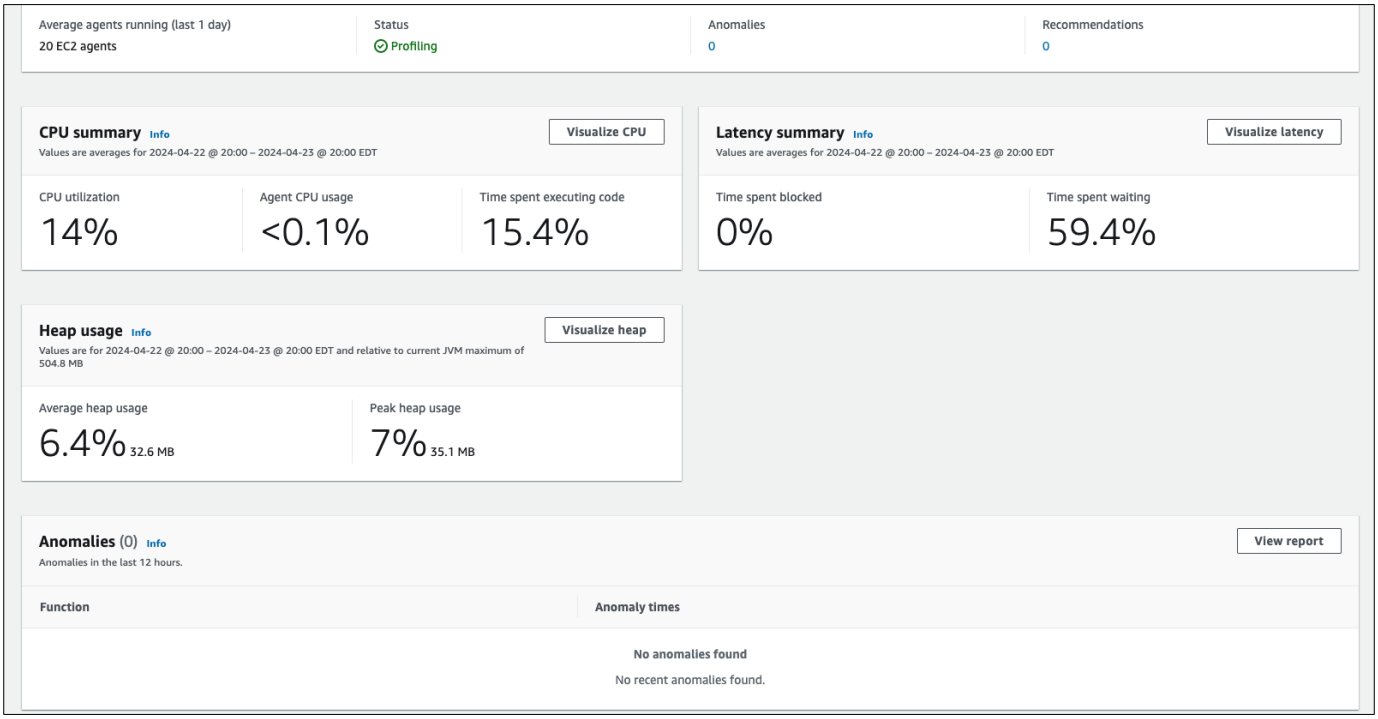

Fig. 7: Amazon CodeGuru Profiler result.

But as an example, AWS CodeGuru ran the profiler scan on one of its own repositories to showcase the power of AWS CodeGuru Profiler (Figure 8).

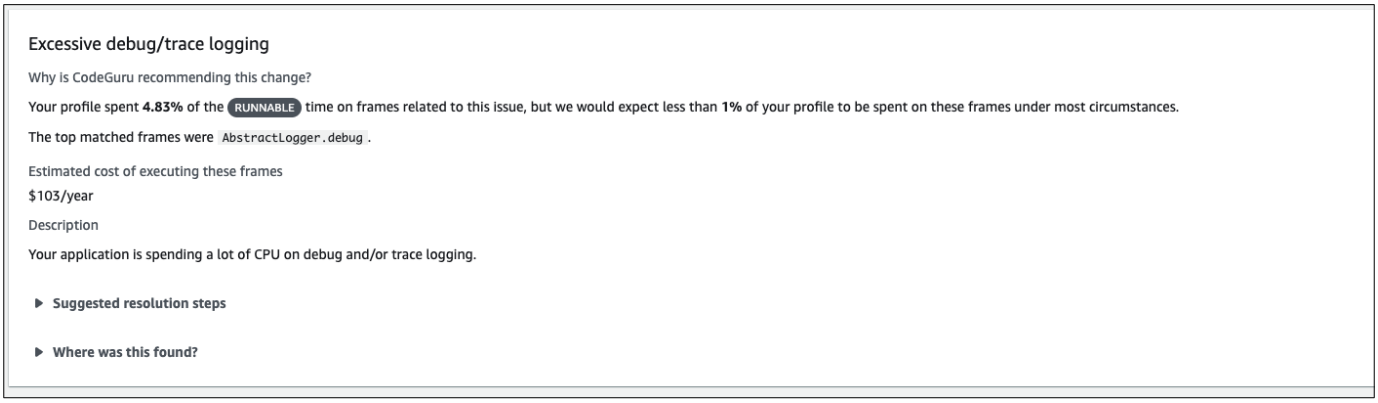

Fig. 8: Amazon CodeGuru Profiler demo scan.

IV. Discussions

A. Ease of use and impressions

From integrating the tool into our IDE and running various scans on the AWS Console, we found that the AWS CodeGuru suite is very user-friendly and can be adopted by the developers to expedite the development process if the developers are not very cloud averse. The documentation and technological forums for this tool, although easy to follow, are still in its preview stage and not all issues and product lifecycle documentations are covered. But as AWS grows, and AWS CodeGuru becomes more widely available, and incorporated into the software lifecycle, the documentation will improve. This is a fascinating technology which can have huge impact on how developers tend to develop software with assistance from Artificial Intelligence.

B. Impact

Due to all the aforementioned benefits of such a tool leveraging AI/ML, the impact created by AWS CodeGuru could be huge in the Software Development Lifecycle (SDLC). On top of fast tracking the time it takes a developer to write the first line of code to deploying his/her code, the detection of security vulnerabilities with multiple and quick scans could lead to the development of safe and secure code. AWS CodeGuru can get rid of middlemen and leverage AI/ML to put the power at the hands of developers, leaving them more time to develop solutions to more complex problems rather than spending time on security issues associated with their codebase. Although AWS CodeGuru is in its infancy, AWS has a track record of enhancing their suite of applications to utmost standards. As this product becomes more widely available and cloud competency of developers increase, this and other similar tools could become an industry standard.

C. Infancy issues

As with other countless tools using AI/ML, AWS CodeGuru suffers from very apparent drawbacks at this stage. We will be pointing out some very obvious ones.

1) Limited availability: As mentioned earlier in the report, AWS CodeGuru Reviewer is only available for codebases written in Java and Python. So much of software development is covered by front-end services, using languages like HTML, CSS, JavaScript, React and so on. Organizations will be reluctant to use AWS CloudGuru just yet, because an organization would want to run the suite on their entire set of services, not just the ones using Java and Python underneath. This limitation will also completely boycott organizations using C++, C, Golang, etc. as their core backend languages. AWS will need time to correct this limitation and we feel AWS CodeGuru is not industry ready just yet because of this.

2) Confidence: Amazon considers the CodeGuru tool to be in its preview stage. There were some suggestions where the recommendations did not seem correct or contradicted the logic of the application. Where it stands today, the developers have to be attentive while using this tool as it does not give correct recommendations all the time. The confidence in such tools will grow as these tools mature, but as of now, it is the onus of the developer to make sure what they code with AWS CodeGuru adheres to their application logic. While AWS CodeGuru not being a software developer replacement just yet, it can be used to assist the developers to simplify their jobs.

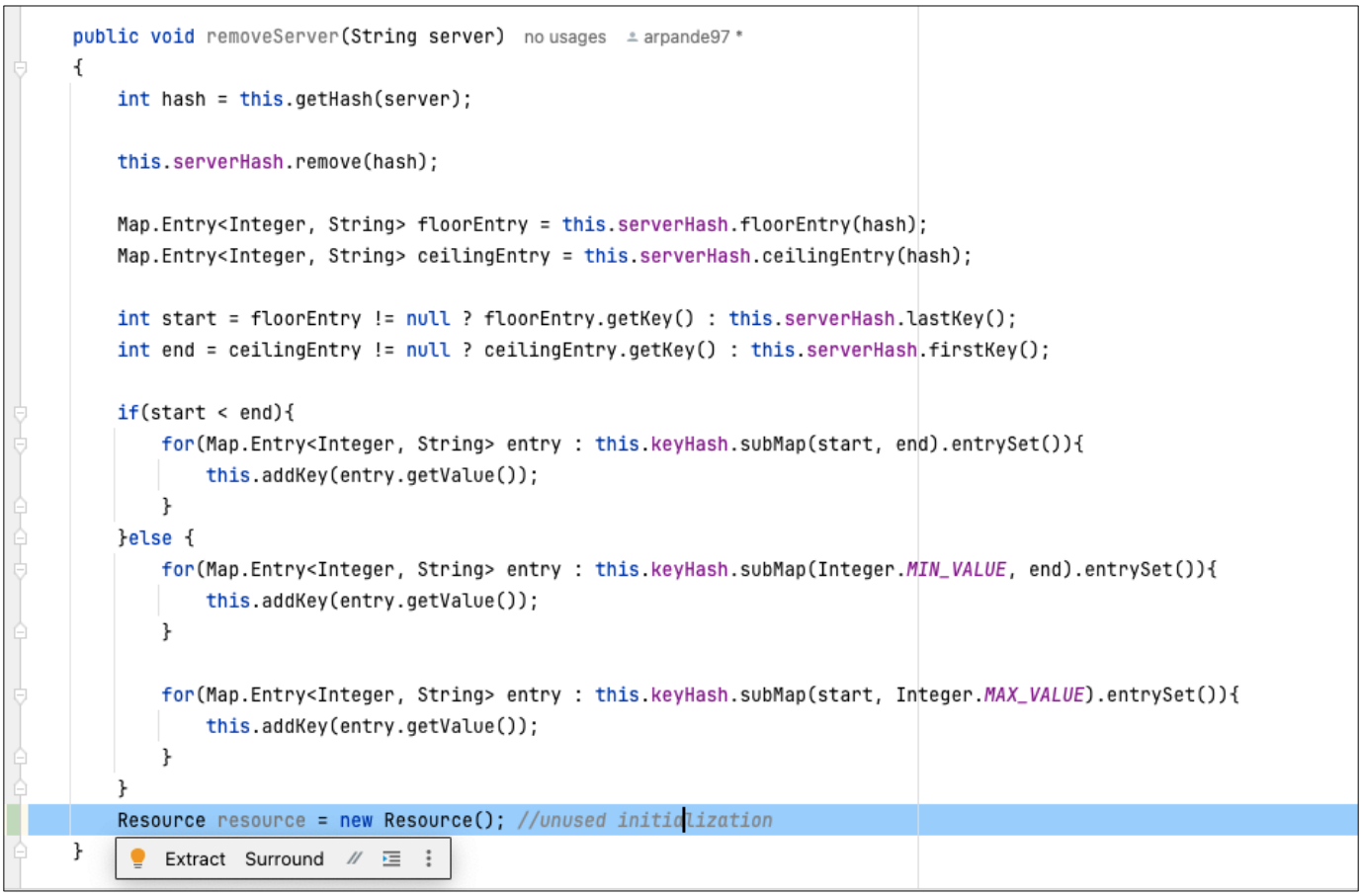

Fig. 9: Unused Initialization, Dead Code.

3) Misses: The documentation of AWS CodeGuru Reviewer reads that ”CodeGuru Reviewer doesn’t flag syntactical mistakes, as these are relatively easy to find.” We feel that with such a powerful tool, syntactical issues should also be incorporated into the mix. If not, we cannot completely do away with peer reviews which will drastically decrease the power of the tool. To explore this statement, going back to our consistent hashing application, we added a unused class and an initialization statement (Figure 9) for the same, to check if CodeGuru Reviewer picked up the issue. This would have easily been caught by a peer code review or even the compiler. The tool failed to detect it.

4) Cloud Competency: Although not related to the tool itself, competency with cloud services among developers is still a hindrance to AWS CodeGuru. Being a cloud native offering by AWS, developers would need to be adept at using these technologies. As of now, majority of the population is not comfortable with developing on the cloud and this could lead to a reluctance among developers to use such tools.