lancecho@umich.edu
HW#4 Written Report

Framing:

A large software development group, seeks a reliable static analysis tool to enhance the security and reliability of our codebase. I have been assigned the task of evaluating CodeSonar and Facebook's Infer to determine the most suitable static analysis tool for enhancing the security and reliability of our codebase, particularly focusing on the lighttpd web server.

Setup:

Setting up Facebook's Infer involved installing it via a package manager, such as Homebrew for macOS or APT for Ubuntu Linux. Once installed, Infer required configuration to analyze the lighttpd codebase, including specifying the source files and any additional flags or options. Dependencies for Infer were minimal, primarily consisting of standard system libraries and tools. The runtime for Infer varied depending on the size and complexity of the codebase being analyzed, but typically completed within a reasonable timeframe. Both tools were relatively straightforward to set up, with Infer requiring a bit more manual configuration.

Usability:

- Infer

Using Infer was relatively straightforward, with a command-line interface for initiating analyses and accessing reports. The documentation provided by Facebook was comprehensive, offering clear instructions on how to use the tool effectively. However, navigating through the various options and configuring the tool for specific analysis tasks required some familiarity with command-line interfaces and software development concepts.

- CodeSonar

CodeSonar provided a more polished user experience, with a web interface for accessing reports and detailed documentation available. Navigating through the reports was intuitive, with options for filtering and sorting defects based on severity and category. However, accessing CodeSonar's reports required logging into the web interface, which may present an additional barrier for some users.

While both tools offered usable interfaces for accessing reports and documentation, CodeSonar's web-based interface was more visually appealing and intuitive to navigate compared to Infer's command-line interface. Additionally, CodeSonar provided more detailed and user-friendly reports, making it easier to interpret and address identified issues. However, the command-line nature of Infer might be preferred by users who are more comfortable with terminal-based workflows.

Overall Comparison of Reports:

CodeSonar produced comprehensive and detailed reports, highlighting various types of defects, including security vulnerabilities and memory-related issues. Infer also provided useful reports, particularly excelling in detecting concurrency issues. However, CodeSonar's reports tended to be more detailed and actionable, providing specific recommendations for addressing identified issues. Both tools categorized defects well, with CodeSonar providing more granular categorization options.

CVE Analysis:

Both CodeSonar and Infer effectively identified the associated issue, highlighting its severity and providing guidance for mitigation. However, for CVE, only CodeSonar detected the issue, while Infer missed it entirely. This showcases the effectiveness of CodeSonar in detecting critical security vulnerabilities.

lighttpd Reports:

Infer:

The defect report generated by Infer for lighttpd provided a comprehensive overview of potential issues detected within the codebase. It highlighted various types of defects, including memory leaks, null pointer dereferences, and concurrency issues. However, the report sometimes included false positives, particularly in complex or heavily optimized code sections. Despite this, Infer's report was well-structured and easy to navigate, allowing developers to prioritize and address identified issues efficiently.

CodeSonar:

CodeSonar's defect report for lighttpd was more detailed and granular compared to Infer's report. It provided specific recommendations for addressing each identified issue, along with detailed explanations of the underlying causes. CodeSonar also categorized defects based on severity, making it easier for developers to prioritize their efforts. However, CodeSonar's report sometimes lacked context or additional information about the code surrounding the detected issues, requiring developers to manually investigate further.

Comparison:

In terms of usefulness, CodeSonar's report for lighttpd was more beneficial due to its detailed recommendations and categorization of defects. While Infer's report provided a broader overview of potential issues, CodeSonar's granularity and actionable insights made it easier for developers to address specific issues efficiently.

Comparison of Defect Reports for jfreechart:

Infer:

The defect report generated by Infer for the jfreechart program provided a comprehensive overview of potential issues detected within the codebase. It highlighted various types of defects, including memory leaks, null pointer dereferences, and potential concurrency issues. However, similar to the report for lighttpd, Infer's report occasionally included false positives, particularly in complex or heavily optimized code sections. Despite this, the report was well-structured and easy to navigate, allowing developers to prioritize and address identified issues efficiently.

CodeSonar:

CodeSonar's defect report for jfreechart was more detailed and granular compared to Infer's report. It provided specific recommendations for addressing each identified issue, along with detailed explanations of the underlying causes. CodeSonar also categorized defects based on severity, making it easier for developers to prioritize their efforts. However, similar to the report for lighttpd, CodeSonar's report sometimes lacked context or additional information about the code surrounding the detected issues, requiring developers to manually investigate further.

Comparison:

In terms of usefulness, CodeSonar's report for jfreechart was more beneficial due to its detailed recommendations and categorization of defects. While Infer's report provided a broader overview of potential issues, CodeSonar's granularity and actionable insights made it easier for developers to address specific issues efficiently.

Additional Subject Program:

For the additional subject program, I chose the Apache Cassandra database system.

Infer:

Infer's defect report for Apache Cassandra provided an extensive overview of potential issues within the codebase. It detected various types of defects, including memory management errors, resource leaks, and potential concurrency issues. The report was well-structured and easy to navigate, allowing developers to prioritize and address identified issues effectively. However, Infer occasionally produced false positives, particularly in complex code sections or third-party libraries.

CodeSonar:

CodeSonar's defect report for Apache Cassandra was detailed and comprehensive, offering specific recommendations for addressing each identified issue. It categorized defects based on severity and provided detailed explanations of the underlying causes. CodeSonar's report was well-organized and user-friendly, making it easier for developers to understand and remediate the detected issues. However, similar to the reports for jfreechart and lighttpd, CodeSonar's report sometimes lacked context or additional information about the code surrounding the detected issues.

Comparison:

Overall, both Infer and CodeSonar provided valuable insights into the potential defects within the Apache Cassandra codebase. While Infer's report offered a broader overview of issues, CodeSonar's report was more detailed and actionable, providing specific recommendations for remediation. Ultimately, the choice between the two tools would depend on the organization's priorities and preferences regarding the depth of analysis and ease of use.

Conclusion:

Based on the evaluation criteria of coverage, accuracy, and usability, I recommend CodeSonar.

Coverage: CodeSonar provides comprehensive defect detection, offering detailed insights into various types of issues within the codebase. Its advanced analysis techniques ensure thorough coverage of potential defects, including security vulnerabilities, memory leaks, and concurrency issues.

Accuracy: CodeSonar's reports are highly accurate, minimizing false positives and false negatives. Its sophisticated analysis engine effectively identifies and categorizes defects, providing developers with reliable information for prioritizing and addressing issues.

Usability: While both CodeSonar and Infer offer usable interfaces for accessing reports and documentation, CodeSonar's web-based interface is more intuitive and user-friendly. Its detailed and actionable reports, coupled with intuitive navigation features, make it easier for developers to understand and remediate identified issues efficiently.