Name: Kaushik Kotian Roll no.:29 Div:D20B Batch:B

Experiment 4

Aim: Study of SAST Tools

Theory:

1. What are SAST tools

SAST stands for Static Application Security Testing. SAST tools are a category of security testing tools used to identify security vulnerabilities in the source code of an application without actually executing the application. These tools perform static analysis of the codebase, looking for potential security flaws, coding errors, and other weaknesses that could be exploited by attackers. However, it's important to note that while SAST tools are valuable for finding certain types of security issues, they should not be the sole security testing method used.

2. Why are SAST tools used

1. Early Detection: SAST tools allow developers and security teams to identify security issues during the development phase, even before the application is deployed. This enables early mitigation and reduces the cost and effort of fixing vulnerabilities later in the development lifecycle.

2. Automation: SAST tools automate the process of scanning and analyzing the codebase, which can be time-consuming and error-prone if done manually. They can quickly analyze large codebases and provide comprehensive reports on potential vulnerabilities.

3. Code Review: SAST tools act as an additional layer of code review. They can identify common coding errors, insecure coding practices, and security issues that developers might overlook during manual code reviews.

4. Compliance: Many industries have specific security standards and regulations that applications must adhere to. SAST tools help in ensuring compliance by identifying security flaws that could violate these standards.

5. Risk Reduction: By identifying and fixing security vulnerabilities early in the development process, SAST tools help reduce the overall risk of security breaches and data leaks in applications.

6. Integrations: SAST tools can be integrated into the development workflow and Continuous Integration/Continuous Deployment (CI/CD) pipelines, allowing for automated security checks as part of the development process.

3. Limitations of SAST Tools

1. Limited Context: SAST tools operate without executing the application, which means they may lack the context of how certain pieces of code are used in the actual runtime environment. Consequently, they might miss vulnerabilities that arise from specific runtime conditions or interactions with other components.

2. Code Complexity: SAST tools may struggle with highly complex or obfuscated code, making it difficult for them to analyze and accurately identify vulnerabilities in such codebases.

3. False Sense of Security: Relying solely on SAST tools for security testing might give a false sense of security, as they cannot identify all possible vulnerabilities. Additional testing methods, like Dynamic Application Security Testing (DAST) and manual penetration testing, are necessary for a more comprehensive assessment.

4. Limited Coverage: SAST tools may not cover certain types of security issues, such as some business logic vulnerabilities or issues stemming from misconfigurations and dependencies.

5. Maintenance Overhead: SAST tools require regular updates to stay current with the latest vulnerabilities and coding practices. Maintaining and updating the tool can be time-consuming and resource-intensive.

4. Tools of SAST:

1. **Klocwork**

Klocwork's Static Application Security Testing (SAST) is a powerful tool designed to detect potential security, quality, and reliability problems in software written in C, C++, C#, Java, JavaScript, Kotlin, and Python. By pinpointing these issues, Klocwork helps organizations adhere to industry standards and regulations.

Tailored for enterprise DevOps environments, Klocwork is versatile and can handle projects of all sizes. It effortlessly integrates with complex development setups and supports a wide range of developer tools, ensuring seamless collaboration and control over the software development process. With Klocwork's Differential Analysis engine, results are delivered swiftly without compromising accuracy, allowing developers to quickly identify and address vulnerabilities.

Moreover, Klocwork seamlessly integrates with CI/CD pipelines, automating continuous compliance checks with every code commit. This means that your software remains guarded against potential vulnerabilities, providing added confidence in the overall security and reliability of your application throughout the development lifecycle.

2. Checkmarx

Checkmarx is a well-known Static Application Security Testing (SAST) tool that is widely used to identify security vulnerabilities in the source code of applications. It helps developers and security professionals analyze their codebase for potential security flaws and weaknesses, allowing them to address these issues before the application is deployed to production.

Key features of Checkmarx include:

Static Code Analysis: Checkmarx performs a thorough analysis of the application's source code, looking for vulnerabilities such as SQL injection, cross-site scripting (XSS), insecure authentication mechanisms, and more.

Integration with Development Workflow: Checkmarx can integrate with popular Integrated Development Environments (IDEs) and code repositories, making it convenient for developers to scan their code as they write it. It can also integrate with Continuous Integration/Continuous Deployment (CI/CD) pipelines to ensure automated security checks throughout the development process.

Wide Language Support: Checkmarx supports a variety of programming languages, including but not limited to Java, C/C++, .NET, JavaScript, Python, Ruby, and more.

Comprehensive Reporting: The tool provides detailed reports that include information about identified vulnerabilities, their severity, and suggested remediation steps, helping developers understand and fix the issues effectively.

Vulnerability Tracking: Checkmarx allows teams to track the status of identified vulnerabilities and manage them throughout the development lifecycle.

Policy Enforcement: Organizations can define their security policies and standards, and Checkmarx can enforce these policies to ensure code compliance.

Continuous Updates: Checkmarx continually updates its vulnerability database to stay current with the latest security threats and coding best practices.

3. Veracode

Veracode's Static Analysis (SAST) product provides developers with fast and automated feedback, seamlessly integrating with popular IDEs, CI/CD pipelines, and work-tracking tools. This allows for efficient scanning and delivers actionable results directly to developers where they work. Prior to application deployment, in-depth policy scans offer clear guidance on identifying and prioritizing security issues, while providing organization-wide views of application security risks and program performance. Additionally, Veracode's patented automatic binary code analysis ensures superior accuracy and coverage by scanning the complete binary code of an application, enabling the discovery and analysis of security flaws more comprehensively than many other tools. With its cloud-based approach, Veracode eliminates the need for expensive software or hardware investments and specialized staff, making it a cost-effective solution for ensuring application security without access to source code for all parts of the application.

4. **SonarQube**

SonarQube is an open-source platform that combines Static Application Security Testing (SAST) capabilities with code quality analysis and continuous inspection. It performs static code analysis on various programming languages, detecting security vulnerabilities, bugs, code smells, and security hotspots in the codebase. With customizable rules and CI/CD integration, SonarQube enables automatic code analysis, reporting, and metrics to aid developers in addressing critical security issues and improving overall code quality. Its extensive plug-in ecosystem further extends its functionality, making it a powerful and widely-used SAST tool in the industry.

Conclusion:
Successfully learned what are SAST tools, it's limitations and some of the tools available in the market.