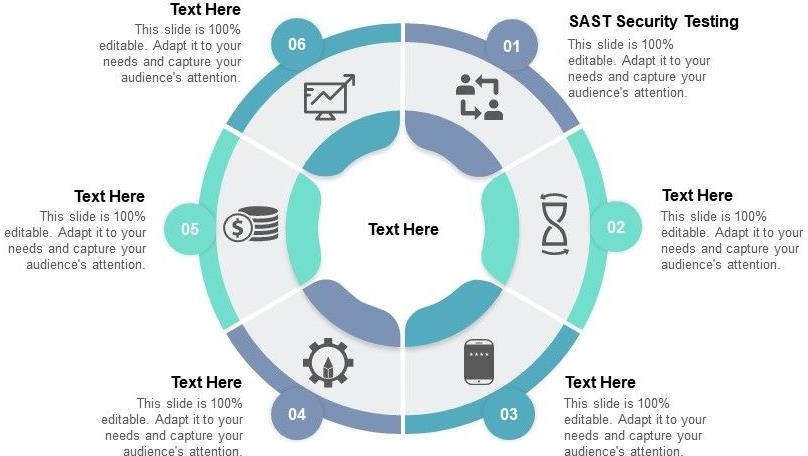
**Write Up Name :** SonarQube

# Theory :

**Write Up**

1. **What is Static Application Security Testing and explain how it is used? Ans :**

* Static application security testing (SAST), or [static analysis](https://www.synopsys.com/software-integrity/security-testing/static-analysis-sast.html), is a testing methodology that analyzes source code to find [security vulnerabilities](https://www.synopsys.com/blogs/software-security/types-of-security-vulnerabilities/) that make your organization’s applications susceptible to attack. SAST scans an application before the code is compiled. It’s also known as white box testing.
* Static application security testing (SAST) is a way to perform automated testing and analysis of a program’s source code without executing it to catch security vulnerabilities early on in the software development cycle. Also referred to as static code analysis, SAST is the process of parsing through the code looking at how it was written and checking for security vulnerabilities and safety concerns.
* One or more sets of coding guidelines like CERT and MISRA are used from the start of development to determine what coding rules to adhere to. Some practitioners also incorporate their own custom rules.
* Because static application security testing tools don’t need a running application to perform an analysis, they can be used early and often in the implementation phase of the software development life cycle (SDLC).



* As a developer is writing code, SAST can analyze it in real-time to inform the user of any rule violations, so you can immediately deal with issues and deliver higher quality applications out of the box while preventing issues at the end of the development process.
* Additionally, as SAST helps you audit code and triage issues during implementation, test automation tools can also easily integrate into development ecosystems where continuous integration/continuous delivery (CI/CD) are part of the workflow that helps assure secure, safe, and reliable code during integration, and before it’s delivered.

# How does SAST work?

* SAST uses a Static Code Analysis tool, which can be thought of like a security guard for a building. Similar to a security guard checking for unlocked doors and open windows that could provide entry to an intruder, a Static Code Analyzer looks at the source code to check for coding and design flaws that could allow for malicious code injection. Some examples of these malicious attacks, according to OWASP, include SQL Injections, Command Injections, and Server-Side Injections, among others.

# State the difference between SAST and DAST?

|  |  |  |
| --- | --- | --- |
| **Sr.**  **No.** | **Static Application Security Testing** | **Dynamic Application Security Testing** |
| 1. | SAST is a type of [White Box security testing.](https://www.geeksforgeeks.org/software-engineering-white-box-testing/) | DAST is type of [Black Box security testing](https://www.geeksforgeeks.org/software-engineering-black-box-testing/). |
| 2. | In SAST, application is tested from inside out. | In DAST, application is tested from outside in. |
| 3. | This type testing is a developers approach of testing. | This type testing is a hackers approach of testing. |
| 4. | No deployed application is required for Static Application Security Testing. | A running application is required for Dynamic Application Security Testing. |
| 5. | Finding vulnerabilities, identifying and fixing bugs is easier in SAST. | Finding vulnerabilities towards end of SDLC. |
| 6. | Fixing vulnerabilities is possible with little cost assistance. | It finds vulnerabilities towards end of SDLC, hence it is expensive to do so. |
| 7. | SAST can not discover issues related run time and environment. | DAST can discover issues related to run time and environment. |
| 8. | Typically it supports all types of software like web applications, web services, thick client. | Typically it only scans apps like web applications, web services but not other types of software. |
| 9. | In this testing, developer has knowledge about  design, application framework and implementation. | In this testing, tester has no knowledge about  application, design, frameworks and implementation that application is built on. |
| 10. | SAST testing requires source code to perform testing operation. | DAST testing does not require source code to perform testing operation. |
| 11. | As it scans static code and performs its testing operation that is why it is called Static Application Security Testing (SAST). | As it scans dynamic code and performs its testing operation that is why it is called Dynamic Application Security Testing (DAST). |
| 12. | This testing is performed in early stages of Software Development Life Cycle (SDLC). | This testing is performed at end of Software Development Life Cycle (SDLC). |
| 13. | In SAST, there is costly long duration dependent on experience of tester. | In DAST, tester is unable to perform  comprehensive application analysis since this is carried our externally. |
| 14. | In SAST, tester is able to perform comprehensive application analysis. | DAST can be done faster as compared to other types of testing due to restricted scope. |

1. **Explain briefly what is the function of SonarQube?**

# Ans :

* SonarQube is a Code Quality Assurance tool that collects and analyzes source code, and provides reports for the code quality of your project. It combines static and dynamic analysis tools and enables quality to be measured continually over time. Everything from minor styling choices, to design errors are inspected and evaluated by SonarQube.
* This provides users with a rich searchable history of the code to analyze where the code is messing up and determine whether or not it is styling issues, code defeats, code duplication, lack of test coverage, or excessively complex code.
* The software will analyze source code from different aspects and drills down the code layer by layer, moving module level down to the class level, with each level producing metric values and statistics that should reveal problematic areas in the source code that needs improvement.



* Sonarqube also ensures code reliability, Application security, and reduces technical debt by making your code base clean and maintainable. Sonarqube also provides support for 27 different languages, including C, C++, Java, Javascript, PHP, GO, Python, and much more. SonarQube also provides Ci/CD integration, and gives feedback during code review with branch analysis and pull request decoration.

# Benefits of SonarQube :

* + Sustainability –

Reduces complexity, possible vulnerabilities, and code duplications, optimising the life of applications.

* + Increase productivity –

Reduces the scale, cost of maintenance, and risk of the application; as such, it removes the need to spend more time changing the code

* + Quality code –

Code quality control is an inseparable part of the process of software development.

* + Detect Errors –

Detects errors in the code and alerts developers to fix them automatically before submitting them for output.

* + Increase consistency –

Determines where the code criteria are breached and enhances the quality

* + Business scaling –

No restriction on the number of projects to be evaluated

* + Enhance developer skills –

Regular feedback on quality problems helps developers to improve their coding skills

# Features of SonarQube:

* + Doesn’t just show you what’s wrong, but also offers quality and management tools to actively helps you correct issues.
  + Focuses on more than just bugs and complexity and offers more feature to help the programmers write code, such as coding rules, test coverage, de-duplications, API documentation, and code complexity all within a dashboard.
  + Regular feedback on quality problems helps developers to improve their coding skills.
  + No restriction on the number of projects to be evaluated.
  + Determines where the code criteria are breached and enhances the quality.