

Miki Lalwani

DI5B/37

Advance Dev.Ops lab

Experiment: 7.

Aim-

To understand static analysis SAST process and learn to integrate Jenkins SAST to SonarQube/Gitlab

Theory-

What is SAST?

Static application system testing (SAST) or static analysis, is a testing methodology that analyzes source code to find security vulnerabilities that make your organization's application susceptible to attack. SAST ~~attacks~~ scans an application before the code is compiled. It's also known as white box testing.

Why is SAST important?

Developers dramatically outnumber security staff. It can be difficult for organizations to find the resources to perform code reviews on even a fraction of its application. A key strength of SAST tools is the ability to analyze 100% of the codebase. Additionally they are much faster than manual secure code reviews performed by

## **Steps-**

### **Integrating Jenkins with SonarQube:**

Windows installation

Step 1 Install JDK 1.8

Step 2 download and install jenkins

<https://www.blazemeter.com/blog/how-to-install-jenkins-on-windows>

**Ubuntu installation**

<https://www.digitalocean.com/community/tutorials/how-to-install-java-with-apt-on-ubuntu-20-04#installing-the-default-jre-jdk>

Step 1 Install JDK 1.8

sudo apt-get install openjdk-8-jre

sudo apt install default-jre

<https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-20-04>

[Open SSH](#)

### **Prerequisites:**

- [Jenkins installed](#)
- [Docker Installed](#) (for SonarQube)
- SonarQube Docker Image

### **Steps to integrate Jenkins with SonarQube**

1. Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.
2. Run SonarQube in a Docker container using this command -

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

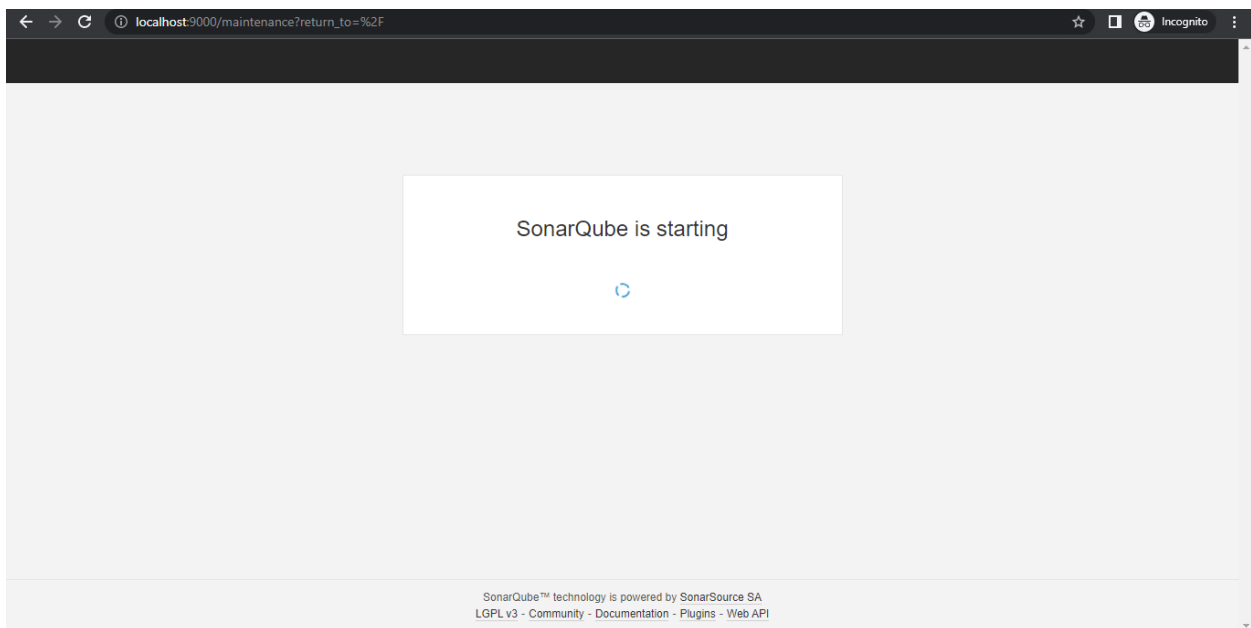
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Admin> docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
9621f1afde84: Pull complete
0da9106727c7: Pull complete
129c5a3f9c32: Pull complete
Digest: sha256:3fa9a76948fab6fafa41950bee256afea943773744723b5e4f38b340643516b9
Status: Downloaded newer image for sonarqube:latest
0239332cf48029d0a9a3991fd9bb95bb082e9172d32b73945b832f70318bb0ad
PS C:\Users\Admin>
```

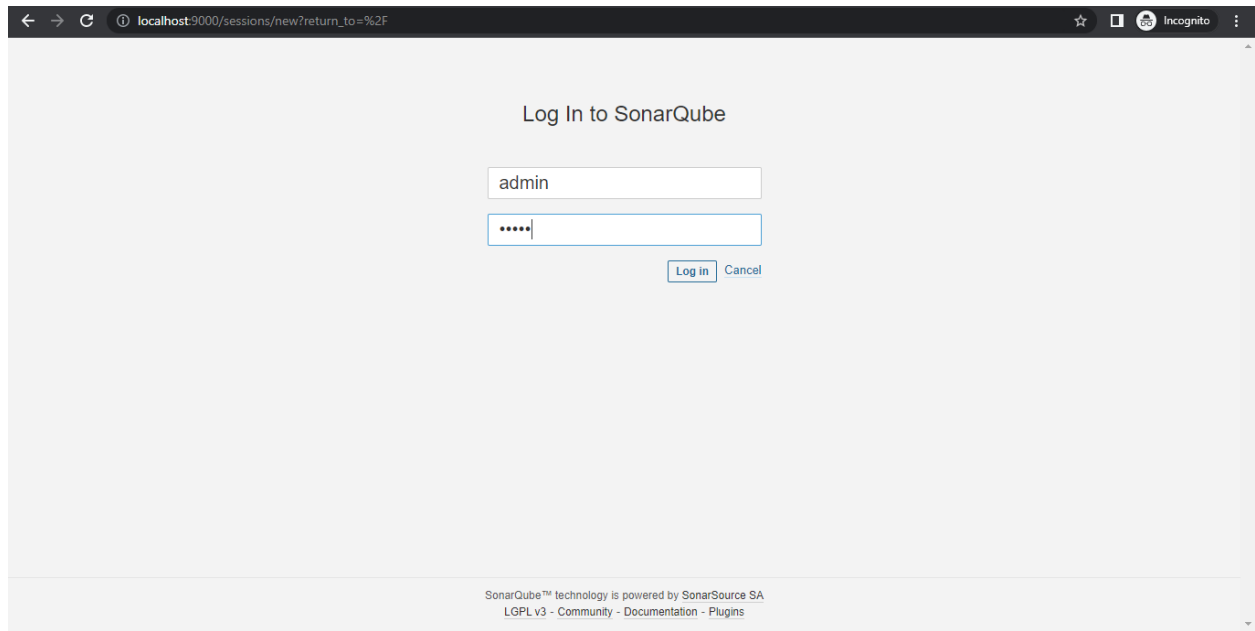
Warning: run below command only once

```
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest
```

3. Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



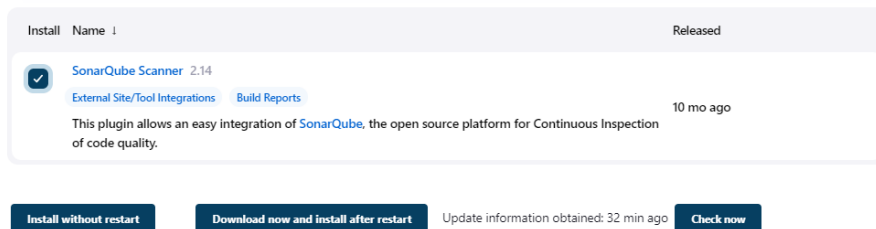
4. Login to SonarQube using username *admin* and password *admin*.



5. Create a manual project in SonarQube with the name **sonarqube**

Setup the project and come back to Jenkins Dashboard.

Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.



- Under Jenkins 'Configure System', look for SonarQube Servers and enter the details.

SonarQube installations

List of SonarQube installations

Name

Server URL

Default is http://localhost:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

- none -

+ Add

Advanced...

Save

Apply


Enter the Server Authentication token if needed.


- Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.
- After the configuration, create a New Item in Jenkins, choose a freestyle project.


Enter an item name

SonarQube

» Required field

**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds.

etc.

OK

- Choose this GitHub repository in Source Code Management.

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

Credentials ?

- none -

+ Add

Advanced...

Save Apply

[https://github.com/shazforiot/MSBuild\\_firstproject.git](https://github.com/shazforiot/MSBuild_firstproject.git)

It is a sample hello-world project with no vulnerabilities and issues, just to test the integration.

10. Under Build-> Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

Analysis properties ?

`sonar.projectKey=sonarqube`  
`sonar.login=admin`  
`sonar.password=student`  
`sonar.sources=C:\ProgramData\jenkins\workspace\SonarQube`  
`sonar.host.url=http://localhost:9000`

11. Go to [http://localhost:9000/<user\\_name>/permissions](http://localhost:9000/<user_name>/permissions) and allow Execute Permissions to the Admin user.

All	Users	Groups	Search for users or groups...	Administer System	Administer	Execute Analysis	Create
				<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
				<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
				<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input type="checkbox"/> Projects
				<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input type="checkbox"/> Projects

4 of 4 shown

12. Run The Build.

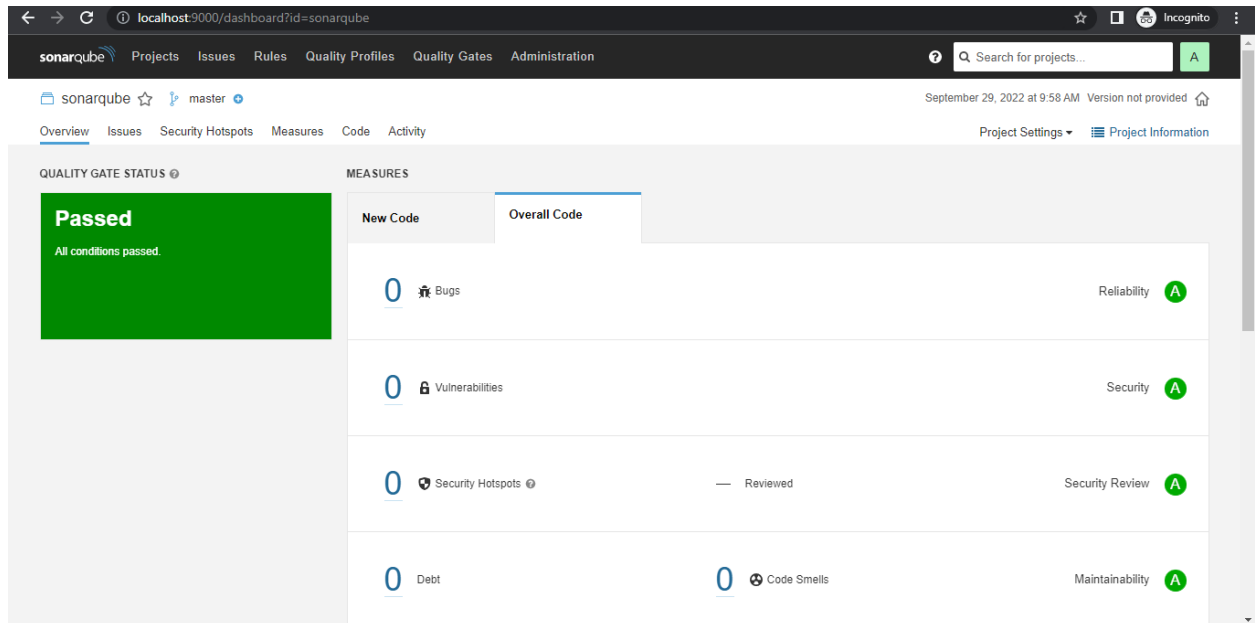
Check the console output.



```
Started by user admin
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\SonarQube
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\SonarQube\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_firstproject.git # timeout=10
Fetching upstream changes from https://github.com/shazforiot/MSBuild_firstproject.git
> git.exe --version # timeout=10
> git --version # 'git version 2.37.2.windows.2'
> git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_firstproject.git
+refs/heads/:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision f2bc042c04c6e72427c380bcae6d6fee7b49adf (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f f2bc042c04c6e72427c380bcae6d6fee7b49adf # timeout=10
Commit message: "updated"
> git.exe rev-list --no-walk f2bc042c04c6e72427c380bcae6d6fee7b49adf # timeout=10
[SonarQube] $ C:\ProgramData\Jenkins\jenkins\tools\udson.plugins.sonar.SonarRunnerInstallation\sonarqube_scanner\bin\sonar-
scanner.bat -Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=sonarqube -Dsonar.login=admin -
Dsonar.host.url=http://localhost:9000 -Dsonar.sources=C:\ProgramData\Jenkins\jenkins\workspace\SonarQube -
Dsonar.password=student -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\jenkins\workspace\SonarQube
WARN: Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'

INFO: Sensor C# [csharp] (done) | time=0ms
INFO: Sensor Analysis Warnings import [csharp]
INFO: Sensor Analysis Warnings import [csharp] (done) | time=0ms
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=0ms
INFO: SCM Publisher SCM provider for this project is: git
INFO: SCM Publisher 4 source files to be analyzed
INFO: SCM Publisher 4/4 source files have been analyzed (done) | time=256ms
INFO: CPD Executor Calculating CPD for 0 files
INFO: CPD Executor CPD calculation finished (done) | time=0ms
INFO: Analysis report generated in 44ms, dir size=119.3 kB
INFO: Analysis report compressed in 17ms, zip size=17.0 kB
INFO: Analysis report uploaded in 486ms
INFO: ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=sonarqube
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AYOHgUcL5ahrBawBcDyB
INFO: Analysis total time: 7.809 s
INFO: -----
INFO: EXECUTION SUCCESS
INFO: -----
INFO: Total time: 8.832s
INFO: Final Memory: 17M/57M
INFO: -----
Finished: SUCCESS
```

13. Once the build is complete, check the project in SonarQube.



In this way, we have integrated Jenkins with SonarQube for SAST.

## Conclusion

In this experiment, we have understood the importance of SAST and have successfully integrated Jenkins with SonarQube for Static Analysis and Code Testing.



humans. These tools can scan millions of lines of code in a matter of minutes. SAST tools automatically identify critical vulnerabilities - such as buffer overflows, SQL injection, cross site scripting and others with high confidence. Thus integrating static analysis into the SDLC can yield dramatic results in the overall quality of code developed.

### Conclusion -

Thus, we successfully understood importance of SAST and integrated with Jenkins with SonarQube for SAST.