

Adv DevOps Practical 7

Aim:

To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Integrating Jenkins with SonarQube:

Prerequisites:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

Steps

to integrate Jenkins with SonarQube

Prerequisites: Make sure you have docker and jenkins installed.
Run **docker -v** to check the docker installation.

Run

1. Open up Jenkins Dashboard on localhost, port 8090 or whichever port it is at for you.

The screenshot shows the Jenkins Dashboard. On the left, there's a sidebar with 'New Item', 'Build History', 'Manage Jenkins', and 'My Views'. The main area displays a table of build history. The table has columns for 'S' (Success), 'W' (Warning), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. The builds listed are 'sahil 7', 'Sahil exp6', 'SahilExp6', and 'sahiljob'. Below the table, there's a 'Build Queue' section showing 'No builds in the queue.' and a 'Build Executor Status' section showing 'Built-in Node' with 1 idle and 2 offline executors.

S	W	Name	Last Success	Last Failure	Last Duration
✓	☀	sahil 7	24 days #2	N/A	96 ms
✓	☀	Sahil exp6	24 days #3	N/A	1 sec
⏸	☀	SahilExp6	N/A	N/A	N/A
✗	☁	sahiljob	N/A	24 days #1	1.5 sec

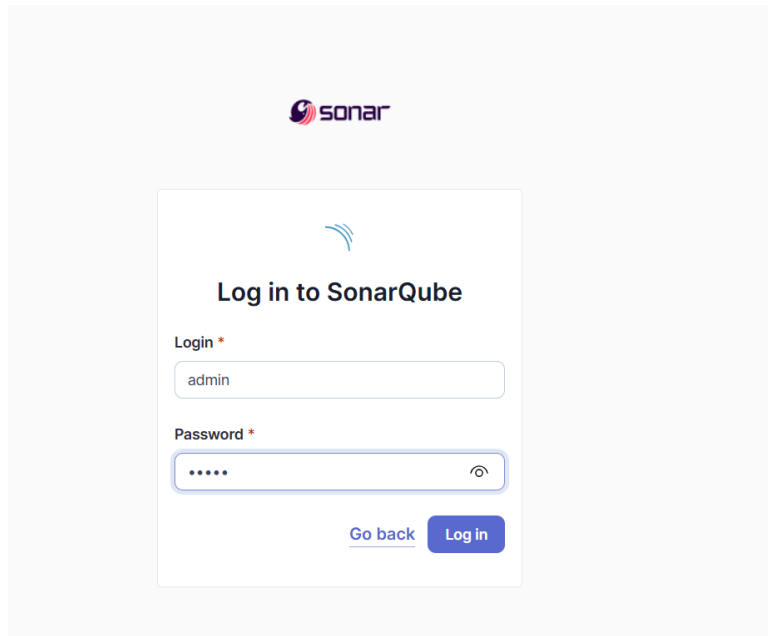
2. Run SonarQube in a Docker container using this command -

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

-----Warning: run below command only once

```
C:\Users\Lenovo>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest 47f6db8dbf2ed99dbe304bc0ebdf47b9d4144c4e4add42055ba44ce231058272
```

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




The image shows the SonarQube login interface. At the top, there is a Sonar logo. Below it, a white box contains the text "Log in to SonarQube" with a blue icon above it. Underneath, there are two input fields: "Login *" with the text "admin" and "Password *" with masked characters ".....". To the right of the password field is an eye icon. At the bottom of the box, there is a "Go back" link and a blue "Log in" button.

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

(do change the password as you cannot use the default one)

Update your password

 This account should not use the default password.

Enter a new password

All fields marked with * are required

Old Password *

New Password *

Confirm Password *

Update


sonarqube

ProjectsIssuesRulesQuality ProfilesQuality GatesAdministrationMore


How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)?
Create your project from your favorite DevOps platform.


First, you need to set up a DevOps platform configuration.

 Import from Azure DevOps


Setup

 Import from Bitbucket Cloud


Setup

 Import from Bitbucket Server

Setup

 Import from GitHub

Setup

 Import from GitLab

Setup

Are you just testing or have an advanced use-case? Create a local project.

Create a local project

5. Create a manual project in SonarQube with the name sonarqube
(Click on create local project)

1 of 2

Create a local project

Project display name *



Project key *



Main branch name *

The name of your project's default branch [Learn More](#) 

Cancel

Next

Setup the project

[Projects](#) [Issues](#) [Rules](#) [Quality Profiles](#) [Quality Gates](#) [Administration](#) [More](#)

Choose the baseline for new code for this project

☒ **Use the global setting**

Previous version
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ **Define a specific setting for this project**

☐ **Previous version**
Any code that has changed since the previous version is considered new code.
Recommended for projects following regular versions or releases.

☐ **Number of days**
Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become pa
Recommended for projects following continuous delivery.

☐ **Reference branch**
Choose a branch as the baseline for the new code.
Recommended for projects using feature branches.

[Back](#) [Create project](#)

And come back to Jenkins Dashboard.

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

Jenkins Shubham Jha

Dashboard > Manage Jenkins > Plugins

Plugins [Install](#)

Updates 29

Available plugins

Installed plugins

Advanced settings

Install	Name	Released
<input checked="" type="checkbox"/>	SonarQube Scanner 2.17.2 External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube , the open source platform for Continuous Inspection of code quality.	7 mo 8 days ago
<input type="checkbox"/>	Sonar Gerrit 388.v9b_f1cb_e42306 External Site/Tool Integrations This plugin allows to submit issues from SonarQube to Gerrit as comments directly.	3 mo 22 days ago
<input type="checkbox"/>	SonarQube Generic Coverage 1.0 TODO	5 yr 1 mo ago

6. Under Jenkins 'Manage Jenkins' then go to 'system', scroll and look for **SonarQube Servers** and click on **add SonarQube** and then enter the details.

Enter the Server Authentication token if needed.(I didn't do it)

In SonarQube installations: Under **Name** add <project name of sonarqube> for me its sonarqube_exp7

In **Server URL** Default is <http://localhost:9000>

SonarQube servers

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

☐ **Environment variables**

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.

+ Add ▼

Advanced ▼

Add SonarQube

ed

7. Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.

Dashboard > Manage Jenkins > Tools

[Dashboard](#) > [Manage Jenkins](#) > [Tools](#)

Gradle installations

[Add Gradle](#)

SonarScanner for MSBuild installations

[Add SonarScanner for MSBuild](#)

SonarQube Scanner installations

[Add SonarQube Scanner](#)

Ant installations

[Add Ant](#)

Maven installations

[Maven installations](#) ▾[✎](#) Edited[Save](#)[Apply](#)

Click on **Add SonarQube Scanner** .

Check the “Install automatically” option. → Under name write any name as identifier →
Check the “Install automatically” option.

SonarScanner for MSBuild installations

Add SonarScanner for MSBuild

SonarQube Scanner installations

Add SonarQube Scanner

☰

SonarQube Scanner

Name

sonarqube_scanner_exp7

☒ **Install automatically** ?

☰

Install from Maven Central

Version

SonarQube Scanner 6.2.0.4584

Add Installer ▾

Add SonarQube Scanner

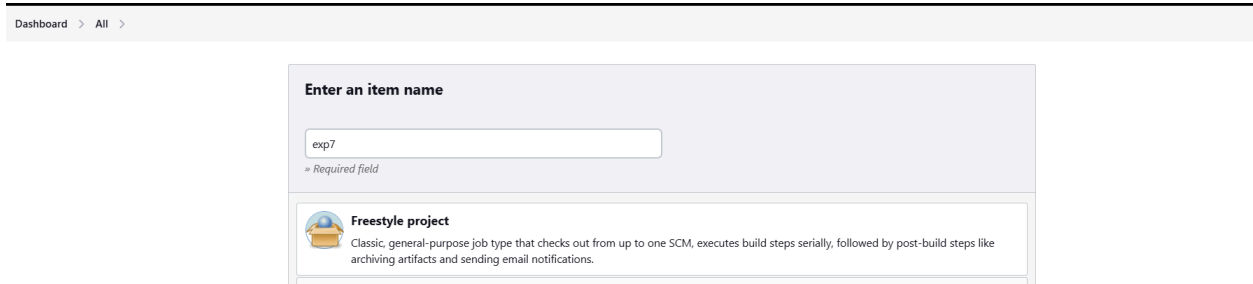
✓ Saved

8. After the configuration, create a New Item in Jenkins, choose a freestyle project.

Dashboard > All >

Enter an item name

= Required field

 **Freestyle project**

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

9. Choose this GitHub repository in Source Code Management.

https://github.com/shazforiot/MSBuild_firstproject

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

Dashboard > exp7 > Configuration

Configure

- General
- Source Code Management**
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Source Code Management

☐ None

☒ **Git** ?

Repositories ?

Repository URL ?

Credentials ?

- none -

+ Add

Advanced

Add Repository

Branches to build ?

Branch Specifier (blank for 'any')

Save Apply

10. Under **Select project** → **Configuration** → **Build steps** → **Execute SonarQube Scanner**, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.

Dashboard > exp7 > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment**
- Build Steps
- Post-build Actions

Filter

- Execute SonarQube Scanner
- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit
- SonarScanner for MSBuild - Begin Analysis
- SonarScanner for MSBuild - End Analysis

Add build step ^

Post-build Actions

Add post-build action v

Save Apply

You will see something like this:

Dashboard > exp7 > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps**
- Post-build Actions

Build Steps

Execute SonarQube Scanner

JDK ?
JDK to be used for this SonarQube analysis
(Inherit From Job) v

Path to project properties ?
v

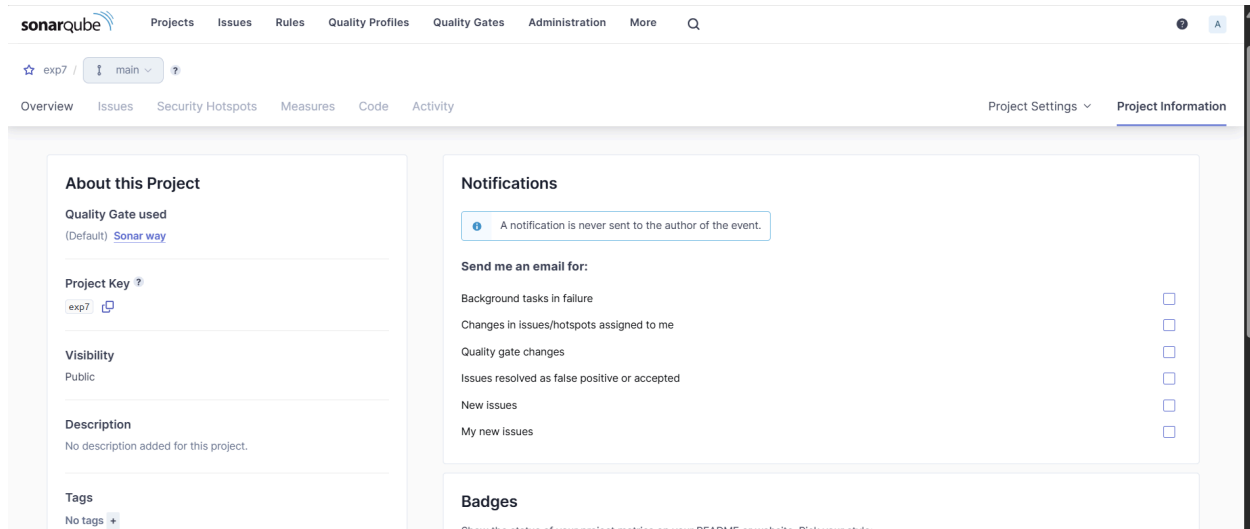
Analysis properties ?
v

Additional arguments ?
v

JVM Options ?
v

Add build step v

Open sonarQube again and go to Project Information appearing in the right side. Click on it and you can copy the project key from About the Project Section.



Use this key in place of <projectKey> in the following code

sonar.projectKey =<projectKey>

sonar.login =admin

sonar.password =<yourpassword for sonar qube>

sonar.host.url =http://localhost:9000

sonar.sources =.

I personally preferred not keeping any spaces after '=' .

The screenshot shows the 'Analysis properties' configuration form in SonarQube. It has a dropdown menu set to '(Inherit From Job)'. Below it is a text input field for 'Path to project properties'. The 'Analysis properties' section is highlighted with a blue border and contains the following text: 'sonar.projectKey =exp7', 'sonar.login =admin', 'sonar.password =2923', 'sonar.host.url =http://localhost:9000', and 'sonar.sources =.'. Below this is an 'Additional arguments' section with a text input field and a dropdown arrow.

Apply and save.

11. Go to sonarQube and go to administration → Security (dropdown) → Global Permissions.

See the administrator below and check the boxes which i checked.

sonarqube

ProjectsIssuesRulesQuality ProfilesQuality GatesAdministrationMoreQ

Administration

ConfigurationSecurityProjectsSystemMarketplace

Global Permissions

Grant and revoke permissions to make changes at the global level. These permissions include editing Quality Profiles, executing analysis, and performing global system administration.

AllUsersGroups

Search for users or groups...

	Administer System ?	Administer ?	Execute Analysis ?	Create ?
<div>sonar-administrators</div> <div>System administrators</div>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input checked="" type="checkbox"/> Projects
<div>sonar-users</div> <div>Every authenticated user automatically belongs to this group</div>	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects
<div>Anyone DEPRECATED</div> <div>Anybody who browses the application belongs to this group. If authentication is not enforced, assigned permissions also apply to non-authenticated users.</div>	<input type="checkbox"/>	<input type="checkbox"/> Quality Gates <input type="checkbox"/> Quality Profiles	<input type="checkbox"/>	<input type="checkbox"/> Projects
<div>A Administrator admin</div>	<input checked="" 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. Go to jenkins and click build:

Dashboard > exp7 >

exp7

Status

Changes

Workspace

Build Now

Configure

Delete Project

SonarQube

Rename

SonarQube

Permalinks

Build History **trend** v

Filter...

#1

Sep 25, 2024, 11:37 AM

Atom feed for all Atom feed for failures

Dashboard > exp7 > #5 > Console Output

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#5'

Timings

Git Build Data

Previous Build

Console Output

Started by user Shubham Jha

Running as SYSTEM

Building on the built-in node in workspace C:\ProgramData\Jenkins\jenkins\workspace\exp7

The recommended git tool is: NONE

No credentials specified

> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\jenkins\workspace\exp7\.git # timeout=10

Fetching changes from the remote Git repository

> git.exe config remote.origin.url https://github.com/shazforiot/MSBuild_FirstProject # timeout=10

Fetching upstream changes from https://github.com/shazforiot/MSBuild_FirstProject

> git.exe --version # timeout=10

> git --version # 'git version 2.45.0.windows.1'

> git.exe fetch --tags --force --progress -- https://github.com/shazforiot/MSBuild_FirstProject +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

[exp7] \$ C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube_scanner_exp7\bin\sonar-scanner.bat -Dsonar.host.url=http://localhost:9000 -Dsonar.projectKey=exp7 -Dsonar.login=admin -Dsonar.host.url=http://localhost:9000 -Dsonar.sources=. -Dsonar.password=2923 -Dsonar.projectBaseDir=C:\ProgramData\Jenkins\jenkins\workspace\exp7

13:47:35.366 WARN Property 'sonar.host.url' with value 'http://localhost:9000' is overridden with value 'http://localhost:9000'

13:47:35.362 INFO Scanner configuration file: C:\ProgramData\Jenkins\jenkins\tools\hudson.plugins.sonar.SonarRunnerInstallation\sonarqube_scanner_exp7\bin\..\conf\sonar-scanner.properties

13:47:35.382 INFO Project root configuration file: NONE

13:47:35.480 INFO SonarScanner CLI 6.2.0.4584

13:47:35.483 INFO Java 21.0.4 Oracle Corporation (64-bit)

13:47:35.413 INFO Windows 11 10.0 amd64

13:47:35.429 INFO User cache: C:\Windows\system32\config\systemprofile\.sonar\cache

13:47:37.015 INFO JRE provisioning: os[jindows], arch[amd64]

13:47:47.097 INFO Communicating with SonarQube Server 10.6.0.92116

13:47:47.665 INFO Starting SonarScanner Engine...

Conclusion:

In this project, I successfully integrated Jenkins with SonarQube to establish a robust automated static application security testing (SAST) pipeline. The setup involved deploying SonarQube using Docker, ensuring smooth container orchestration and efficient resource management. A key component was configuring Jenkins with the appropriate SonarQube plugins, authentication mechanisms, and linking it to a GitHub repository for continuous integration.

One of the challenges I faced was configuring Docker on the Jenkins environment, which required resolving networking issues between the Docker containers and ensuring that the SonarQube server was reachable from Jenkins. Additionally, setting up secure authentication between Jenkins and SonarQube involved troubleshooting token-based authentication and resolving environment path issues, particularly with the **JAVA_HOME** setup for the SonarQube scanner.

After overcoming these obstacles, I integrated the SonarQube scanner as a build step, allowing for continuous code analysis. This setup provided automated detection of code vulnerabilities, code smells, and quality issues. It helped ensure that any new commits triggered immediate analysis, generating detailed reports and promoting continuous improvement in code quality.