Integration SonarQube with Jenkins

Step 1: Install SonarQube Scanner Plugin

- Head over to your Jenkins Server Web portal, click on "***Manage Jenkins***" > "***Manage Plugins***" > Click on the "***Available tab***" then search for SonarQube. And Select "***SonarQube Scanner***" once it shows up in the list of plugins. Click on "***Install without restart***"

Step 2: Generate Token in SonarQube Server

- In this step we are going to generate a token that we will use in Jenkins server to connect to it.
- Click on your ***Admin account Icon*** which will bring up a drop down menu. Click on "***My Account***".
- That will open a new page. On the new page, click on the "***Security***" tab.
- A new "***Tokens***" Under "***Generate Tokens***", put a name you like then
- A new token will be generated. Copy it since we are going to use it in Jenkins next.

Step 3: Configure SonarQube in Jenkins:

- into Jenkins go to "***Manage Jenkins***" > "***Configure System***".
- click ***Add SonarQube***, and add the values you will be prompted to provide.
- SonarQube server URL, that is where your SonarQube server is running at then the "Server authentication token".

http://192.168.1.3:9001

- Open up the Jenkins Credentials Provider
- choose “Secret Text” and choose “Secret Text” and give it a Name/ID that will match and then hit “Add“.
- choose the token we have just added. Once that is done, simply hit “***Apply***” then “***Save***“.

Jenkins

- squ_2077a7e9cbf7524524a713696e19787ef2c747d2
- squ_46a98b9d40038050199857de88cd6eeceebf3876
- squ_3aef020e30f6b636447ce330b479af53e3c72423

jenkins12

- squ_e798c50ee42eb24e478af95e51491fd9e75bcc5e

Step 4: Configure SonarQube Scanner:

- SonarQube scanner :

Is an open-source platform that is used to measure and analyze code quality

is a tool that is used to analyze source code and produce reports that highlight potential code quality issues.

- access the Jenkins Docker container from a bash shell like this

docker-compose exec -it jenkins bash

- in the directory /var/jenkins_home execute commands

wget https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-3.3.0.1492-linux.zip

- unzip the Sonar Scanner binary:

unzip sonar-scanner-cli-3.3.0.1492-linux.zip

- Go to “*Manage Jenkins*” then click on “*Global configuration Tool*“.
- Click on “Add SonarQube Scanner” tab.
- you will need to uncheck “Install automatically” so you can explicitly set SONAR_RUNNER_HOME

Step 5: Configuring Jenkins and SonarQube

- get IP address of host by executing from the host

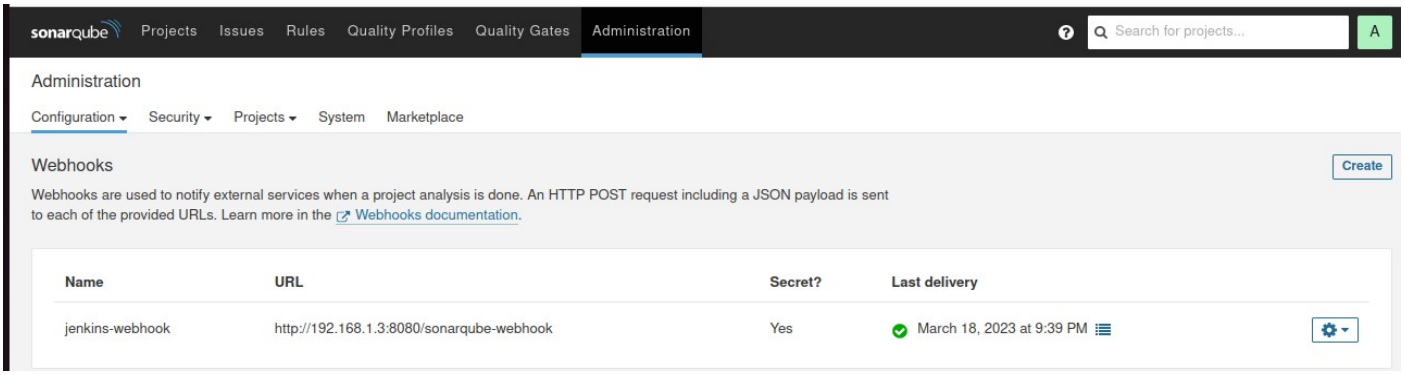
ip add | grep 'inet '

```
root@ubuntu-VivoBook-ASUSLaptop-X515EP-X515EP:~#  ip add | grep 'inet '
    inet 127.0.0.1/8 scope host lo
    inet 192.168.1.3/24 brd 192.168.1.255 scope global dynamic noprefixroute wlo1
    inet 172.20.0.1/16 brd 172.20.255.255 scope global br-d6cd03a8ebfb
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
    inet 172.25.0.1/16 brd 172.25.255.255 scope global br-06de04be53a9
root@ubuntu-VivoBook-ASUSLaptop-X515EP-X515EP:~#
```

This returned 192.168.1.3 which will be used for both Jenkins and SonarQube configuration.

- Note that if you change networks, you will have to update the IP address on Jenkins and SonarQube to be the new host IP. For a more flexible IP assignment that won't need to be updated if the host IP changes, you can leverage networking docker compose
- add webhook in SonarQube to point to Jenkins

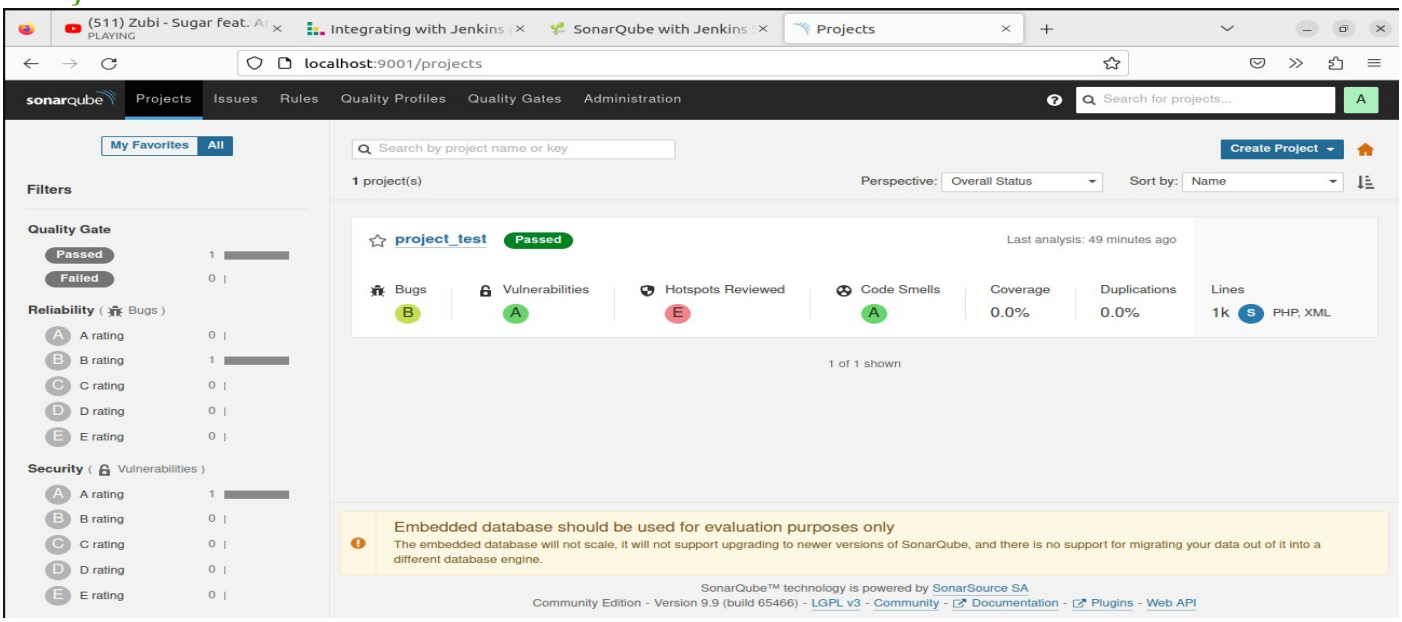
http://192.168.1.3:8080/sonarqube-webhook


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects...
A
Administration
Configuration
Security
Projects
System
Marketplace
Webhooks
Create
Webhooks are used to notify external services when a project analysis is done. An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the Webhooks documentation.
Name
URL
Secret?
Last delivery
jenkins-webhook
http://192.168.1.3:8080/sonarqube-webhook
Yes
March 18, 2023 at 9:39 PM

```
pipeline {
  agent any

  stages {
    stage('Build') {
      steps {
        //sh "composer update"
        sh "composer install"
        sh 'cp .env.example .env'
        sh 'php artisan key:generate'
      }
    }
    stage('test') {
      steps {
        sh "php artisan test "
        sh './vendor/bin/phpunit'
      }
    }
    stage('Code Quality Check via SonarQube') {
      steps {
        script {
          def scannerHome = tool 'sonarqube-scanner';
          withSonarQubeEnv("sonarqube") {
            sh "${tool("sonarqube-scanner")}/bin/sonar-scanner \
            -Dsonar.projectName=project_test \
            -Dsonar.projectKey=project_test \
            -Dsonar.sources=. \
            -Dsonar.host.url=http://172.20.0.1:9001/  \
            -Dsonar.login=squ_04fc4c5b253eb7b2713c7a6b88e44c7c2fad9b23"
          }
        }
      }
    }
    stage("Quality Gate") {
      steps {
        timeout(time: 1, unit: 'HOURS') {
          waitForQualityGate abortPipeline: true
        }
      }
    }
  }
```

}




```
version: "3"
services:
  sonarqube:
    image: sonarqube
    container_name: sonarqube
    restart: always
    environment:
      - SONARQUBE_JDBC_USERNAME=wissem
      - SONARQUBE_JDBC_PASSWORD=Admin123
      - SONARQUBE_JDBC_URL=jdbc:postgresql://postgres:54322/postgres
    ports:
      - "9001:9000"
    links:
      - postgres
    volumes:
      - sonarqube_conf:/opt/sonarqube/conf
      - sonarqube_data:/opt/sonarqube/data
      - sonarqube_extensions:/opt/sonarqube/extensions
      - sonarqube_bundled-plugins:/opt/sonarqube/lib/bundled-plugins
    networks:
      - sonarqube-postgres
  postgres:
    image: postgres
    container_name: postgres_sql
```

```
    restart: always
    environment:
      - POSTGRES_USER=wissem
      - POSTGRES_PASSWORD=Admin123
      - POSTGRES_DB=postgres
    ports:
      - "54322:5432"
    volumes:
      - sonarqube_postgres:/var/lib/postgresql
      - postgres_data:/var/lib/postgresql/data
    networks:
      - sonarqube-postgres
volumes:
  postgres_data:
  sonarqube_bundled-plugins:
  sonarqube_conf:
  sonarqube_data:
  sonarqube_postgres:
  sonarqube_extensions:
networks:
  sonarqube-postgres:
    name: sonarqube-postgres
```