**SONARQUBE**

* It is a open source platform to maintain source code quality. It is written in java. But, can analyze 20 different programming languages.
* Default port = **9000**.
* It analyzes architecture & design, unit tests, potential bugs, coding rules etc.
* You can access sonarqube from web browser with your server ip and sonarqube port.
* You should always use sonar as normal user. Root user can’t sonar.
* One of the main issues with sonar is. Your server should have atleast **4** gm ram. If you have less than 4 gb ram, sonar stops automatically.

**INSTALL SONAR**

* Before we install sonarqube, we need to install java and mysql db.
* After install mysql, Login into mysql with user root. If you have successfully logged into db it means that you have configured mysql correctly.
* Sonarqube uses mysql db to store metrics.
* Now, Download **sonarqube zip** file from the official site and unzip the file.
* If you want to analyze python project with sonarqube, you need to download **sonar-scanner** also and set path for it.

**PATH=/path/to/sonar-scanner/bin:”$PATH”**

* Login into your database and create a database and user for sonarqube as below.

**CREATE DATABASE sonar CHARACTER SET utf8 COLLATE utf8\_general\_ci;**

**CREATE USER 'sonar' IDENTIFIED BY 'Sonar@1234';**

**GRANT ALL ON sonar.\* TO 'sonar'@'%' IDENTIFIED BY 'Sonar@1234';**

**GRANT ALL ON sonar.\* TO 'sonar'@'localhost' IDENTIFIED BY 'Sonar@1234';**

**FLUSH PRIVILEGES;**

**@ to seperate, % from remort login , localhost server login, flush refresh the updates wt we made changes for data base**

* Go to **/sonarqube/conf/sonar.properties**,
* In the database section,

**Uncomment the jdbc.username and password lines.**

**Add your db username and password.**

**Uncomment jdbc.db.url line, for which db you are using and save the file.**

* Now, all the sonar metrics will be saved in the database that you selected.
* By default, sonar can’t start as root, you have to start as normal user. You have to add that user to sonar.sh file.
* Go to **/sonarQube/bin/linux64bit/.** Edit the **sonar.sh** and add this.

**RUN\_AS\_USER=username.**

* Change the ownership and group of sonar dir to the user, from which you want to start sonar.

**chown –R user:user sonarQube**

* Start the service to test whether sonar is working (or) not…

**/sonarqube/bin/linux-64bit/sonar.sh start –** To start sonar.

**/sonarqube/bin/linux-64bit/sonar.sh stop –** To stop sonar.

**/sonarqube/bin/linux-64bit/sonar.sh restart –** To restart sonar.

**/sonarqube/bin/linux-64bit/sonar.sh console –** To see the output.

**./sonar.sh start is used to start service**

* Based on the architecture of your server, the start and stop scripts path will change.
* If you have 64bit system, scripts will be stored in linux-64bit dir.
* If you have 32bit system, scripts will be stored in linux-32bit dir.
* Go to browser, type your ip and sonar port(9000), you can see sonar home page.
* If you want to use your ip (or) hostname instead of localhost in the sonarqube url,
* Go to sonar.properties file,

**In web server section.**

**Sonar.web.host = type your server ip (or) hostname.**

* Restart the sonarqube and now you can access sonar with your ip and sonar port in browser.
* By default, username and password to login to sonar dashboard is **admin** for both.
* You can change the password once you logged into sonar dashboard.

**SONAR-MAVEN INTEGRATION**

* To use sonarqube with maven, you have to install a plugin by making an entry in maven **settings.xml** file.
* Go to settings.xml,
* Under **pluginsGroups** section. Down the second ‘🡪’ mark, paste this.

**<pluginGroup>org.sonarsource.scanner.maven</pluginGroup>**

* In **profiles** section .down the ‘**🡪’** mark and above the activate profiles section, Remove **</profiles>** line and paste this

**<profile>**

**<id>sonar</id>**

**<activation>**

**<activeByDefault>true</activeByDefault>**

**</activation>**

**<properties>**

**<!-- Optional URL to server. Default value is http://localhost:9000 -->**

**<sonar.host.url>**

**http://IP:9000**

**</sonar.host.url>**

**</properties>**

**</profile>**

**</profiles>**

* It will download the plugin from given website above in plugingroup section and it will detect the sonar from the given url.
* After copying this code in settings.xml file, run any maven cmds to see whether we did correct (or) not.
* To test sonar in linux, go to your project **pom.xml** dir and run **mvn sonar:sonar**….It will test the code and give you the errors in the code in sonar GUI dashboard along with your project name.
* Login to your sonar GUI dashboard with your username and password, there you can see your project name. Inside the project name you can see all your code along with errors if there are any.

**SONAR-JENKINS INTEGRATION**

* You can integrate sonarqube with Jenkins for continuous testing. It will test the code quality right after the build job is finished. It will show errors in the code in sonar GUI.
* First, you have to download **sonar-scanner plugin** (sonarqube in old versions).
* Afetr installing the plugin. Go to configure system,

**In sonarqube servers section.**

**Click add sonarqube.**

**Type name.**

**Sonar server url.**

**Sonar version.**

**Sonar account username and password** if it asks (by default, **admin**).

**Click, Save.**

* Go to Global tool configuration,

**In sonar installations section.**

**Click add sonarqubescanner.**

**Type a name.**

**Give Sonarqube home path (if you installed sonar scanner in your server).**

**If not, click install automatically and select the scanner version to install.**

**Click, save.**

* We have configured everything with sonar and Jenkins.
* After configuring sonar in Jenkins, while creating a project.
* In the post build step, select **execute sonarqube scanner.**
* In the analysis properties section, we have to mention analysis parameters, based on these parameters, sonar will scan our project.

**sonar.host.url=**http://yourip:9000/

**sonar.projectName=**myproject (your project name to display in sonar UI)

**sonar.projectKey=**myproject (can be anything, must be unique)

**sonar.projectVersion=**1.0 (can be anything)

**sonar.sources=**path/to/your/code (your code path)

**sonar.language=**java (in my case)

**sonar.login=**username (to login to sonar UI)

**sonar.password=**password (to login to sonar UI)

**sonar.projectBaseDir=**Use this if you need analysis should takes place in different dir rather than default dir(default=/.jenkins/workspace/project). Give write permissions to this directory. It is where sonar.working.directory will be created.

* Click, save and build the project.
* Now, once the build finished successfully, it starts sonar analsys and push the code errors and bugs to sonar dashboard.

**QUALITY PROFILES**

* Quality profile service is central for sonarqube. This is where you define your requirements by defining set of **rules** which need to analyze on top of the particular language.
* Every language has default sonar profile (sonar way), which is used while analyzing.
* You can create your own profile and add your custom rules to that profile to analyze the code.
* You can create as many profiles as you want in sonarqube.
* To create a profile.

**Click on quality profiles tab.**

**Click create.**

**Give a name to profile.**

**Select language to assign the role.**

* Once after the profile creation is finished, you will get a new profile with zero rules.
* You can activate the rules inside the new profile that you have created.
* You can associate projects with this profile. After associating, while analyzing the project, this profile will be executed aganaist the project.
* To associate a profile to project.

**Go to profile to associate.**

**Click manage projects.**

**Click ALL**

**Select projects to associate with this profile.**

**Click, close.**

* You can take backup of a profile by clicking on the backup. It will download an xml file with all the rules.
* You can restore the profile by uploading the xml file to sonar.
* You can copy a profile and create a new profile with the same rules as copied one.
* You can activate and deactivate the rules in profiles as per our requirement.
* There is third-party plugin called find bugs-plugin, we can use this plugin to analyze the project based on this plugin rules.
* We have to download find bugs-plugin from sonar market place and restart the server.
* Go to quality profiles and check for findbugs to know whether find bugs is successfully installed (or) not.
* Once findbugs downloaded, you can set the plugin as default analyzer for the projects. And for every time we analyze the code, it will take the rules form findbugs-plugin.
* You have to select the projects to use this findbugs-plugin for analyzing.

**QUALITY GATES**

* Quality gates are simple Boolean thresholds set on project measures. A project must pass each and every thresholds in order to pass the quality gate as a whole. It defines the quality of your project.
* By default, only one quality gate will be there and used for all the projects, you can create a new one and use it.
* You can create new quality gates and add conditions to that gates to check the projects. you have to assign this quality gate to project, which will use while analyzing the project. If the conditions are passed as we specified, then sonar passes quality gate for the project.
* If the conditions are not met with quality gate, it fails the quality gate for the project and it means there is something wrong in your project code.

**To create a new gate.**

**Click on quality gate tab.**

**Click create.**

**Give a name to gate.**

**Set conditions to gate.**

**Select project to associate with gate.**

**yum install java\* -y**

to inatall java in instance

**java -version**

to see the version wt was the present used by root

**sudo alternatives --config java**

**to** see the versions of java what where present in current instance

**cd /root**

export which is used to show the previous exported paths in a list

sudo alternatives --config java

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64

export JRE\_JRE=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/jre

to set jre home path for primary

**export**

to see whether both jre and java paths exported or not

**echo $JAVA\_HOME**

to see the exported path for perticular home

export PATH=$PATH:/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/bin:/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/jre/bin

in this both java home and jre home

**cd /etc/profile.d/**

**vi java.sh**

we have to create a file with .sh extenton then we have to fallow order

if we place in this directory system will directly read updated java version wt we plased in JAVA\_HOME JAVA\_JRE & PATH

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64

export JRE\_JRE=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/jre

export PATH=$PATH:/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/bin:/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/jre/bin

**chmod +x java.sh**

to add extention for that file

**source java.sh**

which is used to update the file what u placed in profile.d

**my sql instalation**

https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-centos-7

**mysql -u root -p:**

which is used to login to data base from using linux instance

which can be used to connect

**in project**

**vi sonar-project.properties**

sonar.host.url=http://35.154.162.168:9000/

sonar.projectName=flask

sonar.projectKey=1

sonar.projectVersion=1.0

sonar.sources=.

sonar.language=py

sonar.login=admin

sonar.password=admin

to run program go to project and past **sonar-scanner**

**DBS: in database to create separate user and assighn separate particular db to certain user**

CREATE USER 'sumanth' IDENTIFIED BY 'Sumanth@11';

GRANT ALL ON sonar.\* TO 'sumanth'@'%' IDENTIFIED BY 'Sumanth@11';

GRANT ALL ON sonar.\* TO 'sumanth'@'localhost' IDENTIFIED BY 'Sumanth@11';

FLUSH PRIVILEGES;

**Path to keep in java.sh in profil.d**

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/

export JRE\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/jre

export PATH=$PATH:/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/bin:/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.222.b10-0.amzn2.0.1.x86\_64/jre/bin