~~~~~~~~~~~~~~~Sonar Qube Notes~~~~~~~~~~~~~~~~~

Installation -

Sonar Qube Server -

- URL - https://www.sonarqube.org/downloads/
- Download version 7.7 from history…upload is taking time
- After Unzipping Go to this Path - \sonarqube-7.7\bin\windows-x86-64 & then execute startSonar batch file.
- By default it will run on port 9000.

Jenkins-

URL - https://get.jenkins.io/war-stable/

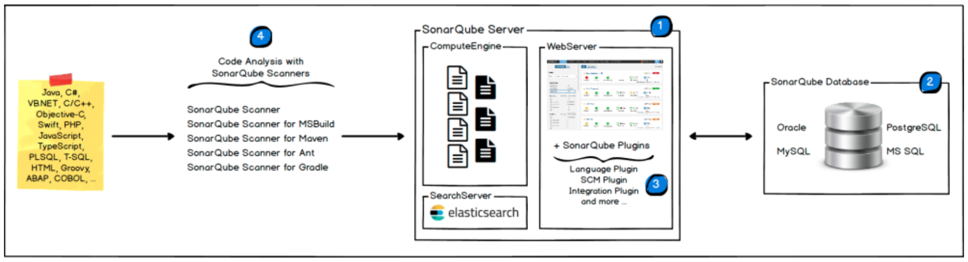
Installation Ref – Windows - https://www.jenkins.io/doc/book/installing/windows/

SonarQube® is an automatic code review tool to detect bugs, vulnerabilities and code smells in your code. It can integrate with your existing workflow to enable continuous code inspection across your project branches and pull requests.

Architecture and Integration

Overview

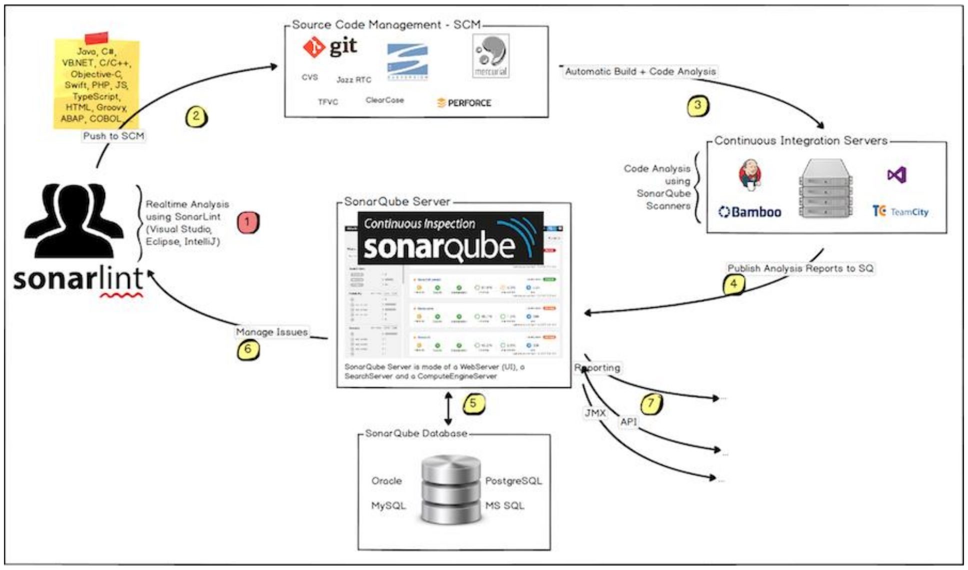
The SonarQube Platform is made of 4 components:


Java, C#, VB.NET, C/C++, Objective-C, Swift, PHP, JavaScript, TypeScript, PLSQL, T-SQL, HTML, Groovy, ABAP, COBOL, ...
4
Code Analysis with SonarQube Scanners
SonarQube Scanner
SonarQube Scanner for MSBuild
SonarQube Scanner for Maven
SonarQube Scanner for Ant
SonarQube Scanner for Gradle
SonarQube Server
1
ComputeEngine
WebServer
+ SonarQube Plugins
Language Plugin
SCM Plugin
Integration Plugin
and more ...
3
SearchServer
elasticsearch
SonarQube Database
2
Oracle
MySQL
PostgreSQL
MS SQL

1. One SonarQube Server starting 3 main processes:
   - Web Server for developers, managers to browse quality snapshots and configure the SonarQube instance
   - Search Server based on Elasticsearch to back searches from the UI
   - Compute Engine Server in charge of processing code analysis reports and saving them in the SonarQube Database
2. One SonarQube Database to store:
   - the configuration of the SonarQube instance (security, plugins settings, etc.)
   - the quality snapshots of projects, views, etc.
3. Multiple SonarQube Plugins installed on the server, possibly including language, SCM, integration, authentication, and governance plugins
4. One or more SonarScanners running on your Build / Continuous Integration Servers to analyze projects

Integration

The following schema shows how SonarQube integrates with other ALM tools and where the various components of SonarQube are used.


Java, C#, VB.NET, C/C++, Objective-C, Swift, PHP, JS, TypeScript, HTML, Groovy, ABAP, COBOL
Push to SCM
2
Source Code Management - SCM
git
CVS
Jazz RTC
TFVC
ClearCase
mercurial
PERFORCE
Automatic Build + Code Analysis
3
Continuous Integration Servers
Code Analysis using SonarQube Scanners
Bamboo
TeamCity
Realtime Analysis using SonarLint (Visual Studio, Eclipse, IntelliJ)
1
sonarlint
SonarQube Server
Continuous Inspection
sonarqube
SonarQube Server is made of a WebServer (UI), a SearchServer and a ComputeEngineServer
Publish Analysis Reports to SQ
4
Manage Issues
6
5
Reporting
JMX
API
7
SonarQube Database
Oracle
MySQL
PostgreSQL
MS SQL

5. Developers code in their IDEs and use SonarLint to run local analysis.
6. Developers push their code into their favourite SCM : git, SVN, TFVC, ...
7. The Continuous Integration Server triggers an automatic build, and the execution of the SonarScanner required to run the SonarQube analysis.
8. The analysis report is sent to the SonarQube Server for processing.
9. SonarQube Server processes and stores the analysis report results in the SonarQube Database, and displays the results in the UI.
10. Developers review, comment, challenge their Issues to manage and reduce their Technical Debt through the SonarQube UI.
11. Managers receive Reports from the analysis. Ops use APIs to automate configuration and extract data from SonarQube. Ops use JMX to monitor SonarQube Server.

Installing from a zip file

Download the SonarQube Community Edition

Unzip it, let's say in *C:\sonarqube* or */opt/sonarqube*

Start the SonarQube Server:

```
# On Windows, execute:C:\sonarqube\bin\windows-x86-xx\StartSonar.bat

# On other operating systems, as a non-root user execute:/opt/sonarqube/bin/[OS]/sonar.sh
console
```

Log in to http://localhost:9000 with System Administrator credentials (admin/admin) and follow the embedded tutorial to analyze your first project.

For Linux

- The user used to start the service is `sonarqube`
- The group used to start the service is `sonarqube`
- The Java Virtual Machine is installed in `/opt/java/`
- SonarQube has been unzipped into `/opt/sonarqube/`
- Because the sonar-application jar name ends with the version of SonarQube, you will need to adjust the `ExecStart` command accordingly on install and at each upgrade.
- The SonarQube data directory, `/opt/sonarqube/data`, and the extensions directory, `/opt/sonarqube/extensions` should be owned by the `sonarqube` user. As a good practice, the rest should be owned by `root`

Once your `sonarqube.service` file is created and properly configured, run

```
sudo systemctl enable sonarqube.servicesudo systemctl start sonarqube.service
```

SonarAnalysis

Once the SonarQube platform has been installed, you're ready to install an analyzer and begin creating projects. To do that, you must install and configure the scanner that is most appropriate for your needs. Do you build with:

- Gradle - SonarScanner for Gradle
- MSBuild - SonarScanner for MSBuild
- Maven - use the SonarScanner for Maven
- Jenkins - SonarScanner for Jenkins
- Azure DevOps - SonarQube Extension for Azure DevOps
- Ant - SonarScanner for Ant
- anything else (CLI) - SonarScanner

Note that we do not recommend running an antivirus scanner on the machine where a SonarQube analysis runs, it could result in unpredictable behavior.

A project is created in the platform automatically on its first analysis. However, if you need to set some configuration on your project before its first analysis, you have the option of provisioning it via Administration options.

What does analysis produce?

SonarQube can perform analysis on 20+ different languages. The outcome of this analysis will be quality measures and issues (instances where coding rules were broken). However, what gets analyzed will vary depending on the language:

- On all languages, "blame" data will automatically be imported from supported SCM providers. Git and SVN are supported automatically. Other providers require additional plugins.
- On all languages, a static analysis of source code is performed (Java files, COBOL programs, etc.)
- A static analysis of compiled code can be performed for certain languages (.class files in Java, .dll files in C#, etc.)
- A dynamic analysis of code can be performed on certain languages.

Mandatory Parameters

Server

Key	Description	Default
`sonar.host.url`	the server URL	http://localhost:9000

Project Configuration

Key	Description	Default
`sonar.projectKey`	The project's unique key. Allowed characters are: letters, numbers, `-`, `_`, `.` and `:`, with at least one non-digit.	For Maven projects, this defaults to `<groupId>:<artifactId>`

Optional Parameters

Project Identity

Key	Description	Default
`sonar.projectName`	Name of the project that will be displayed on the web interface.	`<name>` for Maven projects, otherwise project key. If not provided and there is already a name in the DB, it won't be overwritten
`sonar.projectVersion`	The project version.	`<version>` for Maven projects, otherwise "not provided"

Authentication

If the "Anyone" pseudo-group does not have permission to perform analyses, you'll need to supply the credentials of a user with Execute Analysis permission for the analysis to run under.

Key	Description	Default
`sonar.login`	The login or authentication token of a SonarQube user with Execute Analysis permission on the project.	
`sonar.password`	The password that goes with the `sonar.login` username. This should be left blank if an authentication token is being used.	

`sonar.sources`	Comma-separated paths to directories containing main source files.	Read from build system for Maven, Gradle, MSBuild projects. Defaults to project base directory when neither `sonar.sources` nor `sonar.tests` is provided.
`sonar.projectBaseDir`	Use this property when you need analysis to take place in a directory other than the one from which it was launched. E.G. analysis begins from `jenkins/jobs/myjob/workspace` but the files to be analyzed are in `ftpdrop/cobol/project1`. The path may be relative or absolute. Specify not the the source directory, but some parent of the source directory. The value specified here becomes the new "analysis directory", and other paths are then specified as though the analysis were starting from the specified value of `sonar.projectBaseDir`. Note that the analysis process will need write permissions in this directory; it is where the `sonar.working.directory` will be created.	
`sonar.working.directory`	Set the working directory for an analysis triggered with the SonarScanner or the SonarScanner for Ant (versions greater than 2.0). This property is not compatible with the SonarScanner for MSBuild. Path must be relative, and unique for each project. Beware: the specified folder is deleted before each analysis.	`.sonar`
`sonar.scm.provider`	This property can be used to explicitly tell SonarQube which SCM plugin should be used to grab SCM data on the project	

	(in case auto-detection does not work). The value of this property is always lowercase and depends on the plugin (ex. "tfvc" for the TFVC plugin). Check the documentation page of each plugin for more.	
`sonar.scm.forceReloadAll`	By default, blame information is only retrieved for changed files. Set this property to `true` to load blame information for all files. This can be useful is you feel that some SCM data is outdated but SonarQube does not get the latest information from the SCM engine.	

Scanning a PHP Project

Add sonar-project.properties file

Add properties

- sonar.projectKey=drupal-7.67
    - sonar.projectName=drupal-7.67
    - sonar.projectVersion=1.0
    - #sonar.modules=phpmodule, cssmodule, jsmodule
- phpmodule.sonar.sources=<<path to your php project codebase folder>>
- phpmodule.sonar.language=php
    - sonar.sourceEncoding=UTF-8
- phpmodule.sonar.projectBaseDir=<<project base dir>>

Run below command from cmd – Project Absolute path

sonar-scanner.bat -D"sonar.projectKey=php_project" -D"sonar.sources=." -D"sonar.host.url=http://localhost:9000" -D"sonar.login=8cfebfcfd0999ce43ec5f77a0b5c73064aca0356"

Scanning a Java Maven Project

Just sonarscan required

```
mvn sonar:sonar \

  -Dsonar.projectKey=Sonar_SVN \

  -Dsonar.host.url=http://10.42.206.97:9000 \

  -Dsonar.login=672f0281ad23aea13e5f2ea287bdd1f7a242d373
```

Scanning an Angular project

#installation

mkdir -m 777 nodejs

tar -C nodejs -xvf node-v0.12.14-linux-x64.tar.gz

https://www.digitalocean.com/community/tutorials/how-to-install-node-js-on-a-centos-7-server

/opt/node-v0.12.14

export NODE_HOME=/opt/node-v0.12.14/

export PATH=$NODE_HOME:$PATH

#Mannual run steps

1. Take Checkout
2. Open in Visual tudio Code
3. Open terminal
4. in Same Directory of Project
5. Run command npm install

sonar-project.properties

- sonar.host.url=http://localhost:9000/
- sonar.login=admin
- sonar.password=admin
- sonar.projectKey=ZenCOO_Mobile_Source_Code
- sonar.projectName=ZenCOO_Mobile_Source_Code
- sonar.projectVersion=1.0
- sonar.sourceEncoding=UTF-8
- sonar.sources=.
- sonar.exclusions=**/node_modules/**
- sonar.tests=src
- sonar.test.inclusions=**/*.spec.ts

#Commands which may be used

- npm install
- ng serve
- ionice serve
- npm i
- npm run

in cmd -

chrome.exe --user-data-dir="C:/Chrome dev session" --disable-web-security

https://www.npmjs.com/package/jsonpath

#Commands for Local machine if needed (For Ref.)

npm install jsonpath

$ ionic cordova plugin add cordova-plugin-advanced-http

$ npm install --save @ionic-native/http@4

Note – *For servers we need to configure everything on Jenkins and node installation and set up and copy the node modules folder from local machine to server in workspace over jenkins*

scp node_modules sqadmin@10.42.206.97:/var/lib/jenkins/workspace/Zencfo/zencfo-mobile-source-code/

SCM Integration

Global - Admin login -> Administration -> General Settings -> SCM -> set Credentials

Project Specific – Select Project -> Administration -> General Settings -> SCM -> set Credentials

Create User

Global - Admin login -> Administration ->security -> users -> create user

Enter username , name , password , email id etc.

Customize Quality Gate

Admin login -> Administration -> Quality Gate -> Create /copy Quality gate

Permissions

Project specific - > Administration -> permissions ->

You can make project as Public /private

You can provide required permissions to specifc users

Jenkins -

Installation Links

https://www.oracle.com/in/java/technologies/javase/javase-jdk8-downloads.html ------> jdk-8u251-linux-i586.rpm OR jdk-8u251-linux-i586.tar.gz

https://maven.apache.org/docs/history.html --------> apache-maven-3.6.3-bin.tar.gz ------>/usr/local/apache-maven-3.6.0

https://pkg.jenkins.io/redhat-stable/ --------> jenkins-2.235.1-1.1.noarch.rpm ------------------->/usr/lib/jenkins

Node Js(Version - 12.14.0)- https://nodejs.org/en/download/

Installation Folder

/home/sqadmin/jenkins-2.222.1-1.1.noarch.rpm

Required URLs Access

1) https://repo.maven.apache.org/
2) https://repo.spring.io
3) https://updates.jenkins.io/
4) https://jenkins.io/
5) http://updates.jenkins-ci.org/
6) ftp-nyc.osuosl.org
7) ftp-chi.osuosl.org
8) http://mirror.serverion.com/
9) http://maven.openimaj.org/
10) https://registry.npmjs.org/

Installed SonarQube Scanner

Manage Jenkins -> SonarQube Scanner ->

# User specific environment and startup programs

Setting PATH

```
PATH=$PATH:$HOME/bin

export PATH

export M2_HOME=/opt/apache-maven-3.6.3

export PATH=$M2_HOME/bin:$PATH

export SONAR_SCANNER_HOME=/opt/sonar-scanner-4.2.0.1873-linux

export PATH=$SONAR_SCANNER_HOME/bin:$PATH

export JAVA_HOME=/usr/java/jdk1.8.0_181-amd64/

export PATH=$JAVA_HOME:$PATH
```

Jenkins Configuration-

- SonarqubeScanner installation
- Configure System - > SonarQube Server Details
- Configuration Tools -> Maven from local system
- In Projects -
- Folder - Zenlearn
- Freestyle Project - SVN Checkout
- Configure SCM & in Post Build Projects - Sonar Analysis project
- Create Freestyle Project - Sonar Analysis
- Configure - Build Environment -Select prepare SonarQube Scanner Environment5 & Secret Token
- and in Build - Maven version - Maven & Goals-sonar:sonar & POM Location

Build Pipeline Configuration

a. Build Pipeline Plugin Installed
b. Click on + icon in ZenLearn Folder
c. From here we can create Pipeline view
d. Initial state - ZenLearn >> Zenlearn checkout

Features

1. check-qualitygate
2. Report is generated directly on SonarQubeDashboard using CNES Report
3. Email Configuration
4. SCM configuration
5. Annonymous login disabled
6. Make Project private
7. Create Users
8. Create groups
9. Create Permission Templates
10. Rules Definitions
11. Permissions Categories
12. Quality gate
13. Quality Profile
14. Severity
15. Resolutions
16. and important Filters
17. Custom rules
18. Deactivate Rules

Added OJDBC JAR for Project to build contains Oracle DB-

```
/var/lib/jenkins/.m2/repository/com/oracle

/home/sqadmin/.m2/repository/com/oracle

ojdbc7

cp -r ojdbc7 /var/lib/jenkins/.m2/repository/com/oracle

mvn install:install-file -Dfile=/ojdbc7.jar

-DgroupId=com.oracle

-DartifactId=ojdbc7

-Dversion=12.1.0

-Dpackaging=jar

-DgeneratePom=true
```

Details

#SonarQube Server Details -

- **IP** : 10.42.206.97
- **Hostname** : ze42-v-sonqub01
- **SonarQube Path** : /opt/sonarqube-7.7

#Installations

- Maven Installed & Configured
- NodeJs Installed
- JDK istalled
- Sonar-Scanner Installed
- SonarQube Server installed

#Some Implementations Done are -

- Users Group created namely ZenAssist consisting Shumali and Ranu as 2 users.
- Made All Projects as Private So visible to admin only
- Security improved by enabling "Force User Authentication"
- User Specific and Project specific Permissions Configuration are done to check different scenerios.
- Email Notification Configuration
- SCM Configuration

# Projects Integrated

- **Zenassist -**
- **Zencfo**
- **Zencio**
- **Zencoo**
- **ZenHelp**
- **Zenpulse**

- **Zentalent**
- **Zents**
- **Zenverse**
- **UserInfo-UserProjectInfo-Scheduler**

Directory - /opt/

- apache-maven-3.6.3
- node
- sonarqube-7.7
- sonar-scanner-4.2.0.1873-linux

Setting Path over Server -

- **printenv** – used to view all Env Variables
- **cat ~/.bash_profile** – to open bash_profile file in read mode
- **vi** – for editing the bash file
- Press “i” for insert mode
- Paste the properties for ex.-

```
export M2_HOME=/opt/apache-maven-3.6.3

export PATH=$M2_HOME/bin:$PATH
```

- After pasting / writing the statements, Press **ESC:wq**

Bash_profile -

```
# .bash_profile

# Get the aliases and functions

if [ -f ~/.bashrc ]; then

. ~/.bashrc

fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH

export M2_HOME=/opt/apache-maven-3.6.3

export PATH=$M2_HOME/bin:$PATH
export SONAR_SCANNER_HOME=/opt/sonar-scanner-4.2.0.1873-linux
export PATH=$SONAR_SCANNER_HOME/bin:$PATH
export JAVA_HOME=/usr/java/jdk1.8.0_251-amd64/
export PATH=$JAVA_HOME:$PATH
export NODE_HOME=/opt/node/node-v12.16.3-linux-x64
export PATH=$NODE_HOME/bin:$PATH
```

Errors & Warnings -

[WARNING] SCM provider autodetection failed.

Please use "sonar.scm.provider" to define SCM of your project, or disable the SCM Sensor in the project settings.

ERROR - sonar.java.binaries

```
[ERROR] Failed to execute goal org.sonarsource.scanner.maven:sonar-maven-plugin:3.8.0.2131:sonar (default-cli)
on project zencoo-businessexcellence-utilization-service: Please provide compiled classes
of your project with sonar.java.binaries property -> [Help 1]
```

Resolution - Add sonar.java.binaries property

ERROR - Blame info

```
ERROR: Error during SonarQube Scanner execution
java.lang.IllegalStateException: Error when executing blame for file src/main/java/com/zents/notification/util/PDQueryConstant.java
at org.sonar.plugins.scm.svn.SvnBlameCommand.blame(SvnBlameCommand.java:85)
at org.sonar.plugins.scm.svn.SvnBlameCommand.blame(SvnBlameCommand.java:58)
at org.sonar.scanner.scm.ScmPublisher.publish(ScmPublisher.java:81)
at org.sonar.scanner.scan.ProjectScanContainer.doAfterStart(ProjectScanContainer.java:322)
Caused by: org.tmatesoft.svn.core.SVNAuthenticationException: svn: E170001: Authentication required for '<http://10.42.4.71:80> Subversion Repository'
at
org.tmatesoft.svn.core.internal.wc.SVNErrorManager.authenticationFailed(SVNErrorManager.java:53)
```

Resolution – Add Valid SonarQube Credentials over server SCM settings

ERROR – OJDBC

```
[ERROR] Failed to execute goal on project onboarding-zenhelp: Could not resolve dependencies for

project com.zensar.zenhelp:onboarding-zenhelp:jar:0.0.1-SNAPSHOT: Could not find artifact
com.oracle:ojdbc7:jar:12.1.0

in central (https://repo.maven.apache.org/maven2) -> [Help 1]
```

Resolution

Added OJDBC JAR for Project to build contains Oracle DB-

/var/lib/jenkins/.m2/repository/com/oracle/ojdbc7/12.1.0

Command - cp -r ojdbc7 /var/lib/jenkins/.m2/repository/com/oracle

ERROR: Cannot find module 'typescript'

```
NFO: Sensor SonarTS [typescript]

INFO: Analyzing 204 typescript file(s) with the following configuration file
/var/lib/jenkins/workspace/Zenverse/zenverse-mobile-source-code/tsconfig.json

ERROR: internal/modules/cjs/loader.js:960

ERROR:   throw err;

ERROR:   ^

ERROR:

ERROR: Error: Cannot find module 'typescript'
```

cat tsconfig - typeroots - nodeModules required

Resolution

copy node Modules Folder to respective workspace folder eg -in
/var/lib/jenkins/workspace/Zenverse/zenverse-mobile-source-code (Respective Project Path)

Command used -

scp node_modules sqadmin@10.42.206.97:/var/lib/jenkins/workspace/Zencfo/zencfo-mobile-source-code/

#Properties Used for Analysis -

Java Maven Project-

```
sonar.projectBaseDir=/var/lib/jenkins/workspace/Zents/attendance-scheduler/checkout

sonar.working.directory=/var/lib/jenkins/workspace/Zents/attendance-scheduler/sonar-analysis/.scannerwork

sonar.projectKey=zents-attendance-scheduler

sonar.projectName=zents-attendance-scheduler

sonar.projectVersion=1.0

sonar.sources=src/main/java/

sonar.language=java

sonar.java.binaries=target/classes
```

Mobile- Source-Code -

```
sonar.projectKey=ZenVerse_Mobile_Source_Code

sonar.projectName=ZenVerse_Mobile_Source_Code

sonar.sourceEncoding=UTF-8

sonar.sources=src

sonar.exclusions=**/node_modules/**

sonar.tests=src

sonar.test.inclusions=**/*.spec.ts
```

#JENKINS DEMO

AppName Folder -> Microservice name Folder - >

4 Freestyle Folders for Java Maven Project

- checkout
- build-project
- sonar-analysis
- check-qualitygate

Pipeline - checkout -> build-project -> sonar-analysis->check-qualitygate

**~~~~~~~~~~~~~~~~~~~~~~~Thank You ~~~~~~~~~~~~~~~~~~~~~~~~**

~Ranu Jain