Integrating SonarQube with AWS CodePipeline using AWS CodeBuild

- **About SonarQube:**
- **Licensing:**
- **Prerequisites:**
- **High level architecture:**
- **Solution:**
  - **Connect your repository with SonarQube**
  - **Configure SecretManager**
  - **Configuring AWS CodeBuild**
  - **Set up CodePipeline to verify the SonarQube integration**
- **SonarQube Report:**
- **SonarQube Dashboards:**

About SonarQube:

- Self-managed, automatic code review tool that systematically helps deliver *clean code*.
- Integrates into existing workflow and detects issues in code by performing continuous code inspections of projects.
- Scans the code against coding standards, identifies bugs and even checks for code coverage.
- The tool analyses 30+ different programming languages and integrates into CI pipeline (like aws codebuild, jenkins, buildkite etc)
- Clean as You Code approach helps to maintain high standards and focus on code quality.
- Provides feedback through its UI, email, and in decorations on pull or merge requests (in commercial editions) to notify if there are issues to address.
- Code can only be promoted when it is clean and passes the quality gate.

Licensing:

- Entire suite of Sonar products: SonarQube, SonarCloud, and SonarLint
- SonarQube Community Edition is free.
- All other SonarQube editions are commercial and require a paid license.

- SonarCloud is entirely free for all open source projects. You only pay if you want to analyze private repositories.

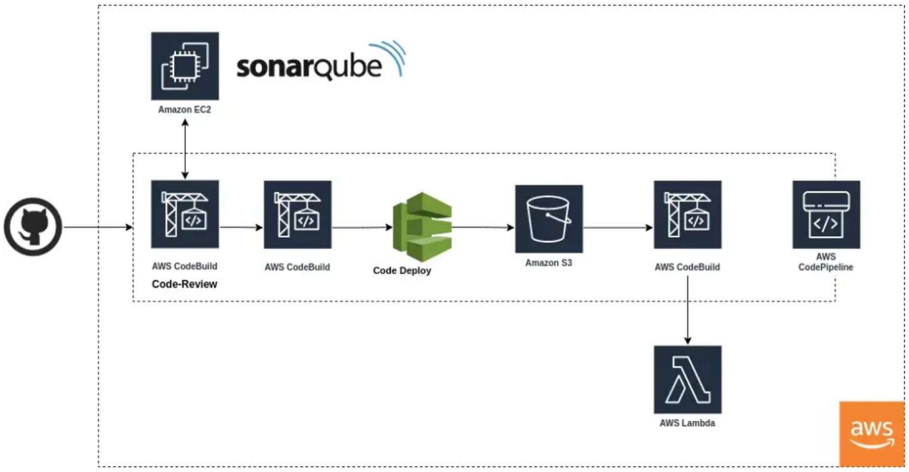
Prerequisites:

- Account credential to login to SonarQube
- AWS Account
- GitHub project to integrate

High level architecture:

Three stage CodePipeline setup to demonstrate the integration with SonarQube

1. **Source stage**, we will use a sample project (gateway) stored in GitHub Enterprise.
2. **Review stage**, we will use AWS CodeBuild project to integrate with SonarQube and perform code quality check.
3. **Build stage**, we will use another AWS CodeBuild project and push the built artifact to S3 bucket.



Solution:

Connect your repository with SonarQube

1. Sign in to GitHub through the SonarQube site using your GitHub credentials
2. Choose **Create a new project** in the SonarQube portal
3. Choose **Add Project Manually**
4. Provide **Project key*** *and* ***Display name****
5. Run Setup and this will Enable a Project Token.

6. Generate a token, to go **User > My Account > Security**. Enter a new Token name and Click **Generate**. Store it for the succeeding steps.
7. Now you are ready to execute the Sonar Scanner locally or via AWS Code Build (Refer below for sample buildspec yaml)

Configure SecretManager

1. Store a new secret via Secrets Manager console - use AWS Secret Manager to store the sonar login credentials. By using Secrets Manager we can provide controlled access to the credentials from CodeBuild.

Configuring AWS CodeBuild

1. Create a *buildspec.yml* file
2. Create a Code build project via CodeBuild console

1. **Sample Buildspec YAML**

```
2. # SonarQube_URL           =  Same across all repo
3. # SonarQube_Access_Token  =  Same across all repo
4. # SONAR_PROJECTKEY        =  Different for each repo and read from
   Environment Variable on CodeBuild Project
5.
6. version: 0.2
7.
8. env:
9.   secrets-manager:
10.      SonarQube_URL: $ENV/sonarqubexxxx
11.      SonarQube_Access_Token: $ENV/sonarqube-xxxxxx
12.
13.
14.  phases:
15.    install:
16.      commands:
17.        - echo "Entering the install phase"
18.        - echo $CODEBUILD_SRC_DIR
19.      runtime-versions:
20.        php: 8.1
21.        java: corretto17
22.      finally:
23.        - java -version
24.
25.
26.    pre_build:
27.      commands:
28.        - echo "Downloading sonar-scanner"
29.        - wget https://binaries.sonarsource.com/Distribution/sonar-
   scanner-cli/sonar-scanner-cli-4.4.0.2170-linux.zip
30.        - unzip sonar-scanner-cli-4.4.0.2170-linux.zip
31.        - mv sonar-scanner-4.4.0.2170-linux sonar-scanner
32.        - chmod -R 775 sonar-scanner
33.        - echo "stage pre_build completed"
```

```
34.        - wget https://sonarcloud.io/static/cpp/build-wrapper-linux-
   x86.zip
35.        - unzip build-wrapper-linux-x86.zip
36.        - chmod -R 775 build-wrapper-linux-x86
37.        - echo "stage pre_build completed"
38.
39.    build:
40.      commands:
41.      - echo "Running SonarQube analysis using sonar-scanner"
42.      - ./sonar-scanner/bin/sonar-scanner
43.       -Dsonar.host.url=$SonarQube_URL
44.       -Dsonar.login=$SonarQube_Access_Token
45.       -Dsonar.projectKey=$SONAR_PROJECTKEY
46.       -Dsonar.c.file.suffixes=-
47.       -Dsonar.cpp.file.suffixes=-
48.       -Dsonar.objc.file.suffixes=-
49.       -Dsonar.sourceEncoding=UTF-8
50.       -Dsonar.java.binaries=.
51.       -Dsonar.qualitygate.wait=true
52.
53.  post_build:
54.    commands:
55.      - echo "Uploading coverage report to SonarQube"
56.      - curl -X POST -H 'Content-Type: application/json' -d @sonarqube-
   generic-coverage.json https://sonarcloud.io/api/coverage/import
57.      - echo "SonarQube analysis completed successfully"
58.
59.  artifacts:
60.    type: zip
     files: '**/*'
```

Note: Quality Gate is a feature in SonarQube that can be configured to ensure coding standards are met and regulated across projects. You can set threshold measures on your projects like code coverage, technical debt measure, number of blocker/critical issues, security rating/unit test pass rate, and more. The last step calls the Quality Gate API to check if the code is satisfying all the conditions set in Quality Gate.

Quality Gate can return four possible responses:

- **ERROR**: The project fails the Quality Gate.
- **WARN**: The project has some irregularities but is ok to be passed on to production.
- **OK**: The project successfully passes the Quality Gate.
- **None**: The Quality Gate is not attached to project.

AWS CodeBuild provides several environment variables that you can use in your build commands. **CODEBUILD_BUILD_SUCCEEDING** is a variable used to indicate whether the current build is succeeding. Setting the value to **0** indicates the build status as **failure** and **1** indicates the build as **success**.

Using the Quality Gate **ERROR** response, set the **CODEBUILD_BUILD_SUCCEEDING** variable to failure. Accordingly, the CodeBuild status can be used to provide response for the pipeline to proceed or to stop.

Set up CodePipeline to verify the SonarQube integration

Switch to your CodePipeline console to create a pipeline for your repository via CodePipeline console
You can integrate SonarQube in any stage in CodePipeline.

This is the output of the AWS Code pipeline..

Output of the AWS Code pipeline

```
[Container] 2023/03/28 13:07:01 Running command unzip sonar-scanner-cli-
4.4.0.2170-linux.zipArchive:  sonar-scanner-cli-4.4.0.2170-linux.zip
creating: sonar-scanner-4.4.0.2170-linux/   creating: sonar-scanner-
4.4.0.2170-linux/jre/   creating: sonar-scanner-4.4.0.2170-linux/jre/conf/
creating: sonar-scanner-4.4.0.2170-linux/jre/conf/security/   creating:
sonar-scanner-4.4.0.2170-linux/jre/conf/security/policy/   creating: sonar-
scanner-4.4.0.2170-linux/jre/conf/security/policy/limited/   creating: sonar-
scanner-4.4.0.2170-linux/jre/conf/security/policy/unlimited/   creating:
sonar-scanner-4.4.0.2170-linux/jre/conf/management/   creating: sonar-
scanner-4.4.0.2170-linux/jre/lib/   creating: sonar-scanner-4.4.0.2170-
linux/jre/lib/server/   creating: sonar-scanner-4.4.0.2170-
linux/jre/lib/security/   creating: sonar-scanner-4.4.0.2170-
linux/jre/lib/jli/   creating: sonar-scanner-4.4.0.2170-linux/jre/lib/jfr/
creating: sonar-scanner-4.4.0.2170-linux/jre/legal/   creating: sonar-
scanner-4.4.0.2170-linux/jre/legal/jdk.internal.le/   creating: sonar-
scanner-4.4.0.2170-linux/jre/legal/java.logging/   creating: sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.management.jfr/   creating: sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.naming.rmi/   creating: sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.unsupported/   creating: sonar-scanner-
4.4.0.2170-linux/jre/legal/java.datatransfer/   creating: sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.pack/   creating: sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.naming.dns/
........
linux/jre/legal/jdk.security.auth/LICENSE -> ../java.base/LICENSE  sonar-
scanner-4.4.0.2170-linux/jre/legal/java.rmi/ASSEMBLY_EXCEPTION ->
../java.base/ASSEMBLY_EXCEPTION  sonar-scanner-4.4.0.2170-
linux/jre/legal/java.rmi/ADDITIONAL_LICENSE_INFO ->
../java.base/ADDITIONAL_LICENSE_INFO  sonar-scanner-4.4.0.2170-
linux/jre/legal/java.rmi/LICENSE -> ../java.base/LICENSE  sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.internal.vm.compiler/ASSEMBLY_EXCEPTION ->
../java.base/ASSEMBLY_EXCEPTION  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.internal.vm.compiler/ADDITIONAL_LICENSE_INFO ->
../java.base/ADDITIONAL_LICENSE_INFO  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.internal.vm.compiler/LICENSE -> ../java.base/LICENSE
sonar-scanner-4.4.0.2170-linux/jre/legal/jdk.jsobject/ASSEMBLY_EXCEPTION ->
../java.base/ASSEMBLY_EXCEPTION  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.jsobject/ADDITIONAL_LICENSE_INFO ->
../java.base/ADDITIONAL_LICENSE_INFO  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.jsobject/LICENSE -> ../java.base/LICENSE  sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.crypto.ec/ASSEMBLY_EXCEPTION ->
../java.base/ASSEMBLY_EXCEPTION  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.crypto.ec/ADDITIONAL_LICENSE_INFO ->
../java.base/ADDITIONAL_LICENSE_INFO  sonar-scanner-4.4.0.2170-
```

```
linux/jre/legal/jdk.crypto.ec/LICENSE -> ../java.base/LICENSE  sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.aot/ASSEMBLY_EXCEPTION ->
../java.base/ASSEMBLY_EXCEPTION  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.aot/ADDITIONAL_LICENSE_INFO ->
../java.base/ADDITIONAL_LICENSE_INFO  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.aot/LICENSE -> ../java.base/LICENSE  sonar-scanner-
4.4.0.2170-linux/jre/legal/jdk.sctp/ASSEMBLY_EXCEPTION ->
../java.base/ASSEMBLY_EXCEPTION  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.sctp/ADDITIONAL_LICENSE_INFO ->
../java.base/ADDITIONAL_LICENSE_INFO  sonar-scanner-4.4.0.2170-
linux/jre/legal/jdk.sctp/LICENSE -> ../java.base/LICENSE  sonar-scanner-
4.4.0.2170-linux/jre/legal/java.scripting/ASSEMBLY_EXCEPTION ->
../java.base/ASSEMBLY_EXCEPTION  sonar-scanner-4.4.0.2170-
linux/jre/legal/java.scripting/ADDITIONAL_LICENSE_INFO ->
../java.base/ADDITIONAL_LICENSE_INFO  sonar-scanner-4.4.0.2170-
linux/jre/legal/java.scripting/LICENSE -> ../java.base/LICENSE[Container]
2023/03/28 13:07:03

Running command mv sonar-scanner-4.4.0.2170-linux sonar-scanner[Container]
2023/03/28 13:07:03
Running command chmod -R 775 sonar-scanner[Container] 2023/03/28 13:07:03
Running command echo "stage pre_build completed"stage pre_build
completed[Container] 2023/03/28 13:07:03
Running command wget https://sonarcloud.io/static/cpp/build-wrapper-linux-
x86.zip--2023-03-28 13:07:03--  https://sonarcloud.io/static/cpp/build-
wrapper-linux-x86.zipResolving sonarcloud.io (sonarcloud.io)...
54.93.168.237, 18.194.43.186, 18.156.88.26Connecting to sonarcloud.io
(sonarcloud.io)|54.93.168.237|:443... connected.HTTP request sent, awaiting
response... 200 Length: unspecified [application/zip]Saving to: ‘build-
wrapper-linux-x86.zip’     0K .......... .......... .......... ..........
..........  204K    50K .......... .......... .......... ..........
..........  204K   100K .......... .......... .......... ..........
..........  204K   150K .......... .......... .......... ..........
.......... 85.7M   200K .......... .......... .......... ..........
..........  106M   250K .......... .......... .......... ..........
..........  149M   300K .......... .......... .......... ..........
..........  204K   350K .......... .......... .......... ..........
..........  109M   400K .......... .......... .......... ..........
.......... 92.2M   450K .......... .......... .......... ..........
.......... 92.5M   500K .......... .......... .......... ..........
..........  109M   550K .......... ..
450M=1.0s2023-03-28 13:07:05 (572 KB/s) - ‘build-wrapper-linux-x86.zip’ saved
[576111][Container] 2023/03/28 13:07:05
Running command unzip build-wrapper-linux-x86.zipArchive:  build-wrapper-
linux-x86.zip   creating: build-wrapper-linux-x86/  inflating: build-wrapper-
linux-x86/libinterceptor-i686.so    inflating: build-wrapper-linux-x86/build-
wrapper-linux-x86-64    inflating: build-wrapper-linux-x86/libinterceptor-
haswell.so    inflating: build-wrapper-linux-x86/libinterceptor-x86_64.so
[Container] 2023/03/28 13:07:05
Running command chmod -R 775 build-wrapper-linux-x86[Container] 2023/03/28
13:07:05
Running command echo "stage pre_build completed"stage pre_build
completed[Container] 2023/03/28 13:07:05 Phase complete: PRE_BUILD State:
SUCCEEDED[Container] 2023/03/28 13:07:05 Phase context status code:  Message:
[Container] 2023/03/28 13:07:05 Entering phase BUILD[Container] 2023/03/28
13:07:05
```

```
Running command echo "Running SonarQube analysis using sonar-scanner"Running
SonarQube analysis using sonar-scanner[Container] 2023/03/28 13:07:05
Running command ./sonar-scanner/bin/sonar-scanner -
Dsonar.host.url=$SonarQube_URL -Dsonar.login=$SonarQube_Access_Token -
Dsonar.projectKey=$PROJECT_KEY -Dsonar.c.file.suffixes=- -
Dsonar.cpp.file.suffixes=- -Dsonar.objc.file.suffixes=- -
Dsonar.sourceEncoding=UTF-8 -Dsonar.java.binaries=.INFO: Scanner
configuration file:
/codebuild/output/src225426879/src/github.*******.com.au/EnterpriseSystems/pi
peline-as-a-service/sonar-scanner/conf/sonar-scanner.propertiesINFO: Project
root configuration file: NONEINFO: SonarScanner 4.4.0.2170INFO: Java 11.0.3
AdoptOpenJDK (64-bit)INFO: Linux 4.14.281-212.502.amzn2.x86_64 amd64INFO:
User cache: /root/.sonar/cacheINFO: Scanner configuration file:
/codebuild/output/src225426879/src/github.*******.com.au/EnterpriseSystems/pi
peline-as-a-service/sonar-scanner/conf/sonar-scanner.propertiesINFO: Project
root configuration file: NONEINFO: Analyzing on SonarQube server 8.4.2INFO:
Default locale: "en_US", source code encoding: "UTF-8"INFO: Load global
settingsINFO: Load global settings (done) | time=159msINFO: Server id:
B3A674CC-AWto78H04qDsnESI5CizINFO: User cache: /root/.sonar/cacheINFO:
Load/download pluginsINFO: Load plugins indexINFO: Load plugins index (done)
| time=46msINFO: Load/download plugins (done) | time=2835msINFO: Loaded core
extensions: developer-scannerINFO: Process project propertiesINFO: Process
project properties (done) | time=1msINFO: Execute project buildersINFO:
Execute project builders (done) | time=4msINFO: Project key: ***INFO: Base
dir:
/codebuild/output/src225426879/src/github.*******.com.au/EnterpriseSystems/pi
peline-as-a-serviceINFO: Working dir:
/codebuild/output/src225426879/src/github.*******.com.au/EnterpriseSystems/pi
peline-as-a-
.....
sensors on projectINFO: Sensor Zero Coverage SensorINFO: Sensor Zero Coverage
Sensor (done) | time=35msINFO: SCM Publisher SCM provider for this project
is: gitINFO: SCM Publisher 34 source files to be analyzedWARN: Shallow clone
detected, no blame information will be provided. You can convert to non-
shallow with 'git fetch --unshallow'.INFO: SCM Publisher 0/34 source files
have been analyzed (done) | time=3msWARN: Missing blame information for the
following files:WARN:   * workflow/helm/helminstallExternal.goWARN:   *
workflow/kube/kubeapply.goWARN:   * workflow/helm/helminstall.goWARN:   *
workflow/kube/kubewatchpodstatus.goWARN:   * workflow/kong/service.goWARN:
* workflow/kong/globalpluginWithTags.goWARN:   *
workflow/kube/kubedelete.goWARN:   * workflow/helm/helmlint.goWARN:   *
workflow/terraform/apply.goWARN:   * workflow/kong/workspaceconfig.goWARN:
* workflow/kube/kubefilterpods.goWARN:   * workflow/tasks/print.goWARN:   *
workflow/git/savevaluecontents.goWARN:   * workflow/git/fetchcontent.goWARN:
* workflow/tasks/delay.goWARN:   * workflow/helm/helmtemplate.goWARN:   *
workflow/definitions/types.goWARN:   *
workflow/workflow/sequential_workflow.goWARN:   *
workflow/kong/rbacuser.goWARN:   * workflow/main.goWARN:   *
workflow/terraform/validate.goWARN:   * workflow/utils/config.goWARN:   *
workflow/tasks/task.goWARN:   * workflow/kong/serviceplugin.goWARN:   *
workflow/terraform/install.goWARN:   * workflow/workflow/workflow.goWARN:   *
workflow/git/saveconfigcontents.go
WARN:   * workflow/selfserviceapi/validatelintcontent.go
WARN:   * workflow/kong/consumer-with-key.go
WARN:   * workflow/git/fetchchanges.go
WARN:   * workflow/kube/kubegetpodlogs.go
WARN:   * workflow/kong/route.go
```

```
WARN:   * workflow/controller/controller.go
WARN:   * workflow/terraform/plan.go
WARN: This may lead to missing/broken features in SonarQube
INFO: CPD Executor Calculating CPD for 34 files
INFO: CPD Executor CPD calculation finished (done) | time=72ms
INFO: Load New Code definition
INFO: Load New Code definition (done) | time=18ms
INFO: Analysis report generated in 141ms, dir size=356 KB
INFO: Analysis report compressed in 64ms, zip size=124 KB
INFO: Analysis report uploaded in 45ms
INFO: ANALYSIS SUCCESSFUL, you can browse ***/dashboard?id=***
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
***/api/ce/task?id=AYcoVIjh7gyYeIWzxcAN
INFO: Analysis total time: 10.326 s
INFO: ------------------------------------------------------------------------
-
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
-
INFO: Total time: 15.614s
INFO: Final Memory: 23M/90MINFO: ---------------------------------------------
---------------------------
[Container] 2023/03/28 13:07:21 Phase complete: BUILD State: SUCCEEDED
[Container] 2023/03/28 13:07:21 Phase context status code:  Message:
[Container] 2023/03/28 13:07:21 Entering phase POST_BUILD
[Container] 2023/03/28 13:07:21 Phase complete: POST_BUILD State: SUCCEEDED
[Container] 2023/03/28 13:07:21 Phase context status code:  Message:
[Container] 2023/03/28 13:07:22 Phase complete: UPLOAD_ARTIFACTS State:
SUCCEEDED
[Container] 2023/03/28 13:07:22 Phase context status code:  Message:
```

SonarQube Report: