REQUIREMENTS AND DESIGN

DOCUMENT

SonarQube-Fortify Plugin

October, 2016

FUNCTIONAL AND TECHNICAL REQUIREMENTS

TABLE OF CONTENTS

GENERAL INFORMATION

PURPOSE

The purpose of this document is to detail expected functionality that the proposed SonarQube-Fortify plugin solution should build. The document also details high-level technical design for the plugin.

SCOPE

This document will outline the functional and high-level technical requirements for the proposed plugin. The requirements are based on development effort that is previously agreed upon and additional detailed discussions with the Vanguard-ITSO team. The design sections, in turn, are based on requirements.

PROJECT REFERENCES

Key documents needed as supporting references to this document are listed below and must be reviewed while referring to the details outlined in this document:

- Plugin development options presentation
- Plugin development signed SOW

POINTS OF CONTACT

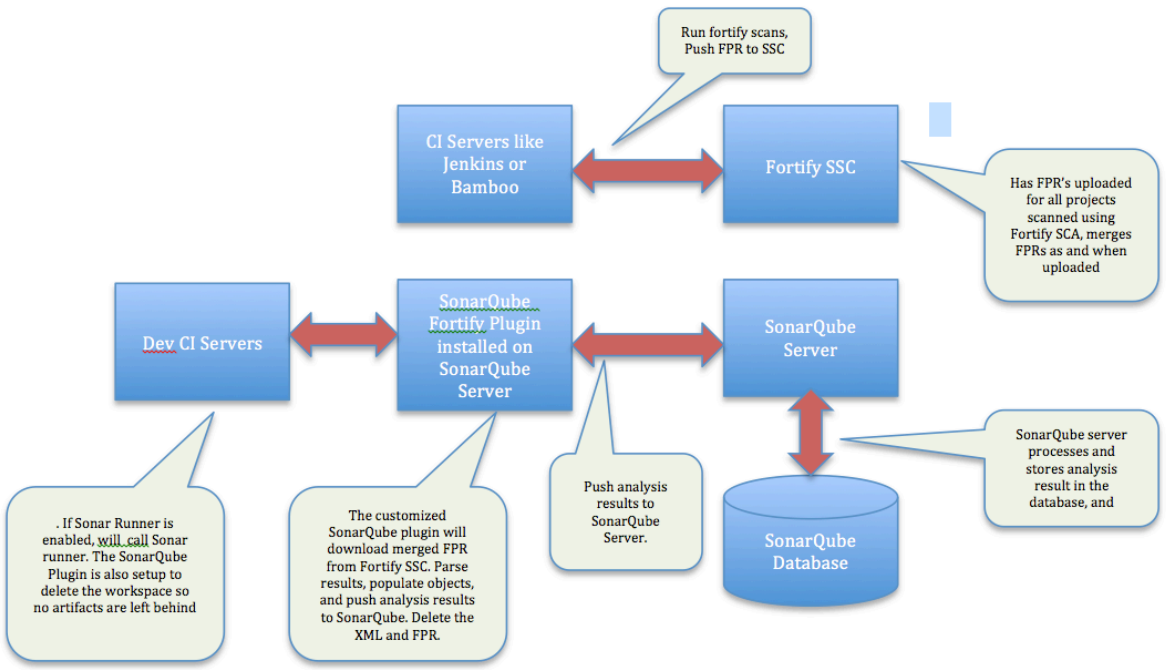
Below is a list of Point of Contacts relevant to this project:

Contact Name	Contact Type	Company	Email	Oversight Function
Susan Wojdula	ITSO Security analyst	Vanguard	Susan_wojdula@vanguard.com	ITSO Vendor manager Point of Contact
Aravind Venkataraman	Managing consultant	Cigital	avenkataraman@cigital.com	Estimation Financial Business/client management
Meera Subbarao	Senior Principal consultant	Cigital	msubbarao@cigital.com	Technical/solutions architect Estimation
Sneha Kokil	Consultant	Cigital	skokil@cigital.com	Technical Requirements, Estimates, Design
Brian Cefali	Associate Consultant	Cigital	bcefali@cigital.com	Development, Testing Support

SONARQUBE FORTIFY ARCHITECTURE & WORKFLOW

The SonarQube Fortify SCA architecture is made up of the following components:

- SonarQube Server
- SonarQube Database
- SonarQube Fortify Plugin
- SonarQube Scanner
- Fortify SSC
- Fortify SCA


Run fortify scans, Push FPR to SSC
CI Servers like Jenkins or Bamboo
Fortify SSC
Has FPR's uploaded for all projects scanned using Fortify SCA, merges FPRs as and when uploaded
Dev CI Servers
SonarQube Fortify Plugin installed on SonarQube Server
SonarQube Server
SonarQube server processes and stores analysis result in the database, and
Push analysis results to SonarQube Server.
SonarQube Database
. If Sonar Runner is enabled, will call Sonar runner. The SonarQube Plugin is also setup to delete the workspace so no artifacts are left behind
The customized SonarQube plugin will download merged FPR from Fortify SSC. Parse results, populate objects, and push analysis results to SonarQube. Delete the XML and FPR.

INTEGRATION:

1. The new SonarQube Fortify plugin is deployed on SonarQube Server
2. The Fortify SCA needs to be installed on CI servers to effectively use the SonarQube Fortify Plugin. This is needed because the SonarQube plugin will be executing on the DEV CI Server. Analysis is running on DEV CI Server – comes to CI Server to pull the data in the current plugin implementation.
3. Jobs are on-boarded to ITSO Jenkins/Bamboo to perform Fortify SCA scans (using the maven-sca-plugin)
4. When Fortify scans are run on CI Servers (Jenkins ITSO server or Bamboo servers), the FPR is uploaded to Fortify SSC automatically.
5. Fortify SSC merges the FPR for each scan when uploaded by CI servers (ITSO Server / Bamboo Server).
6. The DEV servers are configured to run SonarQube analysis either on every build or once per day
7. When SonarQube analysis runs on DEV servers (DEV Software package jobs), it performs the following tasks:
    a. Using FortifyClient which is part of Fortify SCA installation, download the merged FPR from SSC

b. Using ReportGenerator which again is part of Fortify SCA installation, parse the FPR to XML
c. Populate the objects to push the results to SonarQube Server
d. Delete the XML and FPR

SonarQube server displays the results on the dashboard and stores the results in its database

CURRENT AND PROPOSED SYSTEM SUMMARY

The following table shows key features and differences between the current and proposed plugin options.

Key Features	Current	Proposed
Report Readability	PDF reports presented on SonarQube dashboard are difficult to read and understand.	Issues will be added to SonarQube issue list, making them easily accessible and readable to developers.
Scan Results State	Scan results are not triaged or analyzed before presenting on dashboard, therefore, contain large number of false positives, affecting readability of the PDF even more.	Scan results fetched from SSC will be triaged for projects that participate in annual assessment cycle, therefore, eliminating false positives in the top two severity buckets.
Scan Result Source	Refers to Jenkins to download XML reports	Must have a capability to download FPR from SSC
Scan Merge Requirement	Does not have a wait time requirement for report download	May require a configurable wait-time for merged FPR to be available on SSC
SonarQube Rulesets	Does not populate rulesets in SonarQube	Must populates restricted set of rules, involves prefix to the rules to identify their source (i.e. fortify_<rulename>)
SonarQube-Fortify Issue Counts	Does not increment SonarQube issue counts (blocker, critical etc.)	Must find out a way to map Vanguard-specific categories to SonarQube severities and increment the count to show on dashboard
SonarQube-Fortify Issue Drill-down	Does not list Fortify issues on SonarQube	Must populate SonarQube "Issue" objects for the issues to appear in the list.
SonarQube-Fortify Rules and Descriptions	Does not require populating rules and related descriptions in SonarQube.	Must populate some form of issue description for drilled-down issues.

FUNCTIONAL REQUIREMENTS

SUMMARY OF FUNCTIONS

The proposed plugin design will carry out following key functions:

- Download merged scan results (FPR) file from Fortify SSC.
- Extract XML report by applying Vanguard-specific project template and filters.
- Parse the XML report into fields that are required to create an issue on SonarQube and associate it with rule keys with a suitable form of description in the issue details.
- Map Vanguard filters to SonarQube vulnerability categories (Blockers, Major, Info) and add Fortify issues counts to existing issues from other tools.

DETAILED FUNCTIONAL REQUIREMENTS

The sub-sections below elaborate the requirements, specified in the section above.

SCAN RESULTS (FPR) DOWNLOAD

During the annual assessment cycle, projects are regularly triaged on Fortify SSC, removing false positives in the process. Because Fortify SSC centrally holds such analyzed results and merges them with any new scan uploads, the plugin design mandates FPR download from Fortify SSC.

Please note that the results downloaded from Fortify SSC may not be triaged in some cases (such as, project that was not a part of triaging engagement performed before). Consequently, there is still a percentage of false positives for such projects. If triaged, results of the next scan will merge the triaged results, eliminating false positives.

With respect to implementing this functionality, the following requirements must be considered:

- Ensure that the information about assessment status of the project is correctly conveyed to the development teams. This is especially necessary because not all the team participate in annual assessment cycles to have triaged findings available on SSC. (*Dev teams: Please confirm assessment roll-out plan with ITSO*)
- Verify that the project is onboarded on ITSO Jenkins instance and latest scan files are uploaded on SSC.
- Credentials and SSC URL details must be configurable in SonarQube settings.
- Ensure that Vanguard project filter is applied to the downloaded FPRs by default.

XML REPORT EXTRACTION

In the current deployment, SonarQube analysis is not run by ITSO Jenkins instance. The team Jenkins instances (ciserver:xxxxx) invoke SonarQube analysis. Because the proposed implementation will talk only to Fortify SSC as a central repository for both Jenkins-based and Bamboo-based projects, it is imperative that FPR download and report extraction is performed at a central location as well.

With respect to implementing this functionality, the following requirements must be considered:

- Ensure that XML report can be extracted from downloaded FPR file.
- The XML report should serve as input to the plugin sensor that parses and calculates required SonarQube measurements.

SONARQUBE ISSUE DRILL-DOWN AND RULES

Similar to plugins for various other tools, Fortify issues need to be added to the SonarQube issue list that is easily accessible to the developers on the SonarQube dashboard itself. Additionally, Fortify rule keys and related descriptions must be available to be associated with each issue as well as to be configured/enabled in SonarQube rule repository.

Please note that drill-down of issues as comprehensive as Fortify audit workbench or Fortify SSC will not be feasible from performance point of view. Additionally, it will be redundant to replicate all workbench or SSC features in SonarQube.

With respect to implementing this functionality, the following requirements must be considered:

- Ensure that all the custom rules are added to rules configuration along with Fortify out-of-the-box rules.
- Each rule must be accompanied by description that can be helpful to the developer reviewing the issue. The description can be a summary of the issue or a static link to HP documentation on the issue type.
- Each issue must show the code snippet, highlighting the code statement, where security bug is detected.
- Each rule key should clearly identify source of the rule; in this case, Fortify.
- SonarQube should be able to enable/disable Fortify rules as well as configure them as needed in quality gates.

MAPPING TO SONARQUBE SEVERITY CATEGORIES

While Vanguard project teams work with custom confidence-severity buckets for Fortify issues, SonarQube issues have their own severity levels, denoted by – Blocker, Critical, Major, Minor and Info. Because the plugin is going to add to existing SonarQube vulnerability categories, it is necessary to provide mapping between Vanguard custom and SonarQube default categories.

With respect to implementing this functionality, the following requirements must be considered:

- Provide a custom mapping while populating issue counts and issue drill down as given below:

Vanguard Custom	SonarQube Default
High Confidence High	Blocker**
Investigate High	Info
High Confidence Low	Major
Investigate Low	Info

*** During initial roll out of the plugin, High confidence high issues should be configured to go to Critical.*

- Allow the above mapping to be configurable at global and project level.
- Configurable mapping should also include additional filter on category names. For example, user should be able to configure a High confidence high – cross-site scripting issue to Blocker and map all other categories in high confidence high to Critical.

PERFORMANCE CONSIDERATIONS AND LIMITATIONS

Fortify scan result and report stores large number of elements, containing various types of data. Importing all the information, especially, Fortify issue descriptions and recommendations, in each SonarQube issue will impact loading the issue widget severely. Therefore:

- Each issue will be accompanied by description just enough to help developers get their questions resolved.
- Example design choices include static text content with point of contact (ITSO) information and static link to public HP Fortify resources on that particular issue category.

SYSTEM INTEGRATION REQUIREMENTS

The proposed design for SonarQube-Fortify plugin is based on Fortify scan results from central SSC server. With architecture of CI systems, especially Jenkins, at Vanguard, the following are requirements for plugin integration:

- As SonarQube analysis (subsequently, the plugin execution) is only triggered from team Jenkins servers (ciserver:xxxxx) and the new plugin needs to work with XML reports extracted from FPR files, the team Jenkins machines require SCA installation.

  **Please note that the SCA installation on ciserver does NOT intend to run any scans, but will only be used by SonarQube plugin to <u>temporarily</u> download FPR file and extract XML reports to parse.
- Downloaded FPRs and extracted reports must NEVER be saved permanently on ciserver machine.
- SonarQube versions get upgraded on a regular basis on dev as well as prod instances. The plugin must integrate seamlessly with version upgrades.

SOFTWARE AND ACCESS REQUIREMENTS

This sections enlists software required for development, testing and deployment of the plugin. Detailed information on unit, int and prod testing process can be found in *"ITRISKREGISTER\CI-Documents-Config-SourceCode\SonarQube Deployment Guide\Vanguard SonarQube Fortify Plugin User Guide - v1.4.docx"*

DEVELOPMENT

The plugin development should be performed off-site. Therefore, the development and unit testing setup will be arranged by Cigital. On development machine, the following software and/or access is needed:

- Java 7
- Maven
- STS/Eclipse IDE
- BIRT extension for report designing (https://www.eclipse.org/downloads/packages/eclipse-ide-java-and-report-developers/neonr)
- Test setup with Jenkins, Fortify and SonarQube (matching versions at Vanguard Dev instances: 5.1.1 and 5.6.3)
- Vanguard filters in project templates and custom rules information
- Current plugin implementation for adding enhancements

DEPLOYMENT AND TESTING

The plugin deployment and testing is to be performed on-site. The following accesses and contact points are required to do so:

- TeamTrack (to create deployment tickets)
- Configuring Fortify widgets and rules on Dev and Prod instances
- Development setup mentioned in the section above to make Vanguard-specific modifications (e.g. Maven settings.xml file) and fix issues that come up during testing.

HIGH-LEVEL TECHNICAL DESIGN

The sub-sections below detail high level technical design for each of the key requirements.

FPR DOWNLOAD AND XML REPORT EXTRACTION

The design for these functionalities is partly dependent on the location of FPR download. The following considerations affect design decision to have a central location for downloading FPRs and extracting reports.

- Team Jenkins instances run SonarQube analysis independent of ITSO Jenkins servers (ciserver:xxxxx).

- Jenkins as well as Bamboo CI uploads FPR results to common Fortify SSC server.

Accordingly, the following two design choices are available:

1. Use the Fortify SSC VM to execute a script that downloads FPR, extracts report and makes XML report available for the SonarQube plugin.
   Pros
   If chosen, this can be done without any new Fortify SCA installation.
   Cons
   Developing the script, however, will need separate effort for development and testing, which will be addition to the existing timeline.

2. Install Fortify SCA on ciserver VM. (*Note: Fortify scan will not be run on ciserver VM; SCA will only be used for downloading FPRs and report extraction)*
   Pros
   If chosen, report download and XML generation can be integrated in the current process seamlessly, without needing additional development efforts. Also, available license can be used on the ciserver VM.
   Cons
   This will need installing Fortify SCA on ciserver VM.

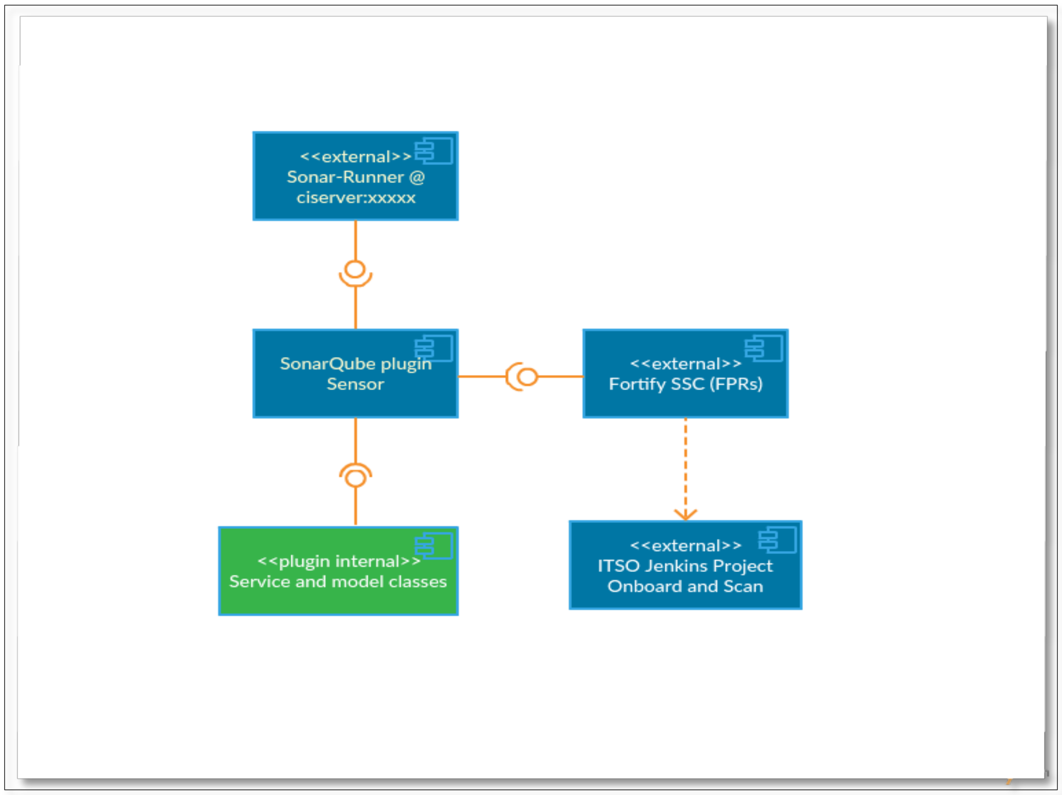
The option #2 is more long term and will require less maintenance than #1.

USER INTERFACE AND DESIGN ELEMENTS

The interface for these two functionalities will be the Jenkins server that invokes SonarQube analysis, which, in turn, invokes the SonarQube-Fortify plugin. The design elements in the plugin include:

- **Plugin properties** – configurable options for Fortify SSC URL and access credentials
- **Plugin sensor** – is invoked when SonarQube analysis is triggered via the Jenkins job being built
- **Plugin service class** - Fortify SCA (*fortifyclient*) command that downloads FPR from configured SSC URL and project data
- **Plugin model class** – Fortify SCA (*fortifyclient*) command that extracts XML report with Vanguard-specific report template

The following high-level component diagram shows various systems and components and their dependencies:



SONARQUBE ISSUE DRILL-DOWN AND RULES

The design for this functionality is governed by the following considerations:

- Issue drill-down can be heavy on overall SonarQube performance. Therefore, a configurable option for enabling/disabling issue drill-down feature should be provided.
- List of Fortify rules involve out-of-the-box as well as custom rules.
- Each rule needs some description/summary that cannot be left blank.

USER INTERFACE AND DESIGN ELEMENTS

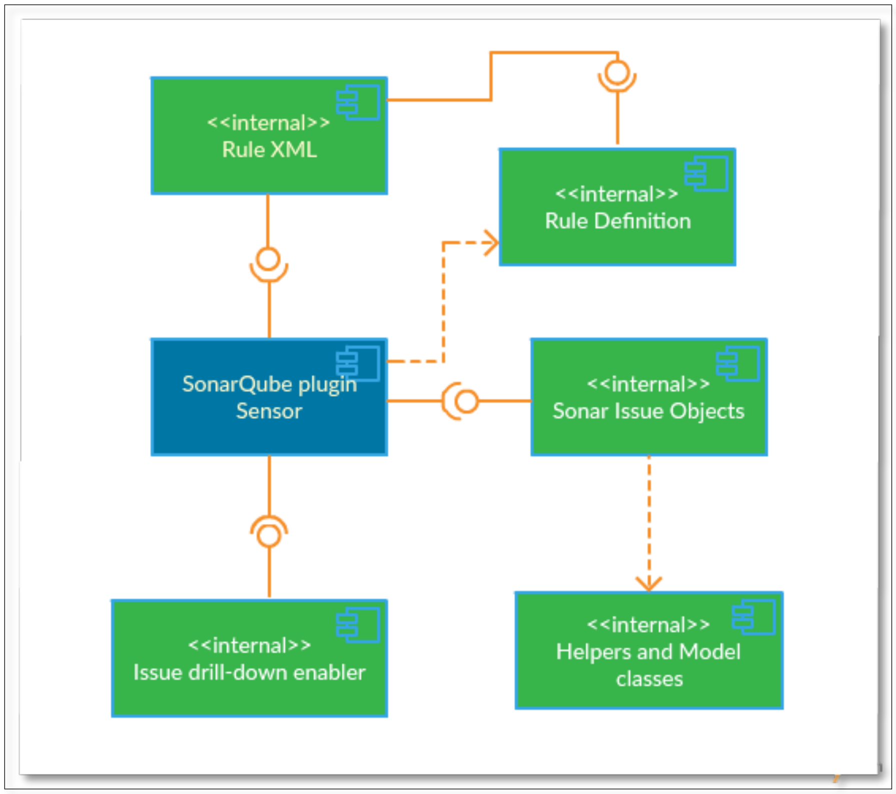
The interface for this functionality is SonarQube dashboard and rule and/or quality gate configuration screens. For issue drill-down to be available for viewing, the project must first be analyzed by invoking SonarQube analysis.

The design elements for this functionality include:

- **Rules XML** – this file contains list of all required Fortify rules, along with their keys and description.
- **Sonar issue objects** – these are responsible for rendering issues on the dashboard. With the help of other internal classes, issue objects must be populated for issue drill-down.

- **Rule definition** – this internal component should read through the rules XML file and import them to Sonar rule sets
- **Issue drill-down enabler** – Boolean configuration option (plugin property) at global as well as project level

The following high-level component diagram shows various systems and components and their dependencies:


<<internal>> Rule XML
<<internal>> Rule Definition
SonarQube plugin Sensor
<<internal>> Sonar Issue Objects
<<internal>> Issue drill-down enabler
<<internal>> Helpers and Model classes

MAPPING TO SONARQUBE SEVERITY CATEGORIES

The design of this functionality is governed by the following considerations:

- Fortify issues will be added to default SonarQube severity categories
- Vanguard-specific severity categories do not have one-to-one mapping on SonarQube severities.
- Mapping between severities is often dictated by project type, risk and team decisions. Therefore, defining configuration options at global and project level becomes necessary.

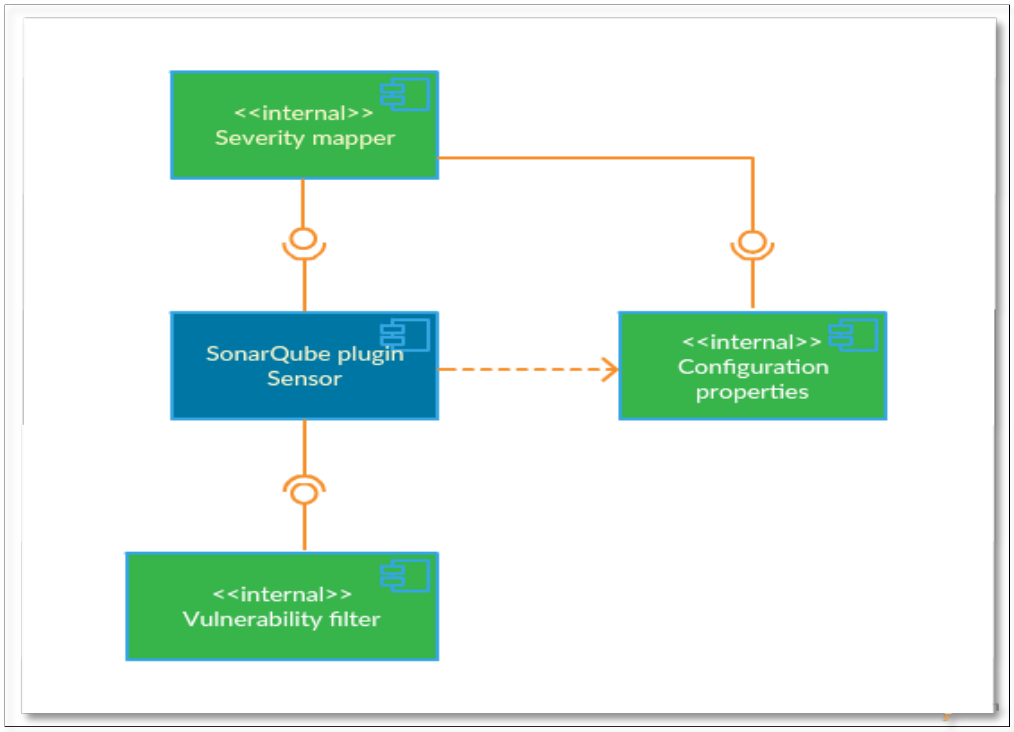
USER INTERFACE AND DESIGN ELEMENTS

The interface for this functionality is mainly SonarQube configuration screen and issue count summary on dashboard.

Accordingly, the design elements for this functionality include:

- **Severity mapper** – this plugin configuration option allows projects to define mapping between Vanguard's custom buckets in Fortify reports and SonarQube default severities.
- **Vulnerability filter** – this plugin configuration option allows projects to define mapping between specific vulnerability categories in combination with Vanguard custom buckets and SonarQube default severities.

The following high-level component diagram shows various systems and components and their dependencies:



APPENDIX: CIGITAL DEVELOPMENT SETUP

This section notes various components and systems setup to replicate test environment for use during off-site development at Cigital. For Fortify VM access and general information, please refer to Gather page.

System	Purpose	URL/Access point	Credentials
Fortify SSC server	Repository for uploaded scan results (FPRs)	http://fortifyssc007.cigital.com:8080/ssc/	Username: Sonar Pwd: S0n@r2016

Fortify SCA	Provides utilities to download FPR and generate XML report	--	--
Jenkins	Hosts Fortify SCA VM as slave, configured to run scans and trigger SonarQube analysis	https://ci-cd-jenkins.cigital.com:8080/	Domain
SonarQube Server	Stores analysis results for plugin, including issue count, drill down and rules.	http://fortifysca007.cigital.com:9000	Username: admin Pwd: admin