Audrey Long

JHU WebSec App Sec Tools
Module Assignment: SonarQube
03/02/2021

VM Installation

You're welcome to complete this assignment on your own machine, but I've also built a base install of an Ubuntu 18 VM you can download here. If you don't already have Oracle's VirtualBox, please install your OS-specific version by downloading it here and import the Ubuntu VM.
Reminder the deliverable will ask you to submit screenshots of your successful SonarQube and SonarScanner installation and scan output

Install SonarQube
Once your Ubuntu VM is up and running (if you're not running it on your own system), please complete the 2-minute install of SonarQube and start the SQ console by following the instructions here. NOTE: SonarQube requires you to have the Java JDK installed as a prerequisite. If you're using the VM, you can install it by running: **sudo apt-get install default-jdk**.

Install SonarScanner
Install SonarScanner by following these instructions.

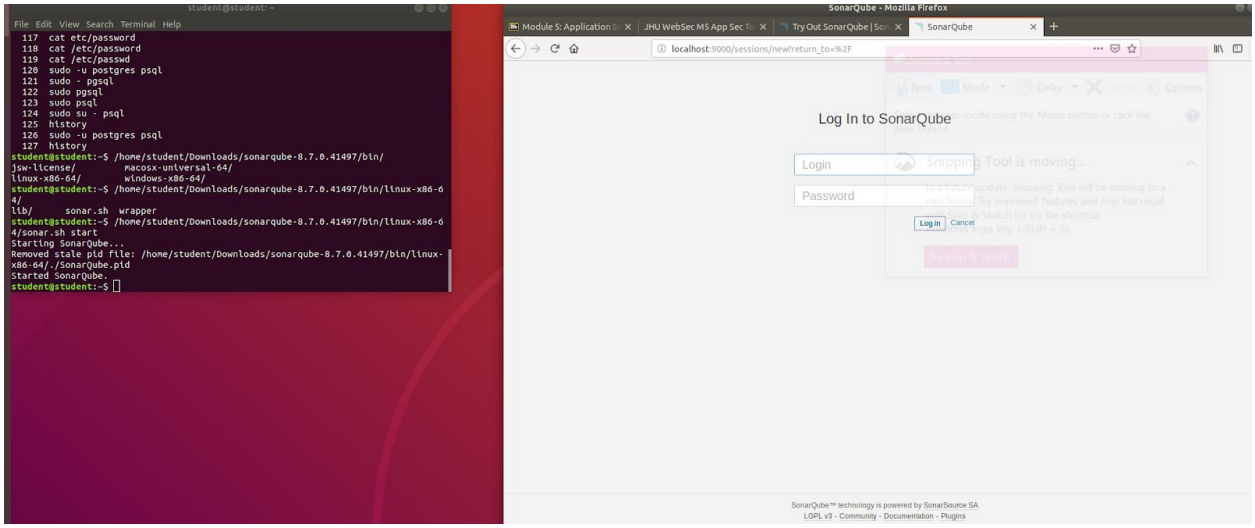

Figure 1: Image of sonarqube up and running

Create a Python Project In your home directory create a folder named **'sonar'** along with a file named **sonar.py** that contains the following source code. The **'sonar.properties'** file you created in the step above should be in this directory as well.

Make sure SonarQube is running by going into the directory where you installed it then navigating to [OS]/bin where [OS] is the directory corresponding to your operating system (linux-x86-64 if you're using the VM) and running: ./sonar.py console.

```
student@student:~/sonar$ sudo /home/student/Downloads/sonar-scanner-cli-4.6.0.2311-linux/sonar-scanner-4.6.0.2311-linux/bin/sonar-scanner -h
INFO:
INFO: usage: sonar-scanner [options]
INFO:
INFO: Options:
INFO:  -D,--define <arg>     Define property
INFO:  -h,--help             Display help information
INFO:  -v,--version          Display version information
INFO:  -X,--debug            Produce execution debug output
student@student:~/sonar$
```

Figure 2: Image of sonar-scanner up and running

Run a Scan

Navigate to the ‘sonar’ directory you created above and initiate a scan: sonar-scanner (if using the VM). Once the scan is complete, login to the SonarQube console through your browser: http://localhost:9000. You should now see 1 analyzed project.

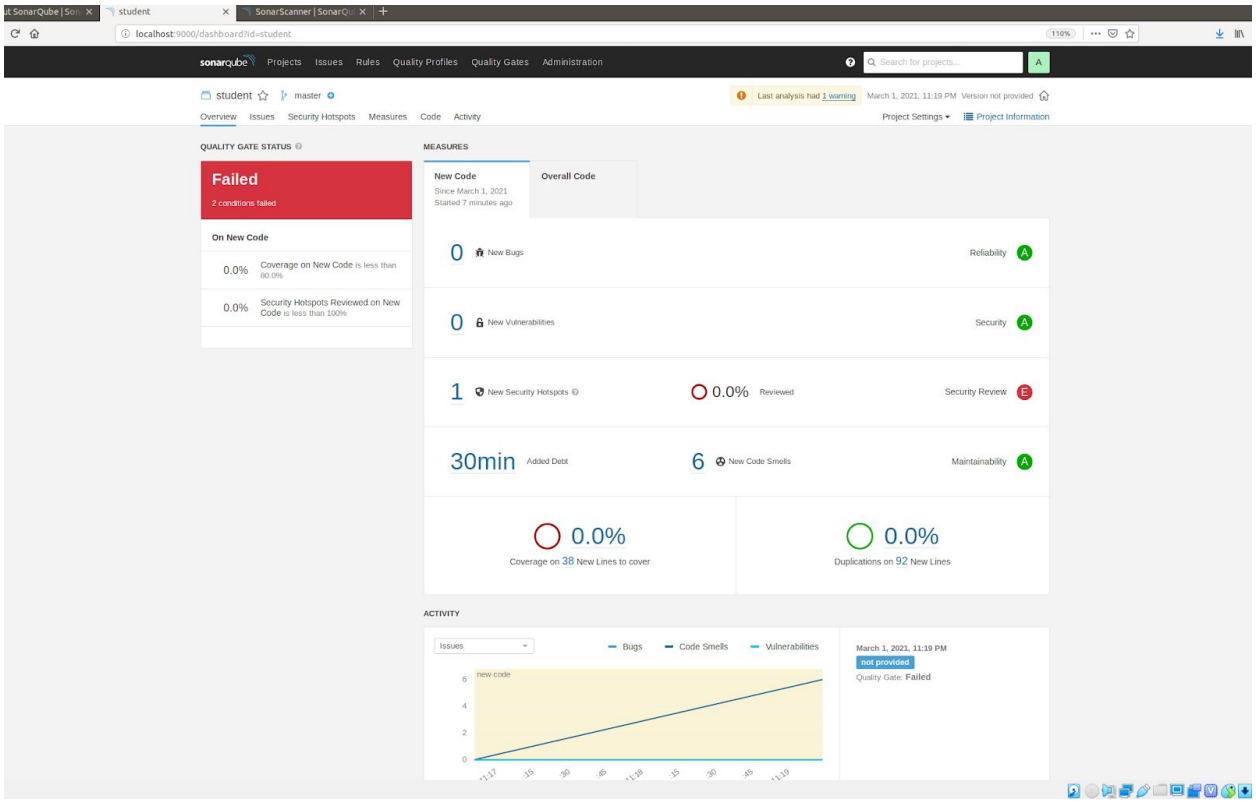

Figure 3: Image of scanned python code

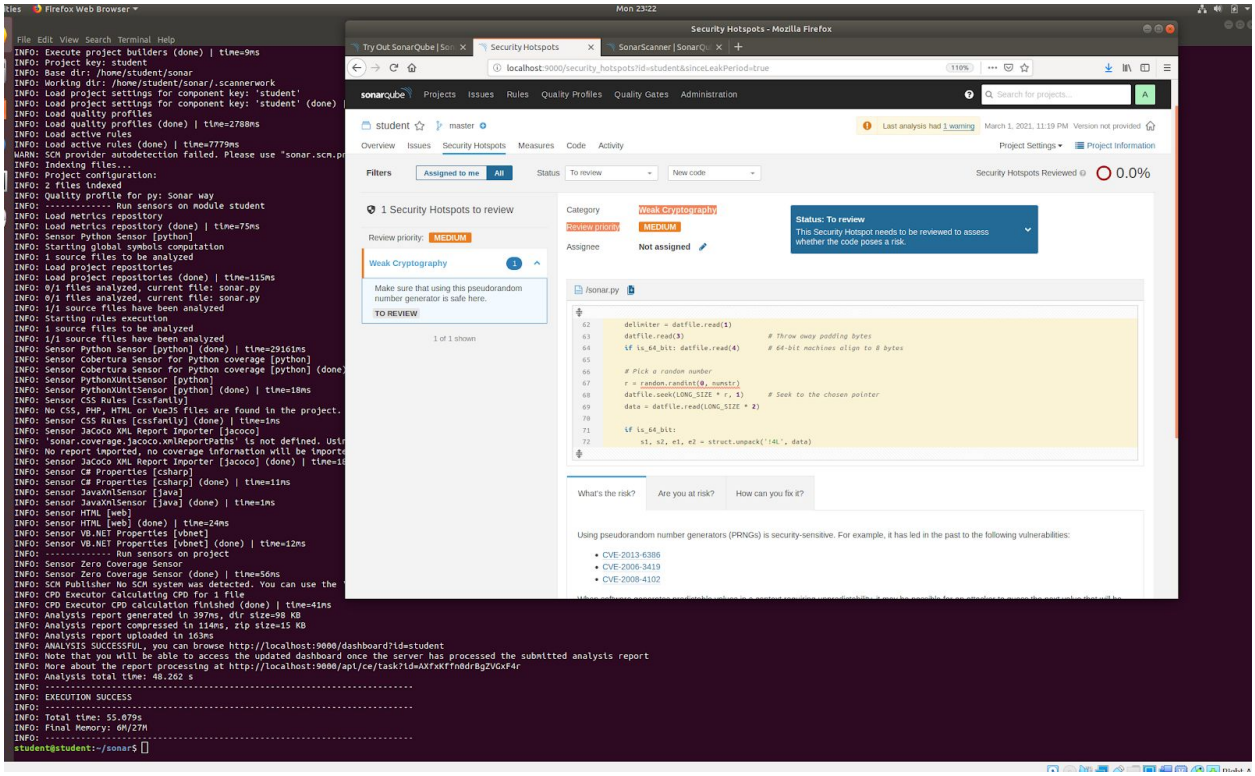

Figure 4: Image of code analysis

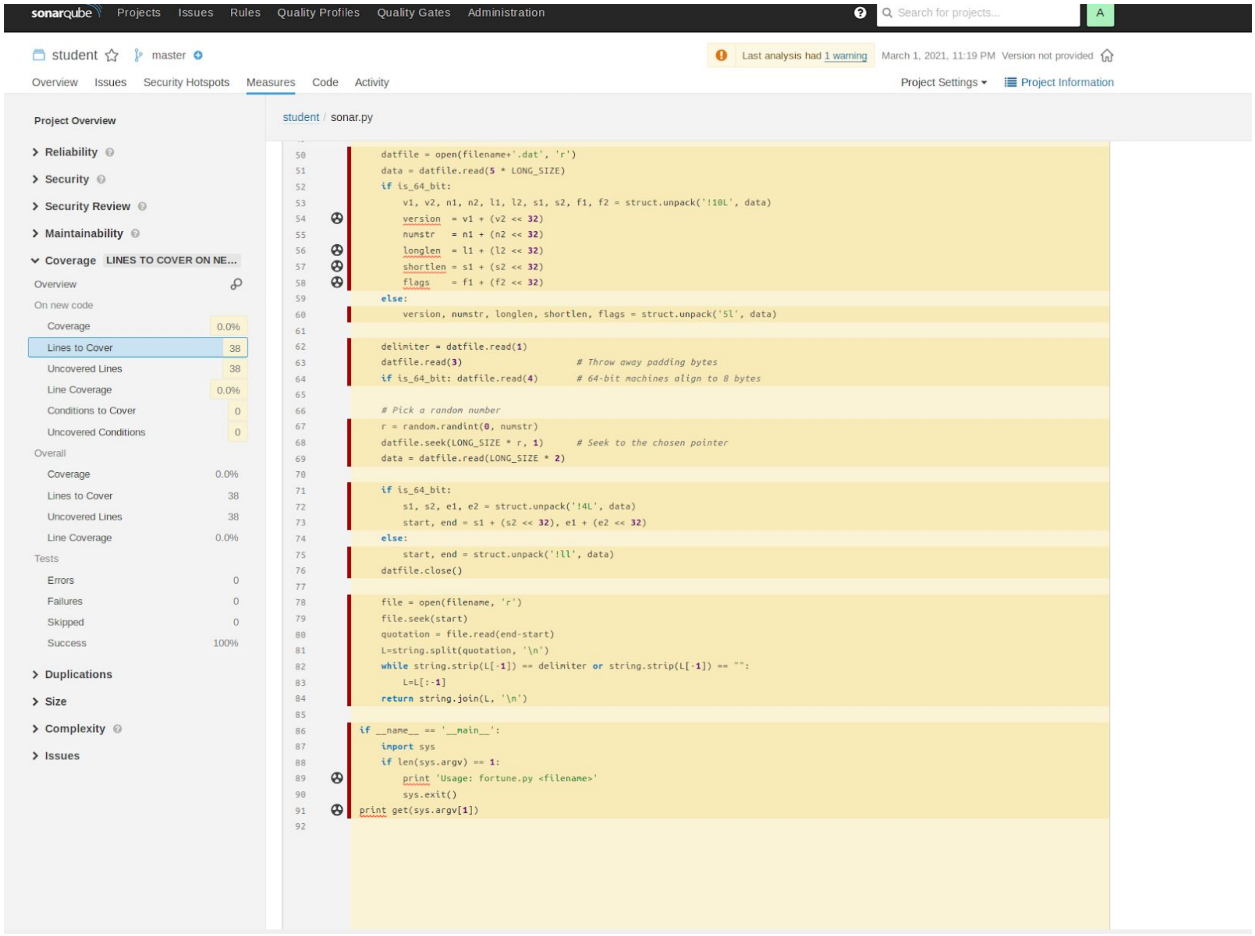

Figure 5: Code smells

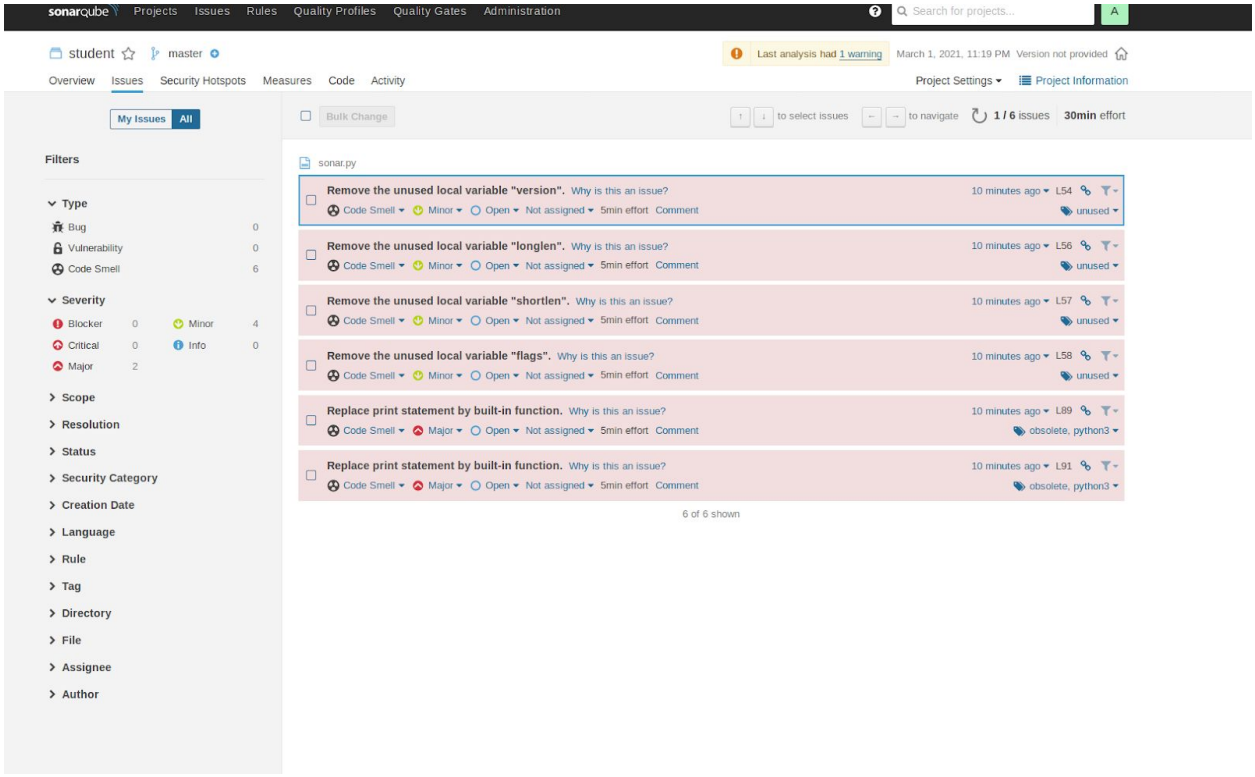

Figure 6: Issues tab

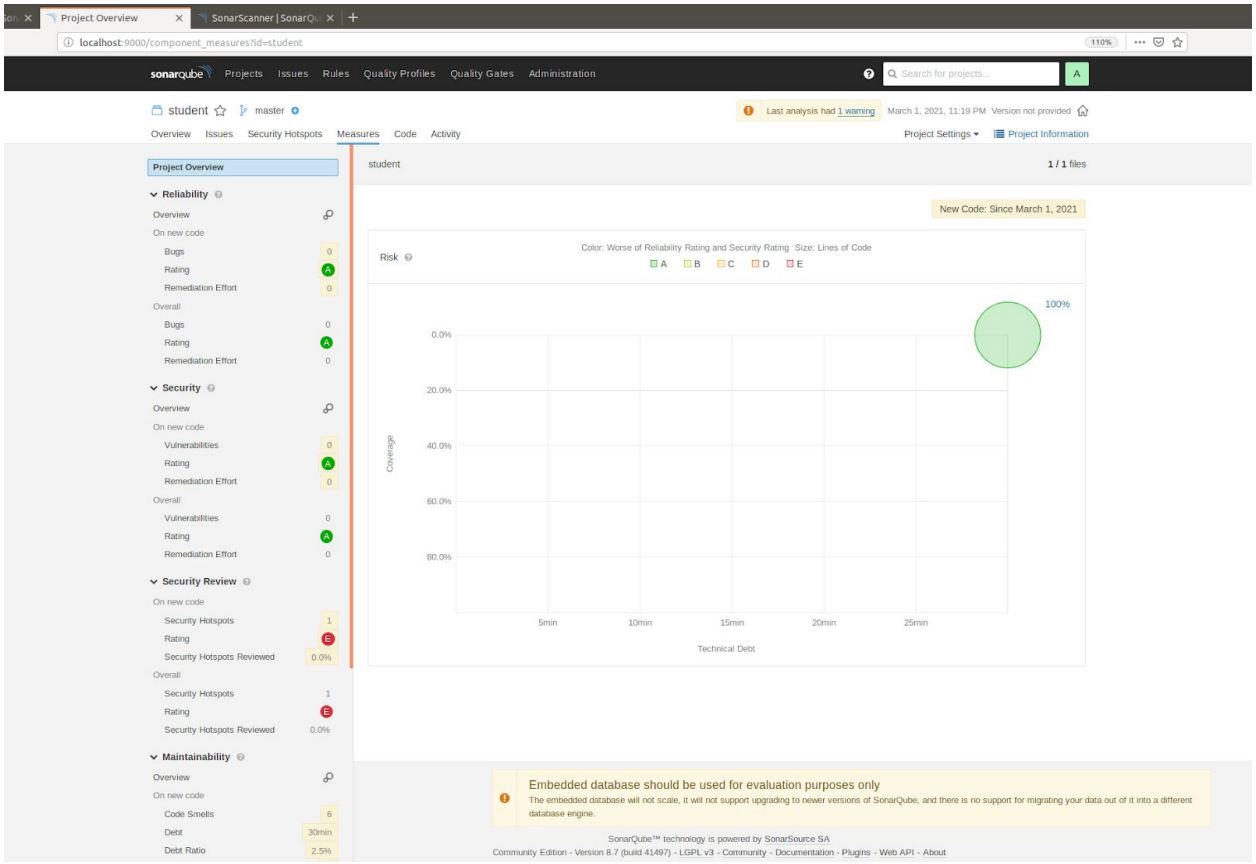

Figure 7: Measures Tab

Deliverables: PDF or Word doc

A. Provide annotated screenshots of:

1. successful SonarQube installation
2. successful SonarScanner installation
3. Resulting scan output of ‘badfortune.py’ code

B. Answer the following questions:

1. What is the difference between SonarQube and SonarScanner?

Sonar scanner is a separate client type application that connected with the sonarqube server will run project analysis with the scanner then send the results to the sonarqube server to process the findings.

2. What is static code analysis and how does it differ from dynamic analysis?

Static code analysis is performed as part of the code review process to ensure the code hitting production is free of bugs, vulnerabilities, and obvious glaring issues normally exploited by adversaries. Generally static code analysis is only performed on the source code whereas dynamic analysis does not access source code and binaries and can find business logic vulnerabilities and vulnerabilities in third-party software interfaces.

3. How many Python rules are run against your source code?
The Python analyzer parses the source code, creates an Abstract Syntax Tree (AST) and then walks through the entire tree. A coding rule is a visitor that is able to visit nodes from this AST. No custom rules have been added but you can see from the output that python cobertuna has been run against the source code.

4. How many Security Hotspot warnings were generated?
During this code review only 1 hotspot was found.

5. For each unique Security Hotspot, describe why it may/may not be a vulnerability and what measures you would take to secure each one.
From the scan results above it looks like the security hotspot is related to a vulnerability stemming from weak encryption.

6. What is a "code smell"?
In computer programming, a code smell is any characteristic in the source code of a program that possibly indicates a deeper problem. [3]

7. How many code smells were found during the scan?
From the figure above sonarqube found 6 code smells.

8. For each unique type of code smell, tell how you would remediate it.
The first four code smells can be remediated by removing unused variables. The last two can be remediated by replacing print statements with built in function to prevent extra information from being disclosed.

9. What is "code coverage"? Why is the code coverage 0% for this scan?
Code coverage is a software testing metric that determines the number of lines of code that is successfully validated under a test procedure, which in turn, helps in analyzing how comprehensively a software is verified. [4] this code coverage is 0% because no testing paths have been traversed.

10. Do a search on other similar tools to SonarScanner, choose one to research, provide a brief overview of how it helps accomplish application security.
Ive chosen the tool coverity since ive used it in the past. Coverity is a fast, accurate, and highly scalable static analysis (SAST) solution that helps development and security teams address security and quality defects early in the software development life cycle. [5] Coverity is a great tool to find memory leaks, buffer overflows, and many other dangerous code that can be exploited by an adversary. The only issue with coverity is that it can take a long time to run on a codebase with a lot of tests.

References

https://blog.setapp.pl/how-to-use-sonarscanner#:~:text=SonarScanner%20is%20a%20separate%20client,SonarQube%20except%20C%23%20and%20VB. [1]

https://owasp.org/www-community/controls/Static_Code_Analysis [2]

https://en.wikipedia.org/wiki/Code_smell [3]

https://www.codegrip.tech/productivity/everything-you-need-to-know-about-code-coverage/ [4]

https://www.synopsys.com/software-integrity/security-testing/static-analysis-sast.html?utm_source=google&utm_medium=cpc&utm_term=&utm_campaign=G_S_Coverity_Exact&cmp=ps-SIG-G_S_Coverity_Exact&gclid=Cj0KCQiA4feBBhC9ARIsABp_nbWwcWgf2FYrnIYyIeLScYG75pzcrNo0myS-b5S-ihZlnj44MdrqRgUaArdjEALw_wcB [5]