

Software Engineering Tools Lab Assignment

No-7

(Module 5- Source code testing using tools)

Q 1. What is Source code analysis? What is its importance?

Source code analysis, also known as static code analysis, is the process of analyzing software source code to identify potential vulnerabilities, defects, and weaknesses. This process involves using automated tools to scan the source code and check for coding errors, security vulnerabilities, and other issues that may impact the software's reliability, performance, or security.

The importance of source code analysis lies in its ability to help identify and prevent potential issues before they can become serious problems. By analyzing the source code early in the development process, developers can identify and fix problems quickly and efficiently, which can help reduce costs and improve the overall quality of the software.

Source code analysis can also help improve the security of software by identifying potential security vulnerabilities that could be exploited by attackers. By identifying and fixing these vulnerabilities early in the development process, developers can reduce the risk of security breaches and protect sensitive data from being compromised.

In summary, source code analysis is an important part of the software development process that can help improve the quality, reliability, and security of software by identifying and addressing potential issues early on.

Q 2. Below are the some important open source tools used in testing the source code, provide the information of below tools with respect to

- a. Owner/ developer
- b. Developed in which language
- c. Brief information/introduction
- d. Language support (applicable for source code written in language)

Name:Sharvari Yashwant Patil
PRN NO:2020BTECS00077

- e. Advantages
- f. Disadvantages

Source code analysis tools-

I. VisualCodeGrepper :

- a. Owner/developer: Olivier Renault
- b. Developed in which language: Python
- c. Brief information/introduction: VisualCodeGrepper is an open-source, cross-platform source code analysis tool that scans code for potential security vulnerabilities. It is designed to be easy to use and can be integrated into existing development workflows. VisualCodeGrepper supports a wide range of programming languages and can be customized to fit specific needs.
- d. Language support: VisualCodeGrepper supports a wide range of programming languages, including C, C++, Java, Python, Ruby, PHP, and more.
- e. Advantages: VisualCodeGrepper is easy to use, customizable, and supports a wide range of programming languages. It can quickly identify potential security vulnerabilities and provide suggestions for fixing them.
- f. Disadvantages: VisualCodeGrepper may produce false positives or miss some vulnerabilities, and it requires some level of technical expertise to use effectively.

II) Rips

- : a. Owner/developer: Johannes Dahse
- b. Developed in which language: PHP
 - c. Brief information/introduction: Rips is an open-source static code analysis tool for detecting security vulnerabilities in PHP applications. It scans PHP code for potential vulnerabilities, including SQL injection, cross-site scripting (XSS), and remote file inclusion. Rips also provides detailed reports and suggested fixes.
 - d. Language support: Rips is specifically designed for PHP code.
 - e. Advantages: Rips is specifically designed for PHP code and can quickly identify potential security vulnerabilities. It provides detailed reports and suggested fixes, making it easier to address security issues.

Name:Sharvari Yashwant Patil
PRN NO:2020BTECS00077

f. Disadvantages: Rips is limited to PHP code and may produce false positives or miss some vulnerabilities.

III) Brakeman

a. Owner/developer: Justin Collins

b. Developed in which language: Ruby

c. Brief information/introduction: Brakeman is an open-source, static analysis security tool for Ruby on Rails applications. It analyzes Rails application code to identify potential security vulnerabilities, including SQL injection, cross-site scripting (XSS), and more. Brakeman also provides detailed reports and suggested fixes.

d. Language support: Brakeman is specifically designed for Ruby on Rails applications.

e. Advantages: Brakeman is specifically designed for Ruby on Rails applications and can quickly identify potential security vulnerabilities. It provides detailed reports and suggested fixes, making it easier to address security issues.

f. Disadvantages: Brakeman is limited to Ruby on Rails applications and may produce false positives or miss some vulnerabilities.

IV) Flawfinder :

a. Owner/developer: David A. Wheeler

b. Developed in which language: C/C++

c. Brief information/introduction: Flawfinder is an open-source static analysis tool for identifying potential security vulnerabilities in C/C++ code. It scans code for potential issues, including buffer overflows, race conditions, and more. Flawfinder also provides a severity rating for each identified issue.

d. Language support: Flawfinder is specifically designed for C/C++ code.

Name:Sharvari Yashwant Patil
PRN NO:2020BTECS00077

e. Advantages: Flawfinder is specifically designed for C/C++ code and can quickly identify potential security vulnerabilities. It provides a severity rating for each identified issue, making it easier to prioritize fixes.

f. Disadvantages: Flawfinder is limited to C/C++ code and may produce false positives or miss some vulnerabilities.

V)Bandit :

a. Owner/developer: PyCQA

b. Developed in which language: Python

c. Brief information/introduction: Bandit is an open-source security tool for Python applications. It performs static analysis on Python code to identify potential security vulnerabilities, including hard-coded secrets, SQL injection, and more. Bandit provides a severity rating for each identified issue

Q 3. Perform source code testing using **Flawfinder** for the code written in 'c' and 'cpp' language given below

Link- <https://github.com/sidp1991/SETAssignment>

Note-use files program1.c and program2.cpp present on above link.

After performing analysis create a report which will contain below points

- a. Number of hits
- b. Potential risks
- c. Suggested alternatives for these risks
- d. Updating the code as per suggestions
- e. Re-execution of code after updating the changes.

Name:Sharvari Yashwant Patil
PRN NO:2020BTECS00077

For program1.cpp

Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.

Number of rules (primarily dangerous function names) in C/C++ ruleset: 222

Examining program1.c

FINAL RESULTS:

program1.c:11: [4] (buffer) strcpy:

Does not check for buffer overflows when copying to destination [MS-banned] (CWE-120). Consider using snprintf, strcpy_s, or strncpy (warning: strncpy easily misused).

program1.c:9: [2] (buffer) char:

Statically-sized arrays can be improperly restricted, leading to potential overflows or other issues (CWE-119!/CWE-120). Perform bounds checking, use functions that limit length, or ensure that the size is larger than the maximum possible length.

ANALYSIS SUMMARY:

Hits = 2

Lines analyzed = 13 in approximately 0.02 seconds (624 lines/second)

Physical Source Lines of Code (SLOC) = 10

Hits@level = [0] 1 [1] 0 [2] 1 [3] 0 [4] 1 [5] 0

Hits@level+ = [0+] 3 [1+] 2 [2+] 2 [3+] 1 [4+] 1 [5+] 0

Hits/KSLOC@level+ = [0+] 300 [1+] 200 [2+] 200 [3+] 100 [4+] 100 [5+] 0

Minimum risk level = 1

Name: Sharvari Yashwant Patil
PRN NO: 2020BTECS00077

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Sharvari\SETAssignment-main\SETAssignment-main> flawfinder program1.c
Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.
Number of rules (primarily dangerous function names) in C/C++ ruleset: 222
Examining program1.c

FINAL RESULTS:

program1.c:11: [4] (buffer) strcpy:
  Does not check for buffer overflows when copying to destination [MS-banned]
  (CWE-120). Consider using snprintf, strcpy_s, or strncpy (warning: strncpy
  easily misused).
program1.c:9: [2] (buffer) char:
  Statically-sized arrays can be improperly restricted, leading to potential
  overflows or other issues (CWE-119!/CWE-120). Perform bounds checking, use
  functions that limit length, or ensure that the size is larger than the
  maximum possible length.

ANALYSIS SUMMARY:

Hits = 2
Lines analyzed = 13 in approximately 0.02 seconds (624 lines/second)
Physical Source Lines of Code (SLOC) = 10
Hits@level+ = [0+] 1 [1] 0 [2] 1 [3] 0 [4] 1 [5] 0
Hits@level+ = [0+] 3 [1+] 2 [2+] 2 [3+] 1 [4+] 1 [5+] 0
Hits/KSLOC@level+ = [0+] 300 [1+] 200 [2+] 200 [3+] 100 [4+] 100 [5+] 0
Minimum risk level = 1

Not every hit is necessarily a security vulnerability.
You can inhibit a report by adding a comment in this form:
// Flawfinder: ignore
Make *sure* it's a false positive!
You can use the option --neverignore to show these.

There may be other security vulnerabilities; review your code!
See 'Secure Programming HOWTO'
```

For program2.cpp

flawfinder program2.cpp

Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.

Number of rules (primarily dangerous function names) in C/C++ ruleset: 222

Examining program2.cpp

FINAL RESULTS:

program2.cpp:9: [2] (buffer) char:

Statically-sized arrays can be improperly restricted, leading to potential overflows or other issues (CWE-119!/CWE-120). Perform bounds checking, use functions that limit length, or ensure that the size is larger than the maximum possible length.

program2.cpp:12: [2] (misc) fopen:

Name: Sharvari Yashwant Patil
PRN NO: 2020BTECS00077

Check when opening files - can an attacker redirect it (via symlinks), force the opening of special file type (e.g., device files), move things around to create a race condition, control its ancestors, or change its contents? (CWE-362).

ANALYSIS SUMMARY:

Hits = 2

Lines analyzed = 25 in approximately 0.02 seconds (1087 lines/second)

Physical Source Lines of Code (SLOC) = 19

Hits@level = [0] 0 [1] 0 [2] 2 [3] 0 [4] 0 [5] 0

Hits@level+ = [0+] 2 [1+] 2 [2+] 2 [3+] 0 [4+] 0 [5+] 0

Hits/KSLOC@level+ = [0+] 105.263 [1+] 105.263 [2+] 105.263 [3+] 0 [4+] 0 [5+] 0

Minimum risk level = 1

```
Windows PowerShell
You can use the option --neverignore to show these.

There may be other security vulnerabilities; review your code!
See 'Secure Programming HOWTO'
(https://dwtweler.com/secure-programs) for more information.
PS C:\Users\Sharvari\SETAssignment-main\SETAssignment-main> flawfinder program2.cpp
Flawfinder version 2.0.19, (C) 2001-2019 David A. Wheeler.
Number of rules (primarily dangerous function names) in C/C++ ruleset: 222
Examining program2.cpp

FINAL RESULTS:

program2.cpp:9: [2] (buffer) char:
  Statically-sized arrays can be improperly restricted, leading to potential
  overflows or other issues (CWE-119/CWE-120). Perform bounds checking, use
  functions that limit length, or ensure that the size is larger than the
  maximum possible length.
program2.cpp:12: [2] (misc) fopen:
  Check when opening files - can an attacker redirect it (via symlinks),
  force the opening of special file type (e.g., device files), move things
  around to create a race condition, control its ancestors, or change its
  contents? (CWE-362).

ANALYSIS SUMMARY:

Hits = 2
Lines analyzed = 25 in approximately 0.02 seconds (1087 lines/second)
Physical Source Lines of Code (SLOC) = 19
Hits@level = [0] 0 [1] 0 [2] 2 [3] 0 [4] 0 [5] 0
Hits@level+ = [0+] 2 [1+] 2 [2+] 2 [3+] 0 [4+] 0 [5+] 0
Hits/KSLOC@level+ = [0+] 105.263 [1+] 105.263 [2+] 105.263 [3+] 0 [4+] 0 [5+] 0
Minimum risk level = 1

Not every hit is necessarily a security vulnerability.
You can inhibit a report by adding a comment in this form:
// flawfinder: ignore
Make *sure* it's a false positive!
You can use the option --neverignore to show these.

There may be other security vulnerabilities; review your code!
```

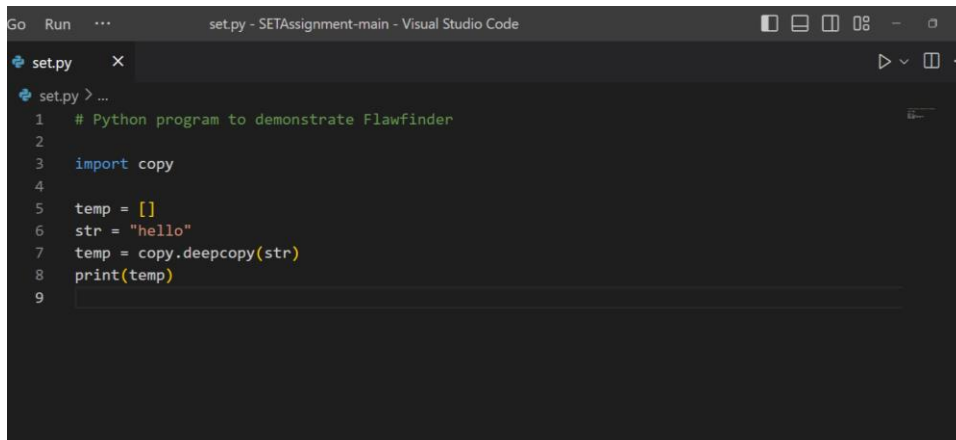
Name:Sharvari Yashwant Patil
PRN NO:2020BTECS00077

Q 4. Perform source code testing using **Bandit** for your code written in 'python' language (use your previous code) for any security flaws

After performing analysis create a report which will contain below points

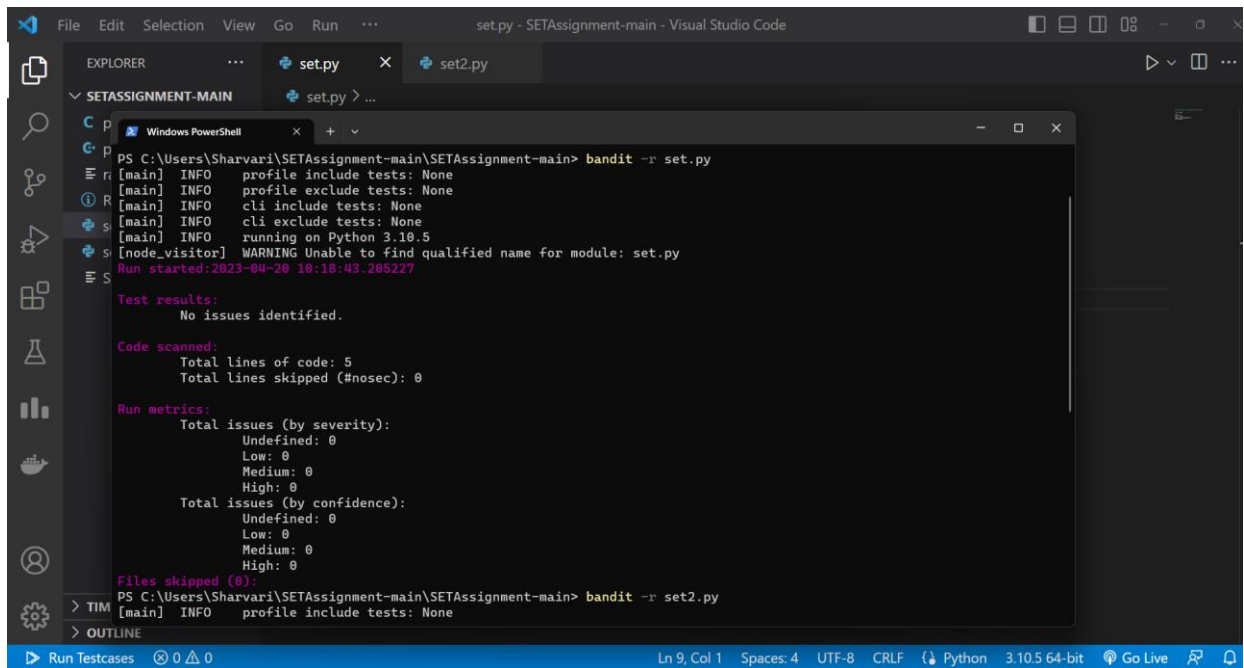
- a. Number of hits
- b. Potential risks
- c. Suggested alternatives for these risks
- d. Updating the code as per suggestions
- e. Re-execution of code after updating the changes.

For set.py



```
Go Run ... set.py - SETAssignment-main - Visual Studio Code
set.py
set.py > ...
1 # Python program to demonstrate Flowfinder
2
3 import copy
4
5 temp = []
6 str = "hello"
7 temp = copy.deepcopy(str)
8 print(temp)
9
```


Name: Sharvari Yashwant Patil
PRN NO: 2020BTECS00077



```
PS C:\Users\Sharvari\SETAssignment-main\SETAssignment-main> bandit -r set.py
[main] INFO profile include tests: None
[main] INFO profile exclude tests: None
[main] INFO cli include tests: None
[main] INFO cli exclude tests: None
[main] INFO running on Python 3.10.5
[node_visitor] WARNING Unable to find qualified name for module: set.py
Run started:2023-04-20 10:18:43.205227

Test results:
  No issues identified.

Code scanned:
  Total lines of code: 5
  Total lines skipped (#nosec): 0

Run metrics:
  Total issues (by severity):
    Undefined: 0
    Low: 0
    Medium: 0
    High: 0
  Total issues (by confidence):
    Undefined: 0
    Low: 0
    Medium: 0
    High: 0
  Files skipped (0):
    PS C:\Users\Sharvari\SETAssignment-main\SETAssignment-main> bandit -r set2.py
[main] INFO profile include tests: None
```

```
[main] INFO profile include tests: None
[main] INFO profile exclude tests: None
[main] INFO cli include tests: None
[main] INFO cli exclude tests: None
[main] INFO running on Python 3.10.5
[node_visitor] WARNING Unable to find qualified name for module: set.py
Run started:2023-04-20 10:18:43.205227
```

Test results:

No issues identified.

Code scanned:

Total lines of code: 5

Total lines skipped (#nosec): 0

Name:Sharvari Yashwant Patil
PRN NO:2020BTECS00077

Run metrics:

Total issues (by severity):

Undefined: 0

Low: 0

Medium: 0

High: 0

Total issues (by confidence):

Undefined: 0

Low: 0

Medium: 0

High: 0

Files skipped (0):