**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 is the process of examining the source code of a software application or system to identify potential vulnerabilities, defects, or bugs. This analysis can be performed using manual or automated techniques, and it typically involves reviewing the code's structure, syntax, and logic.

The importance of source code analysis lies in its ability to identify potential issues before they can cause problems in production. By analyzing the source code, developers can detect and fix errors early in the development lifecycle, reducing the risk of security breaches, data loss, and other issues. Additionally, source code analysis can help developers identify code that is difficult to maintain, leading to better overall code quality and more efficient development processes.

Source code analysis is also essential for compliance with industry and regulatory standards, such as the Payment Card Industry Data Security Standard (PCI DSS) and the Health Insurance Portability and Accountability Act (HIPAA). These standards require organizations to identify and remediate vulnerabilities in their software systems, which can be achieved through source code analysis.

Overall, source code analysis is a critical component of software development and maintenance, as it helps ensure the security, stability, and maintainability of software applications and systems.

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

1. Owner/ developer
2. Developed in which language
3. Brief information/introduction
4. Language support (applicable for source code written in language)
5. Advantages
6. Disadvantages

**Source code analysis tools-**

**I. VisualCodeGrepper**

a. Owner/developer: The VisualCodeGrepper tool is developed by Anand Malik and is released under the GNU General Public License.

b. Developed in which language: Java.

c. Brief information/introduction: VisualCodeGrepper is an open-source static code analysis tool that allows users to search for specific patterns or strings within source code files. The tool is designed to support a wide range of programming languages, including Java, C++, Python, and more.

d. Language support: VisualCodeGrepper supports a wide range of programming languages, including Java, C++, Python, Ruby, PHP, JavaScript, and more.

e. Advantages: VisualCodeGrepper is easy to use and provides fast and accurate results. The tool can be used to search for specific patterns or strings within source code files, making it useful for identifying potential vulnerabilities or defects in code.

f. Disadvantages: VisualCodeGrepper may not be as comprehensive as some other code analysis tools, and it may not be suitable for more complex applications or systems.

**II. Rips**

a. Owner/developer: Rips is developed and maintained by RIPS Technologies GmbH.

b. Developed in which language: PHP.

c. Brief information/introduction: Rips is an open-source static code analysis tool designed to identify security vulnerabilities in PHP applications. The tool analyzes code to identify potential vulnerabilities, such as SQL injection, cross-site scripting, and more.

d. Language support: Rips is specifically designed to analyze PHP applications.

e. Advantages: Rips is a powerful tool that is specifically designed to identify security vulnerabilities in PHP applications. The tool is easy to use and provides detailed reports that make it easy to identify and remediate vulnerabilities.

f. Disadvantages: Rips may not be as comprehensive as some other code analysis tools, and it may not be suitable for applications or systems that are written in languages other than PHP.

**III. Brakeman**

a. Owner/developer: Brakeman is an open-source tool developed by Justin Collins.

b. Developed in which language: Ruby.

c. Brief information/introduction: Brakeman is a static analysis tool that is designed to identify security vulnerabilities in Ruby on Rails applications. The tool analyzes code to identify potential vulnerabilities, such as SQL injection, cross-site scripting, and more.

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

e. Advantages: Brakeman is a powerful tool that is specifically designed to identify security vulnerabilities in Ruby on Rails applications. The tool is easy to use and provides detailed reports that make it easy to identify and remediate vulnerabilities.

f. Disadvantages: Brakeman may not be as comprehensive as some other code analysis tools, and it may not be suitable for applications or systems that are written in languages other than Ruby.

**IV. Bandit:**

a. Owner/Developer: OpenStack

b. Developed in: Python

c. Brief information/introduction: Bandit is a security-focused source code analysis tool for Python. It can identify security issues such as SQL injection, cross-site scripting (XSS), and hardcoded passwords. Bandit can analyze Python source code and report potential security vulnerabilities.

d. Language support: Python

e. Advantages: Bandit is easy to use and can quickly identify potential security vulnerabilities in Python source code. It is also customizable and allows users to define their own security policies. Bandit integrates well with other Python development tools.

f. Disadvantages: Bandit is only suitable for Python source code analysis and cannot be used for other languages. It may also have limitations in identifying more complex security issues.

**V. Flawfinder**

a. Owner/developer: Flawfinder is an open-source tool developed by David A. Wheeler.

b. Developed in which language: Python

c. Brief information/introduction: Flawfinder is a simple static analysis tool that searches for potential security vulnerabilities in source code files. The tool scans source code files and generates a report of potential vulnerabilities, such as buffer overflow, SQL injection, and more.

d. Language support: C, C++, Java, Python, and more.

e. Advantages: Flawfinder is easy to use and requires no prior knowledge of security vulnerabilities. It can quickly identify potential security issues in the source code, making it an essential tool for developers concerned with software security.

f. Disadvantages: Flawfinder is not a comprehensive solution and may not catch all potential security vulnerabilities. It is also a command-line tool, which may be less accessible to some users.

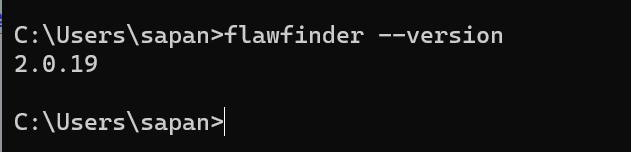
Q 3. Perform source code testing using **Flawfinder** for the code written in ‘c’ and

‘cpp’ language given below

After performing analysis create a report which will contain below points

1. Number of hits (potential security flaws)
2. Potential risks
3. Suggested alternatives for these risks
4. Updating the code as per suggestions
5. Re-execution of code after updating the changes.

INSTALL FLAWFINDER using pip install flawfinder

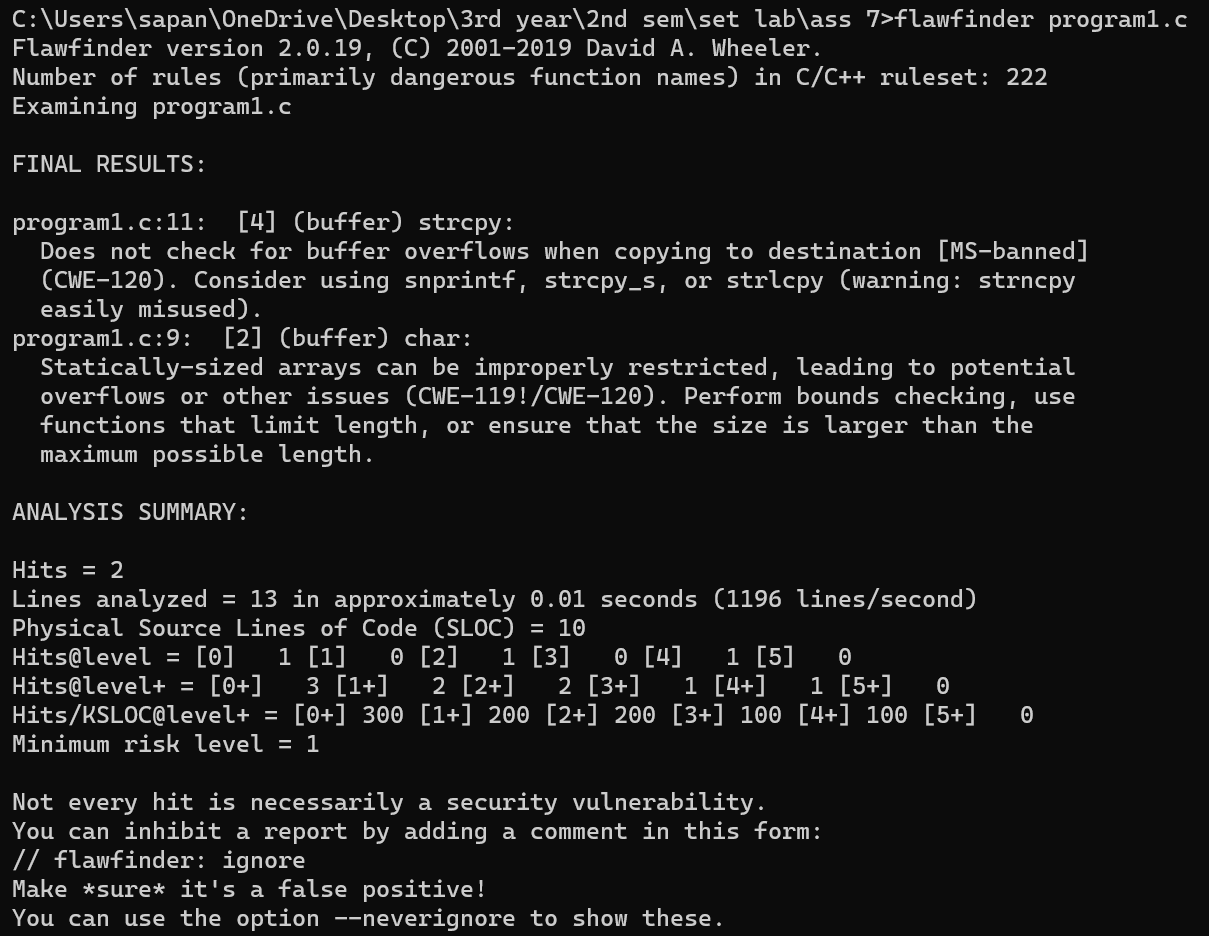


Link- https://github.com/sidp1991/SETAssignment

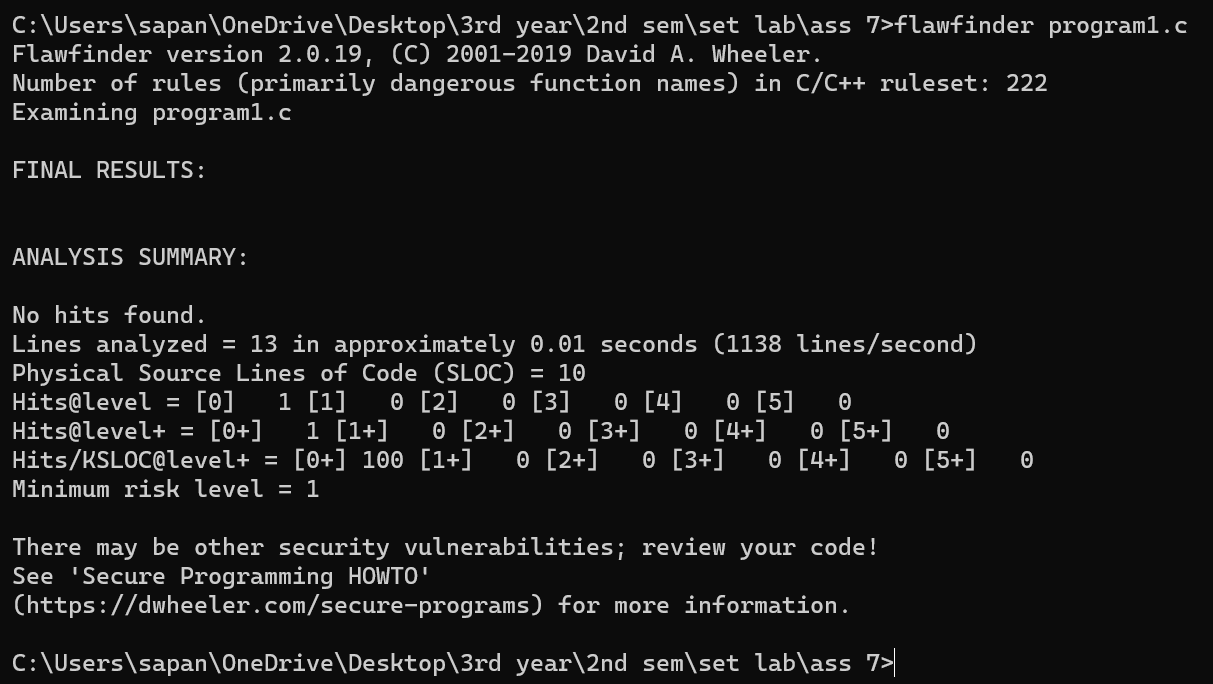
Note-use files program1.c and program2.cpp present on above link.

Go to the required file in cmd and then –

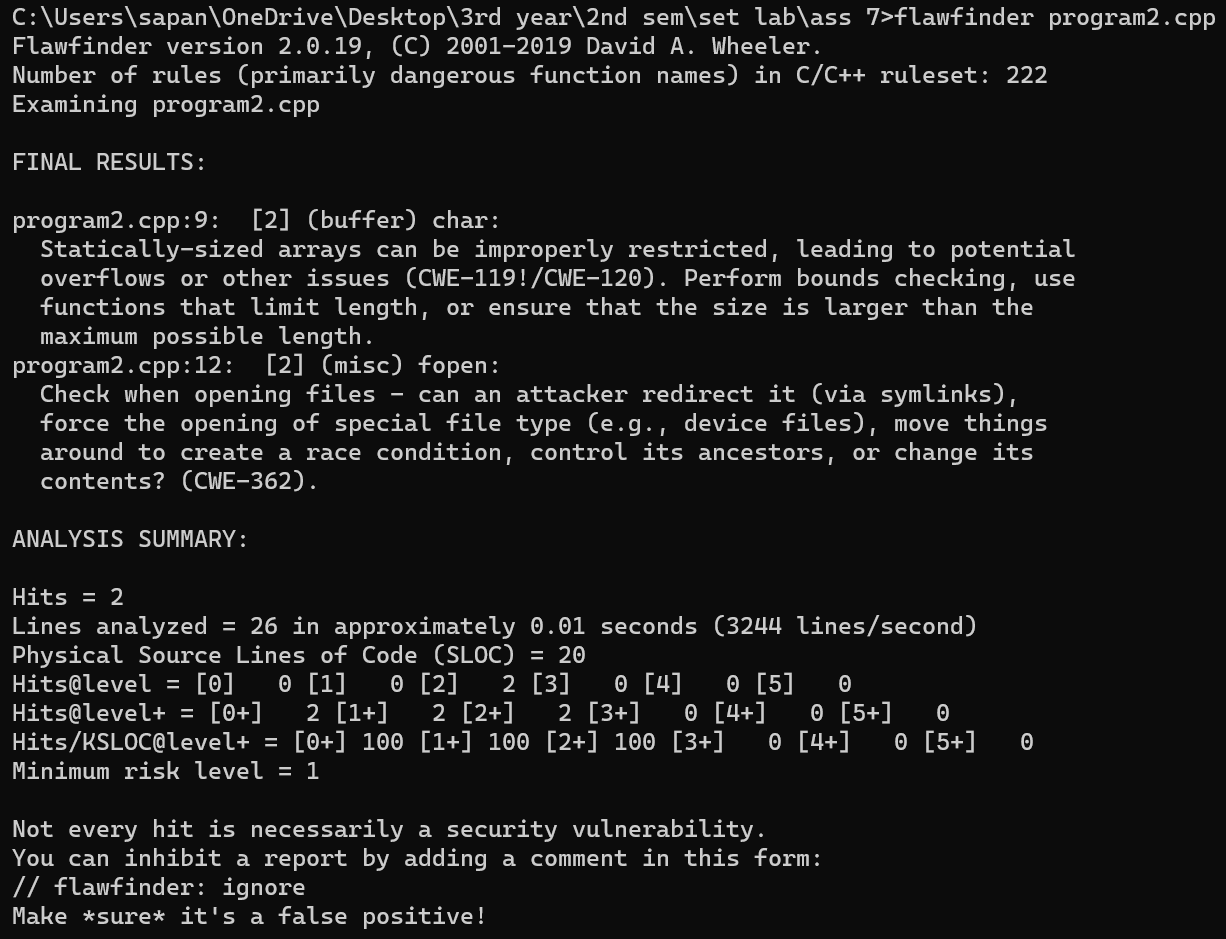
Command – flawfinder program1.c



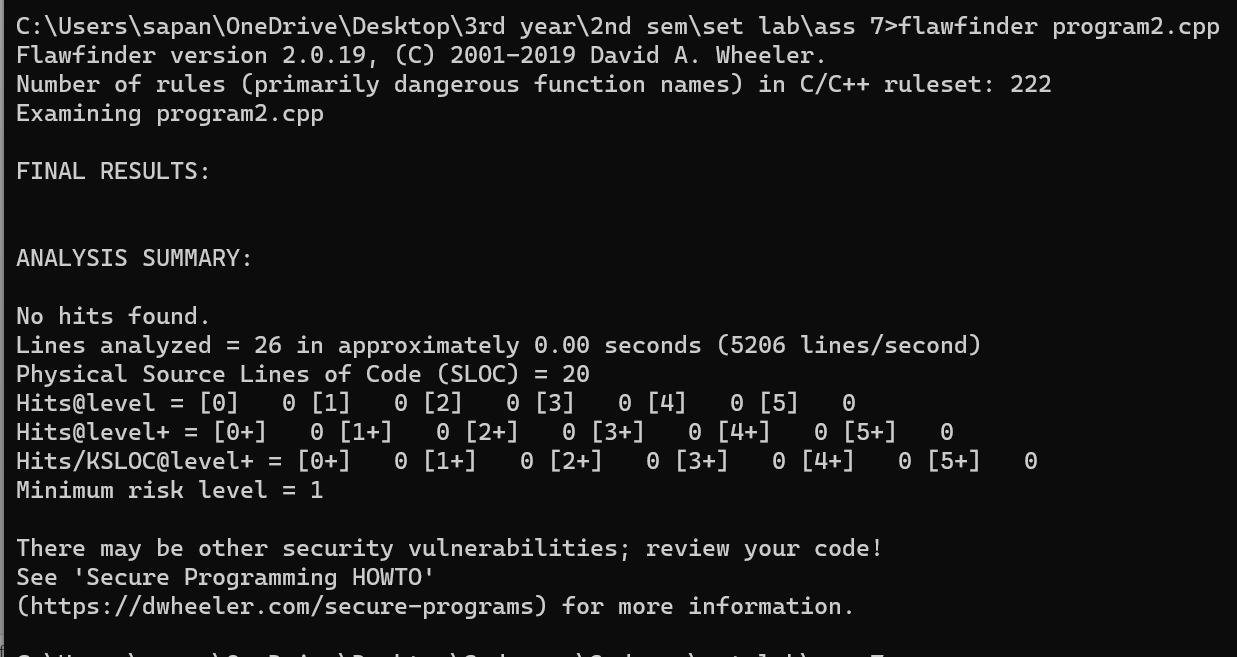
1. Number of hits (potential security flaws) - 2
2. Potential risks - 2
3. Suggested alternatives for these risks – i. use strcpy\_s instead of strcpy as it doesn’t check for buffer overflow, ii. make sure array size is not fixed.
4. Updating the code as per suggestions – using strcpy\_s instead of strcpy and arr[] instead of arr[10].
5. Re-execution of code after updating the changes.



Command – flawfinder program2.cpp

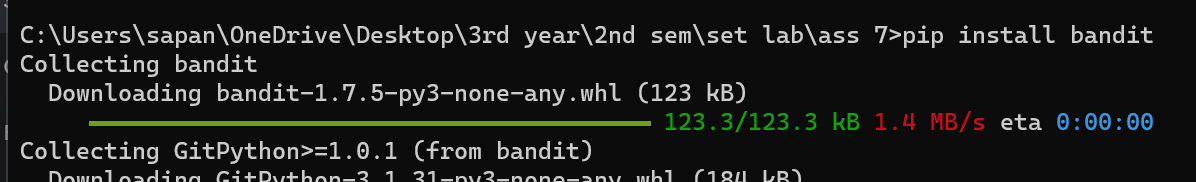


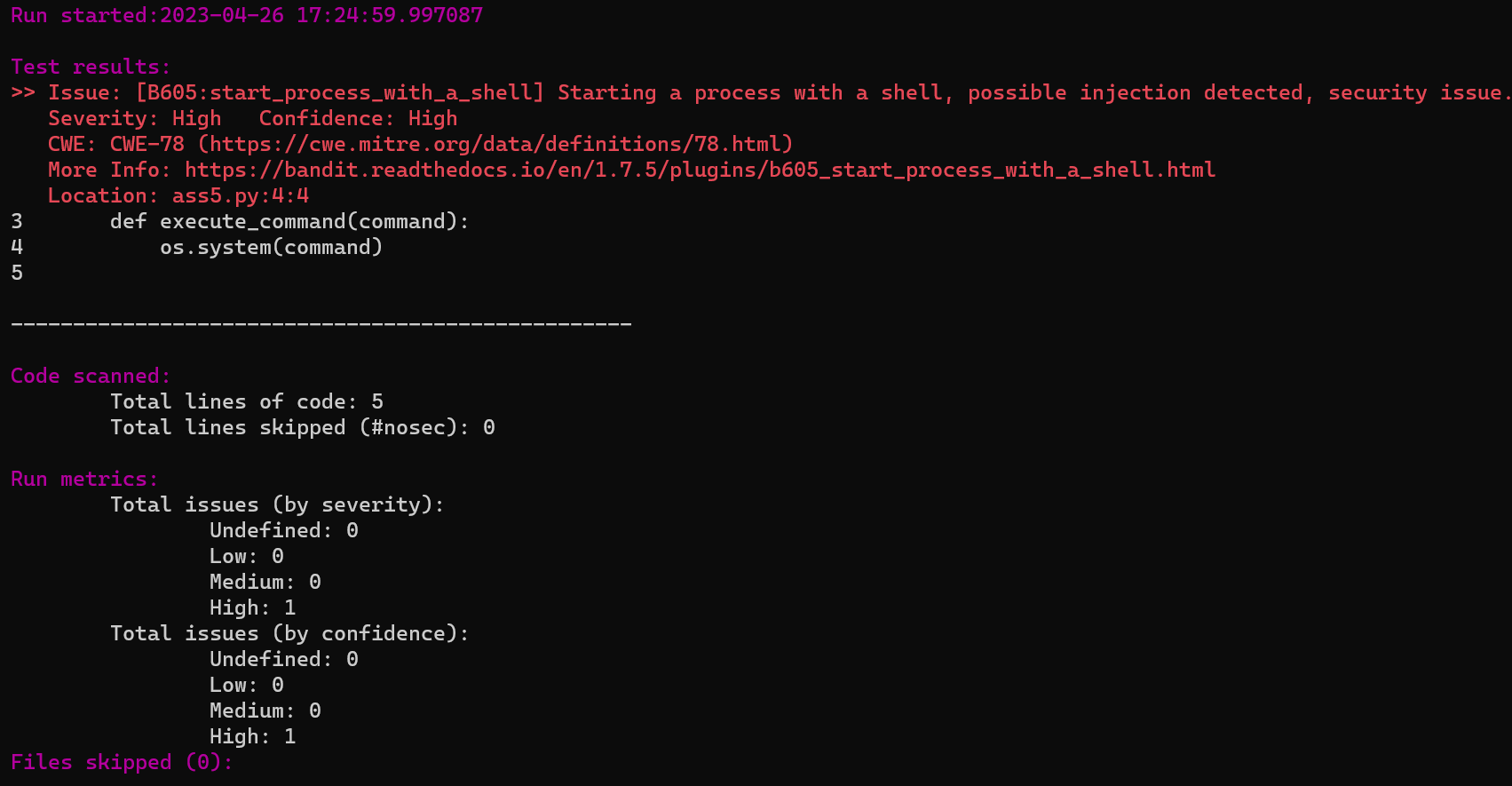
1. Number of hits - 2
2. Potential risks - 2
3. Suggested alternatives for these risks –i. check when opening a file, can an attacker redirect it ii. dont use statically sized array
4. Updating the code as per suggestions – using arr[] instead of arr[10] and using fopen\_s instead of fopen.
5. Re-execution of code after updating the changes.



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

INSTALL BANDIT





After performing analysis create a report which will contain below points

1. Number of hits - 1
2. Potential risks - high
3. Suggested alternatives for these risks
4. Updating the code as per suggestions

execute\_command function uses subprocess.call to run the command specified by the user instead of os.system

1. Re-execution of code after updating the changes.

