Patrick Walsh

Graded Assignment: Homework 4

University of Maryland Global Campus

SDEV 325 – Detecting Software Vulnerabilities

# Executive Summary

Demonstrating insecure interaction between components for CWE-250: Execution with Unnecessary Privileges and CWE-732: Incorrect Permission Assignment for Critical Resource. Each example provides an overview of the security weakness along with code snippets to present the weakness in how it appears. The examples also provide mitigation techniques along with modified code snippets showing where the weakness has been mitigated. Both programs are written in Python.

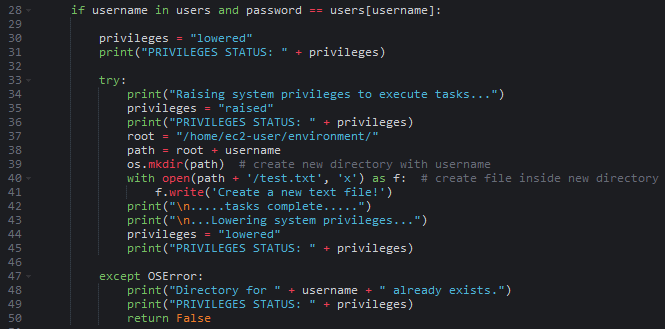
# Example 1 – CWE-250: Execution with Unnecessary Privileges

## Overview

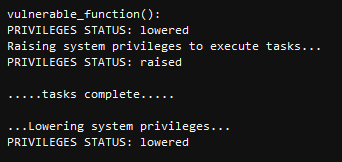
This first example demonstrates a program executing unnecessary privileges and failing to lower privileges under certain situations. If privileges are not lowered as expected, then the program will stay in a state of raised privilege, leaving the program vulnerable to potential exploit and abuse of privileges (). The program is written in Python and simulates a user having to enter a username and password to gain access. If the username exists in the system and the password is correct, then the program executes some file tasks which include creating a new directory and creating a new file in that directory.

## Analysis of the Vulnerability

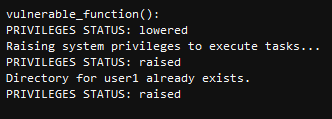
Once a user has been verified with the correct username and password, the program raises system privileges (lines 34-36) in order to create a new directory and create a new file for the user (lines 39-41). However, because of the way the program is written, the program keeps system privileges raised until all tasks have been carried out (lines 43-45).



Below is the expected output when the program runs properly:

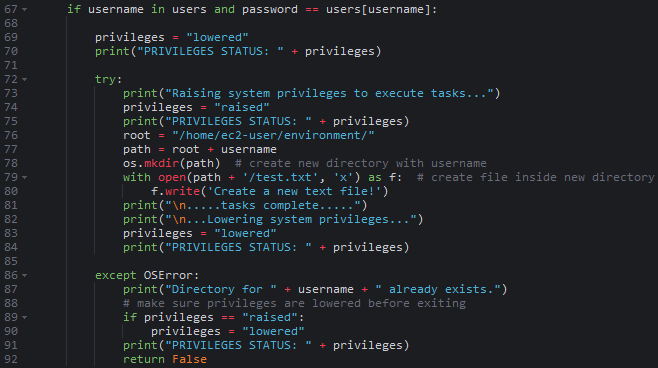


The problem with this setup is when exceptions occur. For example, if the user directory already exists, the program will encounter an OSError when it reaches line 39, causing the except block to be executed (lines 47-50). This causes the program to exit the function with privileges still raised, creating a potential security threat. See output below:

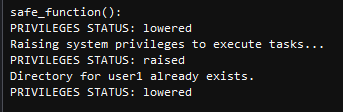


## Mitigation

In order to ensure system privileges are lowered before exiting the function, the function has been modified to check the privilege level upon encountering an OSError (lines 89-91).



This modified code ensures that the program safely exits without leaving the system vulnerable due to excess privileges. See below output:



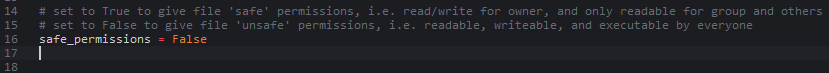
# Example 2 – CWE-732: Incorrect Permission Assignment for Critical Resource

## Overview

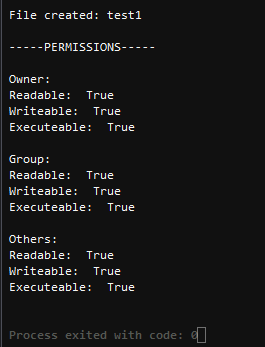
This security weakness occurs when a security-critical resource is set with permissions that allow it to be modified by unintended users (CWE, 2008). This program demonstrates proper and improper permissions assignment for a file. The program is written in Python and lets the user create a new file and set permissions. In one instance the permissions it sets are too wide, making the file potentially vulnerable to manipulation or execution by a malicious or careless user. The program also provides an example of more strict file permissions to prevent abuse or misuse.

## Analysis of the Vulnerability

The program calls a function which creates a new file and sets file permissions. For demonstration purposes, the program allows the user if they want to set the file with ‘safe’ permissions or not.

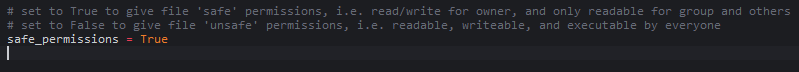


When the file is set with unsafe permissions, it allows everyone to read, write, and execute the file. This could pose a security risk, especially if the file contains sensitive or critical resources (CWE, 2008). Below is the output when the file is created with unsafe permissions:

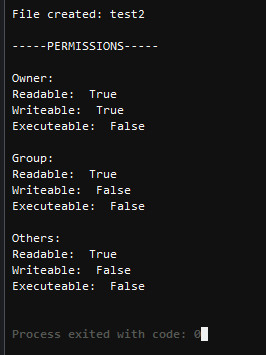


## Mitigation

In order to avoid setting permissions that are too open, the program allows the user to modify the permission settings. When the permissions are set to ‘safe’ it gives the user read write access, but only read access to group and others. This of course, can be tailored to meet the security requirements of the program.



Now when a file is created, it is set with more strict permissions to prevent misuse or abuse. Below is the output:



# References

Common Weakness Enumeration (CWE). (2006, July 19). CWE - CWE-250: Execution with Unnecessary Privileges (4.4). CWE Top 25 Most Dangerous Software Weaknesses. https://cwe.mitre.org/data/definitions/250.html

Common Weakness Enumeration (CWE). (2008, September 8). CWE - CWE-732: Incorrect Permission Assignment for Critical Resource (4.4). CWE Top 25 Most Dangerous Software Weaknesses. https://cwe.mitre.org/data/definitions/732.html