Absolute Path Traversal due to incorrect use of `send_file` call (#… …2025)

Browse files

A path traversal attack (also known as directory traversal) aims to access files and directories that are stored outside the web root folder. By manipulating variables that reference files with “dot-dot-slash (../)” sequences and its variations or by using absolute file paths, it may be possible to access arbitrary files and directories stored on file system including application source code or configuration and critical system files. This attack is also known as “dot-dot-slash”, “directory traversal”, “directory climbing” and “backtracking”.

Common Weakness Enumeration category
CWE - 36

Root Cause Analysis

The `os.path.join` call is unsafe for use with untrusted input. When the `os.path.join` call encounters an absolute path, it ignores all the parameters it has encountered till that point and starts working with the new absolute path. Please see the example below.

```
>>> import os.path
>>> static = "path/to/mySafeStaticDir"
>>> malicious = "/../../../../../etc/passwd"
>>> os.path.join(t,malicious)
'/../../../../../etc/passwd'
```

Since the "malicious" parameter represents an absolute path, the result of `os.path.join` ignores the static directory completely. Hence, untrusted input is passed via the `os.path.join` call to `flask.send_file` can lead to path traversal attacks.

In this case, the problems occurs due to the following code :
https://github.com/ganga-devs/ganga/blob/0c0f9e33b36ee7ead0855f1464f8d4efad26bdbc/ganga/GangaGUI/gui/routes.py#L671

Here, the `path` parameter is attacker controlled. This parameter passes through the unsafe `os.path.join` call making the effective directory and filename passed to the `send_file` call attacker controlled. This leads to a path traversal attack.

Proof of Concept

The bug can be verified using a proof of concept similar to the one shown below.

```
curl --path-as-is 'http://<domain>/job/<int:job_id>/browse///../../../../etc/passwd"'
```

Remediation

This can be fixed by preventing flow of untrusted data to the vulnerable `send_file` function. In case the application logic necessiates this behaviour, one can either use the `werkzeug.utils.safe_join` to join untrusted paths or replace `flask.send_file` calls with `flask.send_from_directory` calls.

Common Vulnerability Scoring System Vector

The attack can be carried over the network. A complex non-standard configuration or a specialized condition is not required for the attack to be successfully conducted. There is no user interaction required for successful execution. The attack can affect components outside the scope of the target module. The attack can be used to gain access to confidential files like passwords, login credentials and other secrets. It cannot be directly used to affect a change on a system resource. Hence has limited to no impact on integrity. Using this attack vector a attacker may make multiple requests for accessing huge files such as a database. This can lead to a partial system denial service. However, the impact on availability is quite low in this case. Taking this account an appropriate CVSS v3.1 vector would be

(AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:L)[https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator?vector=AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:L&version=3.1]

This gives it a base score of 9.3/10 and a severity rating of critical.

References
* [OWASP Path Traversal](https://owasp.org/www-community/attacks/Path_Traversal)
* github/securitylab#669

This bug was found using *[CodeQL by Github](https://codeql.github.com/)*

Co-authored-by: Porcupiney Hairs <porucpiney.hairs@protonmail.com>

develop (#2025)
8.6.1 ... 8.5.10

porcupineyhairs and Porcupiney Hairs committed on May 9

1 parent 0c0f9e3 commit 730e7aba192407d35eb37dd7938d49071124be8c

Showing **1 changed file** with **2 additions** and **2 deletions**.

Split Unified

4 🟩🟩🟥🟥⬜ ganga/GangaGUI/gui/routes.py

```
  import sys
  import datetime
  from functools import wraps
- from werkzeug.utils import secure_filename
+ from werkzeug.utils import secure_filename, safe_join
  from werkzeug.security import generate_password_hash, check_password_hash
  from flask import Flask, request, jsonify, render_template, flash, redirect, url_for, ses
```

```
 from flask_login import login_user, login_required, logout_user, current_user, UserMixin
         return redirect(url_for("job_page", job_id=job_id))

     # Join the base and the requested path
-    abs_path = os.path.join(job_base_dir, path)
+    abs_path = safe_join(job_base_dir, path)

     # URL path variable for going back
     back_path = os.path.dirname(abs_path).replace(job_base_dir, "")
```

0 comments on commit `730e7ab`

Please sign in to comment.