


Merge pull request #2 from porcupineyhairs/FixPathInjection

[Browse files](#)

Fix Path Traversal Vulnerability

 master (#2)



olmax99 committed on May 25

2 parents 0e8a6bd + d49c43a commit 28c985d712d7ac26893433e8035e2e3678fcae9f

Showing 2 changed files with 44 additions and 2 deletions.

Split

Unified

42  .committer.template.md 

```
...      @@ -0,0 +1,42 @@
1      + # Absolute Path Traversal due to incorrect use of `send_file` call
2      +
3      + 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".
4      +
5      + ## Root Cause Analysis
6      +
7      +
8      + 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.
9      + ```
10     + >>> import os.path
11     + >>> static = "path/to/mySafeStaticDir"
12     + >>> malicious = "../.../../.../etc/passwd"
13     + >>> os.path.join(t,malicious)
14     + '../.../../.../etc/passwd'
15     + ```
16     + 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.
17
```

```

18 +
19 + In this case, the problems occurs due to the following code :
20 + https://github.com/olmax99/helm-flask-
    celery/blob/0e8a6bdc4fa5b35fbdda18b27c8e768df8a9bb3c/webapiservice/flaskapi/core/app_setup
    .py#L83
21 +
22 + 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.
23 +
24 +
25 + ## Proof of Concept
26 +
27 + The bug can be verified using a proof of concept similar to the one shown below.
28 +
29 +
30 + ```
31 + curl --path-as-is 'http://<domain>../../../../etc/passwd"'
32 + ```
33 + ## Remediation
34 +
35 + This can be fixed by preventing flow of untrusted data to the vulnerable `send_file`
    function. In case the application logic necessitates this behaviour, one can either use the
    `flask.safe_join` to join untrusted paths or replace `flask.send_file` calls with
    `flask.send_from_directory` calls.
36 +
37 +
38 + ## References
39 + * [OWASP Path Traversal](https://owasp.org/www-community/attacks/Path_Traversal)
40 + * github/securitylab#669
41 +
42 + ### This bug was found using [CodeQL by Github](https://codeql.github.com/)

```

 4  webapiservice/flaskapi/core/app_setup.py 

```

2      2      import socket
3      3
4      4      from flask import current_app, make_response, request
5      - from flask import send_file, render_template
    5      + from flask import send_file, render_template, safe_join
6      6
7      7      from .redis_config import DecodedRedis
8      8      from .redis_conn import FlaskRedis
78     78     def route_frontend(path):
79     79         # ...could be a static file needed by the front end that
80     80         # doesn't use the `static` path (like in `<script src="bundle.js">`)
81     - file_path = os.path.join(current_app.template_folder, path)
    81     + file_path = safe_join(current_app.template_folder, path)

```

82	82	<code>if os.path.isfile(file_path):</code>
83	83	<code> return send_file(file_path)</code>
84	84	<code># ...or should be handled by the SPA's "router" in front end</code>

0 comments on commit `28c985d`

Please [sign in](#) to comment.