Stenographic File Integrity Checker

Name: Prateek(166), Dev Sharma(179)

1. Demo

The tool demonstrates hiding cryptographic hashes of files inside cover files (such as audio WAV files) using steganography. Below is the demonstration flow:

1. Generate SHA256 hash of the file (e.g., report.pdf).

2. Embed this hash into a cover file (e.g., cover.wav).

3. Extract the hidden hash from the stego file (stego.wav).

4. Verify the integrity by comparing the stored hash with the current hash.

Demonstration of modified file detection:
- When the original file is altered, the current hash differs from the stored hash.
- The verification process flags the file as tampered.

2. Report

Embedding and Extraction Logic

The embedding process converts the hash of the target file into binary and hides it within the least significant bits of the cover file's audio samples. During extraction, the process reads the modified LSBs, reconstructs the binary data, and recovers the original hash for comparison.

Potential Limitations

- Limited capacity: The cover file size restricts how many hashes or files can be stored.
- Format dependency: Current implementation supports WAV audio files; other formats may require different encoding strategies.
- Susceptibility to recompression: If cover files are re-encoded or compressed, hidden data may be lost or corrupted.

Ideas for Improvement or Scaling

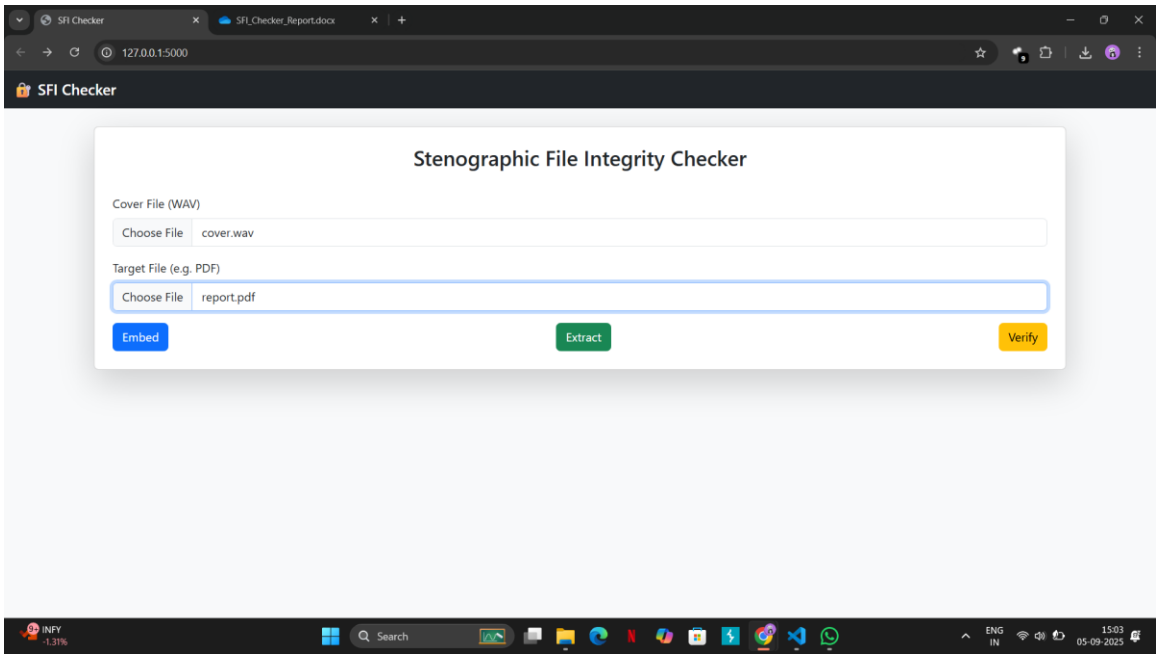
- Extend support to images (PNG, BMP) and other media formats.
- Add encryption before embedding for confidentiality of hashes.
- Enable batch processing of multiple files with a manifest.
- Develop a graphical web interface for user-friendly interaction.
- Explore robust steganography methods resistant to lossy transformations.

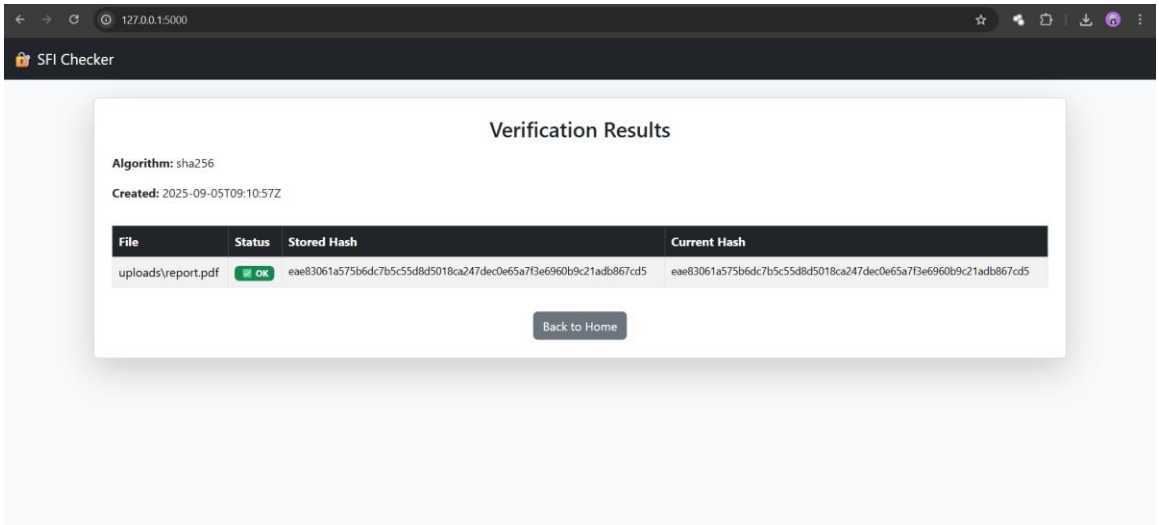
3. Unit Tests

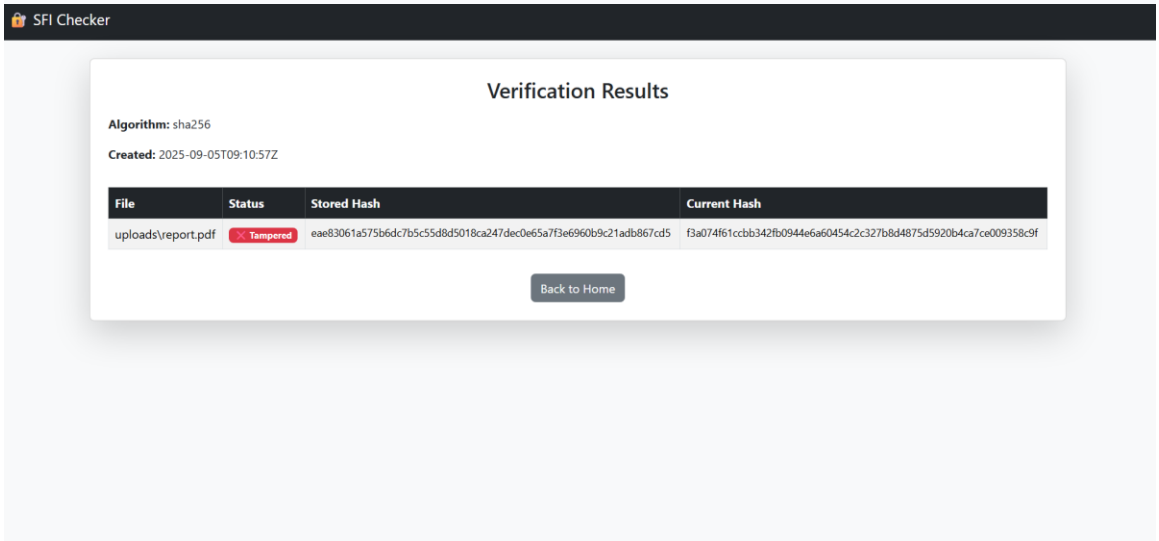
Basic unit tests were implemented to ensure correctness of core functions:

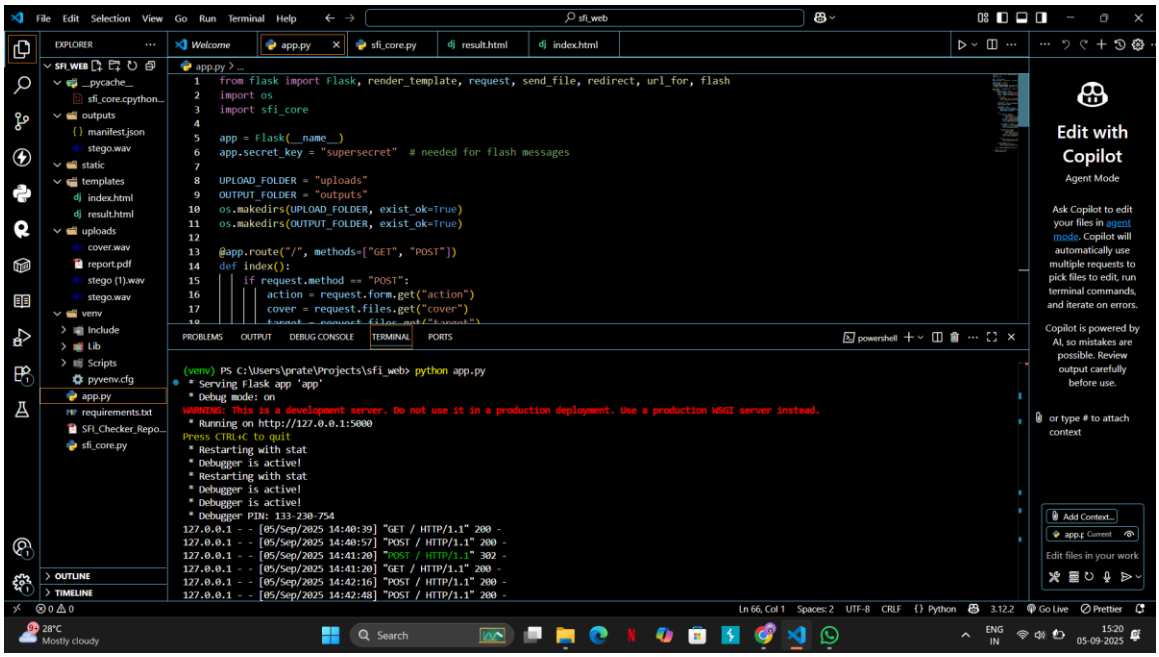
- Test embedding: Verify that hashes can be correctly embedded in cover files.
- Test extraction: Ensure extracted hashes match the original hashes embedded.
- Test verification: Confirm that unchanged files pass verification and modified files are flagged.

4. Proof of concept




127.0.0.1:5000
SFI Checker
Verification Results
Algorithm: sha256
Created: 2025-09-05T09:10:57Z
File
Status
Stored Hash
Current Hash
uploads\report.pdf
OK
eae83061a575b6dc7b5c55d8d5018ca247dec0e65a7f3e6960b9c21adb867cd5
eae83061a575b6dc7b5c55d8d5018ca247dec0e65a7f3e6960b9c21adb867cd5
Back to Home


SFI Checker
Verification Results
Algorithm: sha256
Created: 2025-09-05T09:10:57Z
File
Status
Stored Hash
Current Hash
uploads\report.pdf
Tampered
eae83061a575b6dc7b5c55d8d5018ca247dec0e65a7f3e6960b9c21adb867cd5
f3a074f61ccbb342fb0944e6a60454c2c327b8d4875d5920b4ca7ce009358c9f
Back to Home


File Edit Selection View Go Run Terminal Help
sfi_web
EXPLORER
Welcome
app.py
sfi_core.py
result.html
index.html
app.py > ...
1 from flask import Flask, render_template, request, send_file, redirect, url_for, flash
2 import os
3 import sfi_core
4
5 app = Flask(__name__)
6 app.secret_key = "supersecret" # needed for flash messages
7
8 UPLOAD_FOLDER = "uploads"
9 OUTPUT_FOLDER = "outputs"
10 os.makedirs(UPLOAD_FOLDER, exist_ok=True)
11 os.makedirs(OUTPUT_FOLDER, exist_ok=True)
12
13 @app.route("/", methods=["GET", "POST"])
14 def index():
15 if request.method == "POST":
16 action = request.form.get("action")
17 cover = request.files.get("cover")
SFI_WEB
__pycache__
sfi_core.cpython...
outputs
manifest.json
stego.wav
static
templates
index.html
result.html
uploads
cover.wav
report.pdf
stego (1).wav
stego.wav
venv
Include
Lib
Scripts
pyvenv.cfg
app.py
requirements.txt
SFI_Checker_Repo...
sfi_core.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
(venv) PS C:\Users\prate\Projects\sfi_web> python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Restarting with stat
* Debugger is active!
* Debugger is active!
* Debugger PIN: 133-230-754
127.0.0.1 - - [05/Sep/2025 14:40:39] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [05/Sep/2025 14:40:57] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [05/Sep/2025 14:41:20] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [05/Sep/2025 14:41:20] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [05/Sep/2025 14:42:16] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [05/Sep/2025 14:42:48] "POST / HTTP/1.1" 200 -
Edit with Copilot
Agent Mode
Ask Copilot to edit your files in agent mode. Copilot will automatically use multiple requests to pick files to edit, run terminal commands, and iterate on errors.
Copilot is powered by AI, so mistakes are possible. Review output carefully before use.
or type # to attach context
Add Context...
app.py Current
Edit files in your work
Ln 66, Col 1 Spaces: 2 UTF-8 CRLF Python 3.12.2 Go Live Prettier
28°C
Mostly cloudy
Search
ENG
IN
15:20
05-09-2025