Proof of Concept (PoC) Document – Homoglyph Domain Detector

---

1. Title

Homoglyph Domain Detector

Detecting Unicode-Based Lookalike Domain Attacks

---

2. Introduction

Phishing and social engineering attacks have evolved over time, and attackers now use **Unicode homoglyphs** — characters that *look like* common English letters but are actually different. These characters can deceive even the most aware users by mimicking trusted domains like google.com, facebook.com, etc. This PoC explains a basic but effective script that detects such suspicious domain names.

---

3. Objective

To build a simple tool that takes a domain name as input and detects whether it contains **Unicode homoglyphs** intended to deceive users by mimicking well-known websites.

---

4. Problem Statement

Users often click on domain names that **appear visually correct** but are actually different due to **Unicode manipulation**. For example:

Fake Domain	Real Domain	Difference
google.com	google.com	g = Unicode U+0261, not g
facebook.com	facebook.com	o = Cyrillic o, not English o
arnazon.com	amazon.com	r and n resemble m visually

Such domains are used in phishing to steal credentials, inject malware, or perform scams.

---

5. Why This Tool Is Needed

- Unicode homoglyph attacks are **invisible to the human eye**.
- Even careful users can fall for such traps.
- **Browsers treat Unicode domains as legitimate**.
- Common anti-virus tools don't always flag them.

Therefore, there's a need for a lightweight tool to **flag such lookalike domains**.

---

6. How It Works (Overview)

This tool:

1. **Loads a whitelist** of known trusted domains (e.g., google.com)
2. **Normalizes** the test domain by replacing suspicious Unicode characters with their real ASCII equivalents.
3. **Compares** the normalized domain with trusted domains.
4. **Flags** any domain that looks too similar.

---

7. Approach / Methodology

Step-by-Step:

1. **Define homoglyph mapping**
   Map suspicious Unicode characters (e.g., Cyrillic, Greek) to ASCII.
2. **Normalize domain**
   Replace homoglyphs with their actual characters.
3. **Compare with whitelist**
   Use Python's difflib.SequenceMatcher to compare similarity.
4. **Flag suspicious domains**
   If similarity > 0.8, the domain is flagged as **potentially fake**.

---

8. Code Summary

The project is built using **Python** and includes:

- homoglyph_detector.py: Main script to process domains
- safe_domains.txt: Trusted domain names
- test_urls.txt: Malicious-looking domains to be tested
- output.txt: Final output showing suspicious or safe results

Used Libraries:

- difflib (built-in): to compute similarity
- No external dependencies

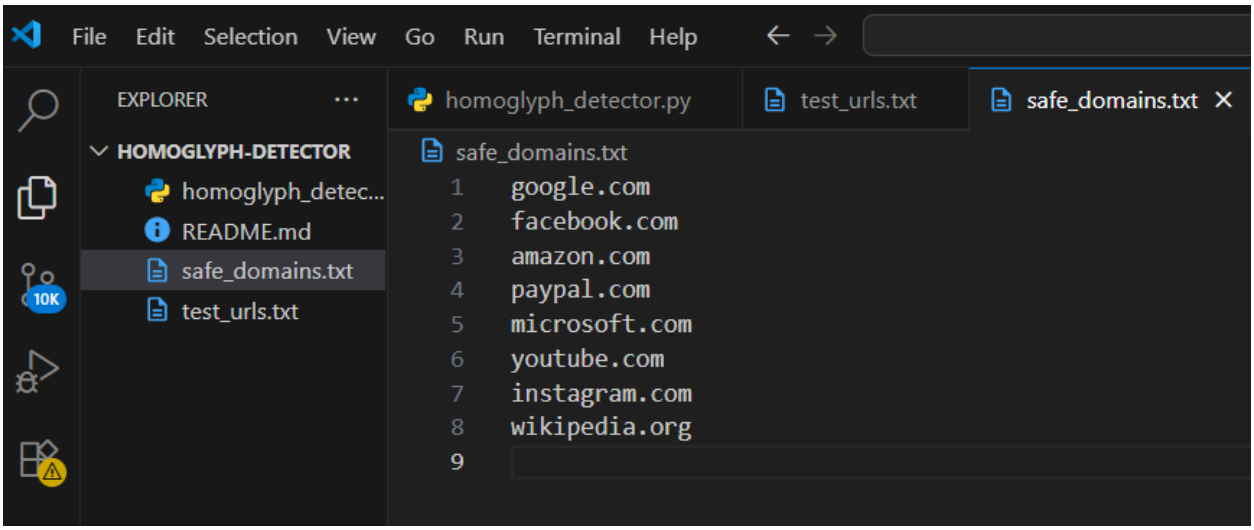
Screenshot Here: terminal_output.png

(Insert your terminal output screenshot here)

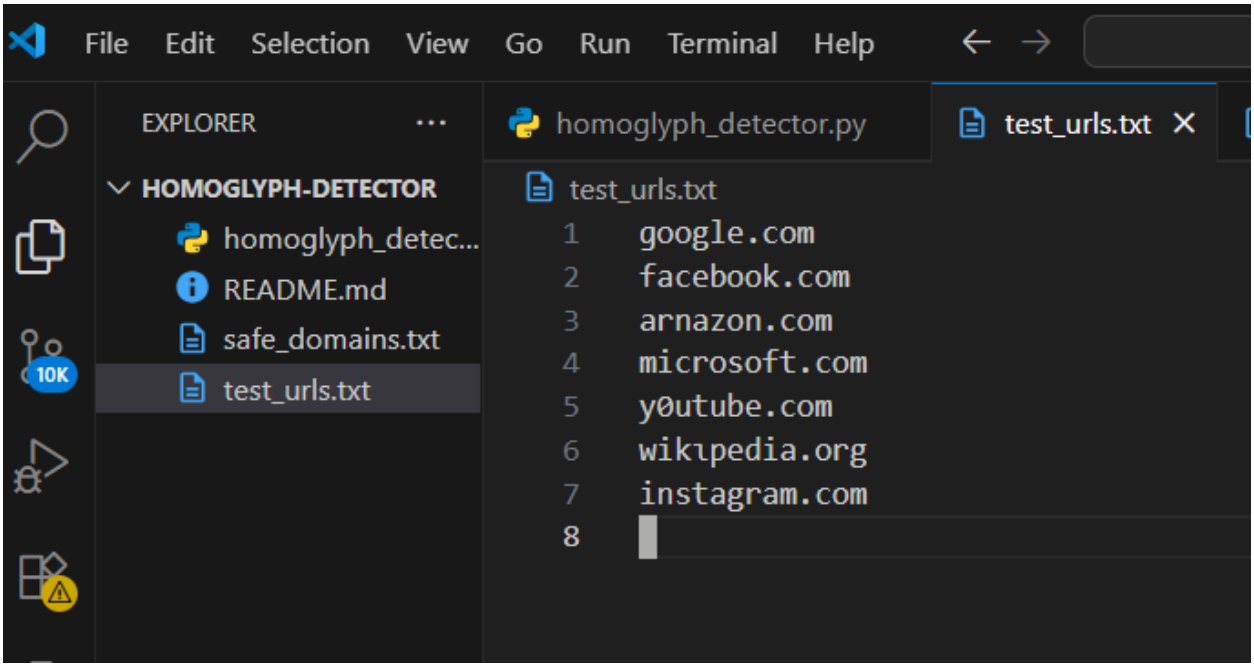
---

9. Input Files

safe_domains.txt

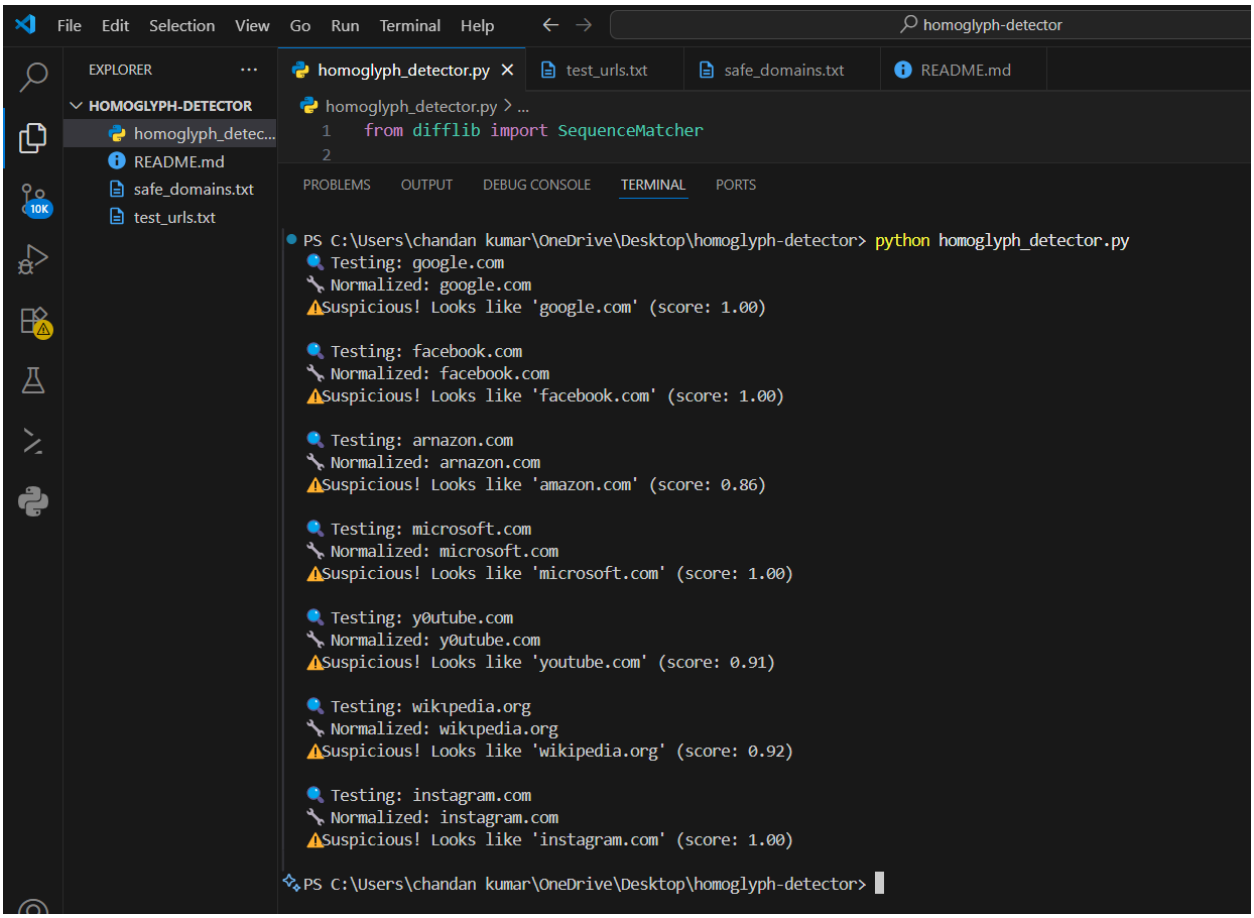


test_urls.txt



---

10. Output Sample (from output.txt)



---

11. Conclusion

This PoC demonstrates how simple Unicode-based homoglyph attacks can be used to trick users. Using a basic Python script, we can proactively detect such phishing domains and raise alerts before the user interacts with them.

This tool is:

- Lightweight
- Extendable
- Effective for personal, enterprise, or email security filters

NAME:- Chandan kumar

Intern I’d :- 183