🔐 Clickjacking Detection Tool — JackReaper

📌 Overview

JackReaper is a Python-based tool that checks whether a website is vulnerable to **Clickjacking attacks** by analyzing security headers such as:

- `X-Frame-Options`
- `Content-Security-Policy (frame-ancestors)`

It also saves the results to a file, including detailed vulnerability explanation and recommended mitigation steps.

🚀 Features

- ✅ Header Analysis (`X-Frame-Options`, `CSP`)
- ✅ Automatic detection of protection levels
- ✅ Generates report with:
  - Vulnerability status
  - Description of Clickjacking
  - Exploitation examples
  - Mitigation techniques
- ✅ Handles Unicode and file encoding errors

📁 Directory Structure

```
clickjacking/
│
├── JackReaper.py        # Main tool script
├── results.txt      # Auto-generated report file
```

⚙️ Requirements

- Python 3.6+
- Libraries:

  ```
  pip install requests
  ```

💻 Usage

```
python JackReaper.py
```

Example Input:

```
Enter the target website URL (with https://): https://www.example.com
```

🧠 How It Works

1. Sends a `GET` request to the target with a custom User-Agent.
2. Extracts headers:
   - `X-Frame-Options` (XFO)
   - `Content-Security-Policy` (CSP)
3. Analyzes:
   - `DENY` or `SAMEORIGIN` in XFO
   - `frame-ancestors` directive in CSP
4. If **no strong protection**, it:
   - Flags as vulnerable
   - Adds PoC explanation
   - Recommends mitigations
5. Saves detailed output to `clickjacking_report.txt`

📄 Sample Output

```
[+] URL: https://www.mlrit.ac.in
[+] X-Frame-Options: None
[+] Content-Security-Policy: None

❌ No X-Frame-Options or CSP frame-ancestors headers found.
🔴 Possible Clickjacking vulnerability detected.

📄 Report saved as: clickjacking_report.txt
```

🛡️ What Is Clickjacking?

Clickjacking is a UI redress attack where a malicious site tricks users into clicking on something different than they perceive, typically by loading the target site in a transparent `<iframe>`.

Example:

```
<iframe src="https://target.com" style="opacity:0.1;position:absolute;"></iframe>
```

Attacker Goals:

- Trick users into performing unintended actions
- Trigger financial transactions or setting changes
- Hijack sessions or permissions

🧯 Mitigation

Add the following headers in HTTP response:

Option 1: Using `X-Frame-Options`

```
X-Frame-Options: DENY
```

or

```
X-Frame-Options: SAMEORIGIN
```

Option 2: Using `Content-Security-Policy`

```
Content-Security-Policy: frame-ancestors 'none';
```

📘 License

MIT License — Free to use for educational and professional security auditing purposes.
