Password Strength Analyzer & Wordlist Generator

Introduction

In today's digital age, password security is crucial to protect sensitive data and online identities. This project aims to provide users with a simple, interactive tool that evaluates password strength and generates custom wordlists based on personal information—helping raise awareness about weak passwords and potential vulnerabilities.

Abstract

This project combines password strength analysis with a personalized wordlist generator to demonstrate how weak or predictable passwords can be easily guessed using common attack patterns. By leveraging user input such as names or birth years, the tool creates realistic wordlists that can be used in penetration testing or educational simulations. A user-friendly GUI ensures accessibility for non-technical users.

Tools Used

1. Python 3.11 – Served as the main programming language, chosen for its simplicity, versatility, and rich ecosystem of libraries suited for both GUI and backend logic.

2. Tkinter and ttk – Used for creating the graphical user interface. `tkinter` provided the core windowing functionality, while `ttk` was used to enhance the visual design with themed widgets.

3. zxcvbn – Integrated to analyze password strength. This library estimates how long a password might take to crack and offers useful feedback based on common patterns.

4. NLTK (Natural Language Toolkit) – Considered as an optional enhancement to generate more intelligent wordlists using basic natural language processing techniques.

5. argparse – Used optionally to enable a command-line interface for users who prefer terminal-based interaction instead of the GUI.

6. Visual Studio Code (VS Code) – Employed as the development environment due to its lightweight nature, extensions, and effective debugging tools.

Steps Involved in Building the Project

1. Environment Setup: Created a virtual environment and installed required libraries (`zxcvbn`, `nltk`, `tkinter`, etc.).

2. Password Strength Function: Used `zxcvbn` to analyze user input and return a strength score, feedback, and estimated crack times.

3. Wordlist Generator: Allowed user input (e.g., name, pet, birthday), applied transformations like adding numbers, using leetspeak (e.g., `a->@`, `e->3`), and stored combinations in a list.

4. Export Utility: Saved the generated list to a `.txt` file under an `/output` folder for use in ethical hacking tools like Hydra or John the Ripper.

5. GUI Development: Created an interactive GUI using Tkinter and `ttk`, styled with emojis and consistent padding for a clean look.

6. Testing: Validated all features using different passwords and inputs to ensure accurate analysis and wordlist output.

Conclusion

This project successfully demonstrates how easily personal information can be transformed into potential passwords, emphasizing the risks of weak and predictable password choices. By combining a real-time strength analyzer with a personalized wordlist generator, the tool not only educates users about password vulnerabilities but also simulates common attack strategies used by hackers. The user-friendly interface ensures accessibility for beginners, while the backend logic remains relevant for cybersecurity learners and professionals. Overall, this tool serves as both an awareness-raising application and a practical resource for understanding the importance of strong, unique passwords in modern digital security.