Password Strength Analyzer & Custom Wordlist Generator

By Tushar Karne

1. Introduction

In today's digital age, weak passwords are a major security risk. Attackers often exploit predictable patterns, reused passwords, or easily guessable personal information. The **Password Strength Analyzer & Custom Wordlist Generator** project addresses this by providing a tool that evaluates password robustness and generates tailored wordlists for authorized security testing. This project emphasizes ethical usage, aiming to improve security awareness and testing practices.

---

2. Abstract

This project delivers a Python-based solution to analyze passwords and generate targeted wordlists. It integrates both CLI and GUI interfaces for flexibility. Password analysis is performed using entropy calculations and the **zxcvbn** library, providing a detailed evaluation including score, entropy, estimated crack time, and feedback. The wordlist generator produces custom combinations from user-provided tokens, incorporating leetspeak, common suffixes, and years to mimic realistic password patterns. The tool is intended for penetration testing, recovery of forgotten passwords, and security awareness.

---

3. Tools Used

- **Python 3.x** – Core programming language.
- **zxcvbn-python** – Password strength evaluation library.
- **NLTK** – Token normalization and word handling.
- **argparse** – Command-line interface support.
- **Tkinter** – GUI implementation.
- **Text/CSV handling** – Exporting generated wordlists.
- **Git/ZIP** – Version control and distribution.

---

4. Steps Involved in Building the Project

Project Setup

- Created a Python package structure (`pw_tool`) with modules for analysis, wordlist generation, CLI, and GUI.
- Added dependencies (`zxcvbn-python`, `nltk`) in `requirements.txt`.
- Prepared README and example wordlist for usage guidance.

Password Analysis Module

- Implemented entropy-based calculations for custom evaluation.
- Integrated **zxcvbn** to enhance accuracy and provide actionable feedback.
- Calculated estimated crack times based on character sets and entropy.
- Developed output formats for CLI and GUI presentation.

Wordlist Generation Module

- Designed a flexible system to accept tokens (names, dates, pets) and produce permutations.
- Added **leetspeak variants**, common suffixes, and year appendages to mimic real-world password patterns.
- Enabled output in `.txt` format for integration with password testing tools.
- Ensured filtering of too-short or invalid entries to maintain quality.

CLI Implementation

- Used `argparse` to provide `analyze` and `gen` subcommands.
- Supported multiple tokens, file output, and optional features like disabling leetspeak or years.
- Printed structured results including score, entropy, crack time, and feedback.

GUI Implementation

- Developed a Tkinter interface for user-friendly interaction.
- Password analysis section with entry field, button, and result display.
- Wordlist generation section with token input, output file selection, and status display.
- Ensured real-time feedback and error handling.

Testing and Validation

- Tested multiple password scenarios to ensure correct scoring and feedback.
- Validated generated wordlists for completeness and pattern coverage.
- Confirmed CLI and GUI produce consistent results.

---

5. Conclusion

The **Password Strength Analyzer & Custom Wordlist Generator** provides a practical and flexible solution to improve password security awareness. By combining entropy analysis, real-world pattern generation, and user-friendly interfaces, it supports both learning and authorized security testing. The project demonstrates how automated tools can enhance cybersecurity practices while promoting ethical usage. Future improvements may include integration with password managers, more advanced attack pattern modelling, and support for cloud-based analysis.