Password Strength Analyzer & Wordlist Generator – Project Report

Abstract

This project presents a Python-based GUI tool that analyzes password strength and generates personalized wordlists. Using the *zxcvbn-python* library, the system evaluates password entropy, pattern weaknesses, and estimated cracking time. The wordlist generator creates customized password lists using user-provided details. This project helps users understand common password vulnerabilities and how targeted wordlists are formed during cyberattacks.

Introduction

Passwords continue to be the most widely used authentication method, yet many remain weak or predictable. Attackers use automated techniques like brute-force and dictionary attacks to exploit such vulnerabilities.
This project aims to educate users about password security by providing:

- A strength analyzer that identifies weaknesses,
- A personalized wordlist generator to demonstrate how attackers guess passwords,
- A simple GUI interface suitable for beginners in cybersecurity.

The tool combines practical password analysis with real cybersecurity concepts.

Tools Used

- **Python 3.11** – Programming language.
- **Tkinter** – Used for building the graphical interface.
- **zxcvbn-python** – Library for password strength estimation.
- **Virtual Environment (venv)** – For dependency isolation.
- **Custom Python logic** – For generating wordlists with variations.

Steps Involved in Building the Project

1. Setting Up the Environment

A virtual environment was created using `venv`, and required packages such as *zxcvbn-python* were installed. Installation errors were resolved by updating pip and configuring trusted hosts.

2. Designing the GUI

Tkinter's Notebook widget was used to create two main sections:

1. **Password Analyzer**
2. **Wordlist Generator**
   Modern styling and ttk widgets were applied to improve usability and layout.

3. Implementing Password Strength Analysis

The analyzer uses zxcvbn to calculate:

- Entropy score
- Weak password patterns (dictionary words, dates, sequences)
- Estimated cracking time
- Suggestions to improve strength

  The results are displayed clearly within the interface.

4. Building the Wordlist Generator

Users provide basic information such as:

- Name
- Pet name
- Birth year

  Optional features include:
- Leet-speak conversions
- Year combinations
- Special character variations

  A preview of the generated wordlist is shown and the full list can be saved as a text file.

5. Enhancing Functionality

Input validation, error handling, and additional UI polishing were added to make the app smoother and more intuitive. The placement, spacing, and theme of widgets were refined for a modern look.

Conclusion

This project successfully delivers a simple yet informative cybersecurity tool. It helps users understand password weaknesses and how attackers construct customized wordlists. The tool is useful for learning, practicing ethical hacking basics, and improving personal password habits. The project can be extended in the future with features like cloud syncing, advanced cracking simulations, or integration with hashing algorithms.