🔐 Password Strength Analyzer with Custom Wordlist Generator

1. Introduction

In today's digital world, the strength and security of passwords play a crucial role in protecting personal and organizational data. Many cyber attacks exploit weak or reused passwords, leading to unauthorized access and data breaches. This project aims to build a Python-based tool that not only evaluates the strength of passwords but also helps generate custom wordlists tailored to users' personal information. This project can aid cybersecurity learners and penetration testers by providing a simulated environment to understand password vulnerability patterns and the concept of social engineering.

2. Abstract

The project is divided into two key modules: (1) a password strength analyzer, and (2) a wordlist generator. The password analyzer uses the `zxcvbn` library to estimate password strength based on entropy, common patterns, and dictionary matches. The wordlist generator allows users to input personal information (e.g., name, birth year, pet name), and then creates variations using common password mutation techniques like capitalization, appending numbers, and leetspeak (e.g., replacing 'a' with '@' or 'e' with '3'). This wordlist can be used in ethical password cracking simulations. Additionally, a GUI interface is provided using Tkinter for ease of use.

3. Tools Used

- **Python 3.13**
- **zxcvbn** – for password strength estimation
- **NLTK** – for text manipulation (optional for future improvements)
- **Tkinter** – for optional graphical user interface
- **argparse** – for command-line usage
- **Text File Output** – for wordlist storage

4. Steps Involved in Building the Project

1. **Project Setup**
   Created a new Python project structure with dedicated files for each functionality.
2. **Password Strength Analyzer**
   - Used `zxcvbn` to analyze the strength of any given password.
   - Displayed a score between 0 (very weak) to 4 (strong).
   - Displayed suggestions for improvement (e.g., avoid common patterns).
3. **Custom Wordlist Generator**
   - Took user inputs like name, birth year, pet name, etc.
   - Created multiple password variants using techniques such as:
     - Reversing strings
     - Appending numbers or years
     - Leetspeak transformations
   - Saved the wordlist to a `.txt` file (e.g., `wordlist.txt`).
4. **GUI (Optional)**
   - Built a simple Tkinter GUI where users can:
     - Analyze password strength
     - Input custom data and generate a wordlist with one click
5. **Testing & Debugging**
   - Tested various password combinations.
   - Ensured the wordlist generator handles edge cases like empty inputs.

---

5. Conclusion

This project allowed the development of a practical and educational tool in the field of cybersecurity. By combining password strength analysis with a customized wordlist generator, the project offers value to learners, ethical hackers, and security testers. The modular approach ensures extensibility—additional features like dictionary-based attacks or password breach checking can be added later. The GUI enhances user-friendliness for non-technical users. Overall, this tool strengthens awareness about password security and demonstrates practical Python skills in cybersecurity.
