Project Report

1. Introduction

The project titled "Password Strength Analyzer with Custom Wordlist Generator" aims to enhance cybersecurity awareness by helping users evaluate the robustness of their passwords. Weak or easily guessable passwords remain a significant vulnerability in digital systems. This tool not only checks password strength but also educates users about common password creation pitfalls and how attackers exploit predictable patterns. Moreover, by generating personalized wordlists, the tool offers insight into how attackers might use social engineering and information gathering to crack passwords. This dual-purpose design makes the tool ideal for both personal use and cybersecurity training.

2. Abstract

This tool integrates a graphical user interface (GUI) with a backend that combines password analysis and custom wordlist generation. Using the zxcvbn password strength estimation library, the analyzer provides real-time feedback on password entropy, guessability, and attack time estimates. Additionally, users can input personal information such as their name, birth year, favorite pet, or other details commonly used in passwords. Based on this input, the tool uses leetspeak conversions and combinatorial logic to generate customized wordlists that mimic how attackers create targeted dictionary attacks. The goal is to highlight the risks of using predictable information in passwords and promote the use of stronger, more secure alternatives.

3. Tools Used

- Python 3.8+ - Core programming language for backend logic.
- Tkinter - Used for building the user-friendly GUI.
- zxcvbn - A powerful password strength estimation library.
- itertools - Used to generate permutations and combinations of input patterns.
- fpdf - For exporting results into a clean and professional PDF report.
- os & sys - For file handling and system-level functions.
- Regex - To validate and sanitize inputs during wordlist generation.

4. Steps Involved in Building the Project

1. Set up the Python environment and install all necessary libraries listed in requirements.txt.
2. Modularize the project using files like analyzer.py, wordlist.py, utils.py, and gui.py.

3. Build the GUI to accept user input for password analysis and wordlist customization.

4. Integrate the zxcvbn library to provide real-time, accurate feedback about password strength and vulnerability.

5. Develop the wordlist generator logic to apply transformations (e.g., leetspeak) and combine user inputs intelligently to mimic realistic attack vectors.

6. Provide export functionality to save analysis results and generated wordlists into .txt or .pdf formats.

7. Test the tool using different password scenarios and user inputs to ensure accuracy and efficiency.

5. Conclusion

The project demonstrates how software can empower users to make informed decisions about password creation and management. By providing visibility into both the strength and guessability of passwords, this tool acts as an educational resource and a training utility. It not only benefits individuals but also serves organizations looking to train employees in basic cybersecurity hygiene. Furthermore, the personalized wordlist feature mimics real-world attack strategies, helping ethical hackers and cybersecurity learners understand the importance of avoiding predictable credentials.

In the future, this tool could be enhanced with additional features such as integration with cloud-based password vaults, brute-force simulation modules, and multilingual wordlist support. Overall, this project fosters better password practices and serves as a foundation for deeper exploration into password security, cracking techniques, and digital defense mechanisms.