“Password Strength Analyzer with Custom Wordlist Generator”

Problem Statement

Weak passwords are a major cybersecurity threat. Many users rely on predictable or reused passwords, making systems vulnerable to brute-force attacks. This project addresses this issue by helping users analyze password strength and generate personalized wordlists for security testing.

Abstract

Passwords remain one of the most commonly used security measures, yet weak and predictable passwords continue to compromise digital security. This project presents a Python-based tool that not only evaluates the strength of user passwords using real-world metrics but also generates customized wordlists based on user-specific data. These wordlists are useful for ethical hacking, penetration testing, and awareness training. By combining password analysis with intelligent wordlist generation using patterns like leetspeak, date variations, and character permutations, the tool empowers users to better understand password vulnerabilities and reinforces the importance of strong password practices.

Introduction

In today’s digital era, securing user accounts and sensitive information is more important than ever. Passwords serve as the first line of defense against unauthorized access, but many users still rely on weak or easily guessable passwords. This creates vulnerabilities that can be exploited by attackers through techniques like brute-force and dictionary attacks.

To address this issue, the **Password Strength Analyzer with Custom Wordlist Generator** project was developed. This Python-based tool helps users evaluate the strength of their passwords using real-world metrics and best practices. Additionally, it can generate personalized wordlists based on user inputs such as name, date of birth, and pet names—common elements found in weak passwords. These wordlists are valuable for ethical hacking, penetration testing, and raising awareness about password security.

“Password Strength Analyzer with Custom Wordlist Generator”

Tool Used

Purpose	**Library / Tool**	**Notes**
Core language	Python	Cross-platform, batteries included
Command-line parsing	argparse	Built-in to the standard library
Strength estimation	zxcvbn-python	Data-driven password strength model

Steps Involved in Building the Project

1. Project scaffolding and CLI argument setup.
2. Integrating zxcvbn for strength checking.
3. Developing helper functions for leetspeak, casing, appending special characters/dates.
4. Using itertools for permutation and combination of base words.
5. Exporting final wordlist to .txt file.
6. Validating inputs and error handling.
7. Documenting usage and examples.