PASSWORD STRENGTH ANALYZER & CUSTOM WORDLIST GENERATOR

Submitted by: Harshita M (Rajalakshmi Institute of Technology)

Introduction

In today's digital age, password security plays a crucial role in protecting personal and organizational data. With the rise of cyberattacks, weak or predictable passwords remain one of the most common vulnerabilities exploited by malicious actors. This project addresses the problem by providing a tool that not only analyzes the strength of a given password but also helps in understanding how easily it can be guessed. Furthermore, the tool enhances cybersecurity awareness by offering users a way to generate custom wordlists based on personal information, which can be used for security auditing, ethical hacking, or penetration testing exercises.

Abstract

This project is designed to serve two core purposes: firstly, to analyze password strength using the zxcvbn library, which provides accurate scoring and feedback based on entropy and pattern recognition; and secondly, to generate a comprehensive custom wordlist by accepting user-specific details such as name, date of birth, pet name, and other keywords. These inputs are transformed into various patterns, including leetspeak substitutions, capitalization variants, and the inclusion of commonly used years like 2002 or 2023. The final tool offers dual usability modes suach as , a Command-Line Interface (CLI) built with argparse for advanced users and a user-friendly Graphical User Interface (GUI) using Tkinter for accessibility. This hybrid approach ensures the tool caters to both technical and non-technical audiences.

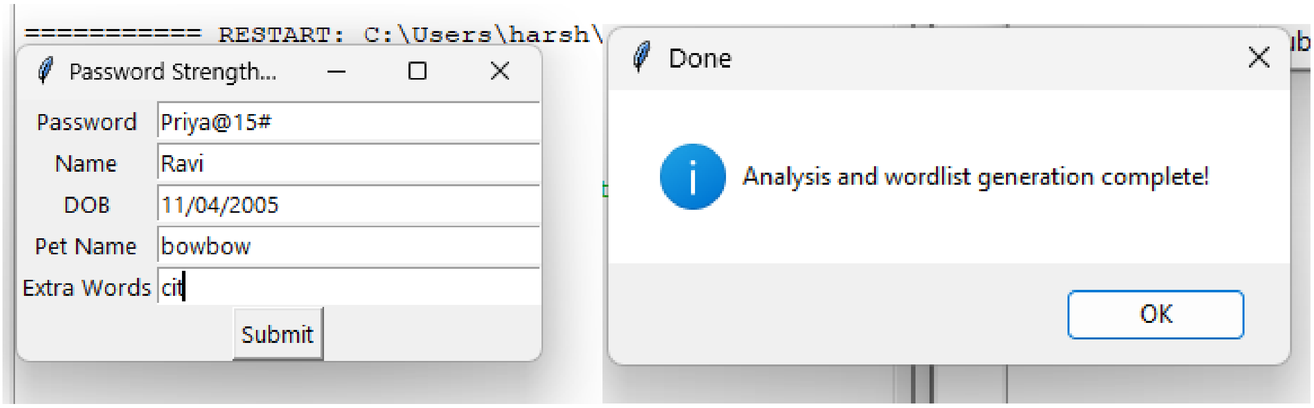
Tools Used

- Python 3 - Core programming language
- Tkinter - GUI for interactive user input
- argparse - Command-line argument parsing
- zxcvbn - Password strength estimation
- NLTK (optional) - Text processing
- itertools - Combination logic
- VS Code - IDE used for development

Steps Involved in Building the Project

The development of the project began with careful planning, during which the main objectives were clearly defined — to analyze password strength and to generate a customized wordlist. It was also decided early on to implement both a Command-Line Interface (CLI) and a Graphical User Interface (GUI) to cater to different user preferences. The password strength module was developed using the zxcvbn library, which evaluates passwords on a scale from 0 to 4 and provides meaningful feedback. This analysis was displayed in the CLI and also shown through message boxes in the GUI. The core feature, the custom wordlist generator, was designed to accept inputs such as the user's name, date of birth, pet name, and additional keywords. These inputs were transformed through multiple patterns, including lowercase, capitalized, leetspeak substitutions (like @ for a and $ for s), and by appending or prepending common years like 2023 and 2002. Duplicate entries were efficiently removed using a Python set. The CLI was built using the argparse module to handle input flags such as -n, -d, and --export, while the GUI was developed using Tkinter for an intuitive and user-friendly experience. Finally, the generated wordlist was exported and saved as a .txt file, making it readily usable for penetration testing or ethical hacking tools.

Output



```
=========== RESTART: C:\Users\harsh\
```

Password Strength...

Password: Priya@15#

Name: Ravi

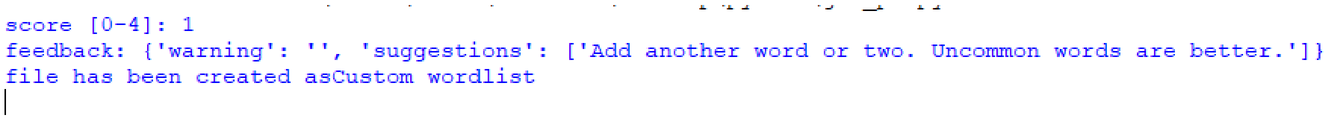
DOB: 11/04/2005

Pet Name: bowbow

Extra Words: cit

Done

Analysis and wordlist generation complete!



```
score [0-4]: 1
feedback: {'warning': '', 'suggestions': ['Add another word or two. Uncommon words are better.']}
file has been created asCustom wordlist
```

Note: Custom wordlist generated is given in the repository

Conclusion

This project successfully demonstrates a practical security tool that aids both users and ethical hackers. The GUI provides simplicity for general users, while the CLI is efficient for developers and testers. It serves as an educational utility in cybersecurity, demonstrating password analysis and brute-force testing preparation techniques.

Future Enhancements

In the future, this tool can be further improved by supporting the inclusion of special symbols and user-defined patterns to create even more complex and realistic password variations. Additionally, the generated wordlist can be enhanced by implementing frequency-based sorting, allowing the most likely password combinations to appear earlier in the list. The strength of the password can be shown GUI. Lastly, the tool could also offer intelligent password suggestions based on the strength analysis, guiding users to create stronger and more secure passwords.