PDF Cracker Tool Using Python

By Inlighn Tech

Objective:

The objective of this project is to develop a Python-based tool that cracks password-protected PDF files using dictionary-based and brute-force attacks. This project will help students understand file handling, password security, multi-threading, and automation in Python.

Project Overview:

PDF files are often protected with passwords to prevent unauthorized access. This project focuses on building a tool that attempts to decrypt such files using a wordlist or by generating possible passwords. The script uses multi-threading for faster password attempts and provides both brute-force and dictionary-based cracking methods.

How the Project Works:

1. **Input Handling: The script accepts command-line arguments specifying the PDF file, wordlist (optional), and brute-force settings.**
2. **Dictionary Attack: If a wordlist is provided, the script tries each password from the list to unlock the PDF.**
3. **Brute-Force Attack: If no wordlist is given, it generates passwords based on user-defined character sets and length constraints.**
4. **Multi-threading: The script uses `ThreadPoolExecutor` to speed up password attempts by running multiple threads in parallel.**
5. **Error Handling: The script manages cases where the PDF is invalid, the password is not found, or required arguments are missing.**

Key Concepts Covered:

- File handling in Python
- Working with PDFs using `pikepdf`
- Implementing dictionary-based and brute-force attacks
- Using multi-threading for improved efficiency

- Exception handling for robust code execution

Step-by-Step Implementation:

1. Install the required libraries: `pikepdf`, `tqdm`.
2. Create a Python script that accepts command-line arguments.
3. Implement a function to read passwords from a wordlist.
4. Implement a function to generate passwords for brute-force attacks.
5. Use multi-threading to test multiple passwords simultaneously.
6. Attempt to open the PDF file using the tested passwords.
7. Display the correct password if found; otherwise, notify the user.
8. Implement error handling for missing files and invalid inputs.

Expected Outcomes:

By completing this project, students will:

- Gain experience in password cracking techniques.
- Understand multi-threading and its impact on performance.
- Learn to work with PDFs programmatically.
- Develop a useful tool for ethical hacking and security research.

Next Steps:

Students should implement their own version of the PDF cracker tool using the outlined concepts. A detailed video lecture will be provided later to demonstrate the correct implementation and solution. This project builds foundational skills for cybersecurity and automation in Python.