Password Strength Tester

IT360 Information Assurance and Security

by

Khalil Elachkham - Mohamed Haroun Cheriha - Mahdi Laafif

April 2023

IT/FIN Junior Students

TUNIS BUSINESS SCHOOL
UNIVERSITY OF TUNIS

Tunis Business School

Ben Arous,TUNISIA.

2022-2023

Contents

Chapter 1

Introduction

The security of user passwords is a critical component of any modern web application or online service. Weak or easily guessable passwords can lead to unauthorized access, data breaches, and other security risks. Therefore, it's essential to implement password strength testing to ensure that users are creating strong and secure passwords.

The OWASP (Open Web Application Security Project) is a widely recognized authority in the field of web application security. They provide guidelines for best practices in password strength testing, including minimum password length, complexity requirements, and the avoidance of common password patterns.

In this project, we will be developing a password strength tester that adheres to the OWASP guidelines. Our system will enforce basic password rules while providing user flexibility and encouraging the use of passphrases over standard passwords. By implementing this project, we aim to enhance the security of user passwords and help prevent unauthorized access and data breaches.

In the following sections, we will provide a clear explanation of the main components and functional flow of our password strength tester, as well as discuss implementation details, user testing, limitations and future work, ethical considerations, and references.

Chapter 2

2.1 Main Concepts

- 1. Password strength testing algorithms: These are mathematical calculations that evaluate the strength of a password. These algorithms are designed to determine how difficult it would be for an attacker to guess or crack a password. OWASP recommends using a combination of techniques, such as entropy calculations, dictionary checks, and rule-based evaluations, to test password strength.

- 2. OWASP guidelines: The Open Web Application Security Project (OWASP) provides guidelines for secure password policies and testing. OWASP provides a comprehensive checklist for testing password strength, which includes many different types of tests, such as testing for common passwords, testing for character repetition, and testing for sequential characters. OWASP guidelines also emphasize the importance of educating users on the importance of strong passwords and password security best practices.

- 3. Password policies: Password policies are the set of rules and requirements that define what constitutes a strong password. OWASP recommends enforcing policies such as minimum password length, required use of special characters, and prohibiting the use of common dictionary words, personal information, or sequential characters. Password policies should be designed to minimize the likelihood of password guessing, cracking, and other types of attacks.

- 4. Passphrases: Passphrases are longer strings of words or sentences that can be used as an alternative to traditional passwords. Passphrases are generally considered more secure than passwords because they are longer and more difficult to guess or crack. OWASP recommends encouraging the use of passphrases in place of traditional passwords to improve password security.

- 5. User interfaces: User interfaces are the graphical or command-line interfaces through which users interact with the password strength tester. OWASP recommends that user interfaces should be designed to be user-friendly and provide clear feedback to the user. The feedback should indicate the strength of the password, which policies the password meets, and which policies the password does not meet.

2.2 Functional Flow

- 1. User enters password: The user inputs their password into the password strength tester.
- 2. Password is analyzed: The tester uses password policies and strength testing algorithms to analyze the password and determine its strength.
- 3. Results are displayed: The tester displays the results of the analysis to the user, typically in the form of a visual indicator (such as a color-coded bar or a thumbs-up/thumbs-down icon) and a numerical score or rating.
- 4. User is given feedback: The tester provides feedback to the user on how they can improve the strength of their password, such as by using a longer password or including more special characters.

2.3 Overview of the main existing solutions

There are several existing solutions available for testing password strength according to the OWASP standards. Here are some of the main ones:

- 1. Password Meter: This free tool checks for common patterns and dictionary words, character types, and password length to test password strength. Its main advantage is that it provides detailed feedback on password strength and areas of improvement. However, its main limitation is that it may not be as accurate in evaluating more complex passwords.
- 2. Microsoft Password Checker: This free tool tests passwords against common patterns and dictionary words, similar to the Password Meter. Its advantage is that it is easy to use

and accessible for Microsoft users. However, its main limitation is that it may not provide as detailed feedback as other solutions.

- 3. LastPass: This popular password manager includes a built-in password strength tester that evaluates passwords based on length, character complexity, and the inclusion of dictionary words and common patterns. Its advantage is that it is integrated with a password manager, making it easier to create and manage secure passwords. However, its limitation is that it requires a paid subscription to access some of its advanced features.
- 4. Dashlane: This password manager includes a password strength tester that evaluates passwords based on length, character complexity, and the inclusion of dictionary words and common patterns. Its advantage is that it is integrated with a password manager, making it easier to create and manage secure passwords. However, its limitation is that it requires a paid subscription to access some of its advanced features.
- 5. KeePass: This free and open-source password manager includes a password strength tester that evaluates passwords based on length, character complexity, and the inclusion of dictionary words and common patterns. Its advantage is that it is free and provides advanced features for managing passwords. However, its limitation is that it may require technical knowledge to use and set up.
- 6. Google Password Checkup: This tool checks for compromised passwords and offers suggestions for improving password strength. Its advantage is that it is easy to use and accessible for Google users. However, its limitation is that it may not provide detailed feedback on password strength and areas of improvement as other solutions.

Overall, these password strength testers provide valuable feedback on password strength and can help users create and manage secure passwords. However, it is important to note their limitations and choose the solution that best fits the user's needs and technical abilities.

2.4 Design of the proposed solution

2.4.1 System Architecture

The Password Strength Tester system can have the following high-level components:

- User Interface: The user interface can be designed to accept user inputs, such as a new password or an existing password to test its strength. The user interface can provide feedback to the user about the strength of the password and any areas of improvement.
- Password Strength Checker: This component can analyze the password entered by the user and check if it meets the required strength standards. This component can be developed using OWASP guidelines.
- Dictionary Lookup: This component can be used to check if the entered password is a common dictionary word or a combination of dictionary words. This check can be performed to identify if the password is susceptible to a dictionary attack.
- Brute Force Attack Simulation: This component can be used to simulate a brute-force attack on the entered password to test its strength. This simulation can be performed using a pre-defined set of rules or configurations.

2.4.2 Data Flow

The data flow between the different components can be as follows:

- User Interface: The user interface can accept the user's password input and send it to the Password Strength Checker component.
- Password Strength Checker: This component can analyze the entered password using OWASP guidelines and determine its strength. The results can be sent back to the user interface component.
- Dictionary Lookup: The Password Strength Checker component can use the Dictionary Lookup component to check if the entered password is a common dictionary word or a combination of dictionary words.

- Brute Force Attack Simulation: The Password Strength Checker component can use the Brute Force Attack Simulation component to simulate a brute-force attack on the entered password to test its strength.
- User Interface: The user interface can display the results of the password strength test to the user.

2.4.3 Messages/Events

The different messages/events that can be exchanged between the components are:

- Password Input: This message/event can be sent from the User Interface component to the Password Strength Checker component, containing the password entered by the user.
- Password Strength Result: This message/event can be sent from the Password Strength Checker component to the User Interface component, containing the results of the password strength test.
- Dictionary Lookup Result: This message/event can be sent from the Dictionary Lookup component to the Password Strength Checker component, containing the results of the dictionary lookup test.
- Brute Force Attack Result: This message/event can be sent from the Brute Force Attack Simulation component to the Password Strength Checker component, containing the results of the brute force attack simulation.

2.4.4 Design

- Class Diagram: The class diagram for the Password Strength Tester system can have the following classes:

  User Interface: This class can have methods for accepting user inputs and displaying results. It can interact with the Password Strength Checker class to get the results of the password strength test.

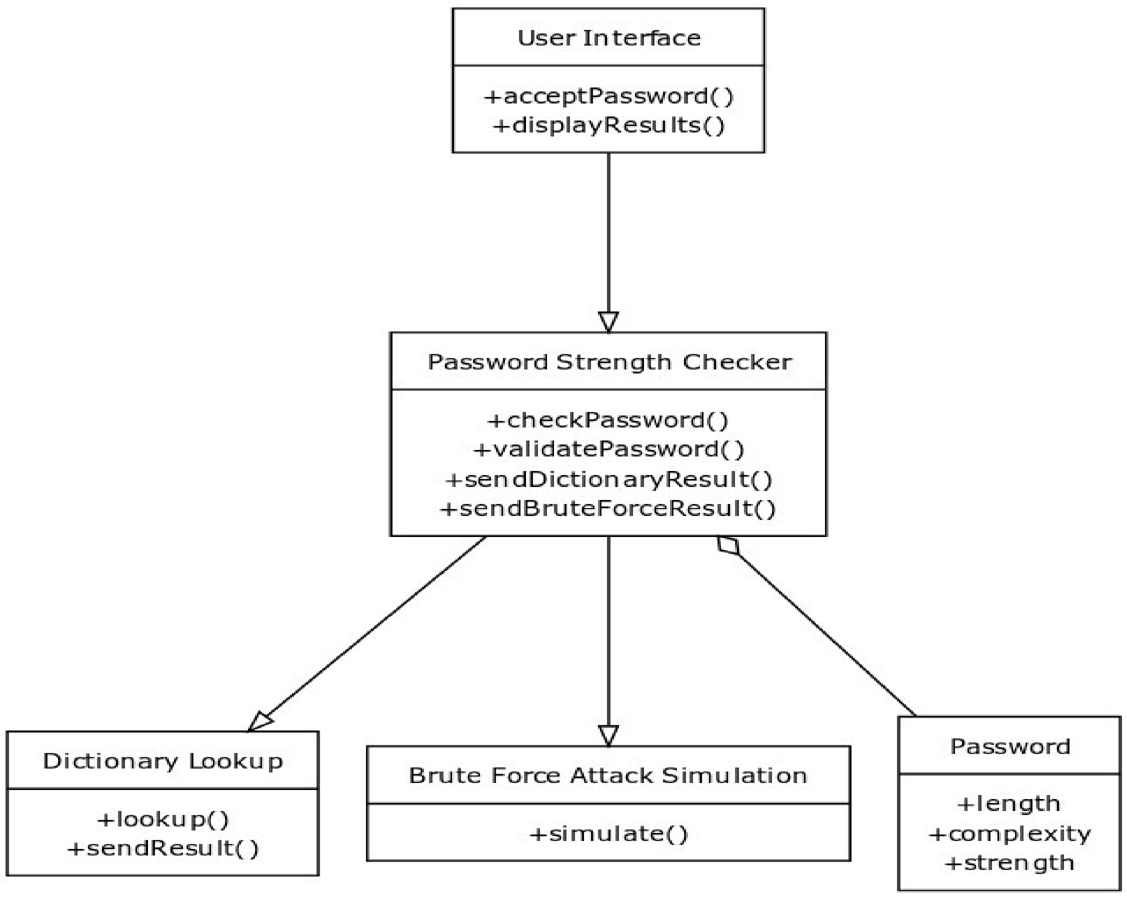
  Password Strength Checker: This class can have methods for analyzing the password entered by the user and determining its strength. It can interact with the Dictionary Lookup and Brute Force Attack Simulation classes to check if the entered password is weak or not.

Dictionary Lookup: This class can have methods for checking if the entered password is a common dictionary word or a combination of dictionary words. It can interact with the Password Strength Checker class to send the results of the dictionary lookup.

Brute Force Attack Simulation: This class can have methods for simulating a brute-force attack on the entered password to test its strength. It can interact with the Password Strength Checker class to send the results of the brute-force attack simulation.

Password: This class can represent the password entered by the user. It can have attributes like length, complexity, and strength.

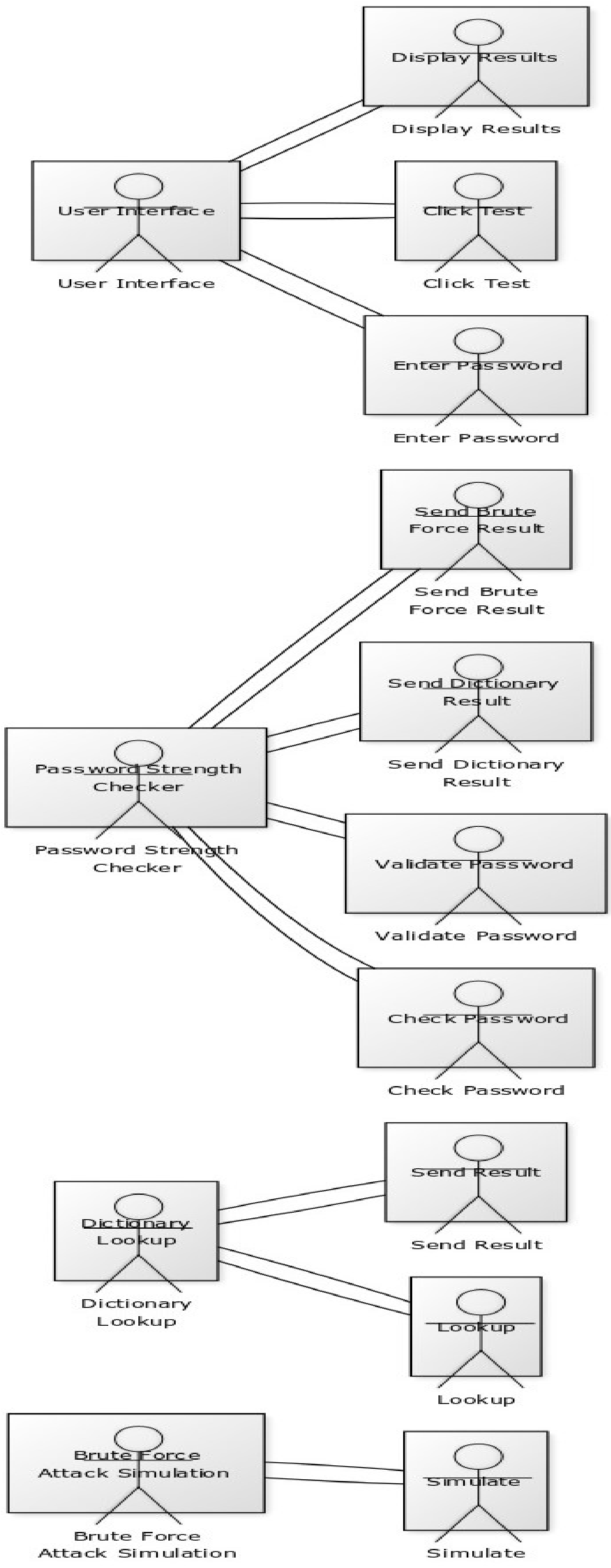
Here’s an example of a class diagram for the Password Strength Tester system:



- The use case diagram shows the interactions between the user interface and the different components of the password strength tester system. The user interface allows the user to enter a password, click the test button, and display the results of the password strength test. The Password Strength Checker component is responsible for checking the password against various rules and algorithms, including dictionary lookups and brute force attack simulations. The Dictionary Lookup component is responsible for checking if the password matches any words in a dictionary, while the Brute Force Attack Simulation component is responsible for simulating a brute force attack on the password.

The use case diagram shows the relationships between the various components of the system and the actions that can be performed by each component. For example, the user interface can enter a password, click the test button, and display the results. The Password Strength Checker can check and validate the password, send results to the dictionary lookup component and brute force attack simulation component. The Dictionary Lookup component can look up the password in a dictionary and send the result back to the Password Strength Checker, while the Brute Force Attack Simulation component can simulate a brute force attack and send the result back to the Password Strength Checker.

Overall, the use case diagram provides an overview of the interactions between the various components of the password strength tester system and the actions that can be performed by each component. It helps to visualize the overall flow of the system and how the different components work together to check the strength of a password. Here's an example of a use case diagram for the Password Strength Tester system:


Display Results
Display Results
User Interface
User Interface
Click Test
Click Test
Enter Password
Enter Password
Send Brute Force Result
Send Brute Force Result
Send Dictionary Result
Send Dictionary Result
Password Strength Checker
Password Strength Checker
Validate Password
Validate Password
Check Password
Check Password
Send Result
Send Result
Dictionary Lookup
Dictionary Lookup
Lookup
Lookup
Brute Force Attack Simulation
Brute Force Attack Simulation
Simulate
Simulate

2.5 Tools and Development Phases

For the development of the Password Strength Tester, we will use the following tools:

- Python: We will use Python to develop the Password Strength Tester.
- Flask: We will use Flask, a micro web framework in Python, to create the web application and handle HTTP requests and responses.
- Pytest: We will use Pytest, a Python testing framework, to write and run tests for the system.
- Git: We will use Git as our version control system to manage the source code.
- Visual Studio Code: We will use Visual Studio Code as our code editor.

Development Phases:

- Planning: We will gather and analyze the project requirements, create a project plan, and define the user stories for the Password Strength Tester.
- Design: We will create a high-level design document that outlines the architecture of the system, its components, and the messages and data exchanged between them.
- Implementation: We will use Python and Flask to implement the system features. We will also integrate Pytest to write and run tests for the system.
- Testing: We will test the system using various test cases to ensure that it meets the project requirements and is secure and reliable.
- Deployment: We will deploy the system to a production environment, ensuring that it is accessible to users and meets the necessary security requirements.

In addition to the tools and development phases mentioned above, we will ensure that the Password Strength Tester system meets the following requirements specified in the project description:

- Minimum password length: We will ensure that the system enforces a minimum password length of at least 8 characters.

- Password complexity: We will include a password complexity checker that requires users to include at least one uppercase letter, one lowercase letter, one number, and one special character.
- Avoiding parts of the username or any public information: We will ensure that the system checks whether the password includes any parts of the username or any public information.
- Passphrases: We will encourage users to use passphrases over standard passwords and provide guidance on how to create strong passphrases.
- Thumbs up when all test cases pass: The password strength tester should only give a thumbs up when all test cases pass, including the tests for minimum length, complexity, avoiding username or public information, and any other relevant tests.

By following the OWASP guidelines and implementing the features mentioned above, we will create a secure and reliable Password Strength Tester that meets the project requirements.

2.6 GitHub repository

link: Password-Strength-Tester

Chapter 3

Conclusion

In summary, a password strength tester is a tool that enforces strong password policies, evaluates password strength using mathematical algorithms, provides user-friendly interfaces for password input and feedback, follows OWASP guidelines, and promotes the use of passphrases over traditional passwords. The functional flow of a password strength tester involves users entering their password, the tester analyzing the password, displaying the results to the user, giving feedback, and optionally allowing users to generate a new password.

Khalil Elachkham-Mohamed Haroun Cheriha-Mahdi Laafif

TUNIS BUSINESS SCHOOL
UNIVERSITY OF TUNIS