Prodigy InfoTech Cyber Security Report

Week 3: Password Complexity Checker

Full Name: Sairaj Turalkar

Date: 16/07/2024

1. Introduction

In the third week of my cybersecurity internship at Prodify Infotech, I was tasked with building a password complexity checker tool. This tool assesses the strength of a password based on various criteria, such as length, presence of uppercase and lowercase letters, numbers, and special characters. It also provides feedback to users on their password strength.

2. Objective

The objective of this task was to develop a tool that evaluates the strength of passwords and provides users with feedback to help them create more secure passwords. The tool should assess passwords based on specific criteria and categorize the password strength as "Very Weak," "Weak," "Moderate," "Strong," or "Very Strong."

3. Methodology

3.1. Password Complexity Checker

The password complexity checker was implemented in Python. The tool evaluates passwords based on the following criteria:

1. **Length**: The password should be at least 8 characters long.
2. **Uppercase Letters**: The password should contain uppercase letters.
3. **Lowercase Letters**: The password should contain lowercase letters.
4. **Digits**: The password should contain digits.
5. **Special Characters**: The password should contain special characters.

Each criterion contributes to the overall strength score, which determines the strength level of the password.

3.2. Python Implementation

The implementation consists of a password_strength function and a main function to interact with the user.

Password Strength Function

The password_strength function evaluates the password based on the defined criteria and provides feedback on how to improve it.

```
def password_strength(password):
    length = len(password) >= 8
    has_upper = any(char.isupper() for char in password)
    has_lower = any(char.islower() for char in password)
    has_digit = any(char.isdigit() for char in password)
    has_special = any(char in "!@#$%^&*()-_+=<>?/|\\{}[]:;" for char in password)

    strength_score = sum([length, has_upper, has_lower, has_digit, has_special])

    if strength_score == 5:
        strength = "Very Strong"
    elif strength_score == 4:
        strength = "Strong"
    elif strength_score == 3:
        strength = "Moderate"
    elif strength_score == 2:
        strength = "Weak"
    else:
        strength = "Very Weak"

    feedback = []
    if not length:
        feedback.append("Password should be at least 8 characters long.")
    if not has_upper:
        feedback.append("Password should include uppercase letters.")
    if not has_lower:
        feedback.append("Password should include lowercase letters.")
    if not has_digit:
        feedback.append("Password should include digits.")
    if not has_special:
        feedback.append("Password should include special characters.")

    return strength, feedback

def main():
    while True:
        password = input("Enter a password to check its strength: ")
        strength, feedback = password_strength(password)

        print(f"Password strength: {strength}")
        for tip in feedback:
            print(f"- {tip}")

        repeat = input("Do you want to check another password? (if yes press 'y' / if no press 'n'):
")
        if repeat.lower() != 'y':
            print("Goodbye!")
            break

if __name__ == "__main__":
    main()
```

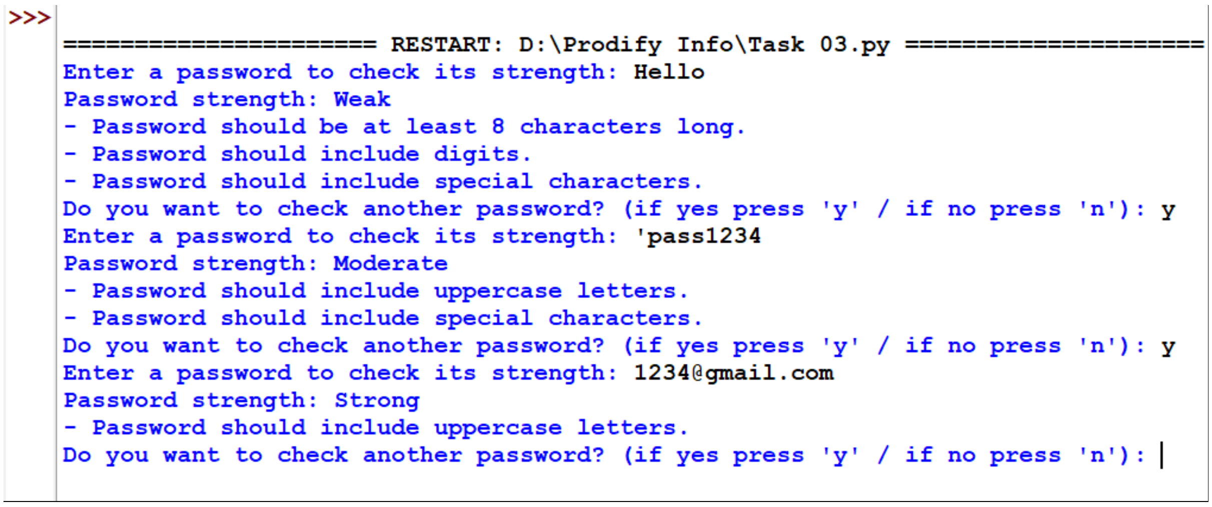
Main Function

The main function provides a user interface for checking password strength.

```
def main():
    while True:
        password = input("Enter a password to check its strength: ")
        strength, feedback = password_strength(password)

        print(f"Password strength: {strength}")
        for tip in feedback:
            print(f"- {tip}")

        repeat = input("Do you want to check another password? (if yes press 'y' / if no press 'n'): ")
        if repeat.lower() != 'y':
            print("Goodbye!")
            break

if __name__ == "__main__":
    main()
```

4. Testing and Results

The password complexity checker was tested with various passwords to ensure it accurately assessed their strength. Here are some examples:



```
>>>
===================== RESTART: D:\Prodify Info\Task 03.py =====================
Enter a password to check its strength: Hello
Password strength: Weak
- Password should be at least 8 characters long.
- Password should include digits.
- Password should include special characters.
Do you want to check another password? (if yes press 'y' / if no press 'n'): y
Enter a password to check its strength: 'pass1234
Password strength: Moderate
- Password should include uppercase letters.
- Password should include special characters.
Do you want to check another password? (if yes press 'y' / if no press 'n'): y
Enter a password to check its strength: 1234@gmail.com
Password strength: Strong
- Password should include uppercase letters.
Do you want to check another password? (if yes press 'y' / if no press 'n'):
```

5. Conclusion

This task provided an opportunity to develop a practical tool for enhancing password security. By implementing the password complexity checker, I gained a deeper understanding of password security practices and user interface design. The tool successfully evaluates password strength and provides constructive feedback to help users create more secure passwords.

6. References

- Python Standard Library: Python Documentation