Information Security and Privacy

Assignment Part-2 REPORT

Name: Hari Kishan Reddy Abbasani
Netid: ha2755

This Assignment is to compromise passwords from the given files present in Password Lists.zip. There are 3 files present in the zip file and each file has a different format written. Let's start with Linked-in

1) Linked In (Result: SUCCESS)

The LinkedIn folder contains a file called SHA1.txt. Therefore, I've come across the SHA1.txt file, which contains hash values. The approach that has crossed my mind for compromising these passwords is a dictionary attack. In this attack, I attempt to hash/encode the most common passwords found in data breaches from internet resources and compare those hash values with the ones in SHA1.txt. If a match is found, I then attempt to print the corresponding passwords and hash values.

SHA1.txt = Provided file
Extract-password.txt = Most commonly used passwords / passwords found in a data breach on the internet

So, I began writing Python code using the hashlib library to encode passwords/plaintext into SHA1 hash values and compare them with the hash values present in the provided SHA1.txt file. After completing the code, I attempted to execute the linkedin.py file multiple times, replacing the plaintext or passwords with those from the internet. Finally, I discovered that some of the hash values matched those from the internet sources listed below:

https://github.com/danielmiessler/SecLists/blob/master/Passwords/2020200_most_used_passwords.txt -

Hashed Password: 96506c3924c68ae55dd8e264eef730aa9dc90c14, Plain Text: jobandtalent

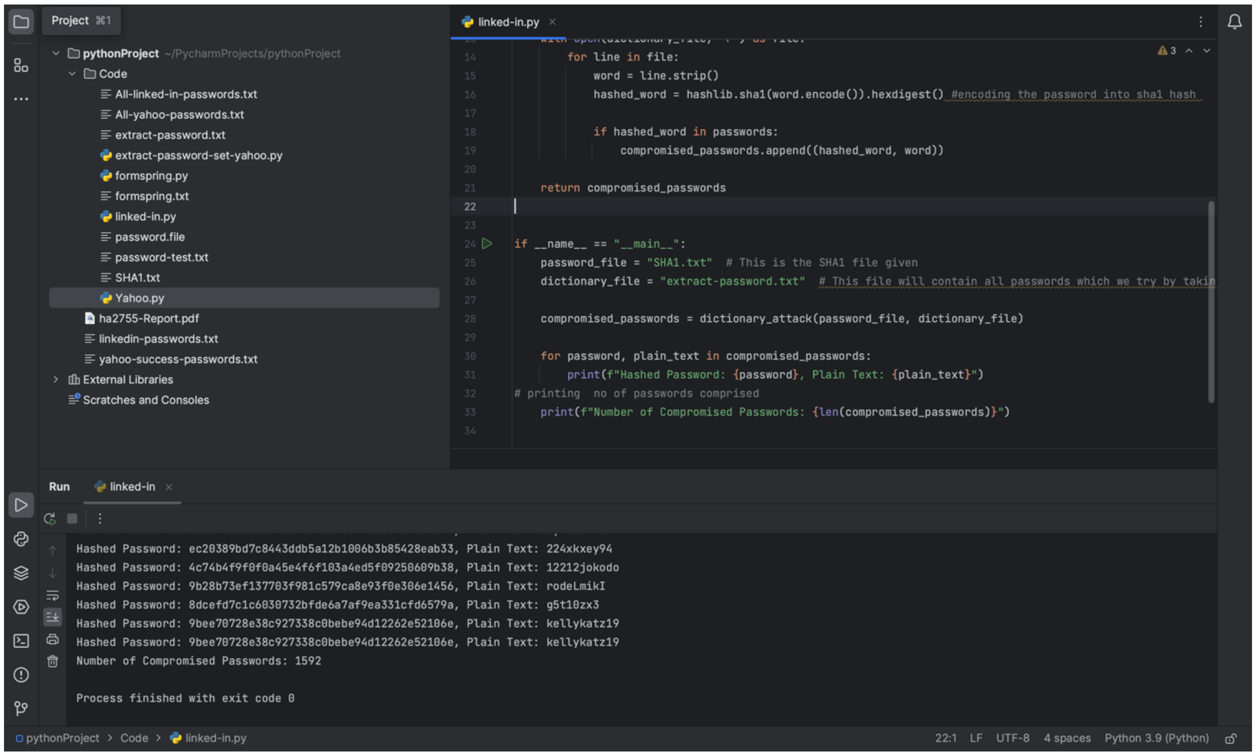
https://github.com/danielmiessler/SecLists/blob/master/Passwords/Leaked-Databases/NordVPN.txt

Hashed Password: ddf38fb37dd307ca61fc754322a8d2bc612f2ca9, Plain Text: Lounaj83

https://github.com/danielmiessler/SecLists/blob/master/Passwords/Leaked-Databases/bible-withcount.txt
Hashed Password: bf1bc9b2d8b899c06774512f3b668d916822ea76, Plain Text: 1 player

Later on, I have used passwords extracted from the Yahoo data breach file (**password.file**). Surprisingly, **I have found 1592 passwords** which were matching the hash values present in SHA1.txt. Please find the list of 1592 linked-in passwords I found in All-linked-in-passwords.txt in the zip folder which I have submitted.

OUTPUT:



```python
            for line in file:
                word = line.strip()
                hashed_word = hashlib.sha1(word.encode()).hexdigest() #encoding the password into sha1 hash

                if hashed_word in passwords:
                    compromised_passwords.append((hashed_word, word))

        return compromised_passwords


if __name__ == "__main__":
    password_file = "SHA1.txt"  # This is the SHA1 file given
    dictionary_file = "extract-password.txt"  # This file will contain all passwords which we try by takin

    compromised_passwords = dictionary_attack(password_file, dictionary_file)

    for password, plain_text in compromised_passwords:
        print(f"Hashed Password: {password}, Plain Text: {plain_text}")
# printing  no of passwords comprised
    print(f"Number of Compromised Passwords: {len(compromised_passwords)}")
```

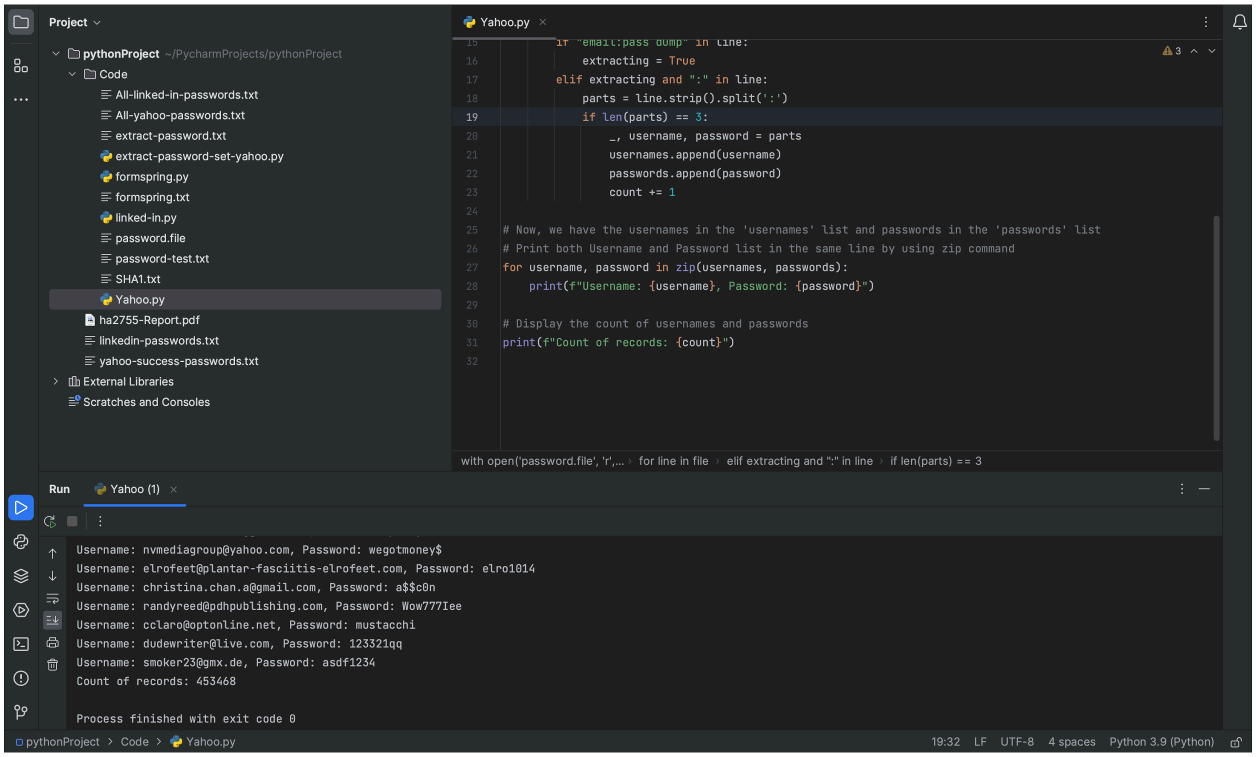
```
Hashed Password: ec20389bd7c8443ddb5a12b1006b3b85428eab33, Plain Text: 224xkxey94
Hashed Password: 4c74b4f9f0f0a45e4f6f103a4ed5f09250609b38, Plain Text: 12212jokodo
Hashed Password: 9b28b73ef137703f981c579ca8e93f0e306e1456, Plain Text: rodeLmikI
Hashed Password: 8dcefd7c1c6030732bfde6a7af9ea331cfd6579a, Plain Text: g5t10zx3
Hashed Password: 9bee70728e38c927338c0bebe94d12262e52106e, Plain Text: kellykatz19
Hashed Password: 9bee70728e38c927338c0bebe94d12262e52106e, Plain Text: kellykatz19
Number of Compromised Passwords: 1592

Process finished with exit code 0
```

2) Yahoo (Result: SUCCESS)

The Yahoo folder contains a file named "password.file." I opened the file and attempted to analyze its format. I quickly identified the usernames and passwords exposed in the file, located within the section labeled "3. email:pass dump (450k users)."

To display the usernames and passwords using a Python script, "yahoo.py," I applied a filter to print the information after the "email:pass dump" section. Additionally, I used the same file to extract only the passwords, which I saved as "passwords-test.txt." I intend to use this file to attempt to compromise the LinkedIn SHA1 file.

OUTPUT:


Project
pythonProject ~/PycharmProjects/pythonProject
Code
All-linked-in-passwords.txt
All-yahoo-passwords.txt
extract-password.txt
extract-password-set-yahoo.py
formspring.py
formspring.txt
linked-in.py
password.file
password-test.txt
SHA1.txt
Yahoo.py
ha2755-Report.pdf
linkedin-passwords.txt
yahoo-success-passwords.txt
External Libraries
Scratches and Consoles
Yahoo.py
Run
Yahoo (1)
with open('password.file', 'r',... › for line in file › elif extracting and ":" in line › if len(parts) == 3
pythonProject › Code › Yahoo.py
19:32 LF UTF-8 4 spaces Python 3.9 (Python)

```python
        if "email:pass dump" in line:
            extracting = True
        elif extracting and ":" in line:
            parts = line.strip().split(':')
            if len(parts) == 3:
                _, username, password = parts
                usernames.append(username)
                passwords.append(password)
                count += 1

# Now, we have the usernames in the 'usernames' list and passwords in the 'passwords' list
# Print both Username and Password list in the same line by using zip command
for username, password in zip(usernames, passwords):
    print(f"Username: {username}, Password: {password}")

# Display the count of usernames and passwords
print(f"Count of records: {count}")
```

```
Username: nvmediagroup@yahoo.com, Password: wegotmoney$
Username: elrofeet@plantar-fasciitis-elrofeet.com, Password: elro1014
Username: christina.chan.a@gmail.com, Password: a$$c0n
Username: randyreed@pdhpublishing.com, Password: Wow777Iee
Username: cclaro@optonline.net, Password: mustacchi
Username: dudewriter@live.com, Password: 123321qq
Username: smoker23@gmx.de, Password: asdf1234
Count of records: 453468

Process finished with exit code 0
```

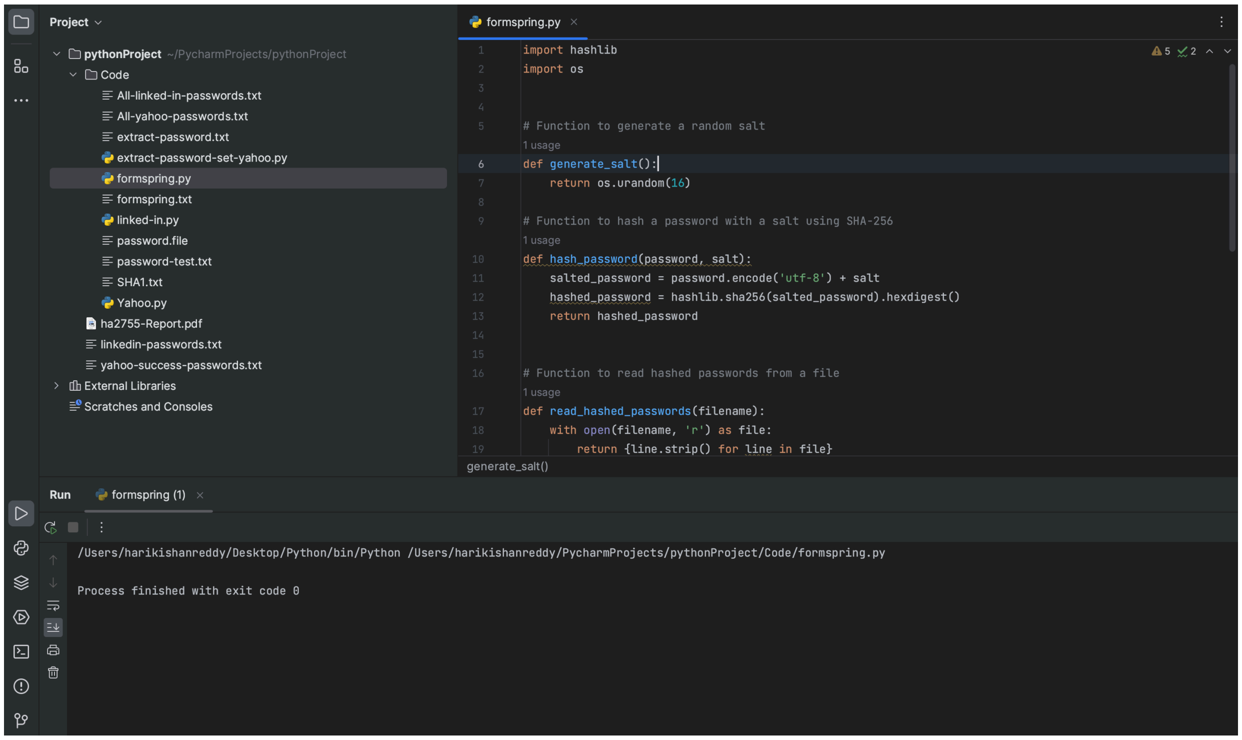
3) Formspring (Result : FAILURE)

I found this as the toughest task to comprise passwords. The Formspring folder has a file called **formspring.txt.** I have found many values and I tried to analyze the format given in the file. I have come to conclusion that these values were hash values due to their consistent length. I started searching for the type of hashing algorithm used to get these hash values. I found that these hash values are formed using a combination of SHA256 and salt as detailed in the below resource:

https://www.securityweek.com/formspring-hacked-420000-passwords-leaked/

I tried using the same dictionary attack approach which I used for linked-in by modifying the code accordingly, but I didn't find a single matched Hashed value after executing the **formspring.py**. The reason I feel is that formspring have used SHA256 and salt (random value from 1-128 in bits by adding), I have used random to randomly generate the number and add it to SHA256 but since I have used random inbuilt function it's difficult to find all the hash values generated with different random values used. I believe that the probability of successfully compromising passwords would increase significantly if the salt values were exposed or provided.

OUTPUT:



```python
import hashlib
import os


# Function to generate a random salt
def generate_salt():
    return os.urandom(16)

# Function to hash a password with a salt using SHA-256
def hash_password(password, salt):
    salted_password = password.encode('utf-8') + salt
    hashed_password = hashlib.sha256(salted_password).hexdigest()
    return hashed_password


# Function to read hashed passwords from a file
def read_hashed_passwords(filename):
    with open(filename, 'r') as file:
        return {line.strip() for line in file}
```

```
/Users/harikishanreddy/Desktop/Python/bin/Python /Users/harikishanreddy/PycharmProjects/pythonProject/Code/formspring.py

Process finished with exit code 0
```

DIFFICULTY IN CRACKING PASSWORDS PROTECTED WITH EACH TYPE OF STORAGE:

Yahoo file name: (password.file) I found this format to find credentials **VERY EASY** as I can directly find the usernames and passwords in the file when compared to other formats.

LinkedIn file name: (SHA1.txt) I consider this format to be of **MODERATE** difficulty for cracking passwords compared to the other two formats.

Formspring file name: (formspring.txt) I find this format **DIFFICULT** to compromise because the format is a combination of SHA256+salt.

File formats:

Yahoo: Direct plain text

LinkedIn: SHA1

Formspring: SHA256+salt