Activity 1: Krit Cholapand 6030070521

1. Write a simple python program to use the word from the dictionary to find the original value of d54cc1fe76f5186380a0939d2fc1723c44e8a5f7 .

Note that you might want to include substitution in your code (lowercase, uppercase, number for letter [‘o’ => 0 , ‘l’ => 1, ‘i’ => 1]).

ThaiLanD

2. For the given dictionary, create a rainbow table (including the substituted strings) using the sha1 algorithm. Measure the time for creating such a table. Measure the size of the table.

time: 5.851572275161743

size: 3644414 entry

3. Based on your code, how long does it take to perform a hash (sha1) on a password string? Please analyze the performance of your system.

1.4960765838623048e-06 sec

My system is I7-8550U

4. If you were a hacker obtaining a password file from a system, estimate how long it takes to break a password with brute force using your computer.

(Please based the answer on your measurement from exercise #3.)

Assume number of character available in password is 94 accordingg to https://www.password-depot.de/en/know-how/brute-force-attacks.htm#:~:text=When%20creating%20a%20password%2C%20the,Special%20characters%20(32%20different).

And maximum length of password is 128

Maximum time it take to crack a password is 94**128*1.4960765838623048e-06 = 5.436496168729334e+246 sec

5. Base on your analysis in exercise #4, what should be the proper length of a password. (e.g. Take a year or longer to break).

$$1.0000000000000000 \times 94^{x} = 2.107913481179283 \times 10^{13}$$

Number line:

Real solution:

$x \approx 6.7526555651638860$

The length should be at least 7 character

6. What is salt? Please explain its role in protecting a password hash.

Salt is some random string that add to password before hash to make hashes of the same password difference. This prevent hacker from knowing user having same password.

Code

1. Cracking Thailand

```
import hashlib

def CheckHash(plain, md5):
    if hashlib.sha1(plain.encode('UTF-8')).hexdigest() == md5:
        print(plain)
        return True
    if hashlib.md5(plain.encode('UTF-8')).hexdigest() == md5:
        print(plain)
        return True
    return False

def Rec(plain, md5, i):
    if i == len(plain):
        return CheckHash(plain, md5)

    char = plain[i].lower()
    subList = []
    if char.isalpha():
        subList += [char.upper(), char]
    else:
        subList += [char]
```

```
    if char == 'o':
        subList.append('0')
    elif char == '0':
        subList.append('o')
        subList.append('O')
    elif char == 'i':
        subList.append('1')
    elif char == 'l':
        subList.append('1')
    elif char == '1':
        subList.append('i')
        subList.append('I')
        subList.append('l')
        subList.append('L')

    for newChar in subList:
        subPlain = plain[0:i] + newChar + plain[i+1:]
        if Rec(subPlain, md5, i+1):
            return True
    return False

filename = '10k-most-common.txt'
md5 = 'd54cc1fe76f5186380a0939d2fc1723c44e8a5f7'

with open(filename) as file:
    for line in file:
        plain = line.strip()
        if Rec(plain, md5, 0):
            break
    else:
        print("not match")
    pass
```

2. Create Rainbow

```
import hashlib
import time

def CheckHash(plain, md5):
    if hashlib.sha1(plain.encode('UTF-8')).hexdigest() == md5:
        print(plain)
        return True
    if hashlib.md5(plain.encode('UTF-8')).hexdigest() == md5:
```

```
        print(plain)
        return True
    return False

def Rec(plain, i):
    if i == len(plain):
        return [plain]

    char = plain[i].lower()
    subList = []
    if char.isalpha():
        subList += [char.upper(), char]
    else:
        subList += [char]

    if char == 'o':
        subList.append('0')
    elif char == '0':
        subList.append('o')
        subList.append('O')
    elif char == 'i':
        subList.append('1')
    elif char == 'l':
        subList.append('1')
    elif char == '1':
        subList.append('i')
        subList.append('I')
        subList.append('l')
        subList.append('L')

    res = []
    for newChar in subList:
        subPlain = plain[0:i] + newChar + plain[i+1:]
        res += Rec(subPlain, i+1)

    return res

filename = '10k-most-common.txt'

start = time.time()

rainbow = []
with open(filename) as file:
```

```
    for line in file:
        plain = line.strip()
        rainbow += Rec(plain, 0)

end = time.time()

print('time:', end-start)
print('size:', len(rainbow))
```

3. Time Average hashing

```
import hashlib
import time
filename = '10k-most-common.txt'

start = time.time()

rainbow = []
i = 0
with open(filename) as file:
    for line in file:
        i += 1
        hashlib.sha1(line.encode('UTF-8')).hexdigest()

end = time.time()
print((end-start)/i)
```