Lab #2: Cryptography
CSE 3801 : Introduction to Cyber Operations
Team: Tyler Dionne

Challenge 1:
Decode the flag:
kqfl{6k_dtz_b9sy_yt_p88u_9_x8hw8y_dtz_rzxy_9qxt_m6i8_6y_kwtr_dtzwx8qk}
Solution:

- Use https://www.dcode.fr/cipher-identifier, paste the encoded text into the cipher identifier. Click on caesar cipher. Paste the text into the box and the flag is displayed on the top left of the screen.

Challenge 2:
The flag.enc in an encoded binary that has been XOR'd with a key. Decode it and return the flag.
Provided file: flag.enc
Solution:

- Use the command 'xxd flag.enc' and we can see many instances of the string 'florida' on the right hand side. We can infer that this is the key that the plaintext has been XOR'd with.
- Then can use python to create a script to XOR the ciphertext with the key and return the flag.

```
from pwn import *
xor_key = b'florida'
with open('flag.enc', 'rb') as encoded_file:
    encoded_data = encoded_file.read()
decoded_data = xor(encoded_data, xor_key)
with open('decoded_flag.bin', 'wb') as decoded_file:
    decoded_file.write(decoded_data)
```

- Can then use the command 'strings decoded_flag.bin' to retrieve the flag.

Challenge 3:
Crack the password for the student. The password is in the format: flag{password}. Given a file hash.txt.
Provided file: hash.txt
Solution:

- Can use the password cracking software johntheripper to solve this challenge.
- We can find the stock wordlist that comes with johntheripper at usr/share/wordlists/john.lst

- We know that the flag will be in the format flag{password} so we must reformat this wordlist. We can do this using sed shown below:

```
$ sed 's/.*/flag{&}/' john.lst > reformatted_john.lst
```

- We can then run the following command to retrieve the flag:

```
$ john -wordlist=reformatted_john.lst hash.txt
```

Challenge 5:
Attack the side channel and capture the flag in time. This connection script makes use of the pwntools library.
Provided files: pyshell.py
Solution:

- Can use a timing based side channel attack to solve this challenge.
- The provied python script shows that when a correct letter is guessed the program will sleep for 0.25. This means that we can figure out each letter by seeing which guess takes the longest amount of time.
- We do not know the length of the flag but we do know that it ends with a '}' so when this is the correct letter we know the flag is complete.
- The following python script will retrieve the flag.

```
from pwn import *
import time
choices = 'abcdefghijklmnopqrstuvwxyz0123456789_}' besttime = -1
bestguess = 'flag{'
beginning = 'flag{'
isend = False
while(not(isend)):
    for c in choices: curr = ''
        curr = beginning + c
        print(curr)
        foo = 0
        r = remote("cse3801-crypto-500.chals.io", 443, ssl=True,
sni="cse3801-crypto-500.chals.io")
        r.recvuntil("Guess the flag >>> ")
        start = time.time()
        r.sendline(curr)
        r.response = r.recvall().decode()
        foo = (time.time() - start)
        r.close()
        if foo > besttime:
            besttime = foo
            bestguess = curr
            if c == '}':
                isend = True
```

```
    beginning = bestguess
print(bestguess)
```