Easy:

**Sanity Check:**

Just scroll down to the bottom of the assignment pdf and obtain the flag.

**Something’s off:**

Since we know the first 4 characters of the flag are CS2107, we already know that “QgGFEL” represents “CS2107” character by character. We then know each character is shifted by 14. Assume [0:25] means A-Z, [26:51] means a-z and [52:61] means 0-9, we can obtain the actual characters.

mapping = {  
 "E" : "0", #confirmed  
 "F" : "1", #confirmed  
 "G" : "2", #confirmed  
 "H" : "3",  
 "I" : "4",  
 "J" : "5",  
 "K" : "6",  
 "L" : "7", #confirmed  
 "M" : "8",  
 "N" : "9",  
 "o" : "a",  
 "p" : "b",  
 "q" : "c",  
 "r" : "d",  
 "s" : "e",  
 "t" : "f",  
 "u" : "g",  
 "v" : "h",  
 "w" : "i",  
 "x" : "j",  
 "y" : "k",  
 "z" : "l",  
 "0" : "m",  
 "1" : "n",  
 "2" : "o",  
 "3" : "p",  
 "4" : "q",  
 "5" : "r",  
 "6" : "s",  
 "7" : "t",  
 "8" : "u",  
 "9" : "v",  
 "A" : "w",  
 "B" : "x",  
 "C" : "y",  
 "D" : "z",  
 "O" : "A",  
 "P" : "B",  
 "Q" : "C", #confirmed  
 "R" : "D",  
 "S" : "E",  
 "T" : "F",  
 "U" : "G",  
 "V" : "H",  
 "W" : "I",  
 "X" : "J",  
 "Y" : "K",  
 "Z" : "L",  
 "a" : "M",  
 "b" : "N",  
 "c" : "O",  
 "d" : "P",  
 "e" : "Q",  
 "f" : "R",  
 "g" : "S", #confirmed  
 "h" : "T",  
 "i" : "U",  
 "j" : "V",  
 "k" : "W",  
 "l" : "X",  
 "m" : "Y",  
 "n" : "Z"  
}  
#def decrypt(cipher):  
def translate(cipher):  
 plain = ""  
 for char in cipher:  
 if (char == '\_' or char == '{' or char == '}'):  
 plain = plain + str(char)  
 else:  
 plain = plain + mapping[char]  
 return plain  
  
print(translate(cipher))

**MAC:**

Go to the directory of the text file, and by Google Search, we know the syntax for openssl to generate the sha256 key of the HMAC is this command:

openssl dgst -sha256 -hmac "CS21072022" text.txt

**Prime Time:**

First, use online tool to factorize the modulus to obtain the value p and q, and then use them to compute the Euler totient. With this information, we can compute the private key (mi), which is the multiplicative inverse of the public exponent mod the modulus. After this, we already have a ciphertext, its private key and the modulus, using this three to obtain the plaintext, and then convert it to hexadecimal so that it can be decoded.

Text

Description automatically generated



**Secret Penguin:**

After Google Search, we know the syntax for openssl to encrypt the png file using AES CBC mode of operation is the following command:

openssl enc -aes-128-cbc -p -K 1234567890abcdef1234567890abcdef -iv abcdef1234567890abcdef1234567890 -in tux.png -out tux.enc

Follow this command to get the sha256 of the encrypted file:

openssl dgst -sha256 tuc.enc

**Medium:**

**Insecure OTP:**

Text

Description automatically generated

The rationale of this question is that a XOR (a XOR k) = k, hence by XOR-ing the first 20 characters of the message and the encrypted message (converted from hex to original string), we can obtain the key, and then by XOR-ing the key and the encrypted message, we can get back the original message.

**Public Password:**

According to the question, the social media is “blue bird”, which implies twitter, so we can obtain the password from that account actually…

Then we go to our Linux system, and enter the command nc cs2107-ctfd-i.comp.nus.edu.sg 4003

By entering the password, we obtain the flag from the server.

**Substitution Cipher:**

According to the question, the encrypted text is believed to be the terms and conditions of a software, hence by observing the encrypted text, we know that



U 🡪 C, T 🡪 S

Then by further inspection, I realize:

Icon

Description automatically generated with low confidence H 🡪 I and M 🡪 V

Also, the order BQUVYONFH represents ABCDEFGHI. Once I try to substitute the known characters, I obtain the flag, which can be read as English text already. Hence, I can know the flag.

**Offline Password Cracking:**First, use “sudo apt-get install john” to download the password cracking tool. Then navigate to the directory of the text file. On that directory, type the command “john –-user=bob stolenshadow.txt” to crack the password. Then use the command “john –show stolenshadow.txt” to show the available password:

bob:abcd1234:1003:1003:Bob,,,:/home/bob:/bin/bash

Text

Description automatically generated

According to the hint, we can only choose one password “abcd1234”.

**Birthday Hash:**

The byte to generate the SHA512 is given as the sum of the event + a random string. Inspired by the birthday paradox, we realized it is possible to have two hash values collide with each other (having exactly the same hash value although the byte strings are different). Hence, we only need to use computer to brute force generate some random hash values and check if got two hash values collide with one another. The code to generate the random hash value is as follow:

Text

Description automatically generated

Text

Description automatically generated

Hard:

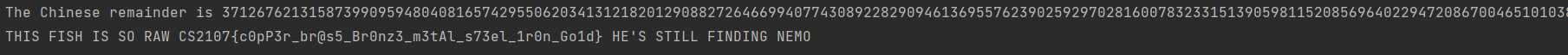
**Copper-rsa**

By searching information from wikepedia, I realized that a coppersmith-attack on rsa consists of these steps:

1. Have a list of x = c mod n (in this case, we have 5 c in c\_arr, and 5 n in n\_arr, the x is the plaintext)
2. Compute N = n1\*n2\*n3\*…\*ni (in this case, i = 5)
3. For each ni, compute Ni, where Ni = N / ni
4. Find multiplicative inverse of Ni mod ni
5. Compute c\_arr[i] \* Ni \* n\_arr[i]
6. Repeat this for all i in range(len(n\_arr)), and add all the calculated value in step 5 together
7. Compute the summation mod N

After we have done all the steps mentioned above, we will know m^3, we then take cube root of this value, and this is the value of the plaintext in rsa.

Now, by solving the given quadratic equation with the help of the value calculated just now and do decoding, we will know the flag.



**Rsa locked doors:**

For door 1, since we have ciphertext c, private key d, and the Euler totient function phi, we can know the e by finding multiplicative inverse of d mod phi. We can also do brute force search to find all possible modulo n. We first obtain the factors of phi, and find all possible combination to get n:

Text

Description automatically generated

For door 2, by inspection, we first know that the public key of Bob and Alice are provided, the public exponent e is 65537 (in hexadecimal) and their modulo n are also given. Also, I realize that each data is 32 bytes, which is able to be decoded using base32 in Python.

Once we start decoding, we will get the data, seq, and the signature like this:

A picture containing text

Description automatically generated

Then we use the signature as the ciphertext to decrypt using the public exponent 65537 and the modulo of Bob. However, it contains a lot of incorrect result, hence we only gather the correct information:

A picture containing table

Description automatically generated

Then, by arranging them with their sequence given, we can get the flag:

A screenshot of a computer

Description automatically generated with medium confidence

Hence, the flag is CS2107{n0\_noiS3\_t00\_d1fficult\_7o\_cLeAn}.