**Chi Seng 1002853**

Challenge description:

**Part I: Shattered**

*Hash Value = “b7a875fc1ea228b9061041b7cec4bd3c52ab3ce3”*

Find the password to enter the website using the hash value given above

Solution

Given the title of the challenge, we know that it is a SHA1 hash that is vulnerable to collision. This information did not matter as I sent the hash into hashkiller and found the password immediately.

Password: letmein

**Part II: RSA**

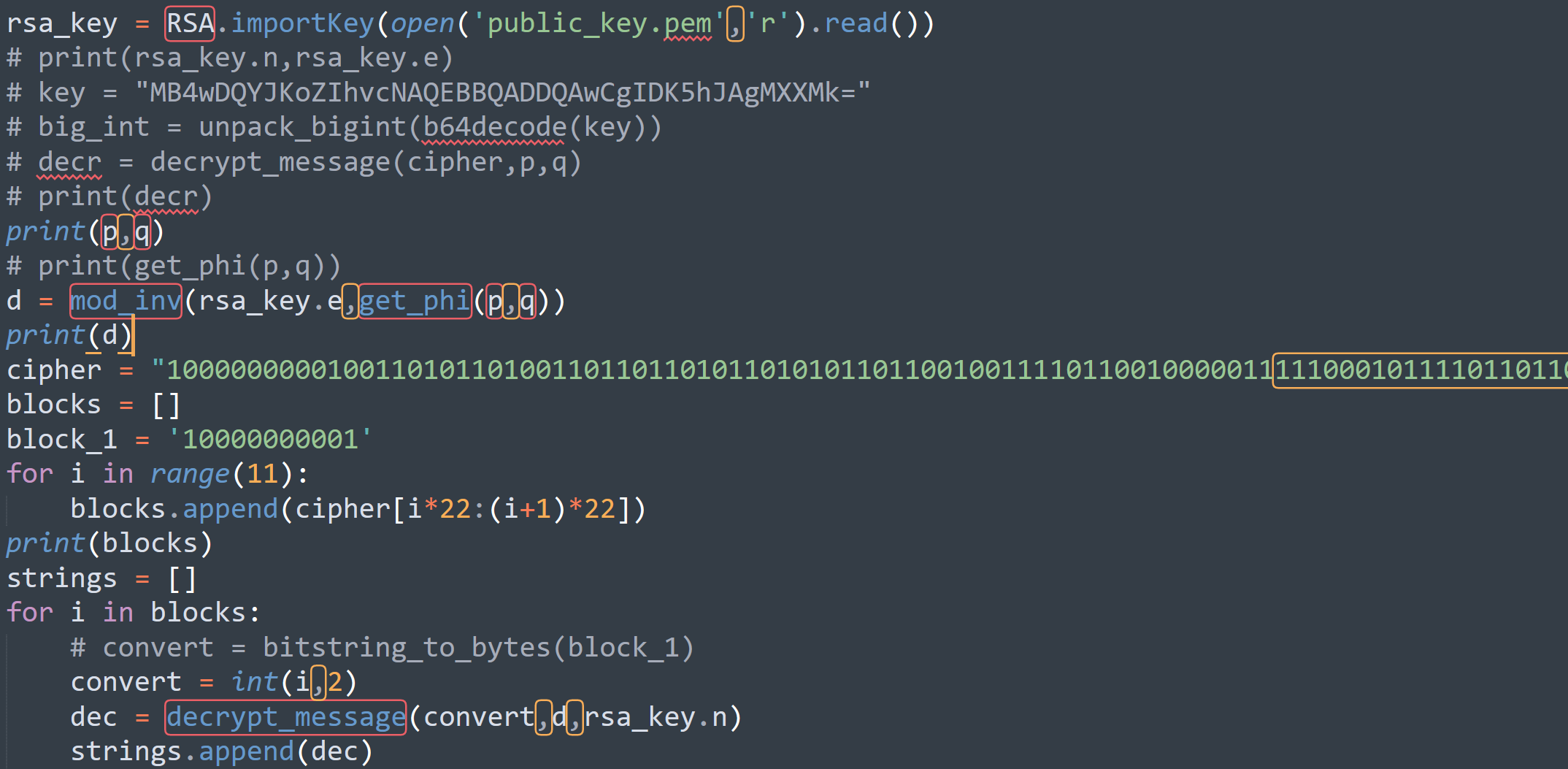
Ciphertext (**in Binary**) = 10000000001001101011010011011011010110101011011001001111011001000001111100010111101101100011011011010110101011001000100100010111011101110000110101011001010110010011110110010000000100100001000101001000010111110111000111100011011011010110101011

With only the public key (public\_key.pem) given and the knowledge that the encryption is using RSA. Get the plaintext from ciphertext.

Solution

Really buggy challenge here as the ciphertext was updated twice and the author did not provide us information that the text needed to be split in 11 chunks to be decrypted. Anyway, reversing the n value gave p and q values of 1579 and 1789 respectively, and getting d through inverse modulus gave us the private key to decrypt the text.

Password : s3cr3t\*c0d3



**Part III: AES key derivation**

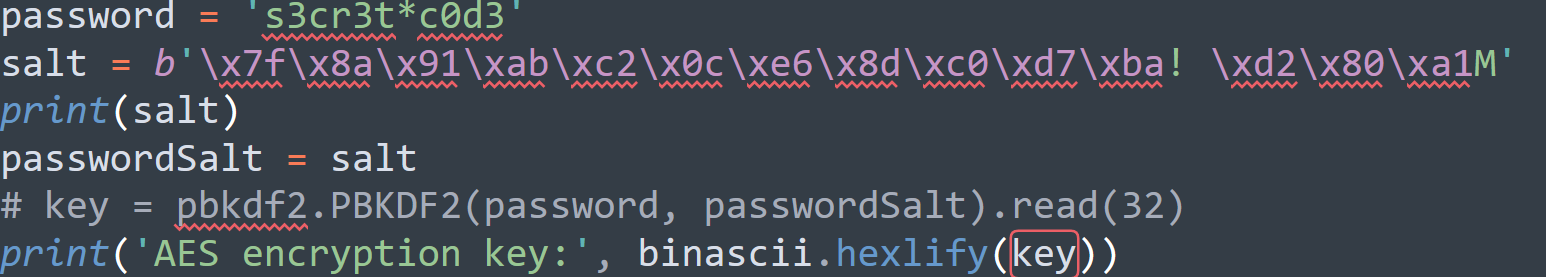
*passwordSalt=b'\x7f\x8a\x91\xab\xc2\x0c\xe6\x8d\xc0\xd7\xba! \xd2\x80\xa1M'*

Using the pyaes and pbkdf2 libraries, password derived from part II and salt given, get the AES encryption key!

Solution

Another buggy challenge as the decrypted key from the password and salt did not work. Import pyaes and pbkdf2 to extract the salt using the library functions.

Stanley gave me the correct key: c531048c47c13a7092d6ad8b36c0cc86bcef6fd4b8063acb7831c4fd2b523d68



**Part IV: Read the Bytes**

Using the text file given, analyse the byte distribution of each line.

Out of all the strings in the text file, there are 4 strings that will stand out.

Did not do this the intended way and “eyeballed” possible strings that could be keys, then using the key and IV given in the next step to decrypt the strings.

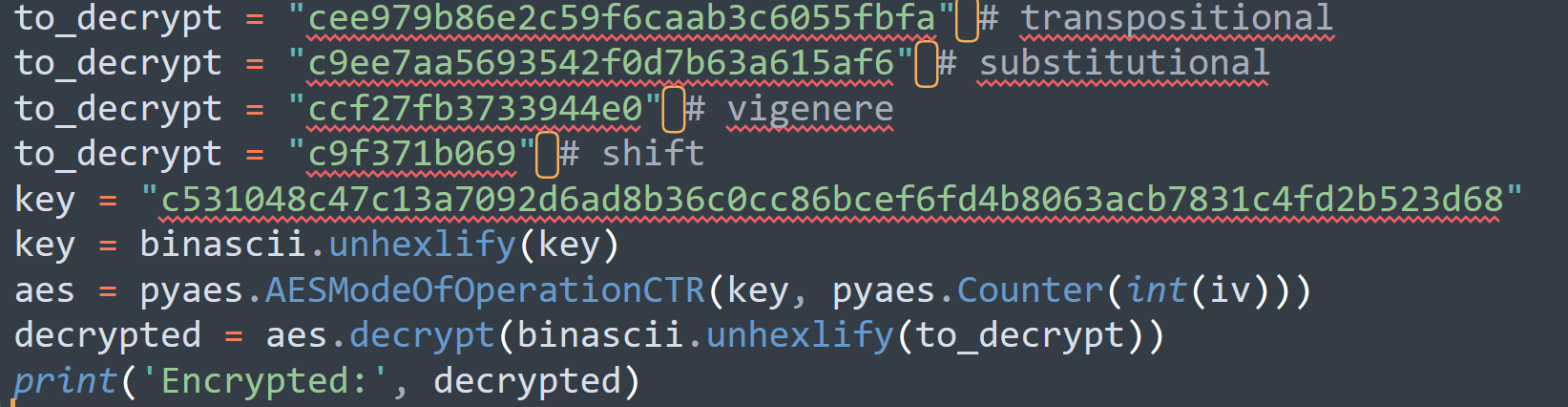
Strings:

cee979b86e2c59f6caab3c6055fbfa

c9ee7aa5693542f0d7b63a615af6

ccf27fb3733944e0

c9f371b069



**Part IV: AES Decryption (CTR Block mode)**

*IV=57116448576878005380785937564945681393249968307171981972 903895716101015138040*

Using the IV given and AES encryption key derived in part II, decrypt the 4 strings you singled out from the entire text file.

Solution

Using the script above we can get the four strings.

Strings:

*Transpositional*

*Subsititution*

*Shift*

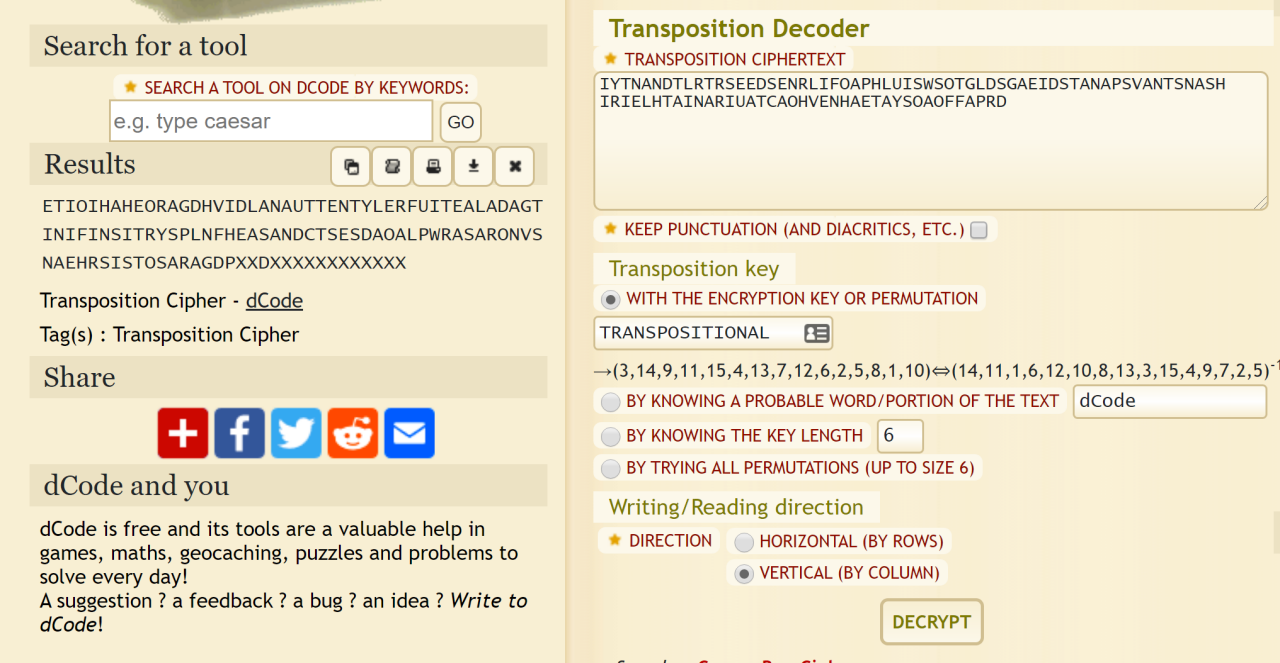
*Vigenere*

**Part V: Final Stage to Decipher**

*Ciphertext= “IYTNANDTLRTRSEEDSENRLIFOAPHLUISWSOTGLDSGAEIDSTANAPSVANTSNASH IRIELHTAINARIUATCAOHVENHAETAYSOAOFFAPRD”*

Using the decrypted text, decrypted the final cipher text!

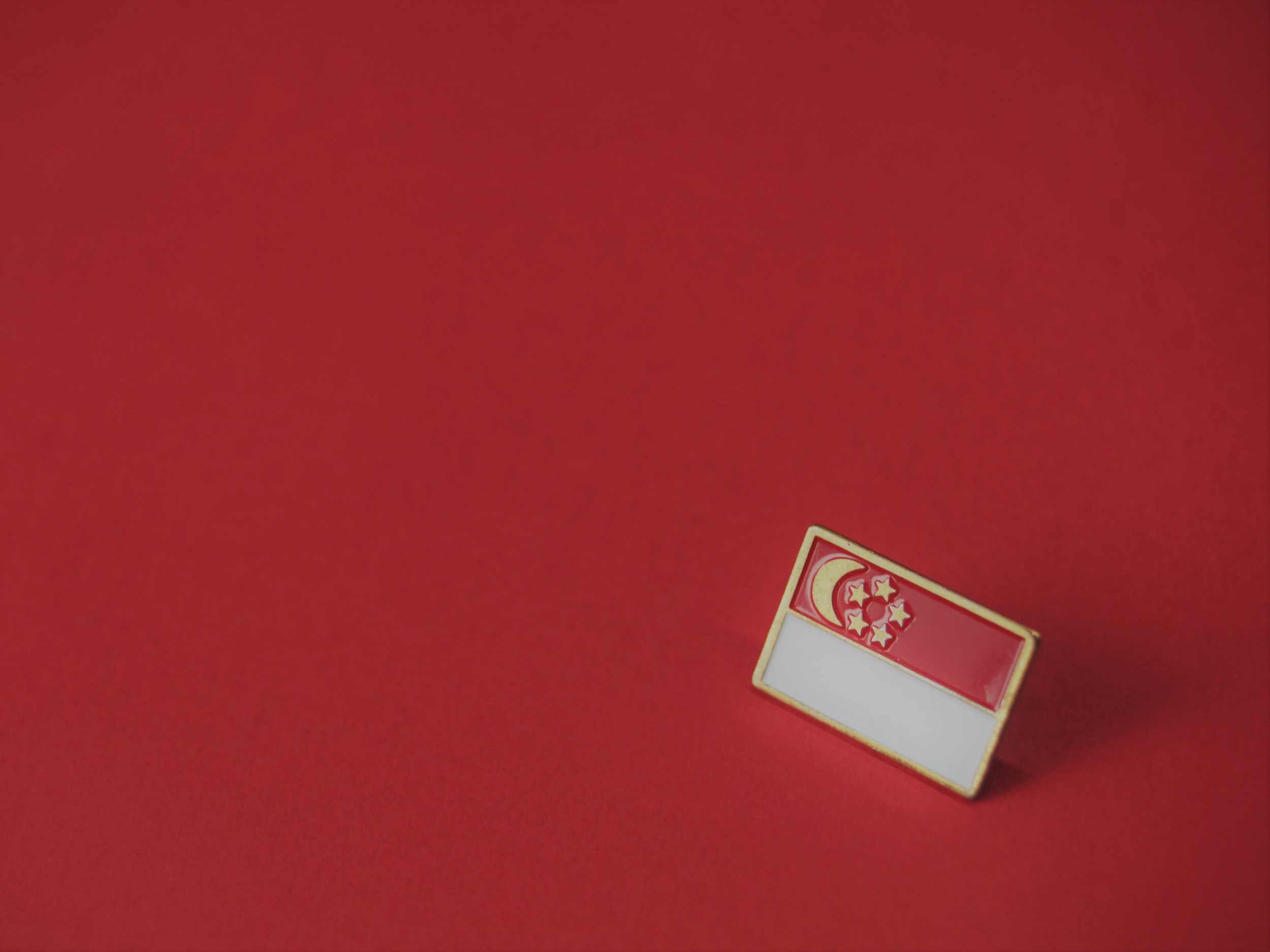
Another slightly buggy challenge, the ciphertext on the web included an extra space which was unintended. Also, using the correct key “transpositional” I was not able to get the correct decrypted text as seen in the screenshot



Got the password from stanley as: THISISTHEFLAGTHATYOUHAVEFINALLYATTAINEDAFTERUSINGTRANSPOSITIONALCIPHERANDRSAANDAESANDPASSWORDSOLVER

And lower cased it to get to the final step: thisistheflagthatyouhavefinallyattainedafterusingtranspositionalcipherandrsaandaesandpasswordsolver

Finally we get the “flag”



Which contains the flag hidden in the page source

Flag: CTF{thisIsTheFinalFlagThatYouAndYourGroupmatesHaveWorkedHardForToGetForTheLastCoupleOfDays:p}

Really buggy challenge and average concept overall. I did not learn much here as reverse engineering did most of the work and the challenges were mostly concepts taught in class, but one good point to note would be that encrypted text have different byte distributions, which can be helpful to distinguish between plain and ciphertext.

-m0zzare11a