Assignment #3 – 'Breaking the Classic Crypto'

Q1> Breaking the Vignere Cipher.

CTF Username: AyushShrivastava

Flag: facfc61267282ed174a9900aeb71f4fd

Explanation:

The Vigenère cipher is a polyalphabetic substitution cipher that is a natural evolution of the Caesar cipher. This cipher encrypts by shifting each letter in the plaintext up or down a certain number of places in the alphabet. The primary weakness of the Vigenère cipher is the repeating nature of its key. If the key's length 'n' *is* correctly guessed, then the cipher text can be treated as 'n' interleaved Caesar ciphers, which can easily be broken individually.

Solution:

1. The first task in breaking Vignere Cipher is identifying the length of the key. We are using the Index of coincidence IoC to find out the length of the key. The index of coincidence (IoC) measures the likelihood that any two characters of a text are the same.
   To find the period of the cipher,
   - We will cut the ciphertext into m slices, containing every m^{th} letter.
   - Then, we find the IoC for each slice and average them. We do this for various choices of 'm'.
   - The smallest 'm' with an average IoC close to 1.7 is our period.

   We now have the length of the key figured out.
2. Since we plan to perform a Statistics attack, we will need the frequencies of the English alphabet. We have Acquired the Engfreq.txt file from PracticalCryptograpghy.com. This file contains the English alphabet count generated from around 4.5 billion characters of English text.
3. Then, construct frequency tables for each ciphertext slice created when the period was found. Those tables are each rotated until they match well to the alphabet frequencies of English. These shifts determine the key.

Code:

A very elaborate code explaining the whole process is available in the Jupyter file.

References:

i. https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher
ii. http://practicalcryptography.com/cryptanalysis/letter-frequencies-various-languages/english-letter-frequencies/
iii. https://www.cipherchallenge.org/wp-content/uploads/2020/12/Five-ways-to-crack-a-Vigenere-cipher.pdf

Q2> Breaking One-Time Pads.

CTF Username: AyushShrivastava

The flag is: 26314018676239997144070650766490

Explanation:

The one-time pad ensures that the ciphertext reveals no information about the plaintext as long as the pad is used only once and the key is kept secret. This key is usually the output of a pseudo-random number generator. The workings of the One-Time pad are pretty simple. One time pad takes the key obtained and XORs the key with the plain text to get the cipher text.

$$Text_{Cipher} = Text_{PlainText} \oplus \ Key$$

Now if the Key is being reused and we know/Guess the plain text, we can try to mount a Known Plain Text attack by XORing the Plain text and Cipher text.

$$Text_{Cipher} \oplus Text_{PlainText} = (Text_{PlainText} \oplus Key\ ) \oplus Text_{PlainText}$$

$$Text_{Cipher} \oplus Text_{PlainText} = Text_{PlainText} \oplus Text_{PlainText} \oplus Key$$

$$Text_{Cipher} \oplus Text_{PlainText} = 0 \oplus Key = Key$$

As $Text_{PlainText} \oplus Text_{PlainText} = 0$, XORing Cipher text and Plain text will give us the key. We will use this key and XOR it with an encrypted message stored at index 0 to obtain the flag.

$$Text_{Flag} = Text_{Cipher[0]} \oplus Key$$

Solution:

We put the plain text as 'thismessageis32bitlongandIamawar' in the CTF server.

The Hex code for the above text is

'0x746869736d657373616765697333326269746c6f6e67616e6449616d61776172'

The cipher hex returned from the CTF server is

'0xa70fbe47076ac21daac94f006e5047de6159e38e667e9ceb897923bb5c072678'

XORing the hex values for Plain text and Cipher text yields. This will be our key.

'0xd367d7346a0fb16ecbae2a691d6375bc082d8fe10819fd85ed3042d63d70470a'

XORing the key obtained with an encrypted Cipher stored at index 0 will yield.

'0x3236333134303138363736323339393937313434303730363530373636343930'

We will obtain the Flag by converting the above hex value to a String value. The flag obtained from this process is '26314018676239997144070650766490'

Code:

A very elaborate code explaining the whole process is available in the Jupyter file.

References:

i. https://textbook.cs161.org/crypto/

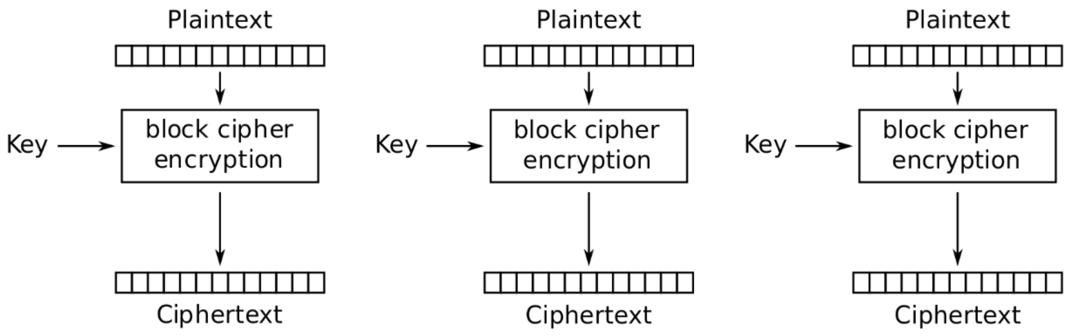
Q3> Breaking the ECB.

CTF Username: AyushShrivastava

The flag is: cf2a6e692ba606517851b39bf6ee5c52

Explanation:

ECB Mode (Electronic Code Book): In this mode the plaintext M is broken into n-bit block $M_1 \cdots M_l$,and each block is encoded using the block cipher: $C_i = E_K(M_i)$. The ciphertext is just a concatenation of these individual blocks: $C = C_1 \cdot C_2 \cdots C_l$.



Electronic Codebook (ECB) mode encryption

Any redundancy in the blocks will show through and allow us to deduce information about the plaintext. For instance, if $M_i = M_j$ then we will have $C_i = C_j$ so ECB mode leaks information about the plaintext. This makes ECB susceptible to Copy-Paste/Cut-Paste attacks.

Solution:

CTF server displays that it uses blocks of 64 bits. On breaking down the received cookies into blocks of 32 characters (64 bits), we can observe that 2nd and 3rd blocks of Admin and our Cookie are the same.

Admin Cookie:
524a1b2a207149ebf8c71e52bce2e75f 1dd3a88a2eb40bda50a10f5eb2beee1e
4318f9765c49735de40e783146382e18 1ed33309ab8e5a252633c06d96138977

My Cookie:
95f700e7eb1003486e0f050768ab1310 1dd3a88a2eb40bda50a10f5eb2beee1e
4318f9765c49735de40e783146382e18 d6eab1e710d8c8a9cc7adfd2b941281c

On closely inspecting the given python code, we can deduce that first block of message provides us the admin rights to access the first IF block since it contains the message similar to "I am yes an Administrator". Second IF contains the Date check which is present in the last part of our cookie. Hence to break the ECB we will use Copy-Paste Attack. We will change the first block of our cookie with the first block of Admin cookie. This breaks the encryptions and allows us to enter the IF blocks releasing the final flag.

The Cookie entered is:
524a1b2a207149ebf8c71e52bce2e75f 1dd3a88a2eb40bda50a10f5eb2beee1e
4318f9765c49735de40e783146382e18 d6eab1e710d8c8a9cc7adfd2b941281c

The flag obtained is cf2a6e692ba606517851b39bf6ee5c52

Additional Code.

i. Link to the folder containing all the below file: (In case any of the link changes)
https://github.com/AYUSHs799/IITGN_CS431

ii. Link to Jupyter file for breaking Vignere Cipher:
https://github.com/AYUSHs799/IITGN_CS431/blob/main/Breaking_Classic_Crypto/CS431%20-%20Vignere%20Break.ipynb

iii. Link for the Text file containing the frequency count:
https://github.com/AYUSHs799/IITGN_CS431/blob/main/Breaking_Classic_Crypto/EngFreq.txt

iv. Link to Jupyter file for breaking One Time Pad:
https://github.com/AYUSHs799/IITGN_CS431/blob/main/Breaking_Classic_Crypto/CS431%20-%20Breaking%20One%20Time%20Pad.ipynb

References.

i. https://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher

ii. http://practicalcryptography.com/cryptanalysis/letter-frequencies-various-languages/english-letter-frequencies/

iii. http://practicalcryptography.com/media/cryptanalysis/files/english_monograms.txt

iv. https://www.cipherchallenge.org/wp-content/uploads/2020/12/Five-ways-to-crack-a-Vigenere-cipher.pdf

v. https://textbook.cs161.org/crypto/