CSE, IIT Kanpur CS641 Modern Cyptology

Assignment

Nikhil Mittal - 17111056 January 23, 2018

Problem 1 : Convert hex to base64

solution Given a string with hexadecimal values need to convert it into base 64 encoding. Using python's bytes class we convert the given hex to bytes object. Then get it's base 64 encoding using in-built base64 package.

Problem 2 : Fixed XOR Write a function that takes two equal-length buffers and produces their XOR combination.

solution In this challenge Hex decode both the strings using the bytes class as in previous challenge and then perform xor operation between them and hex encode the bytes string.

Problem 3 : Single-byte XOR cipher

solution Given string is first hex decoded. Then we need to identify the single character for that we iterate among all the characters ascii - 0 to 255. Each character is xor-ed with the string to obtain original string. Now for all these strings a score is calculated based on the english frequency table and the character which gives string with highest score is selected after sorting!

Problem 4 : Detect single-character XOR One of the 60-character strings in this file has been encrypted by single-character XOR. Find it.

solution It just a simple extension of previous challenge, instead of a single string there are multiple strings. For each string all characters are used and xor-ed, all the resultant strings are sorted based on the score given using the english frequency table as done in previous challenge. So the string with maximum score is returned.

Problem 5 : Implement repeating-key XOR Encrypt stanza, under the key "ICE", using repeating-key XOR.

solution Encrypt the stanza given using the Vigenere cipher, key is given. Then encode the ciphertext and generate its corresponding hex code. Compare this value with the given hex code.

Problem 6 : Break repeating-key XOR. A file has been base64'd after being encrypted with repeating-key XOR. Decrypt it.

solution This was the best part of the set-1!
First using base 64 decoder get the decoded data. Now to break the repeating-key xor, check for all possible key-lengths from 2 to 40 (as suggested). Break into key-length size chunks, for every pair compute hamming distance. Add them and normalize it, save this for every key-length. Then get the key-lengths for the top 3 hamming distances.

Now, for each of the possible key lengths, we try to get generate the key by solving each block considering it was encrypted using single character, and the scoring function of challenge 3 is used to get the best key for each block. We join all these single characters to form the key of that length.
After all the possible keys are generated for each possible key length as shown above, we select the key that gets the best score on decrypting the original ciphertext given. By this method we select key and hence decrypt the file contents.

Problem 7 : AES in ECB mode The Base64-encoded content in a file has been encrypted via AES-128 in ECB mode under the key given. Decrypt it.

solution I used the Python's cryptography library Pycrypto for AES. The key was provided and also the mode, so just had to use the decrypt method under AES module of crypto library to get the plaintext from ciphertext.

Problem 8 : Detect AES in ECB mode In file are a bunch of hex-encoded ciphertexts.One of them has been encrypted with ECB. Detect it.

solution This is simple. For each line in ciphertext, we divide it into 16 byte blocks. Now check if any of the block repeats! If it does then this may be ECB. Under the same 16-byte plaintext block will always produce the same 16 byte ciphertext in ECB. For this I have used sets after splitting into blocks by which we can easily check if any block repeats or not.