

1845479-HW03

Carmignani Federico

November 24, 2021

Abstract

The goal of this homework is to break the AES 256 cipher.

The solution has to be sent to cns@diag.uniroma1.it from the institutional email.

The password to unzip the file with the ciphertext published by Classroom is: 'K6.*%&'.

The deadline is November 24th at 11:59 pm.

1 What is the challenge

The challenge consists in trying to simulate an attack to a given ciphertext using the AES 256 cipher. The objective is to determine the plaintext, a text in English, by attacking the generation of the 256-bits key obtained by using a seed, the right password to retrieve it, through an offline dictionary attack. The used cipher is AES, Advanced Encryption Standard, also known as Rijndael, is a symmetric block cipher algorithm used as a standard by the government of the United States of America. The same algorithm and key are used for encryption and decryption (symmetric encryption) and the key length is 256 bits. The ciphering of the plaintext can be done using modes of operation like CBC. In order to encode a file with AES, the OpenSSL library, a software for applications that secure communications over computer networks, provides a shell command to perform it: `openssl enc -aes-256-cbc -in infile.txt -out outfile.txt.enc -k password`.

The decryption is similar, adding the '-d' flag option and also the '-md md5' option. It is because OpenSSL changed message digest it uses: OpenSSL 1.0.2 still used MD5 and 1.1.0 switched to SHA256, so if the data are encrypted with 1.0.2 or older, MD5 has to be specified as the digest algorithm.

2 How to solve the challenge

The AES encryption usually needs a key and an initialization vector (IV) but it is possible to use also a single password through a derivation algorithm. In this way, we can perform the attack just operating on the password, added with the '-k' flag option.

The dictionary attack consists in trying all the possible passwords to be used contained in a specific list.

I did it using Python as programming language and the implementation will be added to the section 'How to use and run the program'.

It is a dictionary attack that tries all the words contained in the dictionary and uses a simple variation just substituting the "o" with "0". I tried before many other possible variations like: all capital letters, capitalized words, adding a number at the end of the words or making other substitutions like "i" with "1" or "e" with "&". All these attempts didn't lead me to a solution, except for the "o" with the "0", which was the right one. I tried also many dictionaries before focusing on the most famous words in English. It is usual in dictionary attacks to use also more than ten/twenty variations on every single word. So the program stopped when the decryption function was returning 0 and the output file was an ASCII text.

2.1 The steps towards the solution

In order to achieve the solution of the text, I started to generate the passwords taking them from a dictionary found online with the most used words in the English language. In the program, each step a password is taken, the substitution is done, then the OpenSSL shell command is run and if the

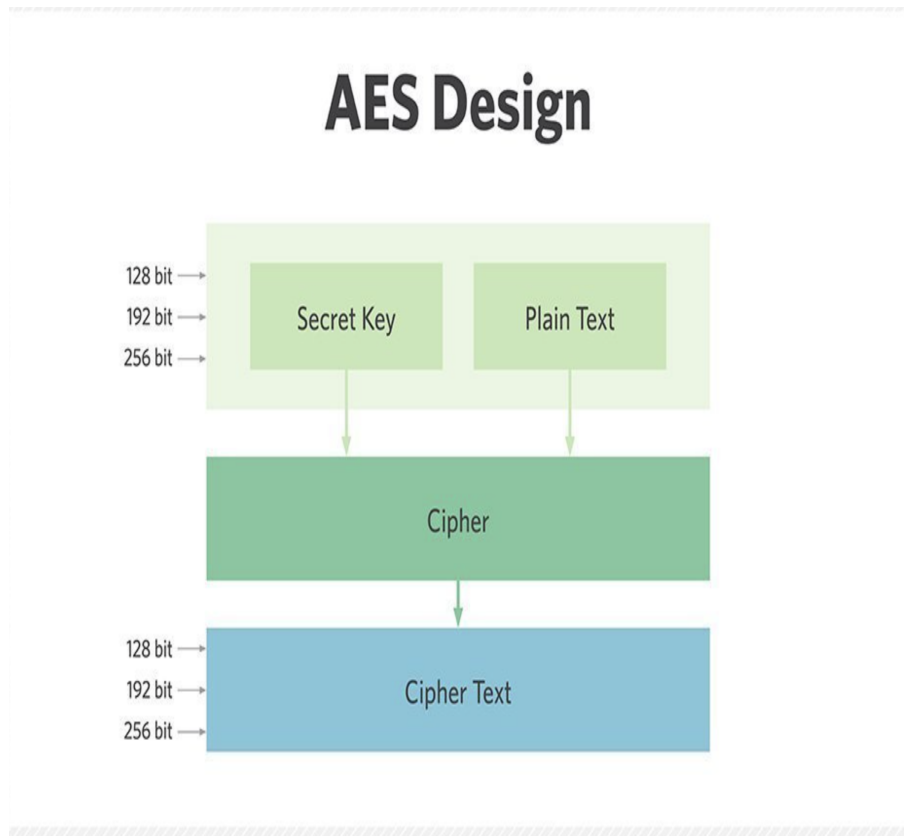


Figure 1: AES idea.

decryption is performed correctly and the file generated is formed by ASCII characters, the program ends printing the password. In order to do it faster, I exploited the parallelism of shells supported by the OS, running in parallel different programs with different possible substitutions. The dictionary used is taken from <https://github.com/first20hours/google-10000-english>.

3 How to use and run the program

The program performs what described before automatically. It has to be executed using Spider or another IDE to run the program or also in the terminal using Python interpreter with this command: 'python 1845479-HW03.py', the important thing is that the file with the ciphertext and the file of the dictionary is in the same folder of the Python script.

3.1 The code

```
#libraries
from subprocess import PIPE, run
import os

#in order to execute shell commands and save their output
def out(command):
    result = run(command, stdout=PIPE, stderr=PIPE, universal_newlines=True, shell=True)
    return result.stdout

#the command to decrypt
```

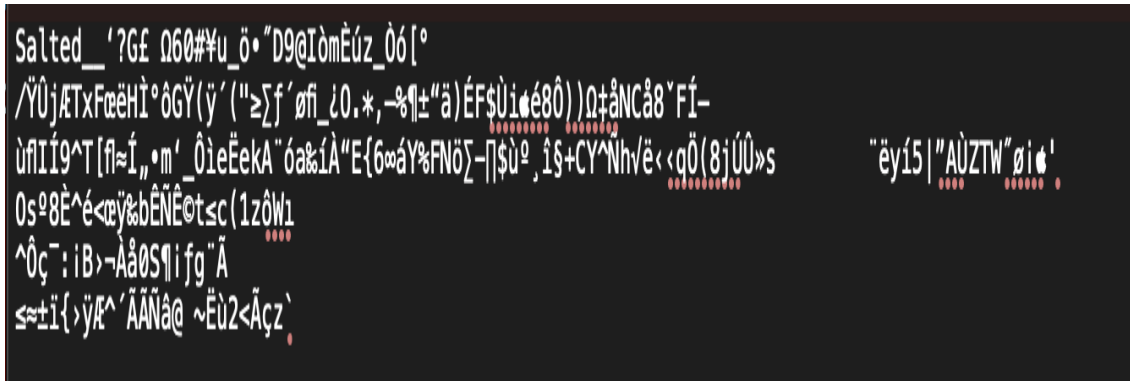


Figure 2: Demo 1.

```
command0= "openssl enc -d -aes-256-cbc -md md5 -in outfile.txt.enc -out file.txt -k "
```

```
#read the file and create a list with all the words from the dictionary
with open("google-10000-english-no-swears.txt") as file:
    lines = file.readlines()
    lines = [line.strip() for line in lines]

#replacing all 'o' with '0' as the hint given
for item in lines:
    print(item)
    item=item.replace('o','0')
    password=item.lower()

#run the command appending that password
command= command0.__add__(password)
res=os.system(command)
print(res)
#if the decrypting process is ok then exit the cycle
if(res!=256):
    print(password)
    #see if the file is ASCII text and in this case exit the program, otherwise
    #a new cycle is started
    my_output = out("file file.txt")
    print(my_output)
    if("ASCII" in my_output):
        break
```

3.2 The program in use

So, after many attempts using different dictionaries and variations, I found the password to decrypt it: *the0rem*.

An illustration is given in Figure 3, where the Python console shows the moment when the password was found.. We can see how the OpenSSL command returns 0 and not 256 (error in decrypting), and the file is ASCII text, in fact it passes the check.

The ciphertext is shown in Figure 2.

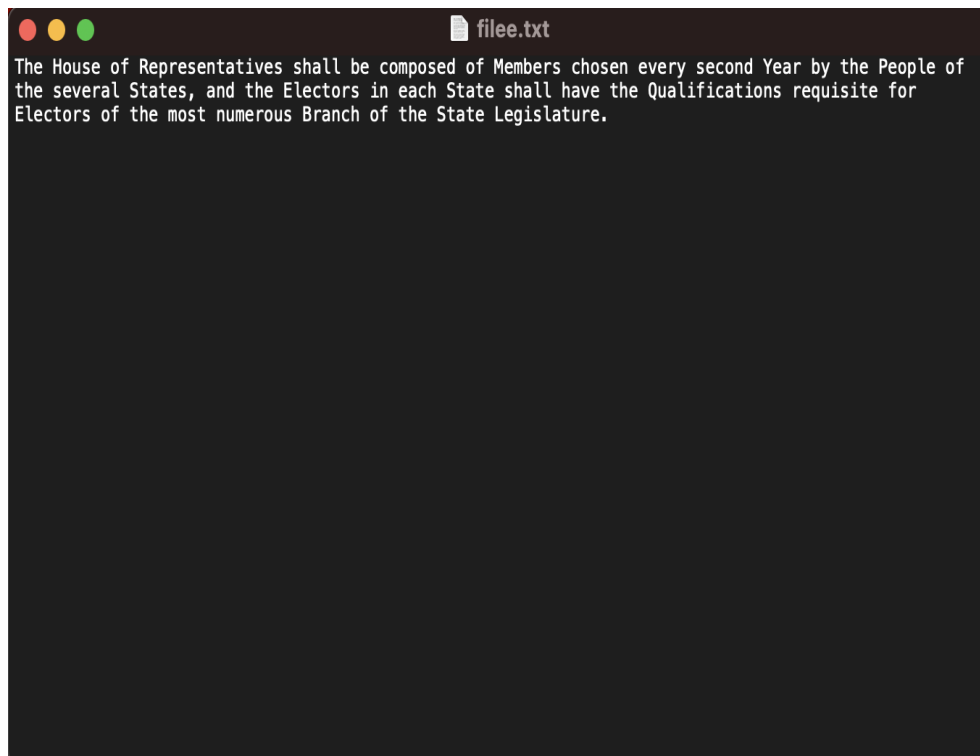
Instead, in Figure 4, it is shown the plaintext produced by the OpenSSL command: *openssl enc -d -aes-256-cbc -md md5 -in outfile.txt.enc -out file.txt -k the0rem*.

```

Using -iter or -pbkdf2 would be better.
bad decrypt
4604182016:error:06065064:digital envelope routines:EVP_DecryptFinal_ex:bad decrypt:crypto/evp/evp_enc.c:610:
256
practitioners
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
bad decrypt
4467109376:error:06065064:digital envelope routines:EVP_DecryptFinal_ex:bad decrypt:crypto/evp/evp_enc.c:610:
256
transcript
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
bad decrypt
4366454272:error:06065064:digital envelope routines:EVP_DecryptFinal_ex:bad decrypt:crypto/evp/evp_enc.c:610:
256
myspace
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
bad decrypt
4553170432:error:06065064:digital envelope routines:EVP_DecryptFinal_ex:bad decrypt:crypto/evp/evp_enc.c:610:
256
theorem
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
0
the0rem
file.txt: ASCII text

```

Figure 3: Demo 2.



```

filee.txt
The House of Representatives shall be composed of Members chosen every second Year by the People of
the several States, and the Electors in each State shall have the Qualifications requisite for
Electors of the most numerous Branch of the State Legislature.

```

Figure 4: Demo 3.