دانشگاه صنعتی شریف

دانشکده مهندسی برق

مبانی رمزنگاری و امنیت شبکه

گزارش پروژه نهایی

استاد: دکتر میرمحسنی

تهیه کننده: کامیار رجبعلی فردی(۹۷۱۰۱۶۶۱)

در این گزارش سعی داریم تا سناریوی زیر را توضیح دهیم و در میان آن درباره ی توابع ایجاد شده توضیح دهیم.

۱. فردی به اسم A در سیستم با user = Alice_2021_^--^ و pass = A!@#$%^A ثبت نام می‌کند.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

2
========== || Home || ==========

username:Alice_2021_^--^

password:A!@#$%^A


successfully registered!
```

۲. A ابتدا میخواهد وارد سیستم شود ولی رمز خود را اشتباه وارد می‌کند و نمی‌تواند وارد شود.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

1
========== || Home || ==========

username:Alice_2021_^--^

password:A!@#$%^B


Access denied ❌
```

۳. A وارد سیستم می‌شود.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

1
========== || Home || ==========

username:Alice_2021_^--^

password:A!@#$%^A


Access granted ✅
```

۴. A دامنه و پسورد های آنها را که در زیر نشان داده شده اند در سیستم ذخیره می‌کند.

User A	
Website	Password
www.gmail.com	Alice123QWE
www.amazon.com	123456

```
========== || User Account || ==========
0 to Go to Home
1 to save your passwords
2 to load your password
3 to change your passwords

1
========== || User Account || ==========

Do you want to add any address and password(YES/NO)? YES

Enter your address: www.gmail.com

Enter your password: Alice123QWE

Address & Password have Saved successfully!

========== || User Account || ==========

Do you want to add any address and password(YES/NO)? YES

Enter your address: www.amazon.com

Enter your password: 123456

Address & Password have Saved successfully!
```

۵. A از سیستم درخواست رمز www.gmail.com را می‌کند تا آنرا برایش نمایش دهد.

```
========== || User Account || ==========
0 to Go to Home
1 to save your passwords
2 to load your password
3 to change your passwords

2
========== || User Account || ==========

Have you forgotten any of your passwords(YES/NO)? YES

Enter your address: www.gmail.com
your Password for  www.gmail.com  :  Alice123QWE
```

۶. A تصمیم می‌گیرد تا رمز وبسایت www.amazon.com را به 6U)qA10By%3SZX$o تغییر دهد.

```
2 to load your password
3 to change your passwords

3
========== || Privacy & Policy || ==========
0 to Go to User Account
1 to change login password
2 to change password of a website

2

Enter your address: www.amazon.com

Enter your new password: 6U)qA10By%3SZX$o
The Password of the Address has changed successfully!

========== || Privacy & Policy || ==========
0 to Go to User Account
1 to change login password
2 to change password of a website

0
========== || User Account || ==========
0 to Go to Home
1 to save your passwords
2 to load your password
3 to change your passwords
```

۷. A با وارد کردن وبسایت های مورد نظر خود سعی تمامی رمز های خود را از سیستم دریافت می‌کند.

User A	
Website	Password
www.gmail.com	Alice123QWE
www.amazon.com	6U)qA10By%3SZX$o

```
========== || User Account || ==========
0 to Go to Home
1 to save your passwords
2 to load your password
3 to change your passwords

2
========== || User Account || ==========

Have you forgotten any of your passwords(YES/NO)? YES

Enter your address: www.amazon.com
your Password for  www.amazon.com  :  6U)qA10By%3SZX$o

========== || User Account || ==========

Have you forgotten any of your passwords(YES/NO)? YES

Enter your address: www.gmail.com
your Password for  www.gmail.com  :  Alice123QWE
```

۸. از سیستم خارج می‌شود.

۹. فردی به اسم B می‌خواهد با نام و رمز A ثبت نام کند ولی با خطا رو به رو می‌شود.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

2
========== || Home || ==========

username:Alice_2021_^--^

password:A!@#$%^A


A user with the same username and password has already been registered!!
```

۱۰. B در سیستم با user = I_M_BOB!! و pass = BOB’s your uncle ثبت نام می‌کند.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

2
========== || Home || ==========

username:I_M_BOB!!

password:BOB's your uncle


successfully registered!
```

۱۱. B وارد سیستم می‌شود.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

1
========== || Home || ==========

username:I_M_BOB!!

password:BOB's your uncle


Access granted ✅
```

۱۲. B دامنه و پسورد های آنها را که در زیر نشان داده شده اند در سیستم ذخیره می‌کند.

User B	
Website	Password
www.google.com	Barking_dog_seldom_bite
ee.sharif.edu	95101234

```
========== || User Account || ==========
0 to Go to Home
1 to save your passwords
2 to load your password
3 to change your passwords

1
========== || User Account || ==========

Do you want to add any address and password(YES/NO)? YES

Enter your address: www.google.com

Enter your password: Barking_dog_seldom_bite

Address & Password have Saved successfully!

========== || User Account || ==========

Do you want to add any address and password(YES/NO)? YES

Enter your address: ee.sharif.edu

Enter your password: 95101234

Address & Password have Saved successfully!
```

۱۳. B رمز ورود خود به سیستم را به enromous_pixel61 تغییر می‌دهد.

```
========== || User Account || ==========

Do you want to add any address and password(YES/NO)? NO
========== || User Account || ==========
0 to Go to Home
1 to save your passwords
2 to load your password
3 to change your passwords

3
========== || Privacy & Policy || ==========
0 to Go to User Account
1 to change login password
2 to change password of a website

1

Enter your new password: enromous_pixel61
Login Password havs changed successfully!
```

۱۴. B از سیستم خارج می‌شود و دوباره سعی می‌کند با رمز قبلی خود وارد شود.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

1
========== || Home || ==========

username:I_M_BOB!!

password:BOB's your uncle


Access denied ❌
```

۱۵. B با رمز جدید وارد سیستم می‌شود.

```
========== || Home || ==========
0 to logout
1 to login
2 to register

1
========== || Home || ==========

username:I_M_BOB!!

password:enromous_pixel61


Access granted ✅
```

۱۶. B تمامی رمز های خود را از سیستم دریافت می‌کند و خارج می‌شود.

```
========== || User Account || ==========
0 to Go to Home
1 to save your passwords
2 to load your password
3 to change your passwords

2
========== || User Account || ==========

Have you forgotten any of your passwords(YES/NO)? YES

Enter your address: www.google.com
your Password for  www.google.com  :  Barking_dog_seldom_bite

========== || User Account || ==========

Have you forgotten any of your passwords(YES/NO)? YES

Enter your address: ee.sharif.edu
your Password for  ee.sharif.edu  :  95101234
```

عکس زیر هم نشان دهنده ی محتوای دیتابیس است:

```
72eb7db81b5e6b26194ef94355ca8c762b51773ca35ba6ce31dd0a15cfcf54df5b349b2841988137d3d8005380c4e6df37497363e0f544faa262fb2f44bc4bb3
8cee0f78da8b5a988620ddda2702f6cd423310ce7732fe3565ad1bf3e316398f
455b5fafd0ac8987ef074d39dafe2ccf3d07a8e3ee51410e2c2cc1456b174bea
ca1d15ad1c627fcb4dff59c986600e8ca5c23b2dfd4699765874091ac6b9a543
60038ad9370b88121a55504c9ae8df3d55ef8504710ac97fecf158f4451d089a

3e5538a444dd6e6be19e9da9377c6e3ca1c6b15925283bed7e5bd4935d82a2290231b5e5203aacfad19a19dbd0324033073792cde048019dc26658765b530936
553851e8a9caf15fb3e25c5e9ab64ad7187bc996a84e2518c211ce9980658c6f
25f139a678476a9f3c25a99a570e169f
f1e269afd3de19b35f8f1c9b54d4e3a8cfab0bdf3158390422c84e6c57f9987a
e099d9b097d988fe116fd7c8af4460867ea87ba657433996f079b585491af102
```

خط اول هش شده ی ترکیبی از رمز و نام کاربری A است. از خط دوم تا پنجم به ترتیب وبسایت و پسورد متناظر به آن بصورت رمز شده(همراه با salt) آورده می‌شود. و بعد از یک کاراکتر \n، اطلاعات کاربرد بعدی شروع می‌شود و الی اخر.

حال درباره ی کد زده شده با پایتون توضیح می‌دهیم. این کد بر اساس یک ماشین حالت نوشته شده است بطوریکه متناسب با هر دستوری که کاربر وارد می‌کند، حالت خود را به یک حالت مناسب تغییر می‌دهد. زمانی که کاربر می‌خواهد در سایت ثبت نام کند تابع add_user فراخوانی می‌شود و پس از ثبت نام کاربر در برنامه به حالت idle بازمی‌گردیم.

```
if state == '0010':
    print('==========','|| Home ||','==========')
    username = input('username:')
    password = input('password:')
    print('\n')
    add_user(username,password)
    state = '0000'
```

تابع add_user بصورت زیر است:

```
def add_user(username,password):
    my_file = open('dataset.txt','r')
    temp = my_file.readlines()
    my_file.close()
    salt = salt_lcg(password + username)
    temp_hash = hash_salt(salt,username+password)
    for i in temp:
        if temp_hash == i[0:-1]:
            print('A user with the same username and password has already been registered!!\n')
            return
    print('successfully registered!\n')
    my_file = open('dataset.txt','a')
    my_file.write(temp_hash+'\n')
    my_file.write('\n')
    my_file.close()
```

در این تابع ابتدا فایل دیتابیس باز شده و محتویات آن خوانده می‌شود سپس با استفاده از تابع پیاده سازی شده salt_lcg از پسورد و نام کاربری یک salt ساخته می‌شود که بصورت زیر است:

```
def salt_lcg(a_string):
    a_byte_array = bytearray(a_string, "ascii")
    byte_list = []
    for byte in a_byte_array:
        binary_representation = int(byte)
        byte_list.append(binary_representation)
    rand = sum(byte_list)
    a = 1140671485
    c = 128201163
    m = 2**24
    for i in range(6):
        rand = (a*rand + c) % m
    return str(rand)
```

این تابع براساس یک مولد همنهشتی خطی با تناوب کامل عمل می‌کند. ورودی این تابع نام کاربری و رمز آن است و سپس کد اسکی این دو رشته ساخته شده و معادل دسیمال آنها با هم جمع می‌شوند و متغیر rand را تشکیل میدهند. حال این متغیر در یک مولد همنهشتی خطی وارد شده و یک خروجی شبه رندوم متناسب با هر کاربر ساخته خواهد شد.

پس از اینکه salt ساخته شد تابع hash_salt که بصورت زیر است ساخته می‌شود:

```
def hash_salt(salt,a_string):
    temp = salt + a_string
    return hashlib.sha512(temp.encode()).hexdigest()
```

در این تابع salt با رشته concat شده و از آنها با استفاده از sha512 هش گرفته می‌شود.

سپس این هش با محتوای دیتابیس مقایسه می‌شود و اگر مشابه آن وجود داشت اخطار داده می‌شود. در غیر این صورت در دیتابیس ذخیره شده و به حالت idle بازگشته می‌شود. حال که کاربر ثبت نام کرد با درخواست ورود تابع authenticate_user را فراخوانی می‌کند و اگر احراز اصالت شد وارد سیستم می‌شود. در غیر این صورت با خطا مواجه می‌شود.

```
if state == '0001':
    print('==========','|| Home ||','==========')
    username = input('username:')
    password = input('password:')
    print('\n')
    (access_loc,salt,master_key) = authenticate_user(username,password)
    if access_loc != -1:
        print('\x1b[1;36;40m' + 'Access granted ✓\n' + '\x1b[0m')
        state = '0011' #user Account
        #print(style.WHITE)
    else:
        print('\x1b[1;31;40m' + 'Access denied ✗\n' + '\x1b[0m')
        state = '0000'
```

```
def authenticate_user(username,password):
    salt = salt_lcg(username+password)
    my_file = open('dataset.txt')
    temp = my_file.readlines()
    my_file.close()
    flag = 0
    for i in range(0,len(temp)):
        if temp[i][0:-1].find(hash_salt(salt,username+password)) != -1:
            flag = 1
            break
    if flag == 1:
        return (i,salt,master_key_generator(salt,username,password))
    else:
        return (-1,-1,-1)

def master_key_generator(salt,username,password):
    temp = hash_salt(salt,username+password+username)
    return temp[0:32]
```

در تابع authenticate_user ابتدا salt ساخته می‌شود و سپس از نام کاربری و رمز عبور به روش توضیح داده شده در بالا هش گرفته شده و با محتویات داخل دیتابیس مقایسه می‌شود. در صورت موفقیت امیز بودن احراز اصالت کاربر، تابع master_key_generator فراخوانی می‌شود و برای کاربر یک master_key که حاصل هش گرفتن salt||user||pass|user می‌باشد ساخته خواهد شد. چون از سیستم رمزگذاری AES با طول کلید ۱۶ بایت استفاده می‌شود

و هش گرفته شده ۲۵۶ بایت دارد، بنابراین تنها ۳۲ کاراکتر اول کلید را درنظر می‌گیریم و از بقیه ی آن صرف نظر می‌کنیم. پس از ساخته شدن master_key، یک تاپل به عنوان خروجی تابع authemticate_user ساخته می‌شود که شامل salt, master_key و موقعیت اشاره‌گر در فایل که کاربر مورد نظر را نشان می‌دهد می‌باشد. پس از خروج از این تابع وارد سیستم می‌شویم.

حال فرض کنید که کاربر می‌خواهد دامنه و پسورد را وارد کند. در این حالت تابع save_URL_PASS فراخوانی می‌شود.

```
if state == '0100':
    while(True):
        print('==========','|| User Account ||','==========')
        command = input('Do you want to add any address and password(YES/NO)? ')
        if command == 'YES':
            address = input('Enter your address: ')
            password_for_address = input('Enter your password: ')
            print('')
            save_URL_PASS(access_loc + 1, master_key, salt, address, password_for_address)
            #pointer += 1
        if command == 'NO':
            state = '0011'
            break
```

```
def save_URL_PASS(pointer, master_key, salt, address, password_for_address):
    my_file = open('dataset.txt')
    temp = my_file.readlines()
    my_file.close()
    temp.insert(pointer , AES_encrypt(master_key,salt + address) + '\n')
    temp.insert(pointer+1 , AES_encrypt(master_key,salt + password_for_address) + '\n')
    my_file = open('dataset.txt','w')
    my_file.writelines(temp)
    my_file.close()
    print('Address & Password have Saved successfully!\n')
    return
```

در این تابع به وسیله ی master_key و سیستم AES، salt||address رمز می‌شود و به عنوان دامنه ذخیره می‌شودو به دنبال آن salt||password_for_address رمز می‌شود و در دیتابیس ذخیره می‌شود.

حال اگر کاربر قصد گرفتن رمز خود متناظر با یک وبسایت را از سیستم را داشته باشد تابع load_URL_PASS فراخوانی می‌شود.

```
if state == '0101':
    while(True):
        pointer = access_loc + 1
        print('==========','|| User Account ||','==========')
        command = input('Have you forgotten any of your passwords(YES/NO)? ')
        if command == 'YES':
            address = input('Enter your address: ')
            load_URL_PASS(pointer, master_key, salt, address)
        if command == 'NO':
            state = '0011'
            break
```

```
def load_URL_PASS(pointer, master_key, salt, address):
    my_file = open('dataset.txt','r')
    temp = my_file.readlines()
    my_file.close()
    cipher_temp = AES_encrypt(master_key,salt+address)
    i = pointer
    while(temp[i] != '\n'):
        if temp[i][0:-1] == cipher_temp:
            plaintext = AES_decrypt(master_key,temp[i+1][0:-1])
            print('your Password for ',address,' : ','\x1b[1;32;40m' + plaintext[len(salt)::] + '\x1b[0m','\n')
            return
        i += 1
    print('You havent any address like this in our database!\n')
    return
```

در این تابع وبسایت مورد نظر کاربر دریافت می‌شود و سپس salt||address با AES و master_key رمز شده و با محتویات موجود در دیتابیس مقایسه می‌شود. در صورت نبود چنین وبسایتی اخطار داده می‌شود و در غیر این صورت رمز متناظر با آن وبسایت به وسیله ی master_key رمزگشایی شده و برگردانده می‌شود.

حال فرض کنید کاربر قصد دارد تا رمز عبور به برنامه را تغییر دهد. در این صورت تابع Change_Login_Password فراخوانی می‌شود که بصورت زیر است:

```
def Change_Login_Password(pointer, new_salt, new_master_key, new_password, salt, master_key, username ,password):
    my_file = open('dataset.txt')
    temp = my_file.readlines()
    my_file.close()
    i = pointer
    temp[i] = hash_salt(new_salt, username + new_password) + '\n'
    i += 1
    while(temp[i] != '\n'):
        temp[i] = AES_encrypt(new_master_key,new_salt + AES_decrypt(master_key, temp[i][0:-1])[len(salt)::]) + '\n'
        i+=1
    my_file = open('dataset.txt','w')
    my_file.writelines(temp)
    my_file.close()
    print('Login Password havs changed successfully!\n')
    return
```

پس از آنکه از کاربر رمز جدید دریافت شد، یک new_salt, new_master_key ایجاد می‌شود و پس از آن تمامی محتویات متناظر با کاربر در دیتابیس با کلید و salt جدید رمز گذاری و هش گرفته می‌شوند.(برای رمزگذاری ابتدا با کلید قدیمی رمزگشایی شده و سپس با کلید جدید رمز می‌شوند.)

حال فرض کنید کاربر قصد دارد تا رمز متناظر با یک دامنه را تغییر دهد در این صورت تابع Change_Web_Password فراخوانی می‌شود.

```
def Change_Web_Password(pointer, master_key, salt, address, new_password):
    my_file = open('dataset.txt','r')
    temp = my_file.readlines()
    my_file.close()
    cipher_temp = AES_encrypt(master_key, salt+address)
    i = pointer
    while(temp[i] != '\n'):
        if temp[i][0:-1] == cipher_temp:
            temp[i+1] = AES_encrypt(master_key, salt + new_password) + '\n'
            my_file = open('dataset.txt','w')
            my_file.writelines(temp)
            my_file.close()
            print('The Password of the Address has changed successfully!\n')
            return
        i += 1
    print('You havent any address like this in our database!\n')
    return
```

پس از دریافت رمز جدید مورد نظر و وبسایت آن از کاربر، در دیتابیس وبسایت مورد نظر جستجو شده و رمز آن را تغییر می‌دهیم.

در هنگام رمزگذاری و رمزگشایی از یک سیستم AES با طول کلید ۱۶ بایت استفاده می‌کنیم که آن را مشابه توضیحات کتاب stallings پیاده سازی کرده ایم که بصورت زیر است:

ابتدا کاراکترهایی را که کاربر وارد می‌کند به ascii تبدیل می‌کنیم. سپس به این رشته ی تولید شده تا اولین مضرب ۱۶ بزرگتر از طول آن pad 0 می‌کنیم. زیرا در کد اسکی 00 نشان دهنده ی NULL است و کاربر قطعا نمی‌تواند این کاراکتر را وارد کند. بنابراین این نوع از padding هیچ مشکلی در رمزگشایی پیام بصورت کاراکتر ایجاد نخواهد کرد و امینت را افزایش می‌دهد. پس از آن این رشته را به بلوک های ۱۶ بایتی تقسیم کرده و هر کدام را به سیستم AES با کلید مطلوب می‌دهیم. اما قبل از آن هر بلوک ۱۶ بایتی را با توابع پیاده سازی شده توسط خودمان به ماتریس های ۴*۴ تبدیل می‌کنیم.

```
def string2hex(a_string):
    return ''.join([format(ord(i),'x') for i in a_string])

def hex2string(a_string):
    return bytearray.fromhex(a_string).decode()

def hex2table(a_string):
    return np.array(textwrap.wrap(a_string,2)).reshape(4,4).transpose().tolist()

def xor_strings(xs, ys):
    a = hex(int(xs, 16) ^ int(ys, 16))[2::]
    if len(a) == 1:
        return '0' + a
    else:
        return a
```

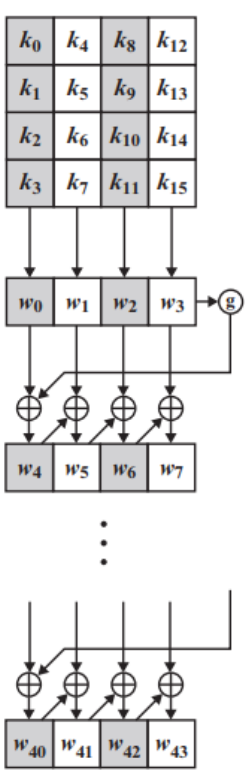
```
def AES_encrypt(key,plain):
    temp = string2hex(plain)
    temp = temp + '0'*((int(len(temp)/32)+1)*32 - len(temp))
    cipher = ''
    for i in range(0,int(len(temp)/32)):
        C = block_AES_encrypt(key, temp[32*i:32*i+32])
        cipher = cipher + C
    return cipher
```

```
def block_AES_encrypt(key,plain):
    W = key_expansion(key)
    table_plain = hex2table(plain)
    num_round = 0
    table_plain = add_round_key(num_round,W,table_plain)
    num_round += 1

    while(num_round <= 10):
        for i in range(0,4):
            for j in range(0,4):
                table_plain[i][j] = S_BOX(table_plain[i][j])
        table_plain = shift_rows(table_plain)
        if num_round <= 9:
            table_plain = mix_column(table_plain)
        table_plain = add_round_key(num_round,W,table_plain)
        num_round += 1
    cipher = ''
    for i in range(0,4):
        for j in range(0,4):
            cipher = cipher + table_plain[j][i]
    return cipher
```

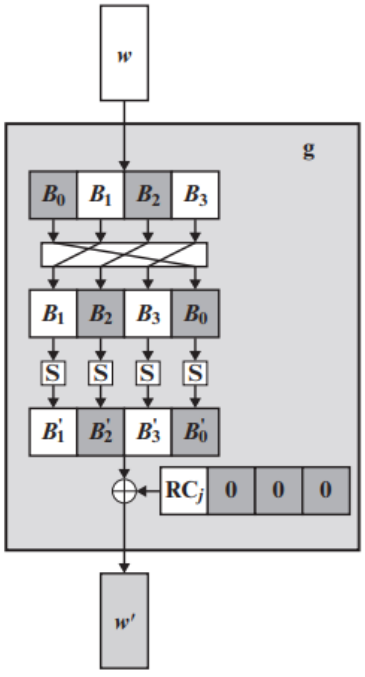
در تابع block_AES_encrypt ابتدا تابع key_expansion فراخوانی می‌شود که از کلید ورودی، ۴۴ کلید می‌سازد.

```python
def key_expansion(key):
    W = [[0 for i in range(0,44)]for j in range(0,4)]
    table_key = hex2table(key) #hex2table(string2hex(key))
    for j in range(0,4):
        for i in range(0,4):
            W[i][j] = table_key[i][j]

    num_round = 0
    while num_round <10:
        for i in range(4*num_round+4,4*num_round + 8):
            B = g(num_round,W)
            for j in range(0,4):
                if i == 4*num_round +4 :
                    W[j][i] = xor_strings(W[j][i-4],B[j])
                else:
                    W[j][i] = xor_strings(W[j][i-4],W[j][i-1])
        num_round += 1
    return W
```

```python
def g(num_round, W):
    B = deque([W[i][4*num_round+3] for i in range(0,4)])
    B.rotate(-1)
    B = list(B)
    B = [S_BOX(B[i]) for i in range(0,len(B))]
    RC = ['01', '02', '04', '08', '10', '20', '40', '80', '1b', '36']
    B[0] = xor_strings(B[0],RC[num_round])
    return B
```



(a) Overall algorithm



(b) Function g

پس از آن تابع add_round_key فراخوانی می‌شود که در round = 0 W0 با بلوک ۴*۴ متن رمز شده xor می‌شود.

```
def add_round_key(num_round,W,table_plain):
    table_key = [[W[i][j]for j in range(4*num_round,4*num_round+4)]for i in range(0,4) ]
    temp = [[0 for i in range(0,4)]for j in range(0,4)]
    for i in range(0,4):
        for j in range(0,4):
            temp[i][j] = xor_strings(table_plain[i][j],table_key[i][j])
    return temp
```

پس از آن در هر round روی بلوک متن الگوریتم های Substitute, shift rows , mix column اجرا می‌شود که هرکدام از آنها در زیر آورده شده است.

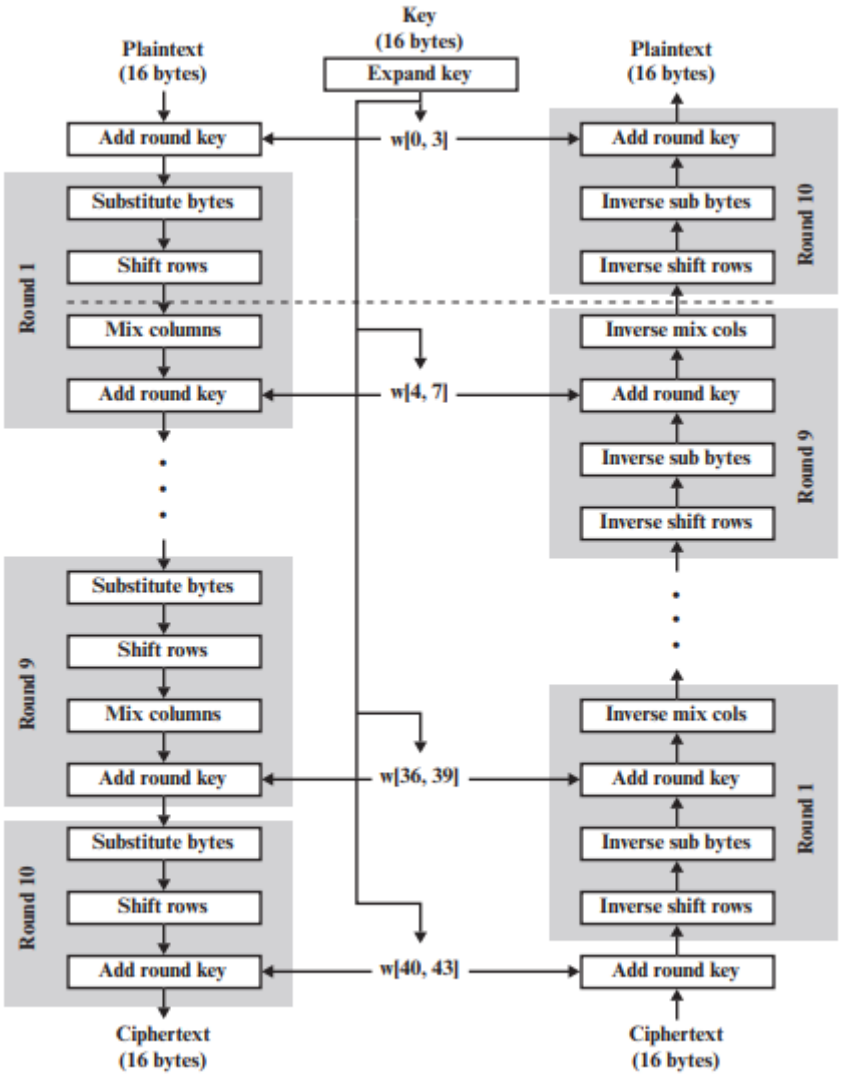
```
def S_BOX(a_string):
    S = [['63', '7C', '77', '7B', 'F2', '6B', '6F', 'C5', '30', '01', '67', '2B', 'FE', 'D7', 'AB', '76'],
         ['CA', '82', 'C9', '7D', 'FA', '59', '47', 'F0', 'AD', 'D4', 'A2', 'AF', '9C', 'A4', '72', 'C0'],
         ['B7', 'FD', '93', '26', '36', '3F', 'F7', 'CC', '34', 'A5', 'E5', 'F1', '71', 'D8', '31', '15'],
         ['04', 'C7', '23', 'C3', '18', '96', '05', '9A', '07', '12', '80', 'E2', 'EB', '27', 'B2', '75'],
         ['09', '83', '2C', '1A', '1B', '6E', '5A', 'A0', '52', '3B', 'D6', 'B3', '29', 'E3', '2F', '84'],
         ['53', 'D1', '00', 'ED', '20', 'FC', 'B1', '5B', '6A', 'CB', 'BE', '39', '4A', '4C', '58', 'CF'],
         ['D0', 'EF', 'AA', 'FB', '43', '4D', '33', '85', '45', 'F9', '02', '7F', '50', '3C', '9F', 'A8'],
         ['51', 'A3', '40', '8F', '92', '9D', '38', 'F5', 'BC', 'B6', 'DA', '21', '10', 'FF', 'F3', 'D2'],
         ['CD', '0C', '13', 'EC', '5F', '97', '44', '17', 'C4', 'A7', '7E', '3D', '64', '5D', '19', '73'],
         ['60', '81', '4F', 'DC', '22', '2A', '90', '88', '46', 'EE', 'B8', '14', 'DE', '5E', '0B', 'DB'],
         ['E0', '32', '3A', '0A', '49', '06', '24', '5C', 'C2', 'D3', 'AC', '62', '91', '95', 'E4', '79'],
         ['E7', 'C8', '37', '6D', '8D', 'D5', '4E', 'A9', '6C', '56', 'F4', 'EA', '65', '7A', 'AE', '08'],
         ['BA', '78', '25', '2E', '1C', 'A6', 'B4', 'C6', 'E8', 'DD', '74', '1F', '4B', 'BD', '8B', '8A'],
         ['70', '3E', 'B5', '66', '48', '03', 'F6', '0E', '61', '35', '57', 'B9', '86', 'C1', '1D', '9E'],
         ['E1', 'F8', '98', '11', '69', 'D9', '8E', '94', '9B', '1E', '87', 'E9', 'CE', '55', '28', 'DF'],
         ['8C', 'A1', '89', '0D', 'BF', 'E6', '42', '68', '41', '99', '2D', '0F', 'B0', '54', 'BB', '16']  ]
    return S[int(a_string[0],16)][int(a_string[1],16)].lower()
```

```
def shift_rows(table_plain):
    temp = [[0 for i in range(0,4)]for j in range(0,4)]
    for i in range(0,4):
        temp[i] = deque(table_plain[i])
        temp[i].rotate(-i)
        temp[i] = list(temp[i])
    return temp
```

```
def multiply(b,a):
    if b == 1:
        return a
    tmp = (a<<1) & 0xff
    if b == 2:
        return tmp if a <= 127 else tmp^0x1b
    if b == 3:
        return tmp^a if a <= 127 else (tmp^0x1b)^a
```

```
def mix(S):
    b = [hex(multiply(2,int(S[0],16)) ^ multiply(3,int(S[1],16)) ^ multiply(1,int(S[2],16)) ^ multiply(1,int(S[3],16)))
         ,
         hex(multiply(1,int(S[0],16)) ^ multiply(2,int(S[1],16)) ^ multiply(3,int(S[2],16)) ^ multiply(1,int(S[3],16)))
         ,
         hex(multiply(1,int(S[0],16)) ^ multiply(1,int(S[1],16)) ^ multiply(2,int(S[2],16)) ^ multiply(3,int(S[3],16)))
         ,
         hex(multiply(3,int(S[0],16)) ^ multiply(1,int(S[1],16)) ^ multiply(1,int(S[2],16)) ^ multiply(2,int(S[3],16)))]
    for i in range(0,4):
        b[i] = b[i][2::]
    return b
```

```
def mix_column(table_plain):
    temp0 = [[0 for i in range(0,4)]for j in range(0,4)]
    for i in range(0,4):
        temp = mix([table_plain[0][i],table_plain[1][i],table_plain[2][i],table_plain[3][i]])
        temp0[0][i] = temp[0]
        temp0[1][i] = temp[1]
        temp0[2][i] = temp[2]
        temp0[3][i] = temp[3]
    return temp0
```



پس از رمزگذاری تمامی بلوک های ۱۶ تایی رمز شده در یک خط نوشته شده و در فایل موجود در دیتابیس ذخیره خواهند شد.
برای رمز گشایی نیز بصورت کاملا مشابه عمل می‌کنیم و با ترتیب کلید برعکس و همچنین عکس الگوریتم های موجود متن مورد نظر را استخراج می‌کنیم. برای جلوگیری از طولانی شدن گزارش از آوردن کد های آنها در این بخش صرف نظر می‌کنیم.