B02

April 29, 2022

[1]:

```
sbox=dict()
sbox['0000']='1001'
sbox['0001']='0100'
sbox['0010']='1010'
sbox['0011']='1011'
sbox['0100']='1101'
sbox['0101']='0001'
sbox['0110']='1000'
sbox['0111']='0101'
sbox['1000']='0110'
sbox['1001']='0010'
sbox['1010']='0000'
sbox['1011']='0011'
sbox['1100']='1100'
sbox['1101']='1110'
sbox['1110']='1111'
sbox['1111']='0111'
```

[2]:

```
mult = {}

mult["0100"] = {"0000" : "0000", "0001" : "0100", "0010" : "1000", "0011" :␣
 ↪"1100",                  "0100" : "0011", "0101" : "0111", "0110" : "1011",␣
 ↪"0111" : "1111",                  "1000" : "0110", "1001" : "0010", "1010" :␣
 ↪"1110", "1011" : "1010",                  "1100" : "0101", "1101" : "0001",␣
 ↪"1110" : "1101", "1111" : "1001"}

mult["0010"] = {"0000" : "0000", "0001" : "0010", "0010" : "0100", "0011" :␣
 ↪"0110",                 "0100" : "1000", "0101" : "1010", "0110" : "1100",␣
 ↪"0111" : "1110",                 "1000" : "0011", "1001" : "0001", "1010" :␣
 ↪"0111", "1011" : "0101",                 "1100" : "1011", "1101" : "1001",␣
 ↪"1110" : "1111", "1111" : "1101"}

mult["1001"] = {"0000" : "0000", "0001" : "1001", "0010" : "0001", "0011" :␣
 ↪"1000",                 "0100" : "0010", "0101" : "1011", "0110" : "0011",␣
 ↪"0111" : "1010",                 "1000" : "0100", "1001" : "1101", "1010" :␣
 ↪"0101", "1011" : "1100",                 "1100" : "0110", "1101" : "1111",␣
 ↪"1110" : "0111", "1111" : "1110"}
```

[3]:

```
def nibblesubs(N,inv=0):
  n=int(len(N)/2)
  left=N[:n]
  right=N[n:]
  l=""
  r=""
  for i in range(n):
    l=l+str(left[i])
    r=r+str(right[i])

  if inv == 0:
      s=sbox[l]+sbox[r]

  else:
    decryptionsbox=dict()
    for k,v in sbox.items():
        decryptionsbox[v]=k
    s=decryptionsbox[l]+decryptionsbox[r]

  output=[]
  for i in s:
    output.append(int(i))

  return output
```

[4]:

```
def shiftrow(N):
  N0=N[:4]
  N1=N[4:8]
  N2=N[8:12]
  N3=N[12:16]

  return N0+N3+N2+N1
```

[5]:

```
def mixcolumns(N):
    N0=N[:4]
    N1=N[4:8]
    N2=N[8:12]
    N3=N[12:16]
    S_00 = exor(N0, [int(x) for x in mult["0100"][getString(N1)]])
    S_01 = exor(N2, [int(x) for x in mult["0100"][getString(N3)]])
    S_10 = exor(N1, [int(x) for x in mult["0100"][getString(N0)]])
    S_11 = exor(N3, [int(x) for x in mult["0100"][getString(N2)]])
    return S_00+S_10+S_01+S_11
```

[6]:

```
def invmixcolumns(N):
    N0=N[:4]
    N1=N[4:8]
    N2=N[8:12]
    N3=N[12:16]
    S_00 = exor([int(x) for x in mult["1001"][getString(N0)]],[int(x) for x in␣
 ↪mult["0010"][getString(N1)]])
    S_01 = exor([int(x) for x in mult["1001"][getString(N2)]],[int(x) for x in␣
 ↪mult["0010"][getString(N3)]])
    S_10 = exor([int(x) for x in mult["1001"][getString(N1)]],[int(x) for x in␣
 ↪mult["0010"][getString(N0)]])
    S_11 = exor([int(x) for x in mult["1001"][getString(N3)]],[int(x) for x in␣
 ↪mult["0010"][getString(N2)]])
    return S_00+S_10+S_01+S_11
```

[7]:

```
def rotatenibble(N):
  n=int(len(N)/2)
  left=N[:n]
  right=N[n:]

  return right+left
```

[8]:

```
def exor(a,b):
  out=[]
  for i in range(len(a)):
    out.append(a[i]^b[i])
  return out
```

[9]:

```
def keyschedule(k):
  #converting string to list for easy calculations
  key=[]
  for i in k:
   key.append(int(i))

  w=[]
  w.append(key[:8])
  w.append(key[8:])
  w.append(exor(exor(w[0],[1,0,0,0,0,0,0,0]),nibblesubs(rotatenibble(w[1]))))
  w.append(exor(w[2],w[1]))
  w.append(exor(exor(w[2],[0,0,1,1,0,0,0,0]),nibblesubs(rotatenibble(w[3]))))
  w.append(exor(w[4],w[3]))
  K0=w[0]+w[1]
  K1=w[2]+w[3]
  K2=w[4]+w[5]
  return K0,K1,K2
```

[10]:

```
def encryption(K0,K1,K2,text):
  t=[]
  for i in text:
    t.append(int(i))
  #Round 0
  round0=exor(t,K0)

  #Round 1"
  nbsub1=nibblesubs(round0[:8])
  nbsub2=nibblesubs(round0[8:])
  nbsub=nbsub1+nbsub2
  sr=shiftrow(nbsub)
  mc=mixcolumns(sr)
  round1=exor(mc,K1)

  #Round 2
  finalnbsub1=nibblesubs(round1[:8])
  finalnbsub2=nibblesubs(round1[8:])
  finalnbsub=finalnbsub1+finalnbsub2
  finalsr=shiftrow(finalnbsub)

  ciphertext=exor(finalsr,K2)
  return ciphertext
```

[11]:

```
def decryption(K0,K1,K2,cipher):
    t=[]
    for i in cipher:
        t.append(int(i))

    #Round 2
    round2=exor(t,K2)

    #Round 1
    sr=shiftrow(round2)
    invnbsub1=nibblesubs(sr[:8],1)
    invnbsub2=nibblesubs(sr[8:],1)
    invnbsub=invnbsub1+invnbsub2
    round1=exor(invnbsub,K1)

    #Round 0
    invmc=invmixcolumns(round1)
    finalsr=shiftrow(invmc)
    finalnbsub1=nibblesubs(finalsr[:8],1)
    finalnbsub2=nibblesubs(finalsr[8:],1)
    finalnbsub=finalnbsub1+finalnbsub2
    plaintext=exor(finalnbsub,K0)

    return plaintext
```

[12]:

```python
def getString(l):
  s=""
  for i in l:
    s=s+str(i)
  return s
```

[13]:

```python
#  k='0100101011110101'
#  plaintext='1101011100101000'
k='1100001111110000'
plaintext='1001110001100011'


Key0,Key1,Key2=keyschedule(k)
cipher=encryption(Key0,Key1,Key2,plaintext)
ciphertext=getString(cipher)

print("16 bit Key:",k)
print("16 bit Plaintext:",plaintext)
print("\n")
print("Ciphertext:",ciphertext)

decrypt=decryption(Key0,Key1,Key2,ciphertext)
decryptedtext = getString(decrypt)
print("Decrypted Text",decryptedtext)
```

```
16 bit Key: 1100001111110000
16 bit Plaintext: 1001110001100011


Ciphertext: 1011110101101001
Decrypted Text 1001110001100011
```

[ ]:

```

```