Experiment No. 3

Title: A5/1

Batch: B2 **Roll No.: 16010420117** **Experiment No.: 03**

Aim: To implement stream cipher A5/1

Resources needed: Windows/Linux

Theory: Pre Lab/ Prior Concepts:

A5/1 employs three linear feedback shifl registers , or LFSRs, whichare labeled X, Y, and Z. Register X holds 19 bits, (x_0,x_1…x_{18})-The register Y holds 22 bits, (y_0,y_1…y_{21}) and Z holds 23 bits, (z_0,y_1…z_{22})·Of course, all computer geeks love powers of two, so it's no accident that the three LFSRs hold a total of 64 bits.

Not coincidentally, the A5/1 key K is also 64 bits. The key is used as the initial fill of the three registers, that is, the key is used as the initial values in the three registers. After these three registers are filled with the key,1 we are ready to generate the keystream. But before we can describe how the keystream is generated, we need to say a little more about the registers X, Y, and Z.

When register X steps, the following series of operations occur:

$$t = x_{13} \oplus x_{16} \oplus x_{17} \oplus x_{18}$$
$$x_i = x_{i-1} \text{ for } i = 18, 17, 16, \ldots, 1$$
$$x_0 = t$$

Similarly, for registers Y and Z, each step consists of

$$t = y_{20} \oplus y_{21}$$
$$y_i = y_{i-1} \text{ for } i = 21, 20, 19 \ldots, 1$$
$$y_0 = t$$

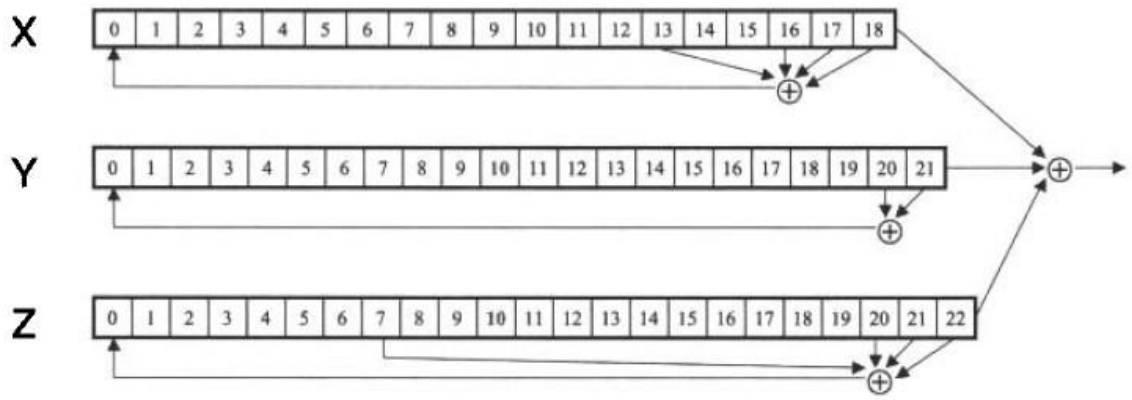
and

$$t = z_7 \oplus z_{20} \oplus z_{21} \oplus z_{22}$$
$$z_i = z_{i-1} \text{ for } i = 22, 21, 20, \ldots, 1$$
$$z_0 = t$$

respectively.

Given three bits x, y, and z, define ma,](x,y, z) to be the majority vote function, that is, if the majority of x, y, and z are 0, the function returns 0; otherwise it returns 1. Since there are an odd number of bits, there cannot be a tie, so this function is well defined.

The wiring diagram for the A5/1 algorithm is illustrated below:



A5/1 Keystream Generator

Procedure / Approach /Algorithm / Activity Diagram:

A. Key Stream generation Algorithm:

At each step: m = maj(x_8, y_{10}, z_{10})

-Examples: maj(0,1,0) = 0 and maj(1,1,0) = 1

If x_8 = m then X steps

-t = $x_{13} \oplus x_{16} \oplus x_{17} \oplus x_{18}$

-$x_i = x_{i-1}$ for i = 18,17,…,1 and x_0 = t

If y_{10} = m then Y steps

-t = $y_{20} \oplus y_{21}$

-$y_i = y_{i-1}$ for i = 21,20,…,1 and y_0 =t

If z_{10} = m then Z steps

-t = $z_7 \oplus z_{20} \oplus z_{21} \oplus z_{22}$

-$z_i = z_{i-1}$ for i = 22,21,…,1 and z_0 = t

Keystreambit is $x_{18} \oplus y_{21} \oplus z_{22}$

Implementation:

Implement the A5/1 algorithm.Encryption and decryption function should ask for key and a input and show the output to the user.

Results: (Program with output as per the format)

Code:

```
alp=['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R
','S','T','U','V','W','X','Y','Z',' ']

def maj(a,b,c):
  lst=[a,b,c]
  a1=a0=0
  for i in lst:
    if(i==0):
      a0+=1
    else:
      a1+=1
  if(a1>a):
    return 1
  else:
    return 0


# print(maj(1,0,0))
def decimalToBinary(n):
  return "{0:b}".format(int(n))

def shiftx(x):
  n=len(x)
  t=x[13]^x[16]^x[17]^x[18]
  i=n-1
  while(i>0):
    x[i]=x[i-1]
    i-=1
  x[0]=t

def shifty(x):
  n=len(x)
  t=x[20]^x[21]
  i=n-1
  while(i>0):
    x[i]=x[i-1]
    i-=1
  x[0]=t

def shiftz(x):
  n=len(x)
  t=x[20]^x[21]^x[22]^x[7]
  i=n-1
  while(i>0):
    x[i]=x[i-1]
    i-=1
  x[0]=t

def keygen(a):
  xi=19
```

```
  x=[0]*xi
  yi=22
  y=[0]*yi
  zi=23
  z=[0]*zi
  x[13]=x[16]=x[17]=x[18]=0
  y[20]=y[21]=0
  z[20]=z[21]=z[22]=z[7]=0

  c=0
  c1=0
  for i in range(64):
    x[c1]=x[13]^x[16]^x[17]^x[18]^a[c]
    c+=1
    if(c1<xi-1):
      c1+=1
    else:
      c1=0
  c=0
  c1=0

  for j in range(64):
    y[c1]=y[20]^y[21]^a[c]
    c+=1
    if(c1<yi-1):
      c1+=1
    else:
      c1=0
  c=0
  c1=0
  for k in range(64):
    z[c1]=z[20]^z[21]^z[22]^z[7]^a[c]
    c+=1
    if(c1<zi-1):
      c1+=1
    else:
      c1=0

  a=[]

  while(c>0):
    t=maj(x[8],y[10],z[10])
    if(x[8]==t):
      shiftx(x)
    if(y[10]==t):
      shifty👍
    if(z[10]==t):
      shiftz(z)
    a.append(x[len(x)-1]^y[len(y)-1]^z[len(z)-1])
    c-=1
  return a

def encrypt(text,key):
  n=len(text)
```

```
  n1=len(key)
  i=0
  s=""
  while(i<n):
    t=decimalToBinary(alp.index(text[i]))
    s+=t
    i+=1
  n2=len(s)
  i=0
  j=0
  ctext=""
  while(i<n2):
    if(j<n1):
      temp=int(s[i])^key[j]
      ctext+=str(temp)
    else:
      temp=int(s[i])^0
      ctext+=str(temp)
    j+=1
    i+=1
  return ctext
lst=[]
a="10101001100001010000101010101010101011101110101011101010101000"
for ch in a:
  lst.append(int(ch))
a=keygen(lst)
text="SOHAM SATISH BHOIR"
ctext=encrypt(text,a)
print("text is ",text)
print("Secret text is ",ctext)
```

Output:

```
text is  SOHAM SATISH BHOIR
Secret text is  1000010000011001001111111010110011111000000101100000101110001000100001
```

Questions:

1) List the stream cipher used in current date along with the name of applications in which those are used.

ChaCha is the most widely used stream cipher in software due to its high speed in real-time applications. It is based on the Salsa20 cipher but it is improved. Google had selected ChaCha20 along with Bernstein's Poly1305 message authentication code in SPDY, whichwas intended as a replacement for TLS over TCP. ChaCha20 is also used forthe arc4random random number generator in FreeBSD,[28] OpenBSD,[29] and NetBSD[30] operating systems. ChaCha20 usuallyoffers better performance than the more prevalent Advanced Encryption Standard (AES)algorithm on systems where the CPU does not feature AES acceleration (such as the AES instruction set for x86 processors). As a result, ChaCha20 is sometimes preferred over AES incertain use cases involving mobile devices, which mostly use ARM-based CPUs.

Outcomes: CO2- Illustrate different cryptographic algorithms for security

Conclusion: (Conclusion to be based on the objectives and outcomes achieved)
Understood the class of stream ciphers and the A5/1 keystream cipher. Wrote a small program to demonstrate the same.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References: Books/ Journals/ Websites:

1. Mark Stamp, “Information Security Principles and Practice”, Wiley.
2. Behrouz A. Forouzan, “Cryptography and Network Security”, Tata McGraw Hill
3. William Stalling, “Cryptography and Network Security”, Prentice Hall