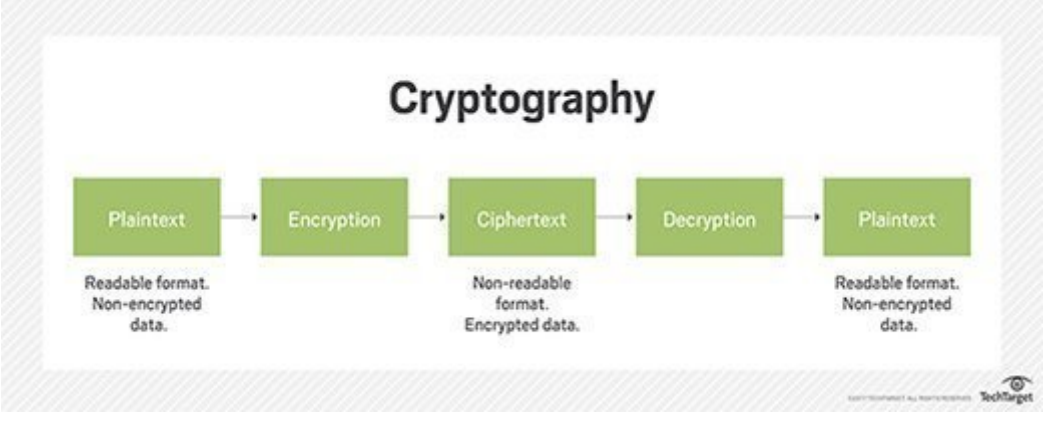
LA Project: Cryptography

Mohammad Mohsin 22837
Syed Owais Ali 23053

Cryptography is the study of encoding and decoding secret messages. Code are called ciphers, uncoded messages are called plaintext, and coded messages are called ciphertext.
The process of converting a plaintext to a ciphertext is called encoding.
The process of converting a ciphertext to a plaintext is called decoding.



1) Polygraphic System:

We will use Polygraphic System which uses Matrix to encode and decode our plaintext.
First we will associate a number with every letter of alphabet.

A	B	C	D	E	F	G	H	I	J	K	L	M
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
1	2	3	4	5	6	7	8	9	10	11	12	13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
14	15	16	17	18	19	20	21	22	23	24	25	26

Then we will suppose nxn matrix A:

In this case we suppose n = 3

$$A = \begin{bmatrix} 0 & 2 & -1 \\ 1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}$$

and compute its inverse A^{-1}

$$A^{-1} = \begin{bmatrix} 1 & 1 & 0 \\ 2 & 1 & 1 \\ 3 & 2 & 2 \end{bmatrix}$$

Both communicating parties should have knowledge of the conversion table, the matrices A and A^{-1}
We can call this a **protocol** *or a set of rules between two parties.*
In order to send an encrypted message, the sending party will first convert text into numbers. Then it will make groups of nx1 vectors. It will then find product of those numbers with A. The result will be cipher text, the encrypted message can now be sent.
The receiving party will receive the cipher text, again make groups of nx1, then it will find product of these vector with A^{-1}*. It will then convert those numbers back to text. Our cipher text is now converted back to plaintext.*

Example 1:

Suppose we want to encode and send the following message: "STUDY LINEAR ALGEBRA"
Using the conversion table above, we will convert this into numbers: 19 20 21 4 25 12 9 14 5 1 18 1 12 7 5 2 18 1.
Then, we divide the numbers into groups of 3 and write each group in the form of a 3×1 *vector.*

$$\begin{bmatrix} 19 \\ 20 \\ 21 \end{bmatrix}, \begin{bmatrix} 4 \\ 25 \\ 12 \end{bmatrix}, \begin{bmatrix} 9 \\ 14 \\ 5 \end{bmatrix}, \begin{bmatrix} 1 \\ 18 \\ 1 \end{bmatrix}, \begin{bmatrix} 12 \\ 7 \\ 5 \end{bmatrix}, \begin{bmatrix} 2 \\ 18 \\ 1 \end{bmatrix}$$

We will now find the product of A with these vectors.

$$A\begin{bmatrix}19\\20\\21\end{bmatrix}, A\begin{bmatrix}4\\25\\12\end{bmatrix}, A\begin{bmatrix}9\\14\\5\end{bmatrix}, A\begin{bmatrix}1\\18\\1\end{bmatrix}, A\begin{bmatrix}12\\7\\5\end{bmatrix}, A\begin{bmatrix}2\\18\\1\end{bmatrix}$$

This will result in the following vectors:

$$\begin{bmatrix}19\\0\\-18\end{bmatrix}, \begin{bmatrix}38\\-34\\-17\end{bmatrix}, \begin{bmatrix}23\\-14\\-18\end{bmatrix}, \begin{bmatrix}35\\-34\\-18\end{bmatrix}, \begin{bmatrix}9\\3\\-14\end{bmatrix}, \begin{bmatrix}35\\-33\\-19\end{bmatrix}$$

Now our message is encrypted: 19 0 -18 38 -34 -17 23 -14 -18 25 -34 -18 9 3 -14 35 -33 -19

Receiving party will divide this group of numbers into 3x1 vectors and compute its product with A^{-1}*:*

$$A^{-1}\begin{bmatrix}19\\0\\-18\end{bmatrix}, A^{-1}\begin{bmatrix}38\\-34\\-17\end{bmatrix}, A^{-1}\begin{bmatrix}23\\-14\\-18\end{bmatrix}, A^{-1}\begin{bmatrix}35\\-34\\-18\end{bmatrix}, A^{-1}\begin{bmatrix}9\\3\\-14\end{bmatrix}, A^{-1}\begin{bmatrix}35\\-33\\-19\end{bmatrix}$$

Reciever has decrypted the cipher text.

$$\begin{bmatrix}19\\20\\21\end{bmatrix}, \begin{bmatrix}4\\25\\12\end{bmatrix}, \begin{bmatrix}9\\14\\5\end{bmatrix}, \begin{bmatrix}1\\18\\1\end{bmatrix}, \begin{bmatrix}12\\7\\5\end{bmatrix}, \begin{bmatrix}2\\18\\1\end{bmatrix}$$

This can now be converted back to the original text: "STUDY LINEAR ALGEBRA"

Example 2:

Suppose we want to encode and send the following message: "HELLO WORLD"
Using the conversion table above, we will convert this into numbers: 8 5 12 12 15 23 15 18 12 4
Then, we divide the numbers into groups of 3 and write each group in the form of a 3×1 *vector.*

$$\begin{bmatrix}8\\5\\12\end{bmatrix}, \begin{bmatrix}12\\15\\23\end{bmatrix}, \begin{bmatrix}15\\18\\12\end{bmatrix}, \begin{bmatrix}4\\X\\X\end{bmatrix}$$

Notice that the number of the letters in the plaintext is not divisible by 3, therefore, we add a dummy number -1 equivalent to space.

$$\begin{bmatrix}8\\5\\12\end{bmatrix}, \begin{bmatrix}12\\15\\23\end{bmatrix}, \begin{bmatrix}15\\18\\12\end{bmatrix}, \begin{bmatrix}4\\-1\\-1\end{bmatrix}$$

We will now find the product of A with these vectors.

$$A\begin{bmatrix}8\\5\\12\end{bmatrix}, A\begin{bmatrix}12\\15\\23\end{bmatrix}, A\begin{bmatrix}15\\18\\12\end{bmatrix}, A\begin{bmatrix}4\\-1\\-1\end{bmatrix}$$

This will result in the following vectors:

$$\begin{bmatrix}-2\\10\\-1\end{bmatrix}, \begin{bmatrix}7\\5\\-4\end{bmatrix}, \begin{bmatrix}24\\-9\\-21\end{bmatrix}, \begin{bmatrix}-1\\5\\-4\end{bmatrix}$$

Now our message is encrypted: -2 10 -1 7 5 -4 24 -9 -21 -1 5 -4

Receiving party will divide this group of numbers into 3x1 vectors and compute its product with A^{-1} *Reciever has decrypted the cipher text.*
This can now be converted back to the original text: "HELLO WORLD"

Coding:

We will use python to implement Polygraphic System

What is numpy?
NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays

In [1]:
```
import numpy as np
```

We will now assign our conversion table and matrix A

Note: Matrix A and conversion table can differ but must be same between two parties

For example: We can take A as 2x2 ,3x3 or even 4x4 Matrix

In [2]:
```
conversion_table = {
    -1:'',1:'a', 2:'b', 3:'c', 4:'d', 5:'e', 6:'f', 7:'g', 8:'h', 9:'i', 10:'j', 11:'k', 12:'l', 13:'m', 14:'n', 15:'o',
    16:'p', 17:'q', 18:'r', 19:'s', 20:'t', 21:'u', 22:'v', 23:'w', 24:'x', 25:'y', 26:'z'}
```

In [3]:
```
A = np.array([
    [0, 2, -1],
    [1, -2., 1],
    [-1, -1, 1]])

A_inverse = np.linalg.inv(A)
```

We will now define our encrypt and decrypt function

In [4]:
```
inv_dict = {value:key for key, value in conversion_table.items()}
def convert_text_to_number(text):
    c = list(text)
    num = []
    for i in c:
        num.append(inv_dict.get(i.lower()))
    return num


def convert_number_to_text(num):
    s =""
    for i in num:
        s += (conversion_table.get(i).upper())
    return s


def encrypt(text):
    text = text.replace(" ","")
    num = convert_text_to_number(text)
    while (len(num)%3 != 0):
        num.append(-1)
    num = np.array(num)
    num = np.reshape(num,(-1,3)).T
    cipher_text = np.dot(A,num)
    cipher_text = np.round(cipher_text.T.flatten())
    return cipher_text


def decrypt(cipher_text):
    num = cipher_text
    num = np.reshape(num,(-1,3)).T
    plain_text = np.dot(A_inverse,num)
    plain_text = plain_text.T.flatten()
    plain_text = np.round(plain_text)
    text = convert_number_to_text(plain_text)
    return text
```

Testing Example 1 with our code:

In [5]:
```
s ="STUDY LINEAR ALGEBRA"
cipher_text = encrypt(s)
print("Cipher Text: ",cipher_text)
text = decrypt(cipher_text)
print("Plaintext after decryption: ",text)
```

```
Cipher Text:  [ 19.   0. -18.  38. -34. -17.  23. -14. -18.  35. -34. -18.   9.   3.
 -14.  35. -33. -19.]
Plaintext after decryption:  STUDYLINEARALGEBRA
```

Testing Example 2 with our code:

In [6]:
```
s ="HELLO WORLD"
num = encrypt(s)
text = decrypt(num)
print("Cipher Text: ",num)
print("Plaintext after decryption: ",text)
```

```
Cipher Text:  [ -2.  10.  -1.   7.   5.  -4.  24.  -9. -21.  -1.   5.  -4.]
Plaintext after decryption:  HELLOWORLD
```

2) Hill Cipher:

Hill Cipher uses the same method as Polygraphic System with additional complexity. The more complex is your algorithm, the more security it ensures.

The encryption algorithm of this method is:

$$C \equiv APmodN$$

where:
A is nxn key matrix
P is the plaintext vector
N is the number of letters in conversion table.

Similarly, The decryption algorithm of this method is:

$$P \equiv A^{-1}CmodN$$

This algorithm converts ciphertext into comlete gibberish.
For instace we take our First example: STUDY LINEAR ALGEBRA
Now our message is converted to ciphertext: 19 0 -18 38 -34 -17 23 -14 -18 25 -34 -18 9 3 -14 35 -33 -19
After taking mod 27 these number will convert to: 19 0 9 11 20 10 23 13 9 35 20 9 9 3 13 8 21 8
Now the text converts to: SISIKTJWMIYTIICMHUH
The text is now completely unreadable

To decrypt we will first multiply these vectors to A^{-1}

$$A^{-1}\begin{bmatrix}19\\0\\9\end{bmatrix}, A^{-1}\begin{bmatrix}11\\20\\10\end{bmatrix}, A^{-1}\begin{bmatrix}23\\13\\9\end{bmatrix}, A^{-1}\begin{bmatrix}35\\20\\9\end{bmatrix}, A^{-1}\begin{bmatrix}9\\3\\13\end{bmatrix}, A^{-1}\begin{bmatrix}8\\21\\8\end{bmatrix}$$

$$\begin{bmatrix}19\\47\\75\end{bmatrix}, \begin{bmatrix}31\\52\\93\end{bmatrix}, \begin{bmatrix}36\\78\\133\end{bmatrix}, \begin{bmatrix}55\\99\\163\end{bmatrix}, \begin{bmatrix}12\\34\\59\end{bmatrix}, \begin{bmatrix}29\\45\\82\end{bmatrix}$$

After taking mod27 of these matrices we will get back our original matrice

$$\begin{bmatrix}19\\20\\21\end{bmatrix}, \begin{bmatrix}4\\25\\12\end{bmatrix}, \begin{bmatrix}9\\14\\5\end{bmatrix}, \begin{bmatrix}1\\18\\1\end{bmatrix}, \begin{bmatrix}12\\7\\5\end{bmatrix}, \begin{bmatrix}2\\18\\1\end{bmatrix} mod27$$

This can now be converted back to the original text: "STUDY LINEAR ALGEBRA"

In order to implement Hill Cipher in our code we have to add (array%27) in our encrypt and decrypt method.

Using More Than One Key in Hill Cipher:

We can use the associative property of matrices to make the coding process more complex and more secure. Therefore; if we have two invertible key matrices AB, and a plaintext column vector P , then the encryption algorithm is

$$C \equiv ABPmodN$$

Similarly, The decryption algorithm of this method is:

$$P \equiv (AB)^{-1}CmodN$$

Generalizing the Algorithm:

In this case we can use n number of invertible matrices to encode or decode any message and the steps will be the same. This means that, if we have the invertible matrices (A B C..... M) then the encryption algorithm will be:

$$C \equiv (ABC.....M)PmodN$$

Similarly, The decryption algorithm of this method is:

$$P \equiv (ABC.....M)^{-1}CmodN$$

Example:

We will encode and decode the following message(I AM SMART) by using the matrices

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} B = \begin{bmatrix} 3 & 2 \\ 4 & 3 \end{bmatrix}$$

The inverse of the following matrices are:

$$A^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & -2 \end{bmatrix} B^{-1} = \begin{bmatrix} 3 & -2 \\ -4 & 3 \end{bmatrix}$$

Using the conversion table above, we will convert this into numbers: 9 1 13 19 13 1 18 20

Then, we divide the numbers into groups of 2 and write each group in the form of a 2×1 vector.

$$\begin{bmatrix} 9 \\ 1 \end{bmatrix}, \begin{bmatrix} 13 \\ 19 \end{bmatrix}, \begin{bmatrix} 13 \\ 1 \end{bmatrix}, \begin{bmatrix} 18 \\ 20 \end{bmatrix}$$

We will now find the product of A and B with these vectors, and then mod27

$$AB \begin{bmatrix} 9 \\ 1 \end{bmatrix}, AB \begin{bmatrix} 13 \\ 19 \end{bmatrix}, AB \begin{bmatrix} 13 \\ 1 \end{bmatrix}, AB \begin{bmatrix} 18 \\ 20 \end{bmatrix} mod 27$$

This will result in the following cipher vectors:

$$\begin{bmatrix} 16 \\ 2 \end{bmatrix}, \begin{bmatrix} 20 \\ 23 \end{bmatrix}, \begin{bmatrix} 2 \\ 14 \end{bmatrix}, \begin{bmatrix} 23 \\ 13 \end{bmatrix}$$

Now the text converts to: PBTWBNWM

In order to decrypt, we will find prduct of ciphertext vectors with A^{-1} *and* B^{-1}*, and then mod27*

$$(AB)^{-1} \begin{bmatrix} 16 \\ 2 \end{bmatrix}, (AB)^{-1} \begin{bmatrix} 20 \\ 23 \end{bmatrix}, (AB)^{-1} \begin{bmatrix} 2 \\ 14 \end{bmatrix}, (AB)^{-1} \begin{bmatrix} 23 \\ 13 \end{bmatrix} mod 27$$

This will decrypt our ciphertext back to plain text which can be converted to our message(I AM SMART)

References:

Cryptography by Means of Linear Algebra and Number Theory by Ajaeb Elfadel
http://i-rep.emu.edu.tr:8080/xmlui/bitstream/handle/11129/1420/ElfadelAjaeb.pdf?sequence=1 (http://i-rep.emu.edu.tr:8080/xmlui/bitstream/handle/11129/1420/ElfadelAjaeb.pdf?sequence=1)

Applications of Linear Algebra:
https://www.math.ucdavis.edu/~daddel/linear_algebra_appl/Applications/applications.html (https://www.math.ucdavis.edu/~daddel/linear_algebra_appl/Applications/applications.html)