Universitetet i Sørøst-Norge

Assignment 1
Subject code: *DCS3100*
Subject name: *Introduction to Data and Cyber-Security*

Rahmat Mozafari

Sunday 29 September, 2019

Contents

List of Figures

Listings

0.1 Concepts illustration of CIA/AIC

What is CIA? CIA stands for Confidentiality Integrity and Authentication/ Availability and this model is a guide for policies information and these three elements are the most crucial components in Security. The CIA can be imagined in as a triangle. Confidentiality are a method and its designed to prevent the information from the reaching the wrong people and making sure that the right person can get it. The integrity is to have the ability to ensure that data is correct and it is not altered from the original sources. The information's such as concerned must be readily and accessible for the user all the times.

The following steps will show the illustration concepts of the CIA between Bob and Alice.

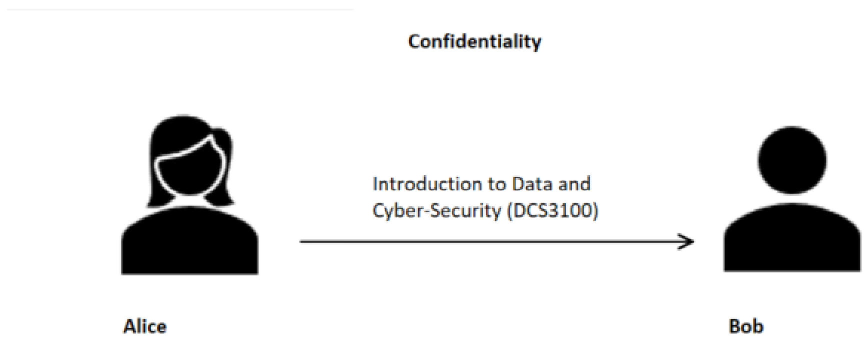

Figure 1: Confidentiality

Lets say Alice want to send a message to Bob, to her friend. The message should only be able to read by Alice and Bob. If a third person view their messages and they shouldn't exchange messages because the information they sharing it can leak and gives a serious consequence for both.

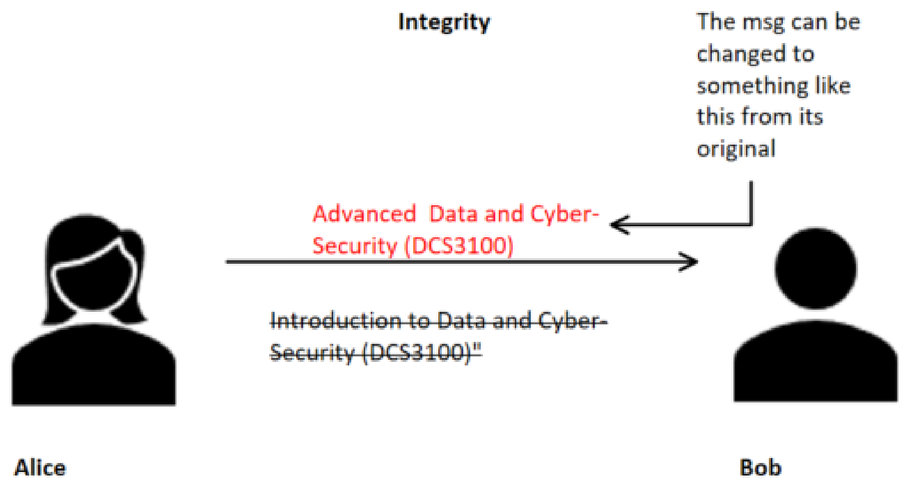

Figure 2: Integrity

Since Bob is receiver and he must be able to verify that the message content is accurate and unchanged. The message content can be modified accidentally or on purpose by a third person.

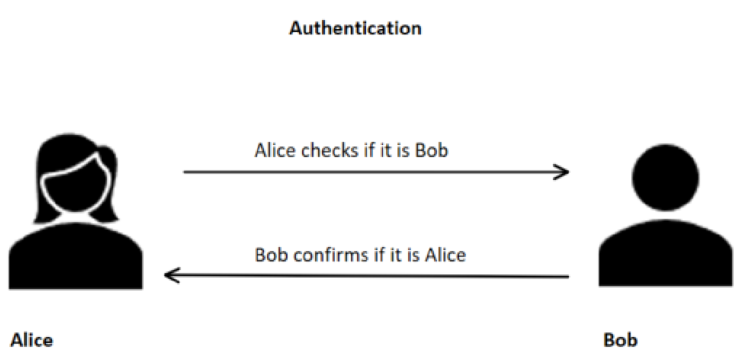

Figure 3: Authentication/ Availability

Alice and Bob should be able to confirm identity of the other party. Alice checks first the identity of the receiver then receiver confirms the sender's identity. If they have any suspect to confirm identity of the other party they have to avoid sending message to each other and figure out another solution.

0.2 Vigenere Cipher

Vigenere cipher is similar to Caesar crypto-system, but in Vigenere we are using several keys instead of just single key. the Vigenere cipher is a form of poly-alphabetic substitution method and this was constructed in the 16th century. This crypto method uses a given word as the private key and the letters in the key define how many character to shift the actual letter in the plain text.

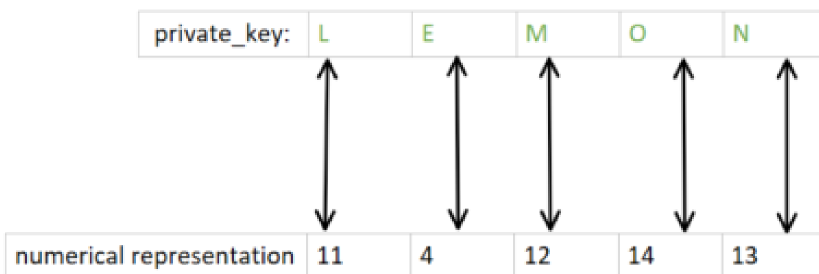

Figure 4: numerical representation

To encrypt Vigenere Cipher we need to use this mathematical formula and it's approximately the same formula as we using for Caesar.

$$C_i(m_i) = (m_i + K_i) mod 26$$

$C_i(m_i)$ is the encrypted character of the cipher text.
m_i is the character of the plain text.
In Vigenere we have to use the `i-th` character of the key for encrypting the `i-th` character.
mod 26 is the length of the English alphabet.
To Decrypt the cipher text to plain text we have to use this formula.

$$D_i(m_i) = (m_i - K_i) mod 26$$

$D_i(m_i)$ is the decrypted character in the cipher text.
To transfer the plain text into the cipher text we use the mathematical formula and using the character in private key in order to transform the letters

in plain text. The first character in plain text will be transformed with the help of the first character if the private key and second letter in the plain text will be transformed with help of the second character in the private key and so on. When all the character in private key are used we start over and do the same operations. After that we can just use Caesar encryption to encrypt the message.

Alphabets with numerical representantions

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
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Private_key	LEMON
The Mathematical formula:	$C_i(m_i) = (m_i + K_i) mod 26$ Private key are used and starting over
	LEM ONLEM ONLEM ONL EMONL EMON LEM ONLE MON
Plain_text:	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
Example operation:	L = 11 T =19 11+19 = 30 mod 26 => 4 and It's E
Cipher_text:	ELQ EHTGW PEZAZ TBI NGACD SHSE ELQ ZNKC PCT

$$D_i(m_i) = (m_i - K_i) mod 26$$

From cipher to plain text:	L = 11 E = 4 4-11 = -7 mod 26 => 19 and it is T And so an ...
Cipher_text:	ELQ EHTGW PEZAZ TBI NGACD SHSE ELQ ZNKC PCT
Plain_text:	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Figure 5: Single key operation

We basically shifting the letter T with as many characters to right as the numerical representation of L and shifting the second character of the plain text with as many letter to right as the numerical representation of second letter in private key etc.

0.3 Single Key Code

```
#alfa = ' abcdefghijklmnopqrstuvwxyz.'
alfa  = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ. '
# vigenere algorithm
#Mathematical formula is:  Ci (mi)  =(mi+ki) mod 28
# mod is  28 with space and .

def en_vigenere(plainText,key):
    #the text we want to encrypt
    plainText = plainText.upper()
    key = key.upper()
    cipherText = ''
    # repesenting the key index as far as key is concerned
    indexKey = 0
    # now we are going to concider all characters in plainText
    for char in plainText:
        #The number of shifts is equal to the index of the char in the alfabet and plus index of the char in the private key
        index = (alfa.find(char) + (alfa.find(key[indexKey]))) % len (alfa) # this is the mathematical operation
        # adding the encrypted char to the cipherText
        cipherText = cipherText +alfa[index]
        # Now I'm concider the next letter and need to increment the key index
        indexKey = indexKey + 1

        # we need to start agin when we have concidered the last letter of key
        if indexKey == len(key):
            indexKey =0
    return cipherText

# Now I'm going to decrypt and using the following formula
# The number og shifts is equal to the index  of the char in the alfabet  and minus index of the char in the key
#Mathematical formula is:  Di (mi)  = (mi-ki) mod 28
def de_vigenere(cipherText, key):
    cipherText = cipherText.upper()
    key = key.upper()
    plainText = ''
    indexKey = 0

```

```
    for char in cipherText:
        index = (alfa.find(char) - (alfa.find(key[indexKey]))) % len(alfa)
        plainText  = plainText + alfa[index]

        indexKey = indexKey +1
        if indexKey ==len(key):
            indexKey =0

    return plainText

if __name__ =="__main__":
    plainText = input("Enter some text to encrypt\n")
    encrypt = en_vigenere(plainText, 'LEMON')
    print("The encrypted message is: %s" % encrypt)
    decrypt = de_vigenere(encrypt, 'LEMON')
    print("The Decrypted message is: %s" % decrypt)
```

Listing 1: Vigenere Cipher with single key

```
This is the output I got when I run the program.
PS C:\Users\m_rah\Desktop\crypto\en-decryption-algorithm\Vigenere> python .\vigenere.py
Enter some text to encrypt
The quick brown fox jumps over the lazy dog.
The encrypted message is: CLQNBDMOYMMV.I.KJ.JMUYYBDKSFSCKXTSMWEJKMOSSM
The Decrypted message is: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG.
```

Listing 2: Output of single key

0.4 Two Keys Code

This is basically the same method I use to encrypt the plain text with two keys. First I encrypting the plain text with help of the first key, when the plain text is encrypted with the first key, then I use the second key to encrypt the encrypted text again with help of the second key. The table is showing the encryption and decryption operation.

Alphabets with numerical representantions

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
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Private_key_1	GREEN
The Mathematical formula:	$C_i(m_i) = (m_i + K_i) mod 26$ Private key are used and starting over
	GRE ENGRE ENGRE ENG REENG REEN GRE ENGR EEN
Plain_text:	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
Example operation:	G = 6 T =19 6+19 = 25 mod 26 => 25 and It's Z
Cipher_text_1:	ZYI UHOTO FEUNR JBD AYQCY FZIE ZYI PNFP HST
Private key_2	WATERMELON
Operation	Z = 25 W = 22 25+22=47 MOD 26 = 21 and its V in Alphabet, etc.
Cipher_text_2:	VYB YYAXZ TRQNK NSP EJEPU FSMV LCT DABP AWK

$$D_i(m_i) = (m_i - K_i) mod 26$$

Decrypting from cipher text to plain text

Private_key_2 = WATERMELON	W= 22 V = 21 21-22 = -1 mod 26 => 25 and it is Z And so an ...
Cipher_text_2:	VYB YYAXZ TRQNK NSP EJEPU FSMV LCT DABP AWK
Cipher_text_1:	ZYI UHOTO FEUNR JBD AYQCY FZIE ZYI PNFP HST
Private_key_1 = GREEN	G=6 Z=25 25-6 = 19 MOD 26 = 19 => T
Plain_text	THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

Figure 6: Two Keys operation

Code

```python
#alfa = ' abcdefghijklmnopqrstuvwxyz.'
alfa  = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ. '
# vigenere algorithm
#Mathematical formula is:  Ci (mi)  =(mi+ki) mod 28
# mod is  28 with space and .

def en_vigenere(plainText,key):
    #the text we want to encrypt
    plainText = plainText.upper()
    key = key.upper()
    cipherText = ''
    # repeseenting the key index as far as key is concerned
    indexKey = 0
    # now we are going to concider all characters in plainText
    for char in plainText:
        #The number of shifts is equal to the index of the char in the alfabet and plus index of the char in the private key
        index = (alfa.find(char) + (alfa.find(key[indexKey]))) % len (alfa) # this is the mathematical operation
        # adding the encrypted char to the cipherText
        cipherText = cipherText +alfa[index]
        # Now I'm concider the next letter and need to increment the key index
        indexKey = indexKey + 1

        # we need to start agin when we have concidered the last letter of key
        if indexKey == len(key):
            indexKey =0
    return cipherText

# Now I'm going to decrypt and using the following formula
# The number og shifts is equal to the index  of the char in the alfabet  and minus index of the char in the key
#Mathematical formula is:  Di (mi)  = (mi-ki) mod 28
def de_vigenere(cipherText, key):
    cipherText = cipherText.upper()
    key = key.upper()
    plainText = ''
    indexKey = 0
```

```python

    for char in cipherText:
        index = (alfa.find(char) - (alfa.find(key[indexKey]))) % len(alfa)
        plainText  = plainText + alfa[index]

        indexKey = indexKey +1
        if indexKey ==len(key):
            indexKey =0

    return plainText

if __name__ =="__main__":
    plainText = input("Enter some text to encrypt\n")
    key_1  =    input("Enter the first key:\n")
    encrypt1 = en_vigenere(plainText, key_1)          # Calling the Encrypting function to encrypt the message with the key 1
    print("The encrypted message with key 1 is: %s" % encrypt1)
    key_2 =     input("Enter the second key:\n")
    encrypt2 = en_vigenere(encrypt1, key_2)           # Encrypting the message with the help of key 2. Calling the same function as I call when I encrypting the message with help of the key 1
    print("The encrypted message wwith the key 2 is: %s" % encrypt2)

    decrypt2 = de_vigenere(encrypt2, key_2)             # Decrypting the message to call the decrypting function, but first I decrypting the text with help of the second to get the encrypt1 text, then I decrypting the encrypt1 to get the plain text
    print("Decrypted message with the key 2 is: %s" % decrypt2)
    decrypt1 = de_vigenere(decrypt2, key_1)
    print("The Decrypted message with the key 1 is: %s" % decrypt1)
```

Listing 3: Vigenere Cipher With two keys

```
PS C:\Users\m_rah\Desktop\crypto\en-decryption-algorithm\Vigenere> python .\vigenere.py
Enter some text to encrypt
The quick brown fox jumps over the lazy dog.
```

```
Enter the first key:
green
The encrypted message with key 1 is: ZYIDB.ZGOMHGS..FWS
    MPJQTDFDZICFILIMRRBAMJDKC
Enter the second key:
watermelon
The encrypted message wwith the key 2 is: TY HSKBRAZBGJCPR.
    BNZJJHXURHIWP ICMBBVMOZDDBG
Decrypted message with the key 2 is: ZYIDB.ZGOMHGS..FWS
    MPJQTDFDZICFILIMRRBAMJDKC
The Decrypted message with the key 1 is: THE QUICK BROWN FOX
    JUMPS OVER THE LAZY DOG.
```

Listing 4: Output of using two keys

0.5 Confusion and Diffusion

They are cryptography technique and purpose with the Confusion is that to make relationship between the statics of the cipher text and the value of the encryption key. On the contrary, diffusion attempts to hide the statistical structure of the plain text through expand out the influence respectively of each individual plain text numeral big piece. They both are properties of operation for secure cipher in cryptography and it was identified by Shannon in 1949. The Confusion is designed/ developed to boots the vagueness of cipher text and make certain that this technique gives no trace about the plain text and the correlation between the encryption key value and the statistics of the cipher text is maintained as complex as achievable. If someone gets control over the statistics of the cipher text and he/she couldn't be able to presume they key. On the other hand the diffusion is the increase the the redundancy of the plain text to cover the structure of the plain text to hinder to attack to calculate the key. The statistical structure of plain text can disappear into long range statistics of the cipher text and that no body can assume the key.