Cryptology

Tutor : Jean-Pierre Zanotti

Laboratory : iMath

Mathis PROVOST
L2PC
2018/2019

Problematic

- **What is the purpose of cryptology in our society and why its evolution is so important ?**

Goal

- Study some encryption, decryption and decyphering algorithms to understand the operation and the utility

Study Axes

- Introduction
- Symmetric-key code (Caesar, Vigenère)
- Asymmetric-key code (RSA)

Tools

- Python 3.6.2
- Python Tkinter


CRYPTOLOGY
01
10

What is cryptology ?

Cryptology is composed of 2 fields of research:

- **Cryptography**

- **Cryptanalysis**

An exemple

Clear text

This is an exemple of encryption and decryption. This is the original message. It is called the clear text, it is lisible whereas the encrypted text is not.

Encryption

Key: « 13 »

Decryption

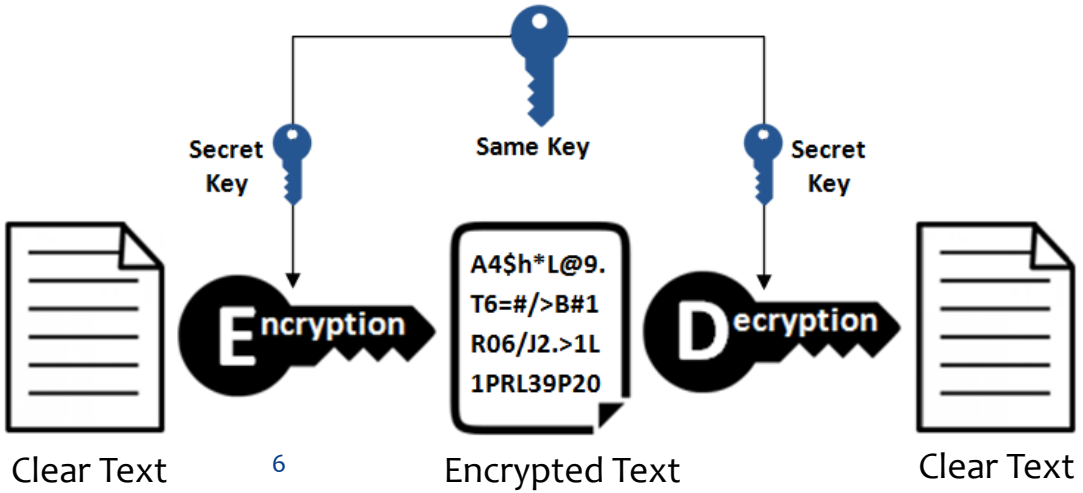
Guvf vf na rkrzcyr bs rapelcgvba naq qrpelcgvba. Guvf vf gur bevtvany zrffntr. Vg vf pnyyrq gur pyrne grkg, vg vf yvfvoyr jurernf gur rapelcgrq grkg vf abg.

Encrypted text

« Symmetric-key » Code

- Symmectric-key codes are codes that use the same key to encrypt and decrypt the text.
- If someone is able to steal the key from one person, he will be able to encrypt and decrypt as he wants, that's the biggest problem of this type of code.

  - Caesar
  - Vigenère



Caesar's Code

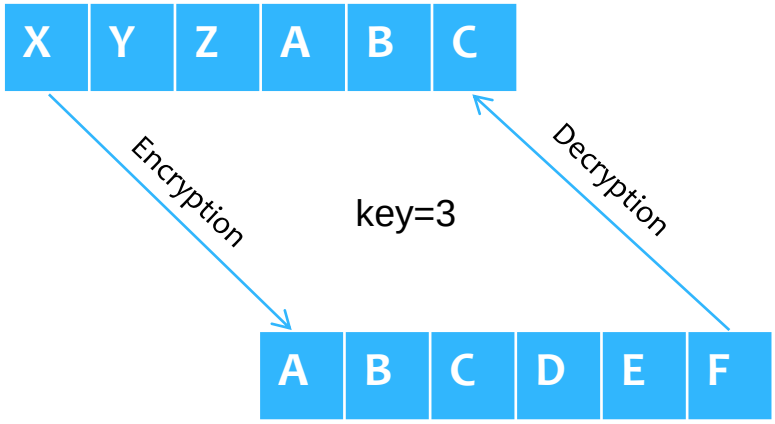
- De/Code
- Decypher

De/Coding with Caesar's Code

This coding method was used by Caesar, which explains the name !

Encryption

- Right shift
- Circular permutation of the alphabet
- 26 possible keys



Decryption

- Left shift
- To know the key

De/Code

Caesar_De_Encryption.py - C:\Users\Mathis\Desktop\UNIV\PPR\L2\Caesar\Caesar_De_Encryp...

File Edit Format Run Options Window Help

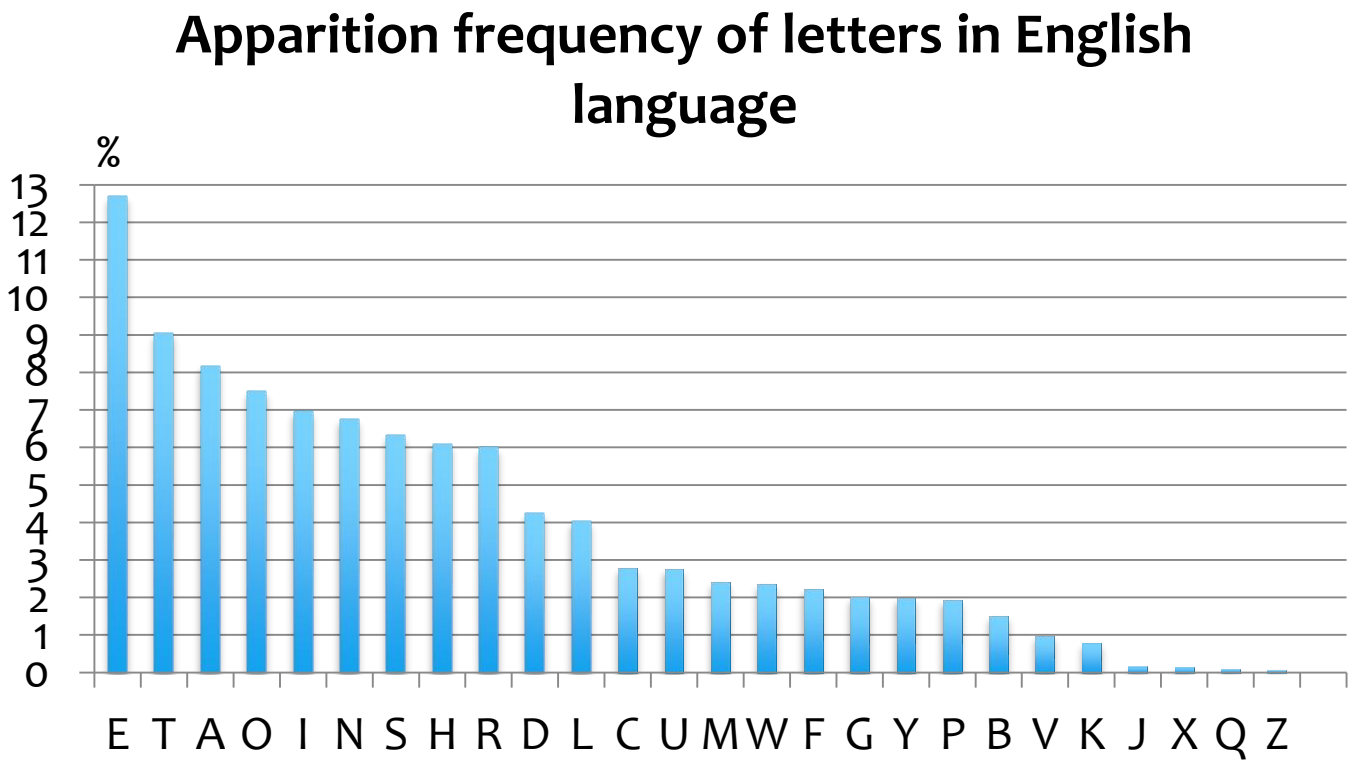
```
a=input("Encryption (E) or Decryption (D) : ")
b=int(input("Clé de chiffrement : "))
Lc=[]
final=''
k=0
i=0
if a=='E':
        c=input("What is the text to encrypt? ")
        while i<len(c):
                Lc.append(c[i])
                if 64<ord(Lc[i])<91 :
                        val= ord(Lc[i])-65
                        val= (val+b)%26
                        val=val+65
                        Lc[i]=chr(val)
                elif 96<ord(Lc[i])<123:
                        val= ord(Lc[i])-97
                        val= (val+b)%26
                        val=val+97
                        Lc[i]=chr(val)
                i=i+1
        while k<len(Lc):
                final=final+Lc[k]
                k=k+1
        print (final)
elif a=='D':
        c=input("What is the text to decrypt ? ")
        while i<len(c):
                Lc.append(c[i])
                if 64<ord(Lc[i])<91 :
                        val= ord(Lc[i])-65
                        val= (val-b)%26
                        val=val+65
                        Lc[i]=chr(val)
                elif 96<ord(Lc[i])<123:
                        val= ord(Lc[i])-97
                        val= (val-b)%26
                        val=val+97
                        Lc[i]=chr(val)
                i=i+1
        while k<len(Lc):
                final=final+Lc[k]
                k=k+1
        print (final)
```

```
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\Mathis\Desktop\UNIV\PPR\L2\Caesar\Caesar_De_Encryption.py
Encryption (E) or Decryption (D) : E
Key : 12
What is the text to encrypt? hello
tqxxa
>>>
 RESTART: C:\Users\Mathis\Desktop\UNIV\PPR\L2\Caesar\Caesar_De_Encryption.py
Encryption (E) or Decryption (D) : D
Key : 12
What is the text to decrypt ? tqxxa
hello
>>>
```


python™

Decyphering Caesar's Code

- Two methods
  - **BruteForce**
    26 possible keys
    Quick method
  - **Frequency analysis**
    Requires a long text

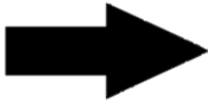


« BruteForce » method

BruteForce.py - C:\Users\Mathis\Desktop\UNIV\PPR\L2\Caesar\BruteForce.py (3.6.4)

File Edit Format Run Options Window Help

```
c=input("What is the text to decypher ? ")
for b in range (1,26):
    final=''
    i=0
    k=0
    Lc=[]
    while i<len(c):
        Lc.append(c[i])

        if 64<ord(Lc[i])<92 :
            val= ord(Lc[i])-65
            val= (val-b)%26
            val=val+65
            Lc[i]=chr(val)

        elif 96<ord(Lc[i])<124:
            val= ord(Lc[i])-97
            val= (val-b)%26
            val=val+97
            Lc[i]=chr(val)
        i=i+1

    while k<len(Lc):
        final=final+Lc[k]
        k=k+1

    print ('Decyphering key :',b,final)
```



```
Type "copyright", "credits" or "license(
>>>
===== RESTART: C:\Users\Mathis\Desktop\U
What is the text to decypher ? tqxxa
Decyphering key : 1 spwwz
Decyphering key : 2 rovvy
Decyphering key : 3 qnuux
Decyphering key : 4 pmttw
Decyphering key : 5 olssv
Decyphering key : 6 nkrru
Decyphering key : 7 mjqqt
Decyphering key : 8 lipps
Decyphering key : 9 khoor
Decyphering key : 10 jgnnq
Decyphering key : 11 ifmmp
Decyphering key : 12 hello
Decyphering key : 13 gdkkn
Decyphering key : 14 fcjjm
Decyphering key : 15 ebiil
Decyphering key : 16 dahhk
Decyphering key : 17 czggj
Decyphering key : 18 byffi
Decyphering key : 19 axeeh
Decyphering key : 20 zwddg
Decyphering key : 21 yvccf
Decyphering key : 22 xubbe
Decyphering key : 23 wtaad
Decyphering key : 24 vszzc
Decyphering key : 25 uryyb
```

Frequency analysis

Frequency analysis.py - C:\Users\Mathis\Desktop\UNIV\PPR\L2\Caesar\Frequency analysis.py (3.6.4)

File Edit Format Run Options Window Help

```
c=input("What is the text to decypher ? ")
L_letter=['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']
L_nbr=[0]*26
for i in range(len(c)):
    for j in range(len(L_letter)):
        if c[i]==L_letter[j]:
            L_nbr[j]=L_nbr[j]+1
print("Here is how many times each letter appeared in the text: ",L_nbr)
maxi=max(L_nbr)
for i in range(len(L_nbr)):
    if L_nbr[i]==maxi:
        index=i
print("The index of the most present letter is : ",index)
print("The letter that appears the most in the encrypted text is : ",L_letter[index])
shift=ord(L_letter[index])-ord("e")
print("The key is: ",shift)
b=shift
Lc=[]
final=''
k=0
i=0
while i<len(c):
        Lc.append(c[i])
        if 64<ord(Lc[i])<91 :
                val= ord(Lc[i])-65
                val= (val-b)%26
                val=val+65
                Lc[i]=chr(val)
        elif 96<ord(Lc[i])<123:
                val= ord(Lc[i])-97
                val= (val-b)%26
                val=val+97
                Lc[i]=chr(val)
        i=i+1
while k<len(Lc):
        final=final+Lc[k]
        k=k+1
print (final)
```

```
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\Mathis\Desktop\UNIV\PPR\L2\Caesar\Frequency analysis.py =
What is the text to decypher ? Ftue fqjf ime qzodkbfqp iuft ftq omqemd oapq iuft
 m wqk qcgmx fa fiqxhq
Here is how many times each letter appeared in the text:  [2, 1, 1, 2, 3, 8, 1,
1, 4, 1, 2, 0, 5, 0, 3, 2, 10, 0, 0, 4, 3, 0, 1, 2, 0, 1]
The index of the most present letter is :  16
The letter that appears the most in the encrypted text is :  q
The key is:  12
This text was encrypted with the caesar code with a key equal to twelve
>>>
```

Vigenère's Code

- De/Code
- Decypher

Vigenère (1523-1596)

	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

Vigenère's square

De/Code with Vigenère's Code

Encryption

- Variable shift
- Poly-alphabetical substitution
- An infinity of keys (25^n)

Clear text : COMPUTERSCIENCE
key : CODE

Text →	C	O	M	P	U	T	E	R	S	C	I	E	N	C	E
Key →	C	O	D	E	C	O	D	E	C	O	D	E	C	O	D
Encrypted text →	F	D	Q	U	X	I	I	W	V	R	M	J	Q	R	I

Decryption

- More complex than Caesar's Code
- Specific case: Vernam's number

Text →	3	15	13	16	21	20	5	18	19	3	9	5	14	3	5
key →	3	15	4	5	3	15	4	5	3	15	4	5	3	15	4
(text+key)%26 →	6	4	17	21	24	9	9	23	22	18	13	10	17	18	9

Vigenère_De_Encryption.py - C:\Users\Mathis\Desktop\UNIV\PPR\L2\Vigenere

File Edit Format Run Options Window Help

```
from math import *
a=input("Encryption (E) or Decryption (D) : ")
b=input("Key: ")
Lc=[]
Lkey=[]
Lf=[]
final=''
k=0
i=0
if a=='E':
        c=input("What is the text to encrypt? ")
        for p in range(len(b)):
                Lkey.append(b[p])
        while len(Lkey)>len(c):
                del(Lkey[-1])
        while i<len(c):
                Lc.append(c[i])
                if 64<ord(Lc[i])<92:
                        val1=ord(Lc[i])-65
                        val2=ord(Lkey[i%(len(b))])-97
                        val=(val1+val2)%26
                        val=val+65
                        Lf.append(chr(val))
                elif 96<ord(Lc[i])<123:
                        val1=ord(Lc[i])-97
                        val2=ord(Lkey[i%(len(b))])-97
                        val=(val1+val2)%26
                        val=val+97
                        Lf.append(chr(val))
                elif Lc[i]==' ':
                        Lf.append(' ')
                i=i+1
        while k<len(Lf):
                final=final+Lf[k]
                k=k+1
        print(final)
#-----------------------------------------------------
elif a=='D':
        c=input("What is the text to decrypt ? ")
        for p in range(len(b)):
                Lkey.append(b[p])
        while len(Lkey)>len(c):
                del(Lkey[-1])
        while i<len(c):
                Lc.append(c[i])
                if 64<ord(Lc[i])<92:
                        val1=ord(Lc[i])-65
                        val2=ord(Lkey[i%(len(b))])-97
                        val=(val1-val2)%26
                        val=val+65
                        Lf.append(chr(val))
                elif 96<ord(Lc[i])<123:
                        val1=ord(Lc[i])-97
                        val2=ord(Lkey[i%(len(b))])-97
                        val=(val1-val2)%26
                        val=val+97
                        Lf.append(chr(val))
                elif Lc[i]==' ':
                        Lf.append(' ')
                i=i+1
        while k<len(Lf):
                final=final+Lf[k]
                k=k+1
        print(final)
```

De/Code

```
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\Mathis\Desktop\UNIV\PPR\L2\Vigenere\Vigenère_De_Encryption.py

Encryption (E) or Decryption (D) : E
Key: informatics
What is the text to encrypt? This text was encrypted with the vigenere code
Bung feqb oif seorrxvwl bwkt mpg dvlseqrx eglr
>>>
 RESTART: C:\Users\Mathis\Desktop\UNIV\PPR\L2\Vigenere\Vigenère_De_Encryption.py

Encryption (E) or Decryption (D) : D
Key: informatics
What is the text to decrypt ? Bung feqb oif seorrxvwl bwkt mpg dvlseqrx eglr
This text was encrypted with the vigenere code
>>>
```

Decyphering Vigenère's Code

Kasiski's test

- Analyse the text
- Decompose the distance
- Find key length
- Decompose the text
- Decode

CS AZZMEQM CO XRWF CS DZRM GFMJECV. X'IMOQJ JC LB NLFMK CC LBM WCCZBM KFIMSZJSZ CS URQIUOU. CS ZLPIE ECZ RMWWTV SB KCCJ QMJ FCSOVJ GCI ZI ICCKS MK QMLL YL'CV ECCJ OKTFWTVM JIZ CO XFWBIWVV IV ACCI CC C'OCKFM JINWWB U'OBKSVUFM

Sequence	Position	Distance	Decomposition
COX	11-140	129	3.43
FCS	16-99	83	83
ZRM	20-83	63	3.3.7
FMJ	24-162	138	2.2.3.3
CLB	37-46	9	3.3
KCC	44-92	48	2.2.2.3
WTV	87-133	46	2.23
CCJ	93-126	33	3.11
ICC	110-155	45	3.3.5
MJI	136-163	27	3.3.3

Kasiski's test

Kasiski.py - C:\Users\Mathis\Desktop\UNIV\PPR\L2\Vigenere\Kasiski.py (3.6.4)

File Edit Format Run Options Window Help

```
def PGCD (a,b) :
    if a <= 0 or b <= 0 :
        print("Error")
    if a == 1 or b == 1 :
        return 1
    if a == b :
        return a
    if a < b :
        return PGCD (a, b-a)
    return PGCD (b, a-b)
#---------------------------------
def KeyLength (text, word = 3) :
    dictionary = {}
    for i in range (0, len (text)-2) :
        t = text [i:i+word]
        if t in dictionary : dictionnary [t].append (i)
        else : dictionary [t] = [i]
    distance = []
    for d in dictionary:
        p = dictionary [d]
        if len (p) > 1 :
            for i in range (0, len (p)-1) :
                distance.append ( p [i+1] - p [i] )
    length = PGCD (distance [0], distance [1])
    for d in distance :
        length = PGCD (length, d)
    return "The key length is :", length

#-------------------------------------------------------------------
def FrequencyAnalysis(c):
    L_letter=['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']
    L_nbr=[0]*26
    for i in range(len(c)):
        for j in range(len(L_letter)):
            if c[i]==L_letter[j]:
                L_nbr[j]=L_nbr[j]+1
    maxi=max(L_nbr)
    for i in range(len(L_nbr)):
        if L_nbr[i]==maxi:
            index=i
    return(L_letter[index])

#|---------------------------------
def Key(encryptedtext,keylength):
    a=""
    key=""
    for j in range(0,keylength):
        for i in range(j,len(encryptedtext),keylength):
            a=a+encryptedtext[i]
            letter=FrequencyAnalysis(a)
            key=key+letter
            a=""
    print("The key is:",key)
```

```
 RESTART: C:\Users\Mathis\Desktop\UNIV\PPR\L2\Vigenere\Vigenère_De_Encryption.py

Encryption (E) or Decryption (D) : E
Key: surf
What is the text to encrypt? this text was encrypted by the code of vigenere wit
h the key surf this text need to be long enough so we can find groups of letters
 that repeat in order to find the length of the key thanks to kasiski test
lbzx nvcl nfk vsulpulyu ts yzy hgxv gz aaavswlv ockm nyj evd mlwx kmam ywrk fyvi
 nf ty qghx whfzyb xg nj wrs zzsv xwgogx iw dykywlj lbry lvuwuk ah tjxvw nf xcei
 nyj fvsyny gz yzy pws yzuepk kt erxambn nvxl
>>>
```

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64 bit (AMD64)]
 on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/Mathis/Desktop/kasiskisa.py ==========
>>> KeyLength("lbzx nvcl nfk vsulpulyu ts yzy hgxv gz aaavswlv ockm nyj evd mlwx
 kmam ywrk fyvi nf ty qghx whfzyb xg nj wrs zzsv xwgogx iw dykywlj lbry lvuwuk a
h tjxvw nf xcei nyj fvsyny gz yzy pws yzuepk kt erxambn nvxl")
The key length is : 4
>>> Key("lbzx nvcl nfk vsulpulyu ts yzy hgxv gz aaavswlv ockm nyj evd mlwx kmam
ywrk fyvi nf ty qghx whfzyb xg nj wrs zzsv xwgogx iw dykywlj lbry lvuwuk ah tjxv
w nf xcei nyj fvsyny gz yzy pws yzuepk kt erxambn nvxl")
The key is: surf
>>>
```

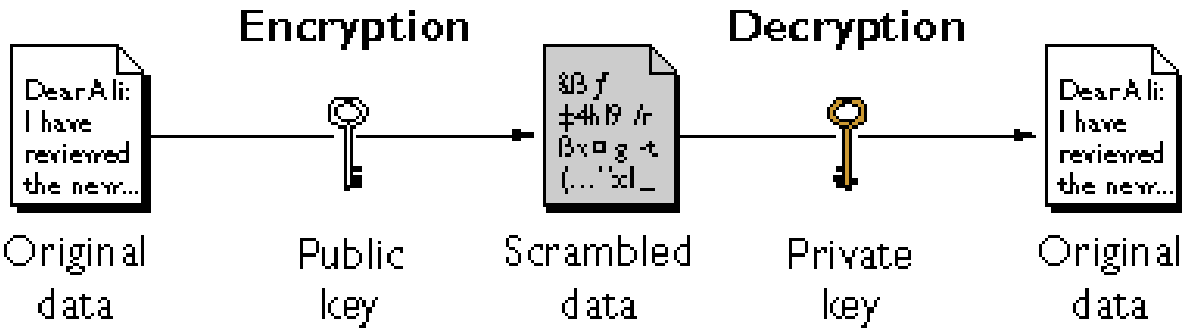
« Asymmetric-key » Code

• Asymmetric-key codes are codes that use differents keys to encrypt and decrypt the text.

•The public key is known from everyone and can only be used to encrypt a text.

•The private key is only known by the creator who decrypts messages.

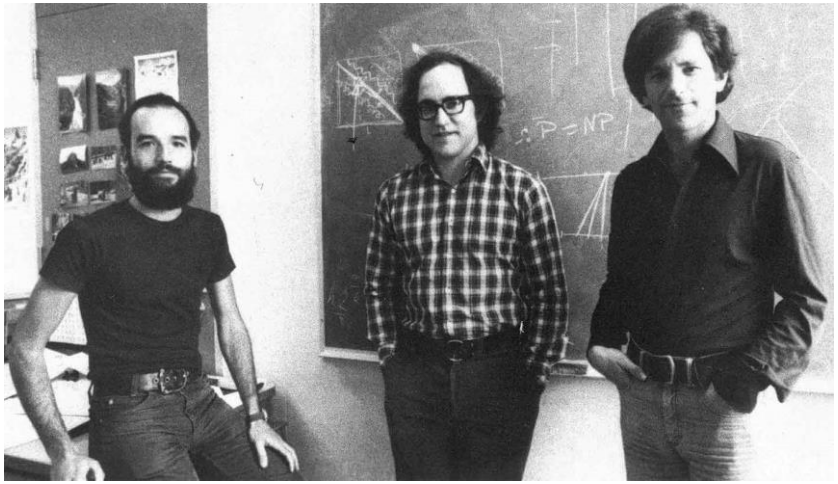
➢ RSA



RSA Code

- De/Code
- Decypher

1977



Rivest / **S**hamir / **A**dleman

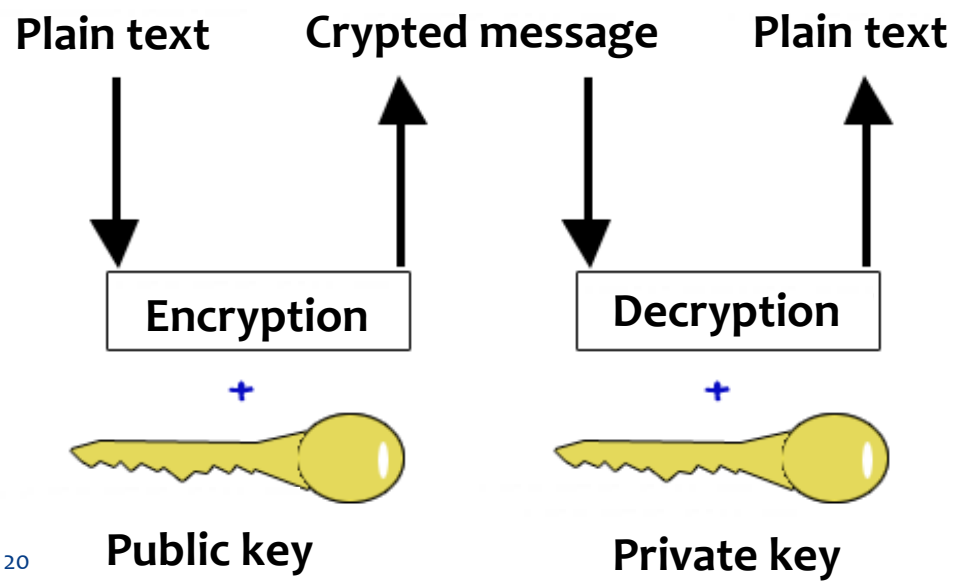
De/Code 1/2

Preparation of the keys

- Choose 2 differents prime numbers : p and q
- n=p*q
- Φ(n)=(p-1)*(q-1)
- Choose e, such as : pgcd(e,Φ(n)) = 1
  - **Euclide's algorithm**
- Choose d such as : d*e – Φ(n) = 1
  - **Bezout's coefficients**

The keys

- Public key : (e,n)
- Private key : (d,n)



De/Code 2/2

Encryption

- **Use public key (e,n)**
- Everyone can encrypt a message
- Transform every letter by a number such as 0≤m<n
- (ex:*A=01 ; B=02 ; … ;Z =26 or ASCII*)
- **x = $m^e \mathbf{mod}(\mathbf{n})$**

m = plain text converted in number
x = crypted text in number

Decryption

- **Use private key (d,n)**
- Only the owner of the private key can decrypt all the messages
- Little theorem of Fermat (improved)

↳ **m = $x^d mod(n)$**

d is the inverse of « e mod(Φ(n)) »

De/Code 1/2

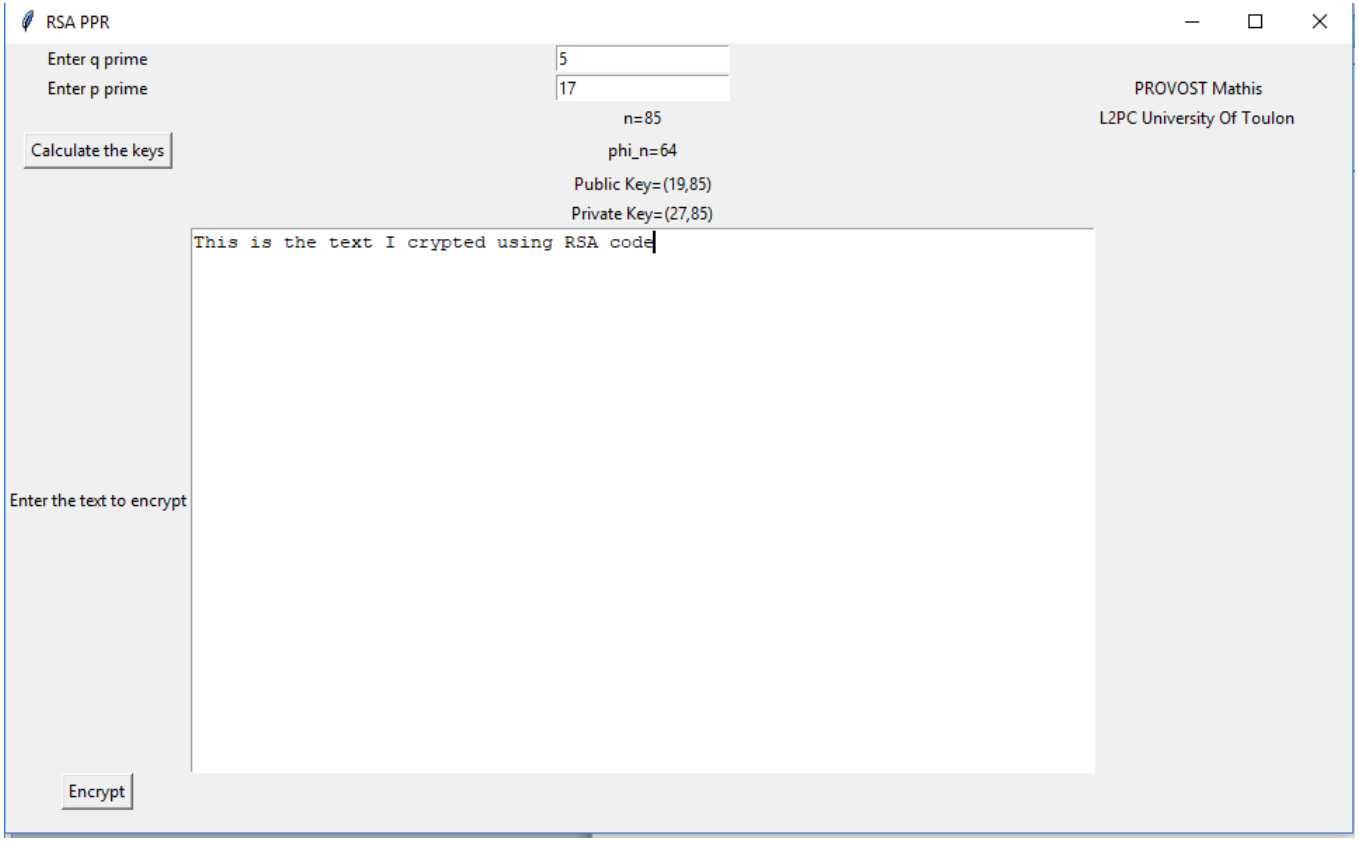
Core code

```
from tkinter import *
from math import *

#Functions
def Calc():
    global n
    pl=int(p.get())
    ql=int(q.get())
    n=pl*ql
    chaine.configure(text='n='+str(n))
    global phiden
    phiden=(pl-1)*(ql-1)
    chainel.configure(text='phi_n='+str(phiden))
    r=0
    global e
    e=0
    i=1
    r=0
    while r==0:
        if((pl<e)and(ql<e)and(e<phiden)and(e/i!=phiden/i)):
            r=1
        i=i+1
        e=e+1
    chaine2.configure(text='Public Key=('+str(e)+','+str(n)+')')
    global d
    d=0
    i=1
    r=0
    while r==0:
        if(e*d%phiden==1):
            r=1
        i=i+1
        d=d+1
    chaine3.configure(text='Private Key=('+str(d-1)+','+str(n)+')')

def wordcrypt():
    motl=str(mot.get(0.,END))
    taille=len(motl)
    i=0
    crypter = " "
    while i<taille:
        ascii = ord(motl[i])
        lettre_crypt = ascii**e%n
        crypter = crypter + str(lettre_crypt) + " "
        i = i + 1
    open("message.txt", "w").write(crypter)
```

Tkinter interface

```
fenl = Tk()
fenl.title('RSA PPR')
#Entry positioning
p = Entry(fenl)
q = Entry(fenl)
mot = Text(fenl)
#Display instructions
pt= Label(fenl, text='Enter p prime')
qt= Label(fenl, text='Enter q prime')
mt= Label(fenl, text='Enter the text to encrypt')
mp=Label(fenl, text='PROVOST Mathis')
fac=Label(fenl, text='L2PC University Of Toulon ')
#Text
chaine= Label()
chainel= Label()
chaine2= Label()
chaine3= Label()
#Buttons
bl= Button(text='Calculate the keys', command=Calc)
b2= Button(text='Encrypt', command=wordcrypt)
bl.grid(row =3)
b2.grid(row =7)
#Text positioning
chaine.grid(row =2, column =1)
chainel.grid(row =3, column =1)
chaine2.grid(row =4, column =1)
chaine3.grid(row =5, column =1)
#Instructions positioning
qt.grid(row =0)
pt.grid(row =1)
mt.grid(row =6)
mp.grid(row =1, column =2)
fac.grid(row =2, column =2)
#Entry areas positioning
p.grid(row =0, column =1)
q.grid(row =1, column=1)
mot.grid(row =6, column=1)
fenl.mainloop()
```

De/Code 2/2



84 59 10 55 43 10 55 43 41 59 16 43 41 16 35 41 43 57 43 24
79 76 48 41 16 60 43 43 55 10 70 52 43 58 77 75 43 24 66 60 16
65

To decrypt : m = $x^d mod(n)$

How to decypher ?

« Classical » way

- We know n , we want to know p and q by decomposing 'n' in prime numbers to calculate the private key d

➡ Take p and q very big, it's really hard to find the decomposition of an astronomous number

Ex: if n=85 -> p and q ? Don't take too long to find out, p=5 and q=17
But if
n=120017095106177134145000034096145147000060017116095073 11
then what are p and q ?

Solution ➡ Change the keys often

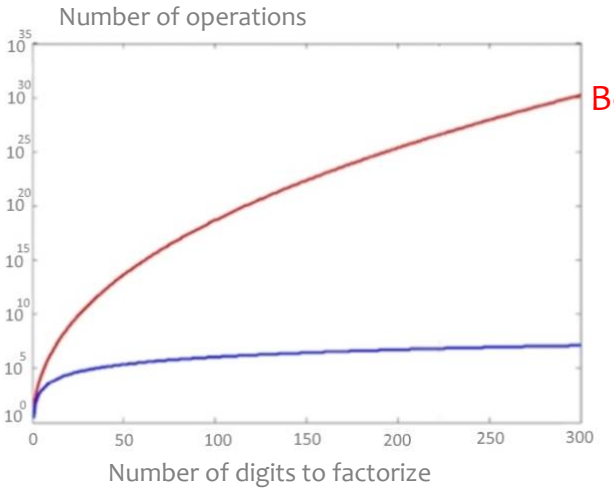
Other ways

Human's mistakes

- Intercept the key when exchanged
- Hack the computer to find the private key

Quantum mechanics
However, more recently the use of quantum computers seems more promising than anything else.

➡ **Shor's algorithm**



Cryptology and its future

Some words to conclude…

- Evolution
- Necessity
- Protection
- Dangerosity

What futur for cryptology ?

Bibliography/Sitography

Informations on cryptology

- Philippe Guillot. « Cryptologie : l'art des codes secrets ». Futura-sciences.com (November 23 2015)
https://www.futura-sciences.com/sciences/dossiers/mathematiques-cryptologie-art-codes-secrets-1817/

- Lucia Sillig. « La cryptographie, un art toujours perfectible ». CourrierSciences (January 12 2011)
https://www.courrierinternational.com/article/2011/01/13/la-cryptographie-un-art-toujours-perfectible

- Frédéric Bayart. « Le lexique de la cryptographie ». Bibm@th.net
http://www.bibmath.net/crypto/index.php?action=affiche&quoi=moderne/vocabulaire

- ANSSI (agence nationale de la sécurité des systèmes d'information)
http://www.securiteinformatique.gouv.fr/autoformations/cryptologie/co/Cryptologie.html

- Simon Singh «Histoire des codes secrets» ISBN Poche: 978-2253150978 (1999)

- Didier Müller. Edition City «Les Codes secrets Décryptés». ISBN: 978-2-35288-544-3 (2007)

Caesar's Code

- Didier Müller. « Chiffre de César ». Nymphomath.com (June 11 2001)
http://www.nymphomath.ch/crypto/cesar/index.html

Vigenère's Code

- « Chiffre de Vigenère ». Wikipédia, (May 18th 2018)
https://fr.wikipedia.org/wiki/Chiffre_de_Vigenère

- « Chiffre de Vigenère ». Dcode, 2018
https://www.dcode.fr/chiffre-vigenere

- Didier-Müller. « Décryptement du chiffre de Vigenère ». Nymphomath.com (April 4th 2001)
https://www.apprendre-en-ligne.net/crypto/vigenere/decodevig.html

- Jean-Pierre Zanotti. « Le chiffrement de Vigenère ». Zanotti.univ-tln.fr,
http://zanotti.univ-tln.fr/crypto/vigenere.html

Kasiski

- « Test de Kasiski ». Numb3rs-Singularity (August 6th 2008)
http://www.numb3rs-singularity.fr/mathematiques/par-nom/test-de-kasiski

- Frédéric Bayart.« Comment vaincre le chiffre de Vigenère? ». *Bibm@th.net*
http://www.bibmath.net/crypto/index.php?action=affiche&quoi=poly/viganalyse

RSA

Douglas Stinson. « Cryptographie, théorie et pratique, 2e édition » (October 6th 2003)
ISBN : 978-2-7117-4800-6
Editor : Vuibert

Thanks to Mr.Jean-Pierre Zanotti, of the laboratory iMath, for his time, his help and his precious advice in the creation of my project on cryptology.