Universidad de Granada

Práctica 2. Criptosistemas afines

Seguridad y Protección de Sistemas Informáticos
Grado en Ingeniería Informática

José María Martín Luque

20 de octubre de 2019

Ejercicios 1, 2 y 4

Para obtener el texto en claro del mensaje cifrado proporcionado en el ejercicio 4 (listado 1) se ha utilizado un ataque basado en la prueba *chi-cuadrado*, para lo que ha sido necesario realizar los ejercicios 1 y 2.

La técnica que subyace tras este tipo de ataque consiste en descifrar el mensaje utilizando todas las posibles claves —en un alfabeto de 26 caracteres son $12 \cdot 26 - 1 = 311$— y comparar mediante un test *chi cuadrado* la frecuencia observada con la que aparecen las letras en dicho posible mensaje descifrado con la frecuencia esperada de dichas letras, es decir, la frecuencia en la que aparece en el idioma en el que sabemos que está escrito el mensaje original.

```
1 BZBTJAVGQGVOTBVGTKFNGQGVMTTNWVYNBFZBATDTBZYNYTBRANBMTDTBTDEVFZBGTYTBNKNGNBWGNXKN
2 BGTWZGVBZTFQGTNQRAWVYTGVFQTGBTXVWGNBFTMVXYTDNSNQVGTBWTTKTFTAWVLRTQNGTDTBTGZANDDT
3 BZIKTNKEVFIGTTAIRBDNYTKNBQZTSNBLRTMZMTATAFZBIVBLRTBBRIFNGZAVBFZBGTINJAVBDVFVKVBY
4 TKMZTUVQNBWVGYTATQWRAVQNDTABZAWTFVGTAKNBZAFTABNBQGNYTGNBYTKVDTNAVWTAJVXVNEZRANMN
5 BWNQGVQZTYNYLRTTCQKVWVXVFZBFVXLRTTBWNBTFIGNYNQVGKNFNAVYTKDGTNYVGYTWVYNBKNBDVBNB
```

Listado 1: Mensaje cifrado

Para ello se ha implementado un programa en Python, mostrado y comentado en el listado 3, que recibe como entrada —o bien solicita al ejecutarse— un mensaje

cifrado bajo un criptosistema afín basado en un alfabeto castellano normalizado de 26 caracteres —es decir, sin la ñ—.

El mensaje descifrado descrito en el listado 2 es un fragmento de la obra del del escritor francés Julio Verne «Veinte mil leguas de viaje submarino». Concretamente pertenece a una intervención del Capitán Nemo en la que responde al profesor Aronnax.

> *Sí, señor profesor. El mar provee a todas mis necesidades. Unas veces echo mis redes a la rastra y las retiro siempre a punto de romperse, y otras me voy de caza por este elemento que parece ser inaccesible al hombre, en busca de las piezas que viven en mis bosques submarinos. Mis rebaños, como los del viejo pastor de Neptuno, pacen sin temor en las inmensas praderas del océano. Tengo yo ahí una vasta propiedad que exploto yo mismo y que está sembrada por la mano del Creador de todas las cosas.*

```
1 sisegnorprofesorelmarproveeatodasmisnecesidadesunasvecesechomisredesalarastrayla
2 sretirosiempreapuntoderomperseyotrasmevoydecazaporesteelementoqueparecesserinacce
3 siblealhombreenbuscadelaspiezasquevivenenmisbosquessubmarinosmisrebagnoscomolosd
4 elviejopastordeneptunopacensintemorenlasinmensaspraderasdeloceanotengoyoahiunava
5 stapropiedadqueexplotoyomismoyqueestasembradaporlamanodelcreadordetodaslascosas
```

Listado 2: Mensaje descifrado

```
1  #!/usr/bin/env python3
2  # archivo:   spsi.p2.py
3  # asignatura: Seguridad y Protección de Sistemas Informáticos
4  # práctica:  Práctica 2
5  # autor:     José María Martín Luque
6
7  import sys
8
9  def texto_a_numeros(texto):
10     """Convierte el texto recibido a una cadena de números.
11
12     Cada número que representa la posición de cada carácter en el alfabeto —sin
13     la —ñ, que es reemplazada por ny.
14
15     El texto se codifica en minúscula.
16     """
17
```

```
18      texto = texto.lower()
19      texto = texto.replace("ñ", "ny")
20
21      numeros = []
22      for c in texto:
23          n = ord(c) - 97
24          numeros.append(n)
25      return numeros
26
27  def numeros_a_texto(numeros):
28      """Convierte la lista de números recibida en un string entendiendo que cada
29      número es la posición de una letra en el alfabeto
30      """
31
32      texto = ""
33      for n in numeros:
34          c = chr(n + 97)
35          texto = texto + c
36      return texto
37
38  def inverso_modulo_n(x, n):
39      """Calcula el inverso de un número 'x' módulo 'n'"""
40      for i in range(n):
41          if ((i*x) % n == 1):
42              return i
43
44  def afin(a, b, x, s):
45      """Función que cifra y descifra mediante un criptosistema afín en un
46      alfabeto de 26 caracteres
47
48      Parámetros:
49      a, b -- claves de cifrado
50      x -- 0 si cifra, 1 si descifra
51      s -- cadena a cifrar
52
53      Cada caracter cifrado 'e' se obtiene mediante la operación
54          e(c) = (a*c + b) % n,
55      donde 'c' es el caracter a cifrar.
56
57      Para descifrar un carácter realizamos la operación
```

```
58          c(e) = (a_p*e + b_p) % n,
59      donde 'e' es el carácter cifrado y 'a_p' y 'b_p' vienen dados por
60          a_p = a^-1 % n,
61          b_p = (-a_p * b) % n.
62
63      Para el cálculo de los inversos módulo 26 se utiliza la función
64      inverso_modulo_n(x, n)
65      """
66
67      numeros = texto_a_numeros(s)
68
69      if (x):
70          for i, num in enumerate(numeros):
71              a_p = inverso_modulo_n(a, 26)
72              b_p = (-a_p * b) % 26
73              numeros[i] = (a_p*num + b_p) % 26
74      else:
75          for i, num in enumerate(numeros):
76              numeros[i] = (a*num + b) % 26
77
78      return numeros_a_texto(numeros)
79
80  """Frecuencias relativas de las letras en el castellano"""
81  f_a = {
82      "a" : 12.53,
83      "b" : 1.42,
84      "c" : 4.68,
85      "d" : 5.86,
86      "e" : 13.68,
87      "f" : 0.69,
88      "g" : 1.01,
89      "h" : 0.70,
90      "i" : 6.25,
91      "j" : 0.44,
92      "k" : 0.02,
93      "l" : 4.97,
94      "m" : 3.15,
95      "n" : 6.71,
96      "o" : 8.68,
97      "p" : 2.51,
```

```
 98     "q" : 0.88,
 99     "r" : 6.87,
100     "s" : 7.98,
101     "t" : 4.63,
102     "u" : 3.93,
103     "v" : 0.90,
104     "w" : 0.01,
105     "x" : 0.22,
106     "y" : 0.90,
107     "z" : 0.52
108 }
109
110 def ataque_chi_cuadrado(s):
111     """Función que descifra un mensaje cifrado mediante un criptosistema afín
112     utilizando la prueba chi cuadrado.
113
114     Parámetros:
115     s -- Mensaje a descifrar
116
117     La técnica que subyace tras este ataque consiste en descifrar el mensaje
118     utilizando todas las posibles claves —en un alfabeto de 26 caracteres son
119     12*26 - 1 = -311 y comparar mediante un test chi cuadrado la frecuencia
120     observada con la que aparecen las letras en dicho posible mensaje
121     descifrado con las del idioma en el que sabemos que está escrito el mensaje
122     original, esto es, la frecuencia esperada de dicha letra.
123
124     La probabilidad de frecuencia característica 'f_a' se ha obtenido de
125     Wikipedia y se puede consultar en
126         https://es.wikipedia.org/wiki/Frecuencia_de_aparici%C3%B3n_de_letras#
                Frecuencia_de_aparici%C3%B3n_de_letras_en_espa%C3%B1ol
127     """
128
129     d = []
130
131     for a in [1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25]:
132         for b in range(26):
133             m = afin(a, b, 1, s)
134
135             """ Las siguientes variables almacenan,
136             'o_m' las frecuencias observadas para cada caracter,
```

```
137             'e_a' las frecuencias esperadas obtenidas a partir de 'f_a', y
138             'r' el grado chi cuadrado del mensaje 'm'.
139             """
140             o_m = {}
141             e_a = {}
142             r = 0
143 
144             for n in range(26):
145                 c = chr(n+97)
146                 o_m[c] = m.count(c)
147                 e_a[c] = f_a[c]/100 * len(m)
148 
149                 r = r + ((o_m[c] + e_a[c])**2)/e_a[c]
150             d.append((r, m))
151 
152     d.sort()
153 
154     return d[0]
155 
156 def main(argv):
157     if len(argv) < 2:
158         c = input("Introduce un mensaje a descrifrar: ")
159         print(ataque_chi_cuadrado(c)[1])
160     else:
161         for c in argv[1:]:
162             print(ataque_chi_cuadrado(c)[1])
163 
164 if __name__ == "__main__":
165     main(sys.argv)
```

Listado 3: Programa que realiza el ataque *chi-cuadrado*