Devoir 1

Par
Louis Pelletier & Charlotte de Lanauze

Présenté à Alain Tapp

Université de Montréal
Cours IFT3275-A-H22 - Sécurité informatique
18 novembre 2021

Question 1.

Factoriser N dans RSA-textbook est un problème difficile. Étant donné l'indice dans la question (Auteur d'Amérique latine), il est donc plus simple d'essayer de choisir le M et d'appliquer M^e mod n avec E =3. Comme M^3 serra considérablement plus petit que N, le mod est inutile dans cette situation.

Nous avons essayé avec quelques noms, dont : Gabriel García Márquez, Mario Vargas Llosa et finalement Jorge Luis Borges.

Jorge Luis Borges fut la bonne réponse, et voilà comment nous y sommes arrivés.

« Jorge Luis Borges »
= [J, o, r, g, e, _, L, u, i, s, _, B, o, r, g, e, s]
= [74, 111, 114, 103, 101, 32, 76, 117, 105, 115, 32, 66, 111, 114, 103, 101, 115]
=
01001010011011110111001001100111011001010010000001001100011101010110100101110011001000000100001001101111011100100110011101100101011001101
= 25329033478623674471733103854802826650995

En appliquant
25329033478623674471733103854802826650995^3

Nous avons :

16250093121834914738489778721283850815023220823910545716502847730384833157965639674158854530545421681485384544525983824875

Comme il est plus petit que N, il est donc inutile de faire mod N. Nous obtenons alors la valeur de C qui est notre cryptogramme. Cette méthode nous a pris 3 essais, ce qui est significativement moins long que d'essayer de factoriser N. Cependant, vu que le mod N n'était pas utilisé, on aurait aussi pu simplement faire la

racine cubique du cryptogramme C, mais cette option n'est normalement pas possible lorsque $M^e > N$.

Question 2.

AES2(x) est vulnérable à une attaque à texte clair connu. Il suffit d'avoir accès à un message en clair x pour pouvoir briser AES2(x) avec au plus 2^{257} évaluations. En effet, nous n'aurions qu'à essayer toutes les 2^{256} clés possibles et pour chacune de ces clés, sauvegarder en mémoire le chiffrement de x à l'aide de cette clé ainsi que le déchiffrement de AES2(x) à l'aide de cette même clé. Puis, on n'a qu'à regarder si un chiffrement de x correspond à un déchiffrement de AES2(x) et on aura alors trouvé le chiffre milieu AESk2(x) et la clé utilisée pour chiffrer x sera k2 et la clé utilisée pour déchiffrer AES2(x) sera k1.

Au lieu d'effectuer une recherche exhaustive qui est 2^{256} fois plus complexe et qui donne une complexité finale de 2^{511}. Avec cette attaque la complexité est seulement doublée. Donc elle est de 2^{257}.

Question 3.

Il est possible d'encrypter un message avec RSA qui n'est pas co-premier avec N. Cependant, cela mène à un gros problème. En effet, cela rend RSA encore plus sécuritaire, au point tel qu'il devient impossible de déchiffrer le message.

Prenons par exemple N = 15 & e = 3.

Avec le message 2.

Nous avons, 2^3 mod 15 = 8.

Avec le message 32.
Nous avons, 32^3 mod 15 = 8.

Nous obtenons le même cryptogramme avec les deux messages différents. Donc, en déchiffrant, il va être impossible de déterminer quel est le bon message. Ce qui peut être très dangereux dans certains cas. Il est possible d'avoir plus de deux messages différents donnant le même cryptogramme aussi bien évidement.

Question 4.

Il est possible de déchiffrer ce cryptogramme pour la raison suivante.
Si deux messages M1 et M2 sont chiffrés avec la même clé k0, alors on a que leurs cryptogrammes C1 et C2 ont la relation suivante : C1 XOR C2 = M1 XOR M2.
On a donc directement de l'information sur M1 et sur M2 et on peut ensuite déduire ces deux messages grâce à la distribution non uniforme des caractères dans la langue française.

Par exemple, en sachant que l'espace est le caractère le plus présent dans la langue française avec une fréquence de 19.3%, on peut calculer la probabilité d'avoir 'espace' XOR 'espace' dans M1 XOR M2 qui est $19.3\%^2$.
En sachant que la fréquence du E est de 13.9%, on peut calculer que la probabilité d'avoir E XOR 'espace' dans M1 XOR M2 est de 2x(19.3% x 13.9%) et ainsi de suite.

On peut aussi deviner plusieurs caractères grâce aux motifs (patterns) de la langue française, par exemple le fait qu'un mot d'une seule lettre sera souvent un « a », qu'un espace ne sera probablement pas suivi d'un autre espace ou d'une ponctuation, qu'une ponctuation sera surement suivie d'un espace, etc.

Nous avons conçu un programme Java qui fait l'analyse statistique du M1 XOR M2 et nous optenons les fréquences

suivantes pour chaque suite de 8 bits :

01100101 0.0013927576
00101100 7.4994646E-4
01100100 0.0013927576
00100011 8.570816E-4
00100010 8.570816E-4
00000001 0.02571245
00000000 0.07220913
10000001 9.6421683E-4
01111110 0.0012856225
01111111 0.001071352
00011011 0.01607028
10000010 2.142704E-4
10001100 0.0024641098
10000011 7.4994646E-4
00000010 0.017784445
01011101 0.0012856225
01010011 0.022176987
01011100 0.001071352
01010010 0.016713092
00100001 4.285408E-4
11010010 1.071352E-4
11011101 1.071352E-4
10101111 1.071352E-4
00100000 0.0012856225
00001101 0.015213199
00000011 0.014998929
00001100 0.025819585
00110101 5.35676E-4
10000111 0.002571245
10011010 0.002571245
10011011 0.0019284337
00110100 2.142704E-4
11000101 4.285408E-4
01001011 0.0022498392
11000100 2.142704E-4
01110010 2.142704E-4
01111100 3.2140562E-4

01110011	2.142704E-4
10011001	5.35676E-4
10000000	8.570816E-4
00001110	0.007285194
01010001	0.004178273
01010000	0.0059995716
00100111	6.4281124E-4
11011111	3.2140562E-4
11011110	1.071352E-4
00101000	5.35676E-4
00101001	5.35676E-4
01100001	1.071352E-4
10101001	2.142704E-4
00001111	0.0103921145
10000100	0.0013927576
10011000	0.0011784872
10000101	0.0016070281
00111001	6.4281124E-4
01001001	0.021641312
01001000	0.0048210844
11001001	0.0039640027
11001010	0.0012856225
01111011	2.142704E-4
01001010	0.0016070281
00111010	5.35676E-4
10110000	1.071352E-4
00111011	1.071352E-4
00010101	0.011892008
10011111	6.4281124E-4
00010100	0.013820441
10000110	0.0016070281
00001000	0.013070495
01010111	6.4281124E-4
01010110	0.0049282196
11100011	3.2140562E-4
01100010	2.142704E-4
11010111	4.285408E-4
01100110	0.0013927576
00101110	5.35676E-4

01100111 1.071352E-4
00101111 6.4281124E-4
00001001 0.014998929
10011110 5.35676E-4
01001111 0.013927577
01001110 0.0209985
01111010 6.4281124E-4
01111000 7.4994646E-4
01111001 0.0013927576
00010110 0.022069853
10011100 0.0022498392
10010011 9.6421683E-4
00010111 0.020034283
10011101 0.0017141632
01010100 0.022391258
00001011 0.019712878
00001010 0.018855795
00101101 6.4281124E-4
01010101 0.020677095
01101111 0.002571245
10010010 2.142704E-4
00110011 2.142704E-4
00111100 6.4281124E-4
00111101 5.35676E-4
01000011 0.008463681
01001100 0.015213199
01000010 0.0077137346
11000011 1.071352E-4
11001100 1.071352E-4
01001101 0.009749304
11001101 1.071352E-4
11001110 0.001071352
01110100 1.071352E-4
01110101 1.071352E-4
00010000 0.019927148
00111110 6.4281124E-4
10010000 2.142704E-4
00010001 0.026462395
10010001 4.285408E-4

01101000 2.142704E-4
00101010 0.004178273
01100011 0.0019284337
01101100 5.35676E-4
01101101 4.285408E-4
00111111 4.285408E-4
01000000 0.003106921
11001111 1.071352E-4
01000001 0.020141419
11001000 0.0012856225
00011100 0.019927148
00010011 0.010820656
00010010 0.01071352
00111000 4.285408E-4
10001010 3.2140562E-4
10001011 0.0013927576
10010110 1.071352E-4
00011101 0.020462824
01011011 5.35676E-4
01011010 0.0018212985
01101110 7.4994646E-4
11010100 2.142704E-4
11010101 3.2140562E-4
00100110 0.001071352
00000101 0.015106064
00000100 0.02121277
00110110 0.0012856225
00110111 3.2140562E-4
01000110 0.0044996785
11000110 2.142704E-4
01000111 0.0037497322
11000111 6.4281124E-4
01110000 7.4994646E-4
00011111 0.013713306
10010101 6.4281124E-4
00011110 0.010927791
10001000 0.0014998929
10010100 5.35676E-4
10001001 6.4281124E-4

01011001 0.0024641098
01011000 0.0034283265
00100101 3.2140562E-4
00100100 1.071352E-4
11011001 1.071352E-4
11011000 5.35676E-4
10100101 1.071352E-4
01101001 7.4994646E-4
00000111 0.026462395
00000110 0.025176773
10001101 0.0011784872
00011010 0.020462824
00110000 4.285408E-4
00110001 0.0011784872
01000101 0.04349689
11000000 7.4994646E-4
01000100 0.010820656
11110011 1.071352E-4
11000010 1.071352E-4
01110111 1.071352E-4
00011001 0.01296336
00011000 0.010820656
00110010 3.2140562E-4
10001110 9.6421683E-4
10001111 0.0012856225
01011111 0.002571245
01011110 0.0022498392
01101010 1.071352E-4
01101011 4.285408E-4
11011011 3.2140562E-4

On peut ainsi voir que la séquence de 8 bits la plus fréquente est ‘00000000’ avec une fréquence de 7.220913%, ce qui veut dire que c’est la proportion des caractères qui sont identiques entre M1 et M2.

Malheureusement, nous n’avons pas réussi à pousser notre analyse statistique assez loin pour pouvoir déchiffrer une partie de M1 ou de M2. Seulement une partie aurait été nécessaire pour

ensuite faire une recherche Google et rapidement trouver le reste du poème.