Devoir 1 - Rapport de l'équipe 13

Nikolas Lévesque (20276665) et Abdelmouhcine Messaad (2151011)

11 novembre 2024

Résumé

Ce rapport a pour but de présenter et justifier les choix que nous avons effectués lors de notre travail sur les questions 1.1, 1.2 et 2. Le rapport se termine par une courte bibliographie.

1 Question 1.1

Le message clair trouvé est **Umberto Eco**!

On cherche le message clair 'M' à partir du cryptogramme 'C' avec la formule du chiffrement RSA-textbook :

$$C = M^e \ mod N$$

1.1 Démarche

Étant donné que l'exposant 'e' est vraiment petit et que le message 'M' est suffisamment petit pour que 'M' au cube soit plus petit que 'N', on peut supposer que 'C' est égal à 'M' au cube. On peut donc obtenir 'M' en calculant la racine cubique entière de 'C'. Une fois 'M' obtenu, on le convertit en une chaîne de caractères **en inversant le processus de str_to_int**. Puisque qu'on a supposé que 'M' cube est inférieur à 'N', pas besoin de tenir compte du modulo 'N'.

Quand des petits exposants publics sont utilisés sans padding avec un message court, trouver le message est trivial, d'où l'importance d'utiliser OAEP comme vu en cours.

2 Question 1.2

Le message clair trouvé est **Marcel Proust**

Pour la question 1.2, impossible d'utiliser la même stratégie qu'à la question précédente. Le site dcode est lui-même incapable de déchiffrer le message sous prétexte que l'attaque de Wiener, de la décomposition en Facteurs premiers et d'autres ont tous échoués.

2.1 Première démarche sans succès

Pour déchiffrer le message de la façon initialement envisagée par l'équipe, nous aurions eu besoin de la clé privée 'd' (inverse multiplicatif de 'e' modulo $\varphi(N)$), où $\varphi(N)$ est la fonction d'Euler de N. Pour calculer $\varphi(N)$, il aurait fallu factoriser N en ses facteurs premiers p et q :

$$N = p \times q$$

Étant donné la taille de 'N', il serait trop long de le factoriser dans un temps raisonnable. Si e divisait n, on pourrait déduire :

$$p = e = 173$$

$$q = \frac{N}{e}$$

'q' nous permettrait de trouver $\varphi(N)$:

$$\varphi(N) = (p-1) \times (q-1)$$

puis, de calculer la clé privée 'd' avec l'inverse multiplicatif de 'e' modulo $\varphi(N)$. Il serait maintenant possible de déchiffrer 'M' :

$$M = C^d \; mod N$$

et de le convertir en message clair avec int_to_str(n) comme nous l'avons fait avec la question précédente.

2.2 Seconde démarche

Malheureusement, 'e' ne divise pas 'N', ce qui implique qu'on doive factoriser (ce que nous allons éviter de faire, car il y a une méthode plus simple). Sachant que le message clair est le nom d'un personnage célèbre, nous avons eu l'idée d'opter pour la technique de l'attaque par force brute. L'objectif est de tester une liste de noms d'auteurs célèbres pour trouver lequel correspond au message chiffré donné. Cette technique fonctionne lorsque le message clair est prévisible et appartient à un ensemble restreint comme c'est le cas ici.

La technique est simple. D'abord, on met sur pied une banque de données de choix possibles. Ensuite, suffit de passer au travers de la liste avec une boucle pour transformer, puis comparer chaque nom de la liste avec le cryptogramme donné.

Les choix possibles sont convertis en entier avec la fonction **str_to_int** donnée dans le devoir qui encode le texte en concaténant les codes binaires UTF-8 de chaque caractère.

On trouve ensuite le cryptogramme du choix qui nous intéresse :

$$C = M^e \; mod N$$

Finalement, reste à comparer le cryptogramme trouvé avec le cryptogramme du message chiffré.

3 Question 2

Le programme doit être capable de retrouver le message clair sans connaître la clé de substitution utilisée pour le chiffrement. Pour ce faire, nous allons utiliser la fréquence d'apparition des lettres.

3.1 Première démarche non concluente

Comme première stratégie, nous avons décidé d'utiliser l'analyse fréquentielle. L'idée principale était d'associer les octets les plus fréquents du message chiffré aux caractères les plus fréquents du texte en français. Notre hypothèse est que les caractères les plus fréquents dans le texte clair correspondront aux octets les plus fréquents dans le message chiffré.

Pour commencer, on s'assure que le texte est en UTF-8 et que toutes les lettres sont en minuscules pour éviter de compter 'A' et 'a' comme des caractères différents. On passe ensuite au travers du texte en comptant le nombre d'occurrences de chaque lettre.

On divise ensuite la chaîne binaire en segments de 8 bits (un octet) et compte les occurrences de chaque octet dans le message chiffré. On peut ensuite substituer les octets du message chiffré et les caractères du texte de référence en se basant sur leurs fréquences respectives.

Cette façon de faire suppose que le chiffrement est une simple substitution monoalphabétique des octets, ce qui n'est pas exactement le cas. Ce programme ne fonctionne donc pas pour des chiffrements plus sophistiqués. L'analyse pourrait être améliorée en prenant en compte les fréquences de paires, mais voyant que le résultat n'était pas concluant, nous avons décidé de tout changer.

3.2 Seconde démarche plus prometteuse

Déçu par notre première tentative, nous avons décidé d'utiliser les fréquences standard des lettres, des bigrammes et des trigrammes. Pour commencer, on analyse les fréquences des caractères dans le message chiffré pour obtenir une estimation des fréquences des lettres chiffrées. Ensuite, on utilise les fréquences standard pour créer un mapping initial entre les lettres chiffrées et les lettres dans le message clair. L'algorithme du recuit simulé utilisé maximise le score et permet d'obtenir le meilleur déchiffrement possible en échangeant aléatoirement des lettres dans le mapping. Le score total est aussi influencé par le nombre de mots français valides dans le texte déchiffré.

La fonction score_mapping calcule un score pour un mapping donné en se basant sur la fréquence des bigrammes et trigrammes dans le texte déchiffré. Les n-grammes connus augmentent le score, tandis que les n-grammes inconnus le diminuent.

Malheureusement, même avec toutes ces modifications le programme ne fonctionne pas parfaitement. Au stade actuel, nous obtenons une suite de lettres qui ne forment pas de vrais mots. L'idée d'intégrer un dictionnaire de mots français pour améliorer la fonction de score était bonne, mais nous ne savons plus quoi changer pour arriver au résultat souhaité.

4 Bibliographie

Question 1.1 :
Factorisation de N en p et q
Décodeur en ligne

Question 1.2 :
Liste de 50 auteurs célèbres

Question 2 :
Liste de mots français
Méthode de substitution en Python
Algorithme de recuit-simulé
Analyse des fréquences d'un texte