Introduction à la sécurité informatique
IFT3725 - IFT6271- Automne 2024 - Devoir 1

Par étudiants:
Yongkang He 20220607
Wanting Teng 20179470

Table des matières

Introduction à la sécurité informatique
IFT3725 - IFT6271- Automne 2024 - Devoir 1

Question 1

Clé publique Question 1.1

Idée :
- Lorsque e=3, pour récupérer M à partir de C, on peut calculer la racine cubique entière de C. Ainsi, l'inverse de la racine cubique permettra de calculer M
- Une fois que l'on trouve M, on le convertira en chaine de caractères

Bref, nous avons introduit 1 fonction principale et 2 fonctions auxiliaires :

1. Décrypter un message chiffré (decrypt_msg)
    1. Calculer la racine cubique entière de C (integer_nth_root)
    2. Convertir un entier x en une chaîne de caractères (int_to_str)

Réponse : Umberto Eco

```
# Appel de la fonction
decrypt_msg(C, e)
✓ [174] 14ms
 M: 10327524770482866052172273 5
 message : Umberto Eco
```

Introduction à la sécurité informatique
IFT3725 - IFT6271- Automne 2024 - Devoir 1

Clé publique Question 1.2

Idée :

- Lorsque e=173, cette valeur étant grande, on ne peut plus passer par la fonction précédemment
- Nous allons donc utiliser **d**, qui est la clé privée. Pour cela, nous devons d'abord trouver l'indice d'Euler.
- Pour trouver l'indice d'Euler, il nous faudra les valeurs de **p** et **q**.

* Calculer **p** et **q** prend un temps considérable sans matériel spécifique, car les grands nombres sont difficiles à factoriser.

Résultat p et q à partir de https://www.alpertron.com.ar/ECM.HTM est:

172 219604 291138 178634 924980 176652 297603 347655 313304 280071 646410 523864 939208 855547 078498 922947 475940 487766 894695 848119 416017 067844 129458 299713 889703 424997 977808 694983 717968 420001 033168 722360 067307 143390 485095 229367 172423 195469 582545 920975 539060 699530 956357 494837 243598 213416 944408 434967 474317 474605 697904 676813 343577 310719 430442 085422 937057 220239 881971 046349 315235 043163 226355 302567 726074 269720 408051 461805 113819 456513 196492 192727 498270 702594 217800 502904 761235 711809 203123 842506 621973 488494 670663 483187 137290 546241 477681 096402 483981 619592 515049 062514 180404 818608 764516 997842 633077 157249 806627 735448 350463 (603 digits) = 10 715086 071862 673209 484250 490600 018105 614048 117055 336074 437503 883703 510511 249361 224931 983788 156958 581275 946729 175531 468251 871452 856923 140435 984577 574698 574803 934567 774824 230985 421074 605062 371141 877954 182153 046474 983581 941267 398767 559165 543946 077062 914571 196477 686542 167660 429831 652624 386837 205668 069673 (302 digits) × 16 072629 107794 009814 226375 735900 027158 421072 175583 004111 656255 825555 265766 874041 837397 975682 235437 871913 920093 763297 202377 807179 285384 710653 976866 362047 862205 901851 662236 346478 131611 907593 556712 816931 273229 569712 475372 911901 098151 338748 315919 115594 371856 794716 529813 251490 644747 478936 580257 043048 672231 (302 digits)

Bref, nous avons introduit 1 fonction:

- Fonction de déchiffrement RSA (rsa_decrypt)
  - Vérification si ce sont les bonnes p et q (ils sont unique)
  - Calculer phi_N
  - Calculer l'inverse modulaire de e pour obtenir d (modinv) *fct fourni*
  - Calculer de l'exponentiation modulaire (modular_pow) *fct fourni*
  - Convertir un entier x en une chaîne de caractères

Réponse: Marcel Proust

```
# Appel de la fonction
try:
    message = rsa_decrypt(C, e, p, q)
    print("Message converti en texte:", message)
except Exception as ex:
    print("Erreur lors du déchiffrement:", ex)
✓ [180] 35ms
```

```
Message converti en texte: Marcel Proust
```

Introduction à la sécurité informatique
IFT3725 - IFT6271- Automne 2024 - Devoir 1

Question 2

Elle est entièrement sur github

https://github.com/IFT3275-Securite-Informatique/devoir-1-cryptographie-yongkang-wanting

Bibliographie

- https://www.alpertron.com.ar/ECM.HTM (site pour trouver p et q)
- https://fr.wikipedia.org/wiki/Factorisation_de_Lenstra_par_les_courbes_elliptiques