

1 Descriptif

L'objectif de ce défi est d'implémenter une méthode permettant de chiffrer un message entier en utilisant la clé publique et l'algorithme de RSA. On rappelle que pour chiffrer un long message, il suffit de découper ce message en morceaux et de chiffrer chacun des morceaux indépendamment.

Dans le test, les messages à chiffrer feront exactement 252 bits.

2 Protocole

1. Une fois la connexion établie, le serveur commence par envoyer un premier message annonçant le début du défi :

– Debut du defi : Chiffrer –

Ce message n'attend pas de réponse.

2. Le serveur envoie ensuite une série de triplets (M, N, e) où M est un mot binaire de 252 bits et N et e des nombres binaires.
3. Pour chaque triplet (M, N, e) , le serveur doit recevoir en retour M chiffré avec la clé (N, e) (sous forme binaire).
4. Après chaque réponse, le serveur enverra un message commençant par "OK" ou "NOK" suivant si la réponse est correcte ou non.
5. A la fin du défi, le serveur enverra un message indiquant "Defi valide" ou "Defi echoue!". Aucune réponse n'est attendue.
6. Le serveur terminera la communication par le message "FIN", votre client devra alors fermer la socket. Aucune réponse n'est attendue.

3 Exemple de communication

Voici un exemple (incomplet) d'une communication pour ce défi. Dans cet exemple les "<" et ">" indiquent le sens de transfert de chaque message et ne doivent pas être présents dans la communication.

```

< -- Debut du defi : Chiffrer --
< 00101001001011111111001100000001000000110100001011011000011011...
< 10000001010100101001001011011110110100000010101110000111001001...
< 00010010101000011011010000000010110110000011110000010111101000...
> 01100000110011001101111100000000101011000000011011010011001101...
< OK
< 0110110110101111111111001001101010101100011100100110000011100...
< 1101001101010111010001001101000100010001010011100011001011101...
< 1000110011110010010000000100011011000101111011001100011010010...
> 0101101101101001001000010000001011111010001010010110011100101...
< OK
< 01010101001100010110101000111001111011011100000011000001011100...
< 111000111010000101001001111111111000100111010000110000101101...
< 100100100100101111110010000000010111010000011111001110111010...
> 100110000101110111011001110011010010110011100101111000000101...
< OK

```