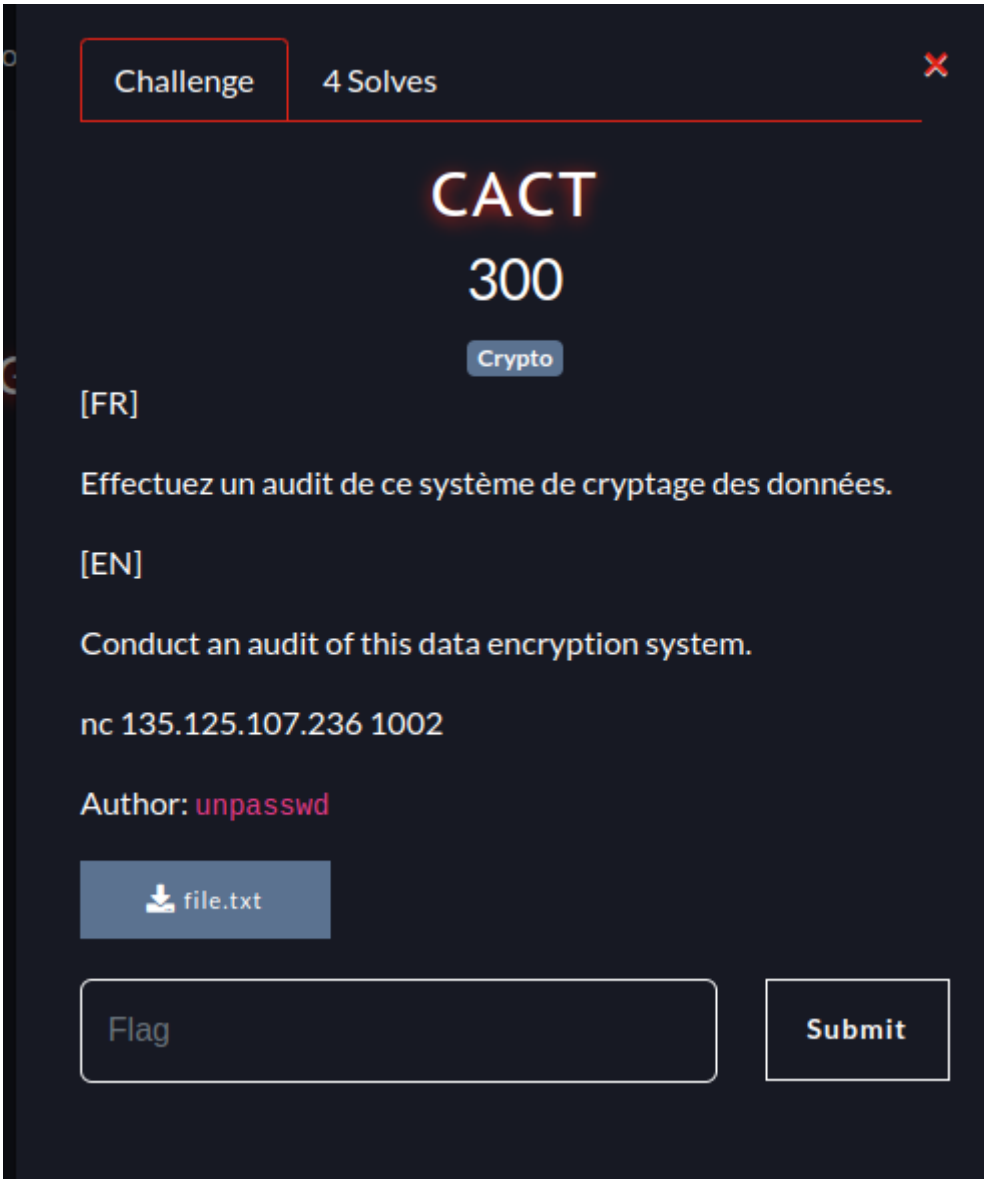
Writeup du challenge CACT de la phase de qualification pour le Hackerlab2024

Nom d’utilisateur : takeoff
E-mail : robertohoungbo@gmail.com
Affiliation : IFRI
Pays : Bénin



Le but du challenge est d’effectuer un audit sur le système de cryptage de données.

Le contexte du challenge ressemblait beaucoup à un challenge que j'avais fait auparavant. Dans l'ensemble nous avons un fichier qui contient la clé publique (n et e) puis le ciphertext, et en plus nous pouvons soumettre des messages chiffrés à une instance netcat qui se charge de les déchiffrés excepté le ciphertext donné.

Pour résoudre le challenge j'ai d'abord chifrré le message clair « 2 » avec la clé publique puis j'ai essayé de le décoder avec l'instance et cela a fonctionné.

```
Python 3.12.3 (main, Apr 10 2024, 05:33:47) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> n = 46136304259497491075027547433989825755344684837680418819600728221218462382962397187618591610752848585750694181667790597428002311390565813868762231574774 23
>>>
>>> e = 65537
>>>
>>> print(pow(2, e, n))
934039981946309233440318424088145186324354767900820320569449421750260940333311210515314308001383322917135891826569635754800287855670426395034265443121613
>>>
```

```
(my_env) roberto@roberto-HP-Pavilion-x360-Convertible:~/RsaCtfTool$ nc 135.125.107.236 1002

Déchiffrement...
 Je ne reçois que les types long et je déchiffre tous les messages chiffrés, à l'exception de celui donné dans le contexte.

 Message chiffré : 934039981946309233440318424088145186324354767900820320569449421750260940333311210515314308001383322917135891826569657548002878557840263950342654431216 13

 Message clair :  2
```

Ensuite j'ai stocké la valeur du message chiffré de « 2 » dans une variable f que j'ai multiplié au ciphertext c le tout congru à n.

```
>>> f = 934039981946309233440318424088145186324354767900820320569449421750260940333311210515314308001383322917135891826569635754800287855670426395034265443121613
>>>
>>> c = 158588007140076018598521363818402521591719452298312848482650484126791952360340411799030157315854406989675003448617556011397817522262957833745512396343217 3
>>>
>>> d = pow(c * f, 1, n)
>>> print(d)
404610516426945436097102253300576096309335372594641465439747112679511833635418153405707201147612653113260882635327405679570583440108506450157094649879132 8
```

J'ai ensuite envoyé le message chiffré obtenu au serveur qui me l'a décodé.

```
Message chiffré : 40461051642694543609710225330057609630933537259464146543974711267951183363541815340570720114761265311326088263532740
567957058344010850645015709464987913328

Message clair :  308854690474584056852956624639382604403419007051347091782379358448881888290439044522178093135850234

Message chiffré : ^C
```

J’ai pris le message décodé par le serveur qui en réalité est le ciphertext déchiffré multiplié par mon message chiffré (2 en clair si vous vous en rappelez). Il me suffit donc là de prendre le résultat renvoyé par le serveur et de le diviser par 2 puis de faire un long_to_bytes sur ça et j’aurai mon flag.

```
>>> v = 308854690474584056852956624639382604403419007051347091782379358448881888290439044522178093135850234
>>> v = v//2
>>> v
154427345237292028426478312319691302201709503525673545891189679224440944145219522610890465679251 17
>>>
roberto@roberto-HP-Pavilion-x360-Convertible:~/RsaCtfTool$ source ~/my_env/bin/activate
(my_env) roberto@roberto-HP-Pavilion-x360-Convertible:~/RsaCtfTool$ python3
Python 3.12.3 (main, Apr 10 2024, 05:33:47) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> from Crypto.Util.number import *
>>> long_to_bytes(154427345237292028426478312319691302201709503525673545891189679224440944145219522610890465679251 17)
b'HLB2024{CCTA_Congratulation_h4ck3r_81955}'
>>>
```

FLAG : HLB2024{CCTA_Congratulation_h4ck3r_81955}

Merci !