Cryptographie - KeyExchange - 75 points

Kévin DUVERGER

Table des matières

1 Résolution KeyExchange :

Avant de commencer ce challenge, quelques rappels sur Diffie-Hellman. L'idée de cet échange de clé est qu'Alice va prendre un a aléatoire puis elle va envoyer $g^a \pmod p$ à Bob. Bob va ensuite prendre un b aléatoire, envoyer son $g^b \pmod p$. Finalement, les 2 parties pourront chacune calculer la clé $K = H(g^{ab})$.

Commençons donc par nous connecter au serveur, voici ce qu'il nous envoie :

```
Welcome to KeyExchange !
In this session, we will use the following parameters :
    p = ...
    g = ...

Here is my public share : ...
Now give my yours       :
```

Grâce à cette première partie, nous pouvons récupérer $g, p, g^a \pmod p$!

La seconde partie consiste à envoyer une share publique au serveur :

```
b = random.randint(2, p - 2)
B = pow(g, b, p)
```

Une fois cela fait, nous pouvons voir ce que le serveur nous envoie :

```
We can now compute K with K = H(g^ab.to_bytes(1024, big)), H being SHA2-256"
And use this key to encrypt / decrypt with AES-ECB-256
Here is the encrypted flag : ... !
```

Grâce à cela, nous avons les instructions pour continuer et récupérer le flag !

Voici donc le code qui nous permet de calculer la clé et déchiffrer le message :

```
s = pow(A, b, p)
K = hashlib.sha256(s.to_bytes(1024, byteorder="big")).digest()
print("The flag is : " + str(AES.new(K, AES.MODE_ECB).decrypt(bytes.fromhex(encryptedFlag))))
```

Ce qui nous permet de conclure !

Pour finir, le code final de la solution est le suivant :

```
import random
import hashlib
import os, sys, socket
from Crypto.Cipher import AES

server = "146.59.227.136"
port = 23402

ma_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

try:
    ma_socket.connect((server, port))
except Exception as e:
    print("Problème de connexion", e.args)
    sys.exit(1)

# Getting the data
ligne = ma_socket.recv(4096).decode("utf-8")
data = ligne.split("\n")
p = int(data[2][5:])
g = int(data[3][5:])
A = int(data[5][26:])

# Computing our values
b = random.randint(2, p - 2)
B = pow(g, b, p)
s = pow(A, b, p)
ma_socket.sendall((str(B) + "\n").encode("utf-8"))

# Last part
ligne = ma_socket.recv(4096).decode("utf-8")
encryptedFlag = ligne.split("\n")[3][29:]
encryptedFlag = encryptedFlag[:-2]
K = hashlib.sha256(s.to_bytes(1024, byteorder="big")).digest()
print("The flag is : " + str(AES.new(K, AES.MODE_ECB).decrypt(bytes.fromhex(encryptedFlag))))

ma_socket.close()
```