Week 5 - 21/10/2022

Exercício 1

Como a cifragem por stream cipher consiste num XOR do plaintext com a stream cipher. Pelas propriedades da operação XOR sabemos que se aplicarmos duas vezes a função XOR com o mesmo valor, a função não se altera. Assim sabemos que se:

```
C1 = M1 XOR S e C2 = M2 XOR S,
sendo C1 a cifra de M1 e C2 a cifra de M2,
aplicando a stream cipher S.
```

então:

```
C1 XOR C2 = (M1 XOR S) XOR (M2 XOR S) = M1 XOR M2
```

Os pacotes contém mensagens em ASCII cifrados e cada caracter em ASCII tem sempre o valor menor que 128, ou seja o primeiro bit é sempre 0, sabemos ainda que 0 XOR 0 é sempre 0, logo para saber se as cifras não usaram a mesma stream cipher basta verificar que há caracteres com valores maiores ou iguais a 128.

Assim criamos o seguinte código para resolver:

```
from data import *
def solve(a,b):
    for x in zip(a,b):
        if int(x[0] ^^ x[1])>= 128:
            return false

def find():
    for i in range(0, len(packets)):
        for j in range(i+1, len(packets)):
            if(solve(packets[i], packets[j])):
                print (i, 'e', j)
    return
print('Posição dos pacotes no array do ficheiro data.py que usaram a mesı
find()
```

Exercício 2

a)

Como c = RC4(v||k) XOR m, sendo || a concatenação, e o Bob recebe (v||c), o Bob pode recalcular m fazendo:

```
m = RC4(v||k) XOR c
```

já que o XOR tem a seguinte propriedade:

```
A = B XOR C <=> C = B XOR A
```

b)

Basta verificar o mesmo valor v, ou seja os mesmo primeiros 80 bits, quando estes forem iguais as key stream usadas são iguais, pois o RC4 é determinístico e é usada sempre a mesma chave k de 128 bits.

c)

Como k é sempre igual, o que determina a unicidade da key stream é o v de 80 bits. O número de mensagens é 2^(80/2)=2^40, pois apartir da mensagem 2^40 a probabilidade de dois valores v random se repetirem é maior que 50%, por causa do paradoxo do aniversário.