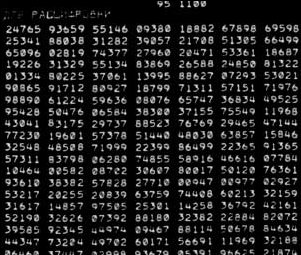


**Criptografia**

**TP2**



João Miguel da Silva Alves (83624)

Paulo Jorge Alves (84480)

**MESTRADO INTEGRADO EM ENGENHARIA BIOMÉDICA**

**INFORMÁTICA MÉDICA 2020/2021**

O *one-time-pad* usa uma chave aleatória tão grande quanto a mensagem, de modo a que a chave não precise de ser repetida. Além disso, a chave é usada para cifrar e decifrar uma única mensagem, sendo por fim descartada. Cada nova mensagem exige uma nova chave com o mesmo tamanho. Esta cifra é inquebrável. Produz uma saída aleatória que não possui qualquer relacionamento estatístico com o texto claro. Como o criptograma não contém informação sobre o texto limpo, simplesmente não existe um meio de quebrar o código.

No entanto, neste exercício, sabe-se que dois criptogramas resultaram da encriptação com a mesma chave. A regra geral diz que nunca se deve usar a mesma chave mais de uma vez. Caso contrário, a cifra começa a ficar vulnerável. Este é um exemplo deste caso.

Posto isto, considerando C1 um criptograma e C2 outro criptograma, e sendo m1 e m2 as duas mensagens que foram cifradas com a mesma chave k, sabe-se que:

Assim, de modo a eliminar k:

Como C1 e C2 foram encriptados com a mesma chave, a subtração (módulo 26) dos dois criptogramas vai mostrar um padrão de letras “A” (correspondente ao número 0).

Para descobrir os dois criptogramas que foram encriptados com a mesma chave usou-se o código Python abaixo:



Figura 1 - Código Python

Assim, para resolver o problema em causa, a ideia foi fazer todas as subtrações (módulo 26) possíveis entre os 20 criptogramas, e por fim fazer a contagem das ocorrências da letra “A” em cada um. O texto com maior quantidade de “A”, vai permitir saber quais os criptogramas resultaram da encriptação com a mesma chave.

Começou-se por colocar todos os criptogramas numa lista *text.* Percorrendo essa lista 2 vezes (de forma a operação a realizar seja efetuada entre todos os elementos da lista, ou seja, entre todos os criptogramas), e transformando também essas strings em listas (linhas 8 e 9 do código da figura 1), realiza-se a subtração (módulo 26) entre estas 2 listas.

Para cada uma dessas listas, faz-se a subtração (módulo 26) elemento a elemento, como por exemplo, querendo subtrair (em modulo 26) o “A” pelo “C”:

Desta forma, usando as funções Python ord() e chr() para o efeito, obtém-se uma lista com todas as subtrações (módulo 26) possíveis.

De seguida, percorrem-se todos os elementos desta lista (sublistas), e registam-se as ocorrências da letra “A” através de um dicionário. No fim é impressa uma lista com os dicionários, em que cada dicionário corresponde às ocorrências de “A” em cada texto, e todos os textos são escritos num ficheiro resultado.txt.

Após analisar todos os dicionários, verifica-se que a letra A aparece 371 vezes (valor máximo de ocorrências em todos os textos) num dos textos (figura 2), sendo esse o texto o corresponde à subtração do criptograma 6 (“LXBVRMY…”) com o 14 (“HCOGIIGQ…”).



Figura 2 - Texto, onde se verificou o maior número de “A”, resultante da subtração do criptograma 6 com o 14.

Assim, chega-se à conclusão de que os dois criptogramas que foram encriptados pela mesma chave correspondem ao criptograma 6 e ao criptograma 14.

Desta forma provou-se que a cifra *one-time-pad* só é inquebrável se a chave for usada uma única vez e por fim descartada. Caso contrário, o adversário consegue ter pistas credíveis sobre a mensagem original, que podem levar à sua decifragem.