Exercício 2

Para realizar este exercício foi necessário descompilar a aplicação Lab2-2.apk utilizando o apktool. Foi feito uma script bat para agilizar este processo, que se encontra com o nome de decode_apk.bat .

Após isto fomos ao ficheiro .\Lab2-2\smali\sg\vantagepoint\uncrackable1\MainActivity, para analisarmos o método verify(View v), neste método apercebemo-nos que era chamado um método da classe a.smali, e neste método estavam definidos a chave e a frase a codificar, bem como a parte do algoritmo de causar algum ruído nessas variáveis. Abaixo estão essas duas variáveis no ficheiro.

```
.locals 5

const-string v0, "8d127684cbc37c17616d806cf50473cc" -- chave p cifrar em base64

const-string v1, "5UJiFctbmgbDoLXmpL12mkno8HT4Lv8dlat8FxR2GOc=" -- frase cifrada
```

Depois disso, apercebemo-nos que era chamada uma codificação da chave para hexadecimal nesse mesmo ficheiro, mas após disso, eram em conjunto chamados a chave codificada, e a frase encriptada por um método a(). Este método já se encontrava noutro diretório ..\Lab2-2\smali\sg\vantagepoint\a , e aqui foi onde encontrámos os algoritmos utilizados para codificar a junção da chave e da frase, que estão apresentados na fotografia abaixo.

```
90    new-instance v0, Ljavax/crypto/spec/SecretKeySpec;
01
12    const-string v1, "AES/ECB/PKCS7Padding"
23
34    invoke-direct {v0, p0, v1}, Ljavax/crypto/spec/SecretKeySpec;-><init>([BLjava/lang/String;)V
45
56    const-string p0, "AES"
67
78    invoke-static {p0}, Ljavax/crypto/Cipher;->getInstance(Ljava/lang/String;)Ljavax/crypto/Cipher;
8
```

Após isto, seguimos a forma como as implementações estavam concretizadas em Smalli, e tentámos igualar a ordem, tendo em conta que tínhamos os algoritmos utilizados, bem como a ordem dos mesmos.