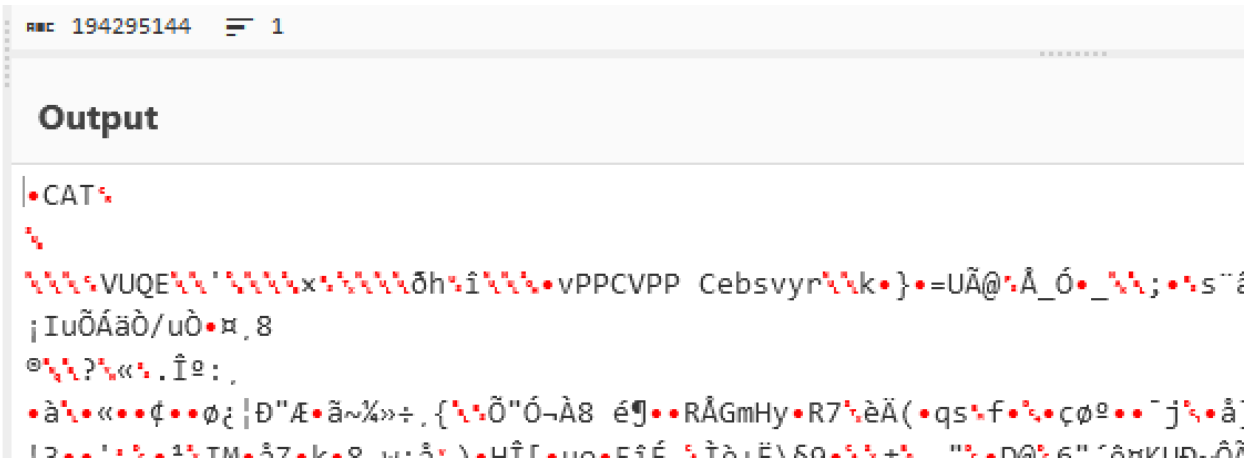
Info:

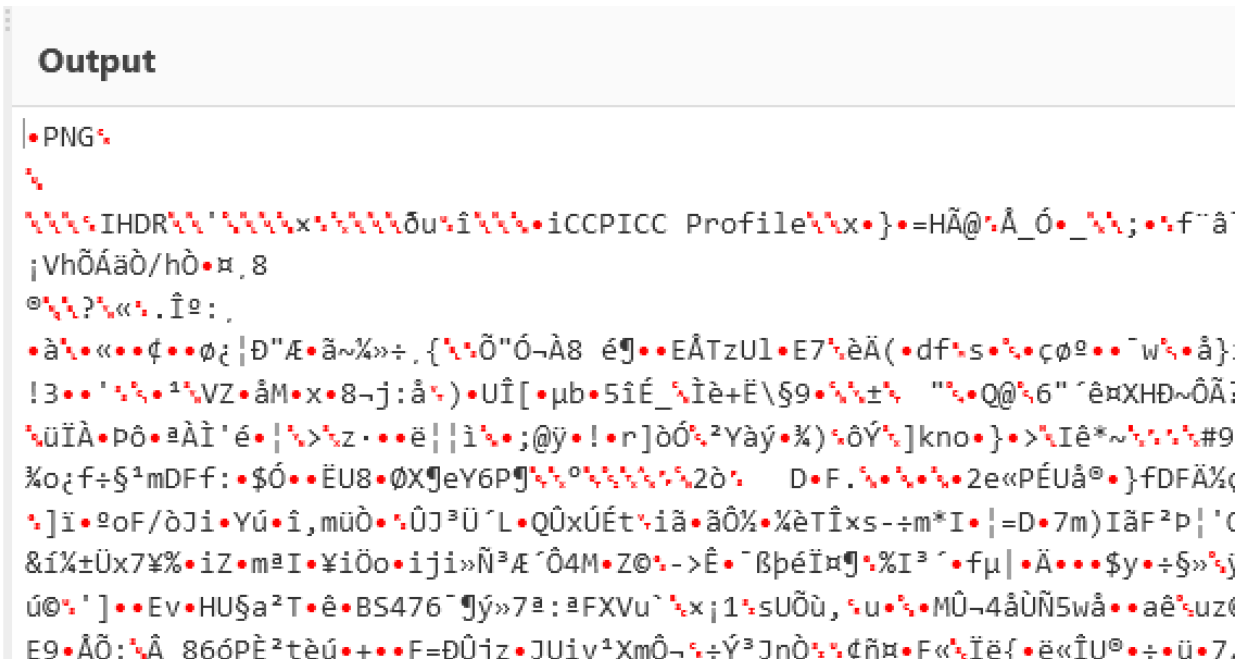
So many layers....

---

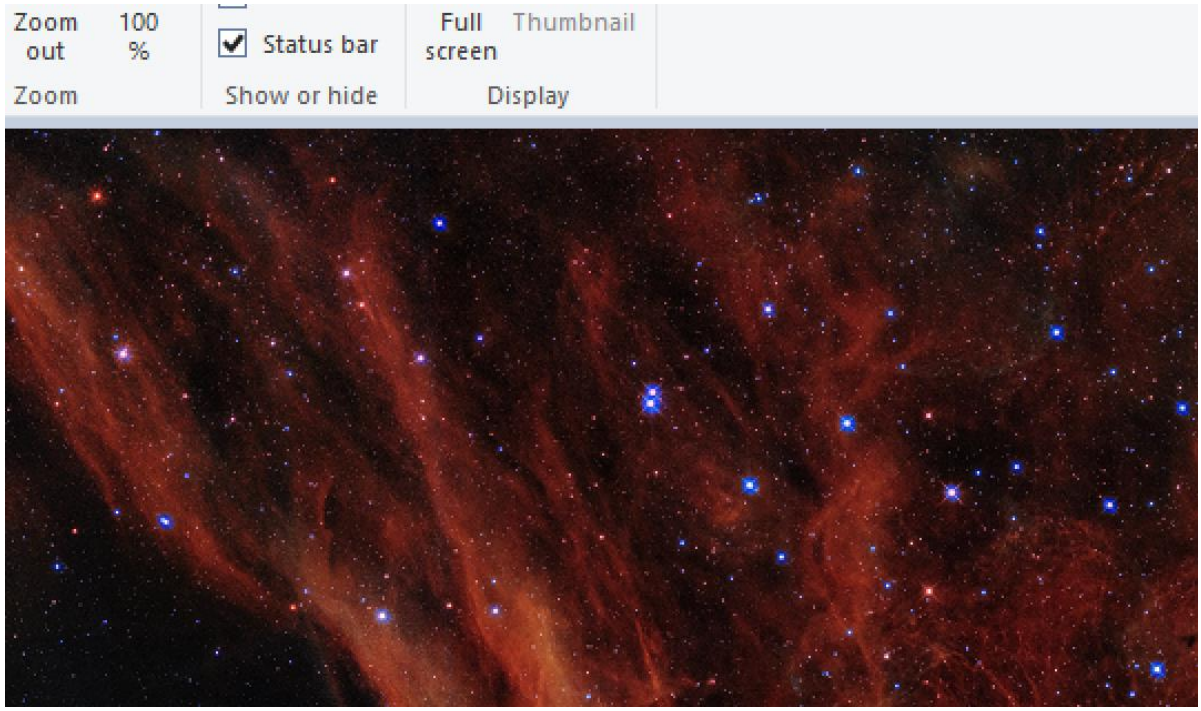
1. Checking the file with DiE shows that is simple plaintext
2. We can use tools like cyberchef https://gchq.github.io/CyberChef to read the file content (notepad is struggling with the size)
3. Looking at the text inside it seems to be simpler encoding with no weird characters. Without any more info we can try the more common ones like base64. Base64 gives somehow structured output. This could be the right direction. The “CAT” could be some sort of file descriptor



4. Going with the theory that “CAT” is a file descriptor it seems that characters are just shifted. One famous shifting cipher is ROT13 and ROT47. ROT13 gives the following result:



5. This looks promising as this seems to be a .png. To check if we are on the right path we can try downloading it and see what it looks like
6. We downloaded a valid picture

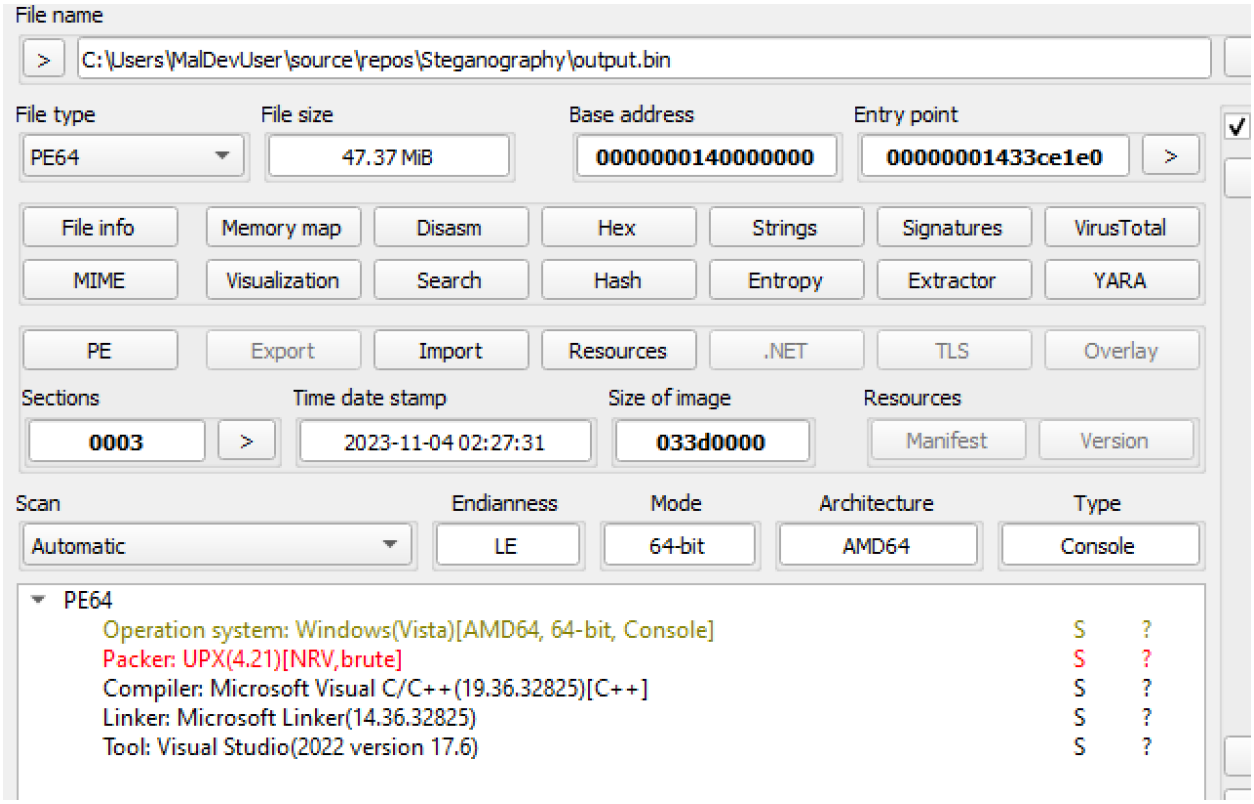


7. As there is no visible flag anywhere the next best guess is that data is hidden in the picture itself Very likely the LSB (least significant bit) has been used to store data without visibly changing the picture. To check that theory we can use tools such as https://github.com/ragibson/Steganography
   Turns out we were right:

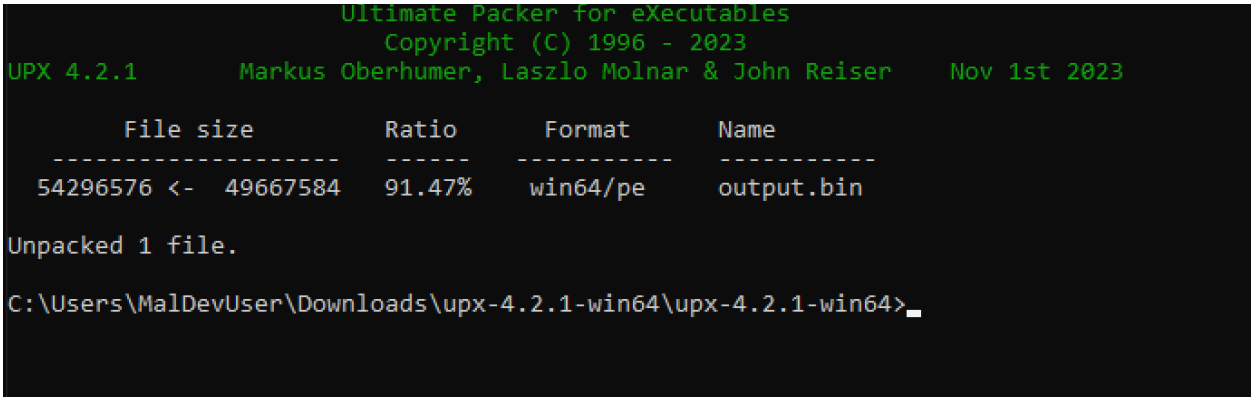
```
C:\Users\MalDevUser\source\repos\Steganography>stegolsb steglsb -r -i download.png -o output.bin
Files read                      in 15.18s
49667584 bytes recovered        in 10.28s
Output file written             in 0.27s

C:\Users\MalDevUser\source\repos\Steganography>
```

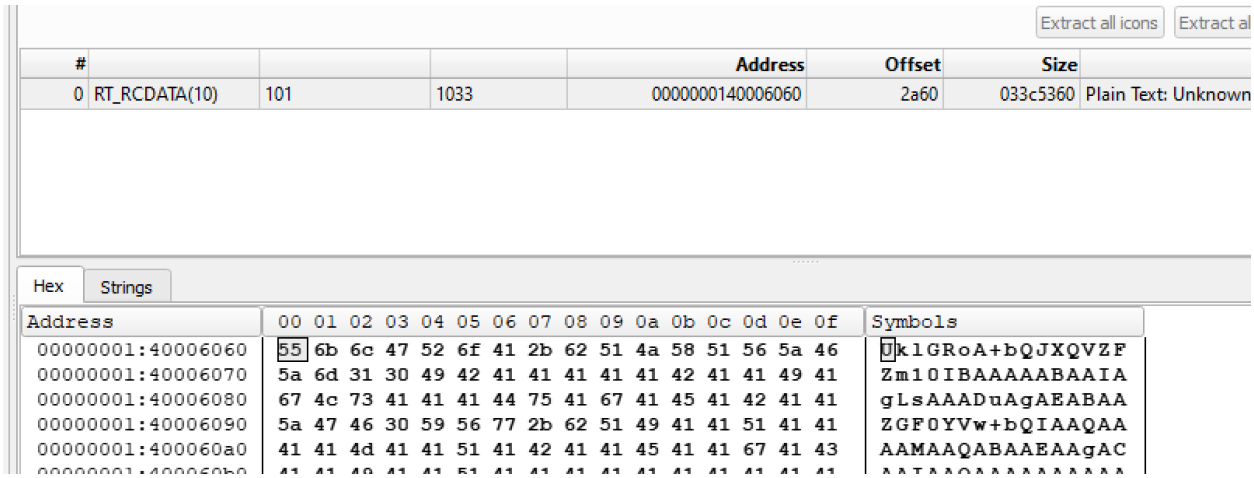
8. The output.bin seems to an UPX-packed executable

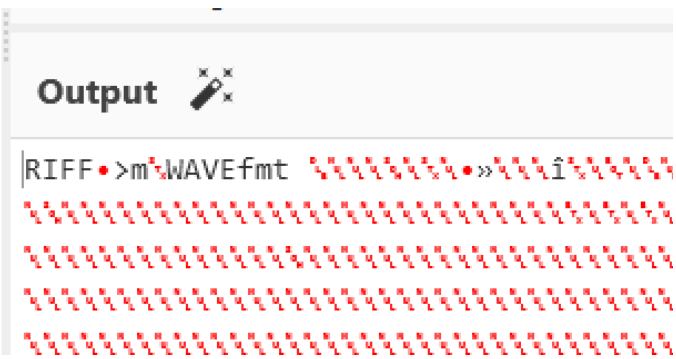


9. To unpack UPX we can use the tool UPX itself and simply unpack it



10. Checking the executable we can see that it contains a very big resource file. First best guess without knowing we check for base64 encryption again and we are right.



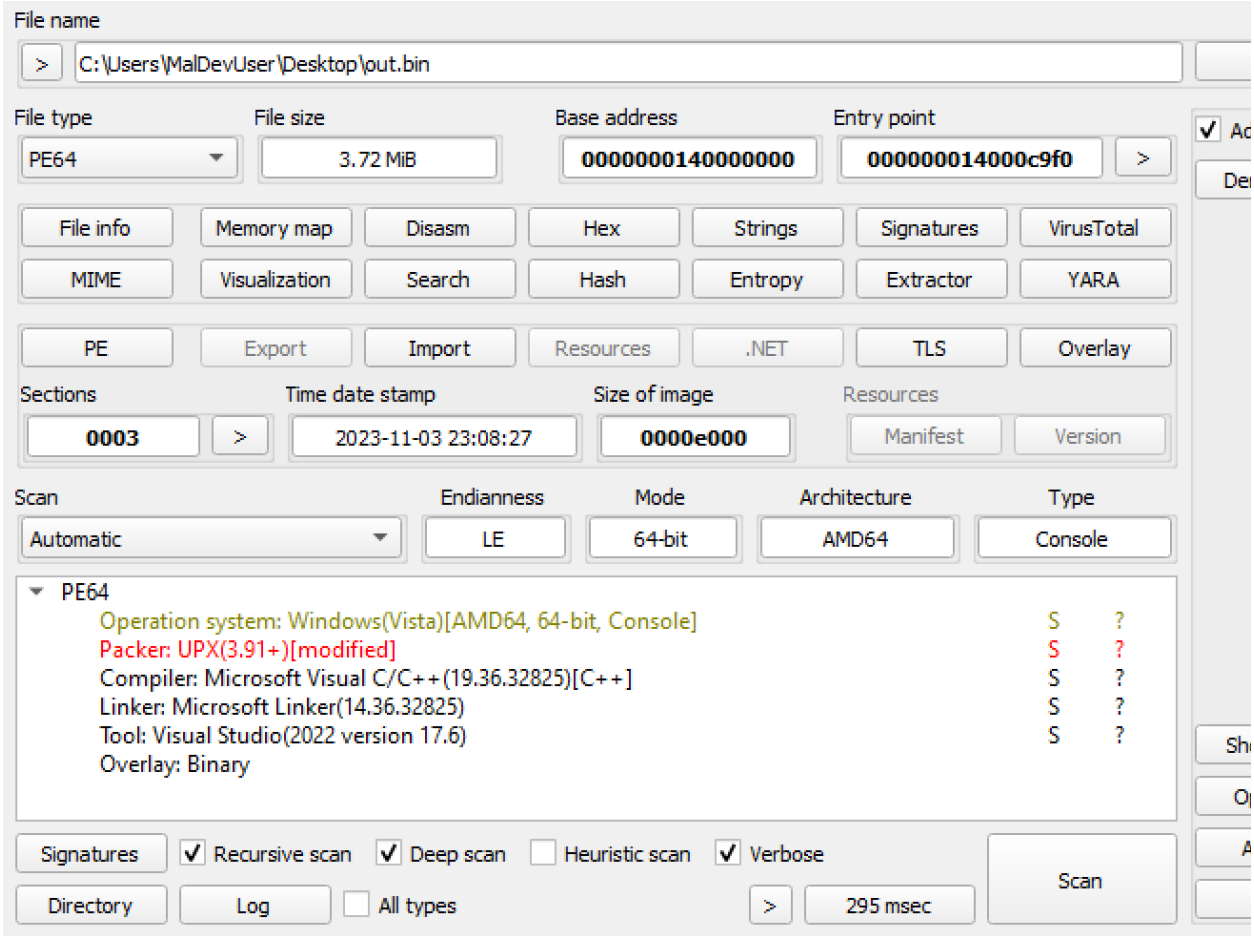


11. This time we get a .wav file. As it turns out its Rick Astley – Never gonna give you up
    As this is obviously not the flag we need to dig deeper. We can use the same tool again we used to recover the hidden data in the .png file but this time target the audio-file. As we do not know how big the hidden data is we try to recover as much as possible

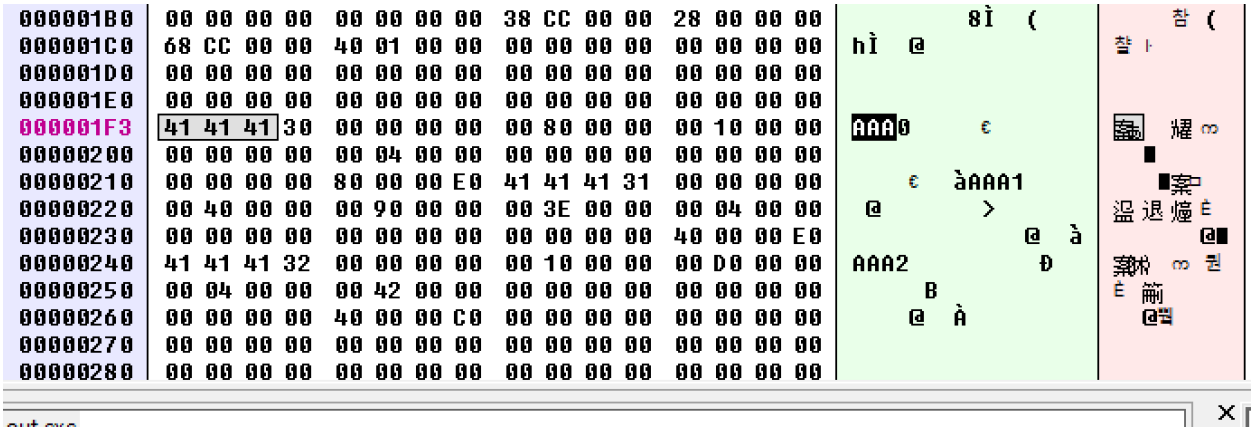
```
C:\Users\MalDevUser\source\repos\Steganography>stegolsb wavsteg -r -i download.wav -o out.bin -b 3900000
Files read                      in 0.02s
Recovered 3900000 bytes         in 0.18s
Written output file             in 0.02s

C:\Users\MalDevUser\source\repos\Steganography>
```

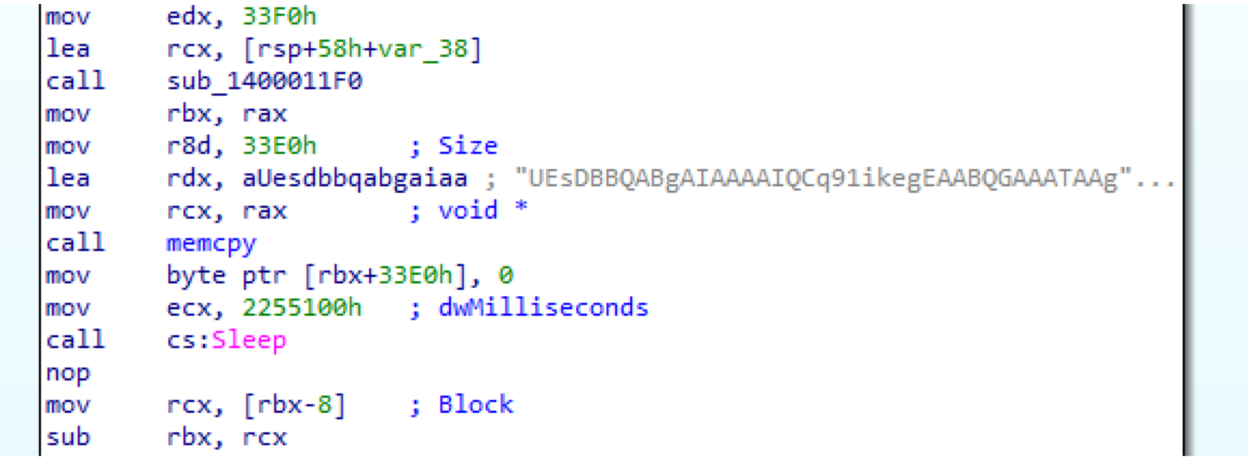
12. The recovered file is another executable and also packed with UPX however this time UPX is unable to recover the original file



13. Checking the file in a hex editor we can clearly see that the strings “UPX” have been replaced with “AAA”. For upx to work we replace them again with UPX.

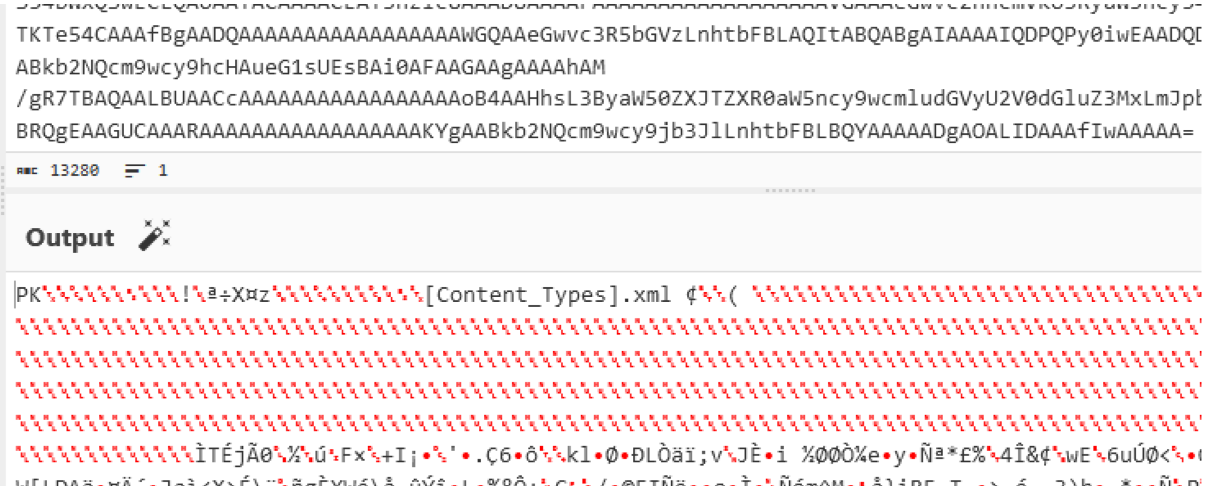


14. Again we can see that the binary is holding a very large base64-string and then sleeps for a very long time. We can decode that base64-string and see what is contains



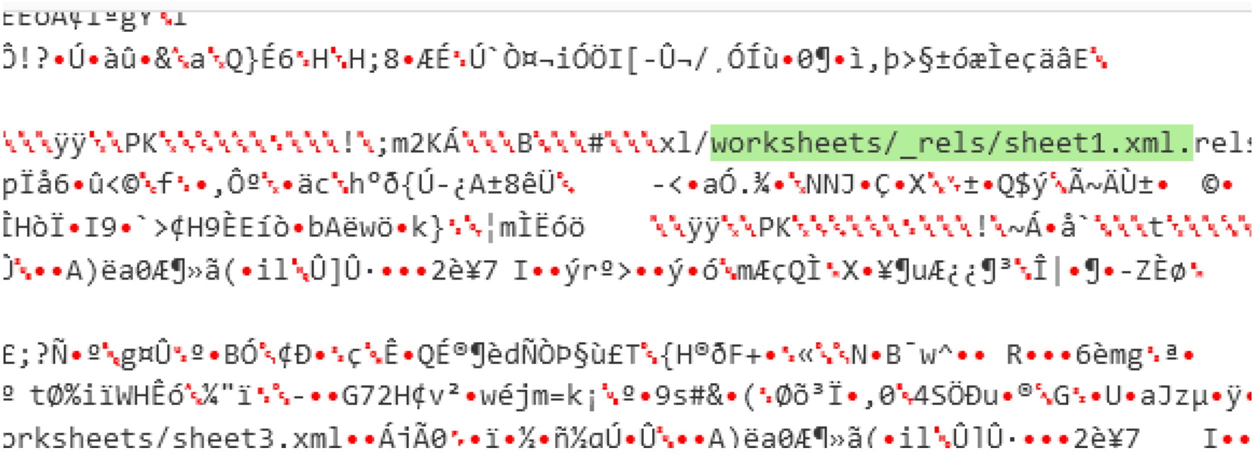
```
mov     edx, 33F0h
lea     rcx, [rsp+58h+var_38]
call    sub_1400011F0
mov     rbx, rax
mov     r8d, 33E0h      ; Size
lea     rdx, aUesdbbqabgaiaa ; "UEsDBBQABgAIAAAAIQCq91ikegEAABQGAAATAAg"...
mov     rcx, rax        ; void *
call    memcpy
mov     byte ptr [rbx+33E0h], 0
mov     ecx, 2255100h   ; dwMilliseconds
call    cs:Sleep
nop
mov     rcx, [rbx-8]    ; Block
sub     rbx, rcx
```

15. It looks like we recovered an excel-file



```
TKTe54CAAAfBgAADQAAAAAAAAAAAAAAAAAAWGQAAeGwvc3R5bGVzLnhtbFBLAQItABQABgAIAAAAIQDPQPy0iwEAADQD
ABkb2NQcm9wcy9hcHAueG1sUEsBAi0AFAAGAAgAAAAhAM
/gR7TBAQAALBUAACcAAAAAAAAAAAAAAAAAAoB4AAHhsL3ByaW50ZXJTZXR0aW5ncy9wcmludGVyU2V0dGluZ3MxLmJpb
BRQgEAAGUCAAARAAAAAAAAAAAAAAAAAAKYgAABkb2NQcm9wcy9jb3JlLnhtbFBLBQYAAAAADgAOALIDAAAfIwAAAAA=
```

```
PK!ª÷X¤z[Content_Types].xml ¢(
ÌTÉjÃ0½ú Fx+I¡'.Ç6ôkl ÐLÒäï;vJÈi %ØØÒ%e yÑª*£%4Î&¢wE6uÚØ<
```

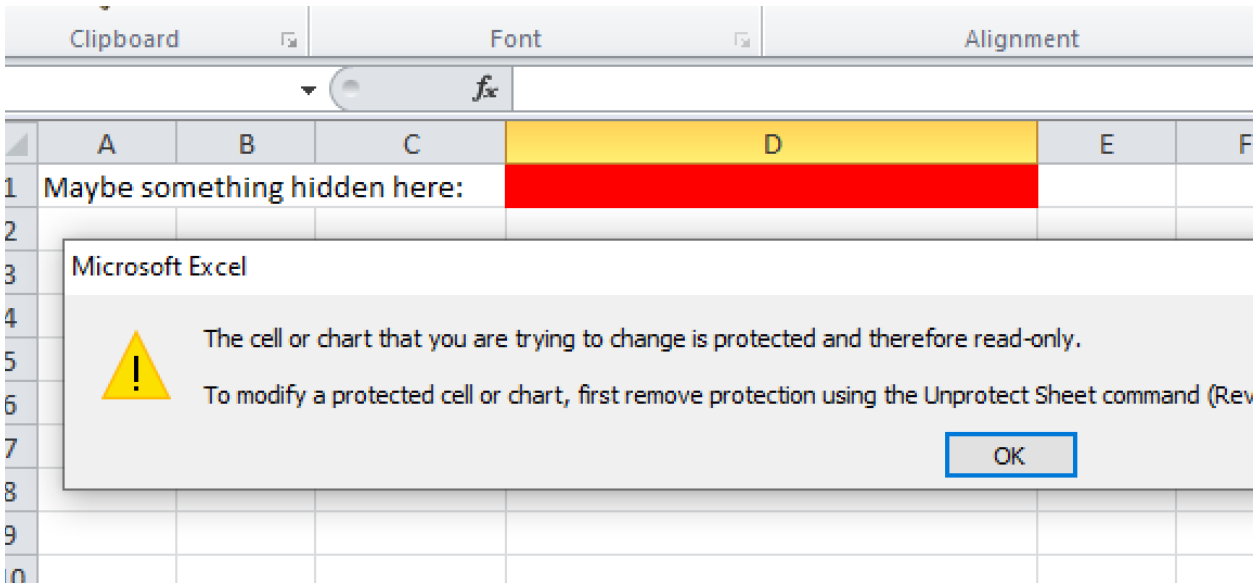


```
Ͻ!? Ú àû &aQ}É6HH;8 ÆÉÚ`Ò¤¬iÓÖI[-Û¬/¸ÓÍù 0¶ ì,þ>§±óæÌeçäâE

ÿÿPK!;m2KÁB#xl/worksheets/_rels/sheet1.xml.rel
pÏå6 û<©f ,Ôº äch°ð{Ú-¿A±8êÜ     -< aÓ.¾ NNJ Ç X± Q$ýÃ~ÄÙ±  ©
ÌHòÏ I9 `>¢H9ÈEíò bAëwö k}¦mÌËóö     ÿÿPK!~Á å`t
Ͻ A)ëa0Æ¶»ã( il Û]Û· 2è¥7 I ýrº> ý ómÆçQÌX ¥¶uÆ¿¿¶³Î| ¶ -ZÈø

E;?Ñ ºg¤Ûº BÓ¢Ð çÊ QÉ©¶èdÑÒÞ§ù£T{H®ðF+ «N B¯w^  R 6èmgª
º tØ%iïWHÊó¼"ï- G72H¢v² wéjm=k¡º 9s#& (Øõ³Ï ,04SÖÐu ®G U aJzµ ÿ
orksheets/sheet3.xml ÁiÃ0 ï ½ ñ½gÚ Û A)ëa0Æ¶»ã( il Û]Û· 2è¥7    I
```

16. However we are unable to read the content as they are pw-protected



17. We can circumvent this by simply deleting the pw from the worksheet. Open up the file with 7zip, navigate to xl\worksheets\sheet1.xml and edit the file. Delete the whole “sheetProtection” part and save the file

```
eetData><sheetProtection password="A7CB" sheet=
```