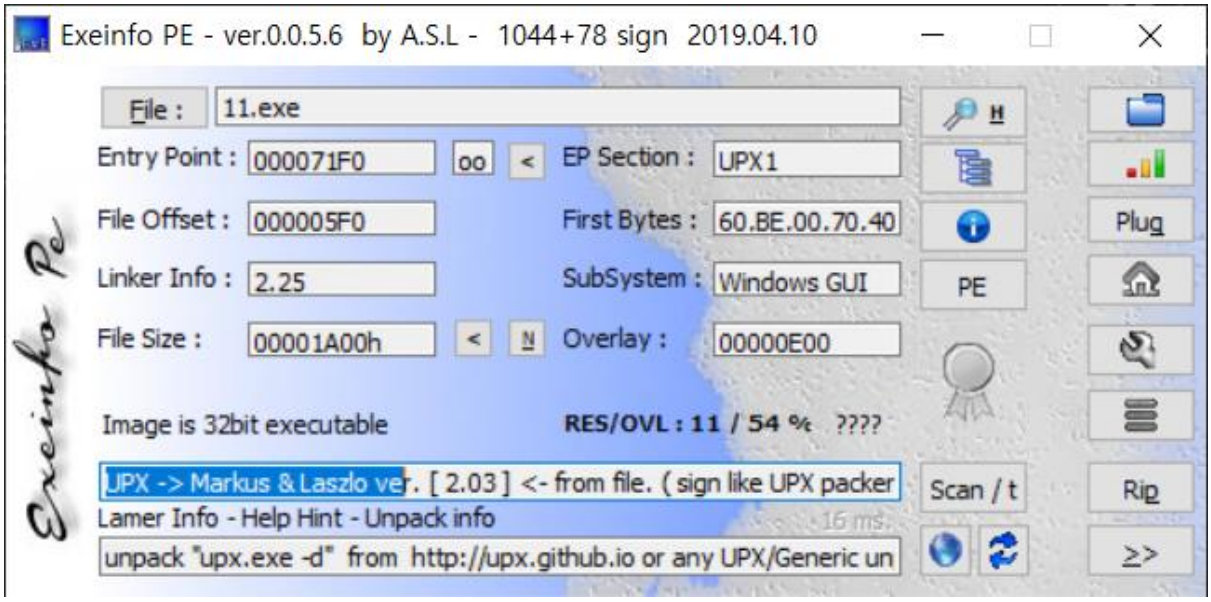
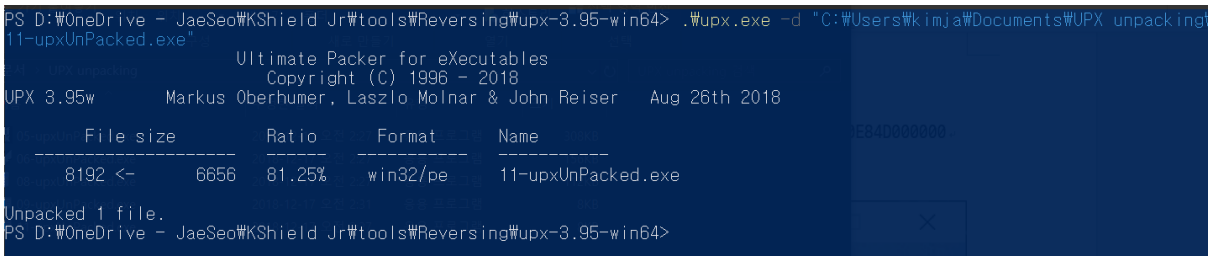
11.exe - OEP를 찾으시오. Ex) 00401000 / Stolenbyte 를 찾으시오.

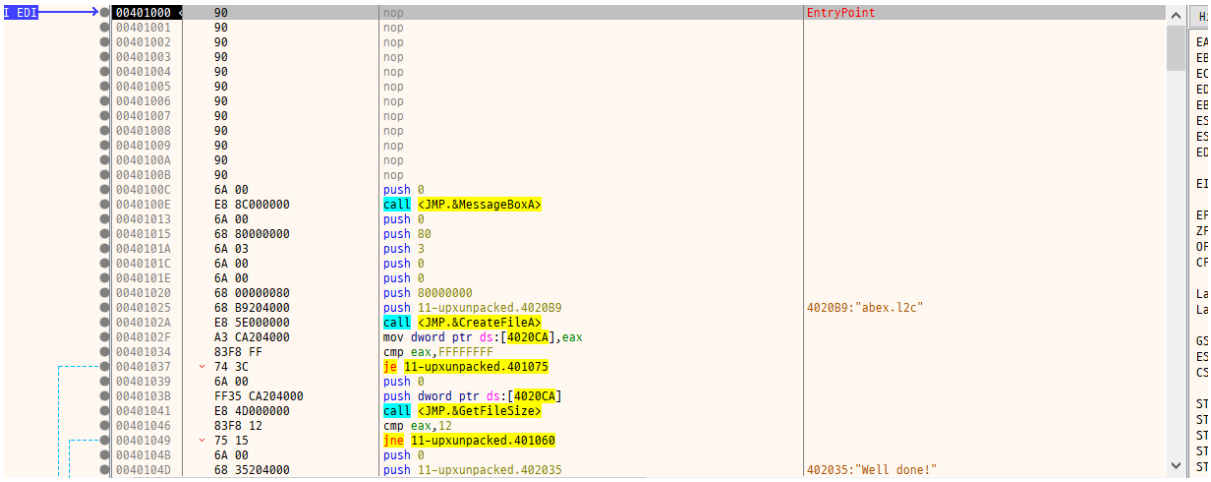
Ex) FF35CA204000E84D000000 정답인증은 OEP+ Stolenbyte Ex) 00401000FF35CA204000E84D000000

프로그램 PE 분석을 일단 진행하여 본다.



UPX로 Packing 되어 있는 것을 알 수가 있는데 UPX로 Unpacking을 진행한다.





UPX로 Unpacking된 파일을 X32dbg로 열어보니 NOP으로 되어 있는 부분이 있다.

Stolen Byte가 있으니 Packing된 파일을 통해 분석을 해본다.

Address	Disassembly
0040736D	popad

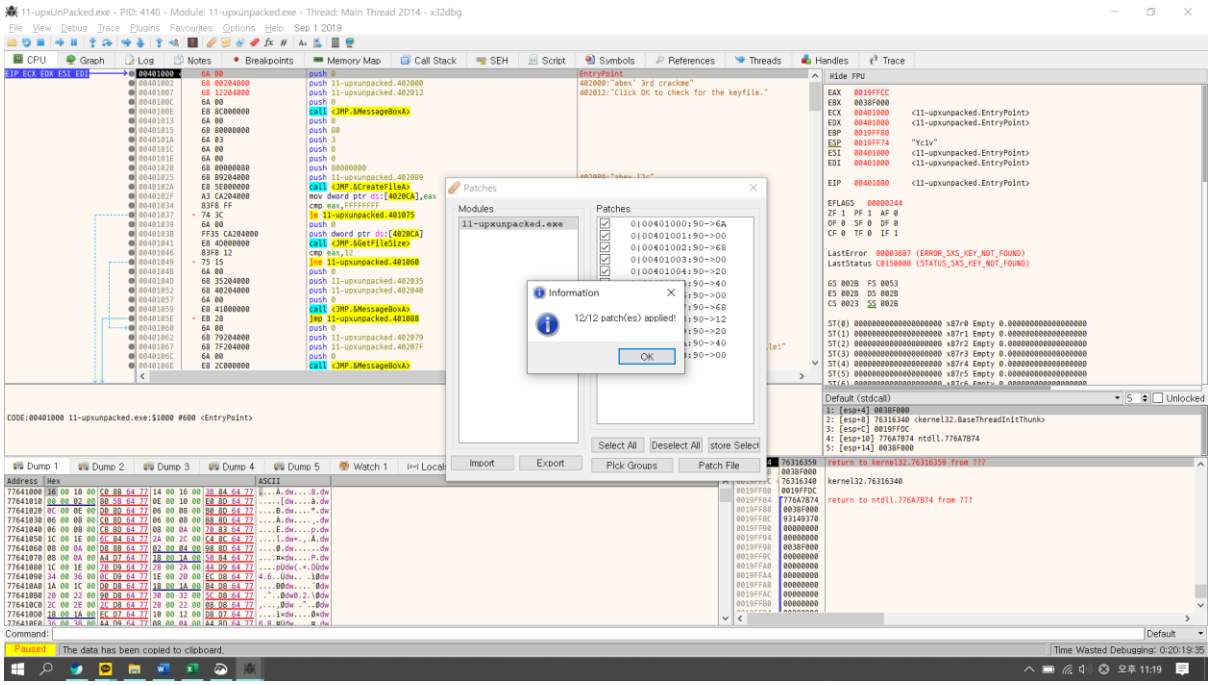
Upx로 unpacking 끝난 이후 POPAD로 레지스터를 복구하는 곳을 찾아 분석한다.

```
0040736C    58              pop eax
0040736D    61              popad
0040736E    6A 00           push 0
00407370    68 00204000     push 11.402000                  402000:"abex' 3rd crackme"
00407375    68 12204000     push 11.402012                  402012:"Click OK to check for the keyfile."
0040737A    8D4424 80       lea eax,dword ptr ss:[esp-80]
0040737E    6A 00           push 0
00407380    39C4            cmp esp,eax
00407382    75 FA           jne 11.40737E
00407384    83EC 80         sub esp,FFFFFF80
00407387    E9 809CFFFF     jmp 11.40100C
```

Popad 이후 push를 하는 모습을 볼 수가 있는데 아직 OEP로 들어가기 전에 작동하는 것을 보아 Stolen byte로 추정이 된다. 그리고 JMP 11.40110C로 OEP로 들어가는 모습을 확인할 수 있다.

```
00401000    6A 00           push 0                          EntryPoint
00401002    68 00204000     push 11-upxunpacked.402000      402000:"abex' 3rd crackme"
00401007    68 12204000     push 11-upxunpacked.402012      402012:"Click OK to check for the keyfile."
0040100C    6A 00           push 0
0040100E    E8 8C000000     call <JMP.&MessageBoxA>
00401013    6A 00           push 0
00401015    68 80000000     push 80
0040101A    6A 03           push 3
0040101C    6A 00           push 0
0040101E    6A 00           push 0
00401020    68 00000080     push 80000000
00401025    68 B9204000     push 11-upxunpacked.4020B9      4020B9:"abex.l2c"
0040102A    E8 5E000000     call <JMP.&CreateFileA>
```

Unpacking 된 파일 Nop 부분에 Stolen Byte를 집어넣어 정상적인 프로그램으로 바꾼다.



정답: 004010006A0068002040006812204000