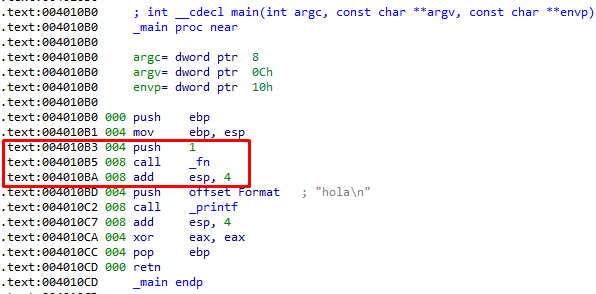
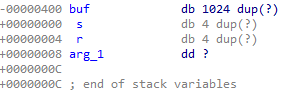
Abrimos el **ABOX** en **IDA**, lo principal que vemos aquí es que hay una función que recibe como parámetro un número entero 1.



Dentro de esta función **fn** vemos una variable local **buf,** más el argumento **arg\_1** que se le paso por parámetro.



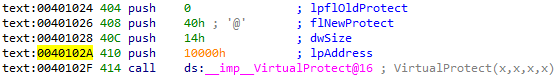
**buf** tiene un tamaño de **0x400 (1024) bytes**

****

Se carga una librería **nsvcrt.dll** la cual viene en conjunto con el **abo**, aquí vemos que no guarda el resultado en ningún lado.

****

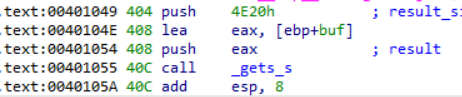
Hay un **VirtualProtect,** que vemos que en el primer argumento se le está pasando un cero, cuando debería ser una dirección donde se guardará la vieja protección.



vemos que hay un función **gets\_s** que es una versión más segura que la **gets.**

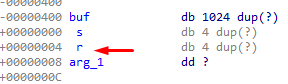
| char \*gets\_s(  char \*buffer,  size\_t sizeInCharacters ); |
| --- |

En este caso se le puede pasar **0x4E20** caracteres que se guardaran en el **buffer.**



En principio pisaremos la **dirección de retorno**, y así podremos controlar el flujo del programa.

Esto lo podemos lograr **desbordando** la variable **buffer** que se llenará con la función **gets\_s**.



Tenemos que darle permiso de **ejecución al stack,** y para lograr esto usaremos **VirtualProtect**, a la cual usando **ROP** llenaremos todos sus **argumentos.** [DEP y ROP](https://docs.google.com/document/d/1OatMNN7Q0QbMUpbaKk67raUxB-fOjsGf9_qFvBtMSbM/edit?usp=sharing)

Así tenemos que dejar los registros, para luego hacer un **Pushad retn**

EDX=**040** #new protect

ECX=**0x10000** #puntero escribible

EDI= puntero a **RET**

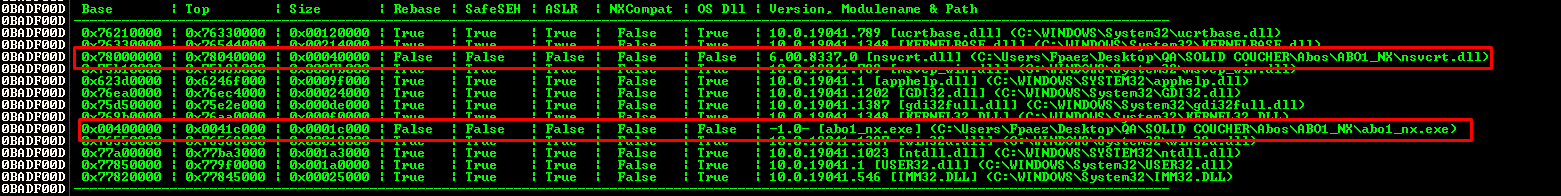
EBX=**1** #size

ESI= Puntero a **VirtualProtect**

EBP= puntero a **JMP ESP** ya que allí retorna y salta después del **VirtualProtect**.

Estos serán los argumentos de **VirtualProtect.**

**-------**

****

Ahi vemos que el módulo que carga y el mismo ejecutable no tienen ni ASLR ni NX (PERO SI EL EJECUTABLE TIENE DEP o.O ?¿)

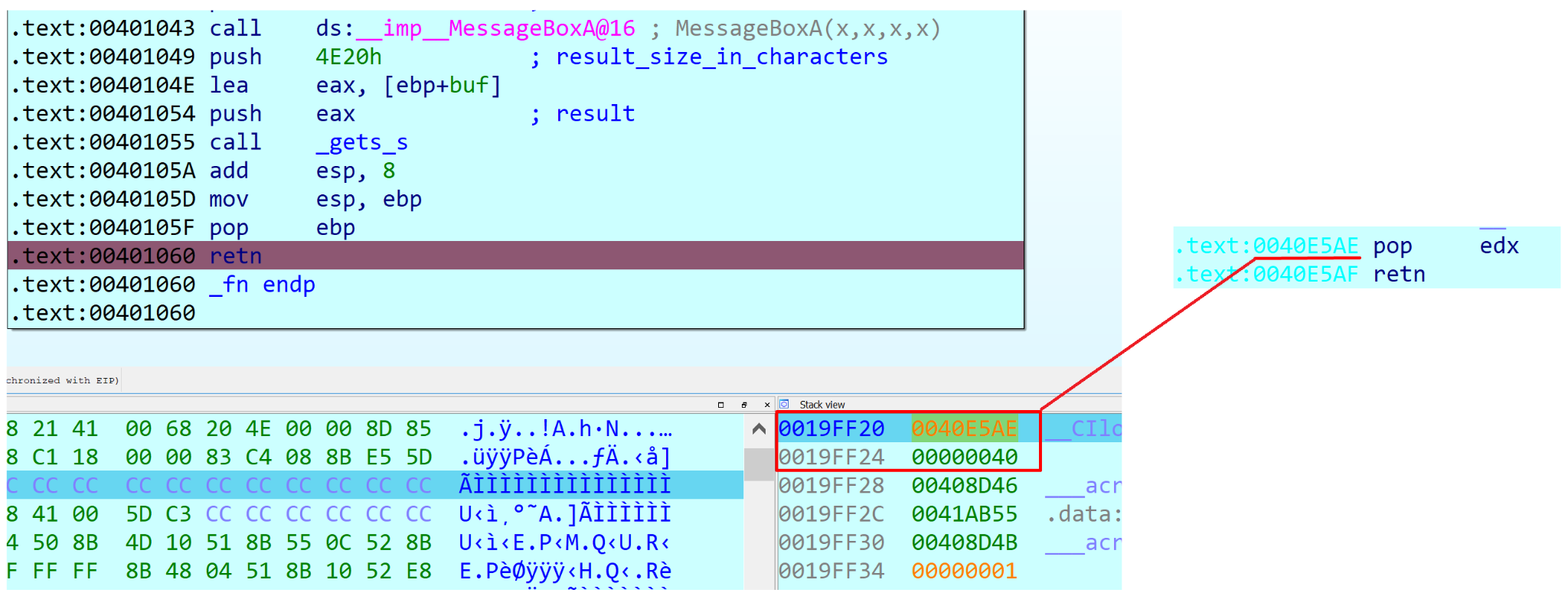
Bueno lo único que tenemos que hacer ahora es acomodar los argumentos de **VirtualProtect**, y hacer que todo funcione jeje… Una vez se ejecute el VirtualProtect debemos chequear que nos retorne EAX en 1, lo que significa que en este caso el stack ahora tiene permisos de ejecución!!!

Vamos acomodar los argumentos,

1. EDX=**040** #new protect
2. ECX=**0x10000** #puntero escribible
3. EDI= puntero a **RET**
4. EBX=**1** #size
5. ESI= Puntero a **VirtualProtect**
6. EBP= puntero a **JMP ESP** ya que allí retorna y salta después del **VirtualProtect**.

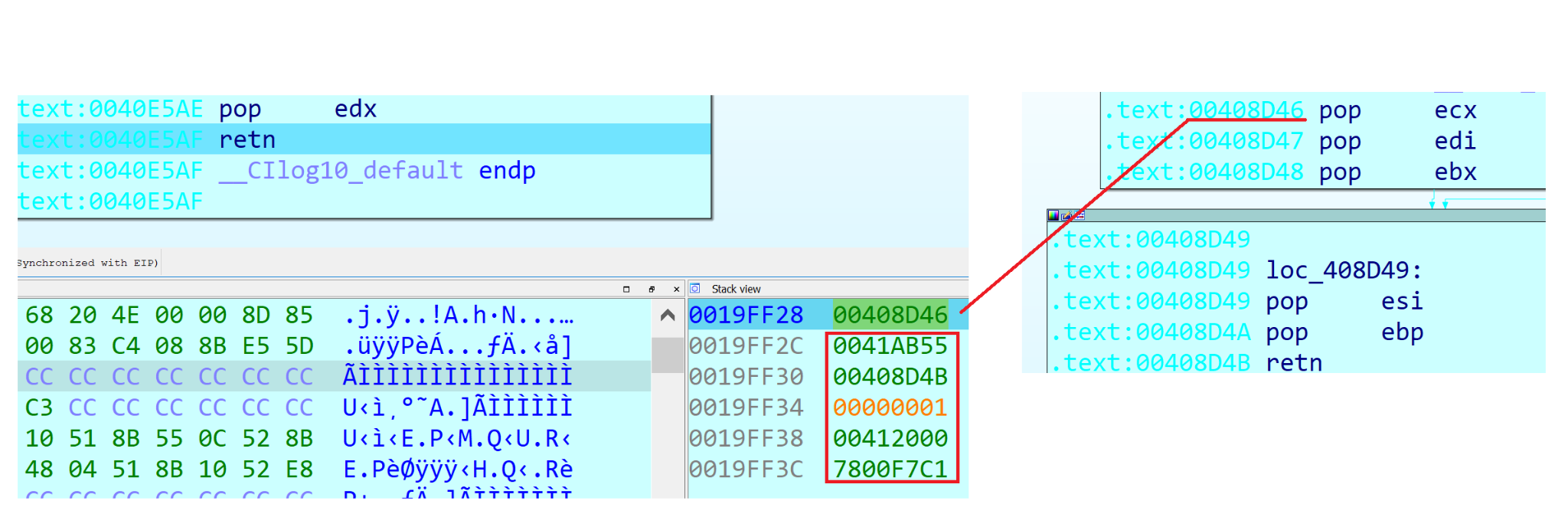
Primero necesitamos meter en el registro **edx** el valor **0x40**, que al momento de llegar al **retn** ya lo deberíamos tener en la pila porque se lo pasamos nosotros con toda la fruta…

Okey, y para meter este valor en el registro **edx** necesitamos un **pop edx / retn**

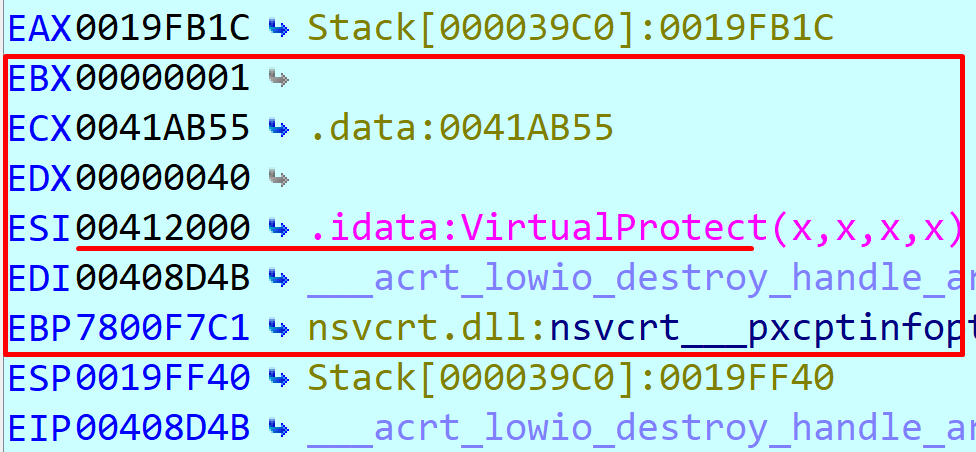
****

Como se observa primero saltará al **pop edx**, lo ejecutará y sacará el valor **40** y lo pondrá en **edx.**

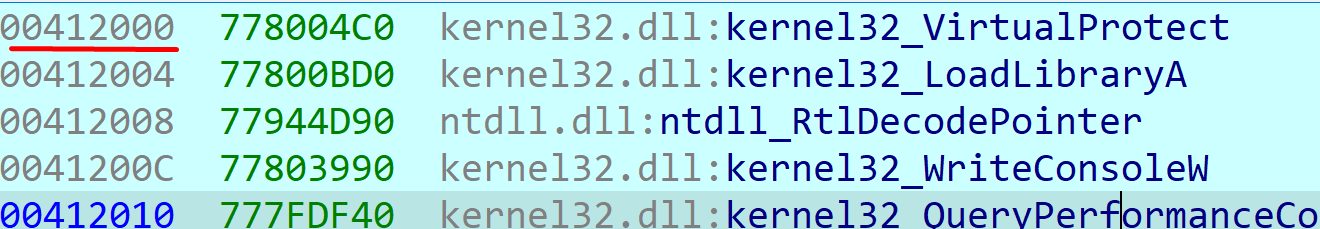
Al resto de los valores podemos sacarlos con un multiple pop que encontre en el código.



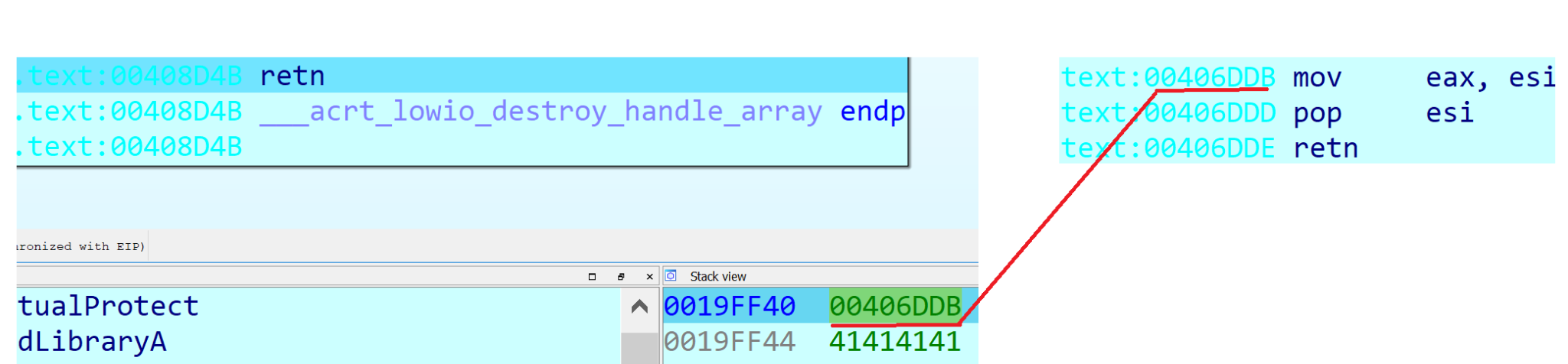
y nos van a quedar los registros asi:



En **ESI** yo puse la dirección en la **IAT**

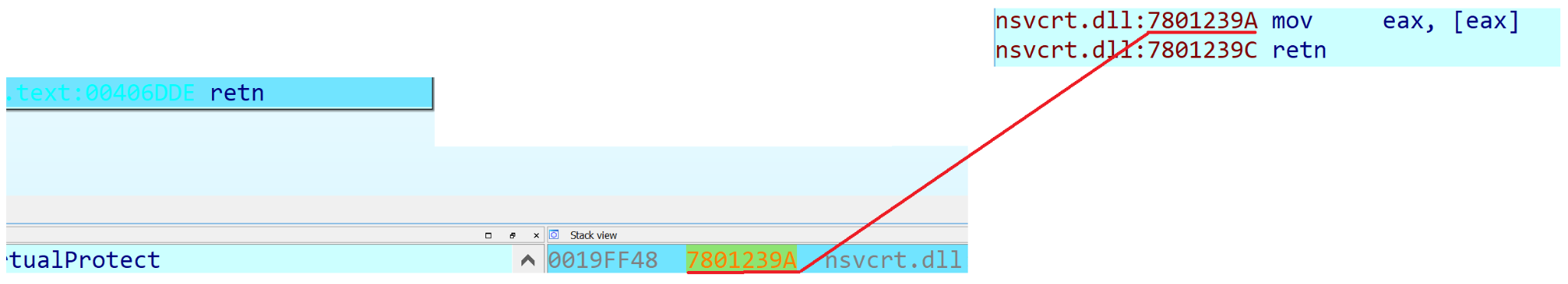
****

Mi objetivo ahora es hacer que en **ESI** quede la dirección de **VirtualProtect,** para ello veo que el registro **eax** lo tengo al pedo, así que voy a moverlo ahí por ahora.



Vemos que hay un POP ESI, y que me va a machacar lo que tenía ahí, vamos a poner un 0x41414141 ….

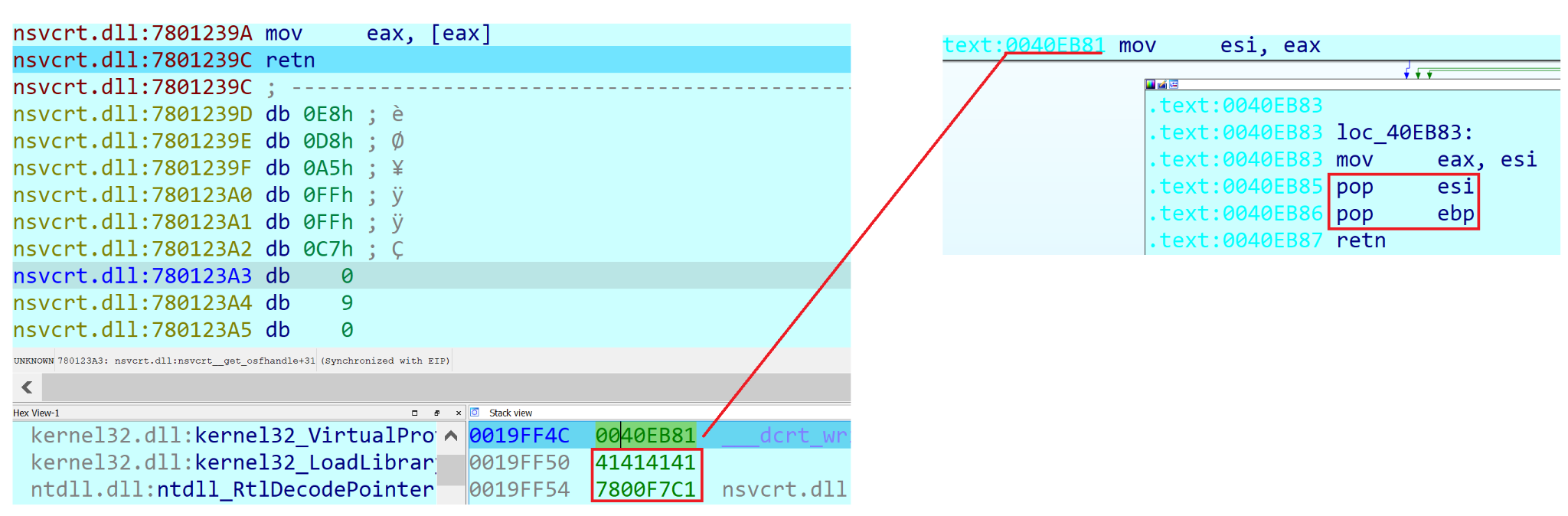
Lo que tenemos que hacer ahora es mover el contenido de **eax** en **eax**, y asi obtendremos la direccion de **VirtualProtect**.



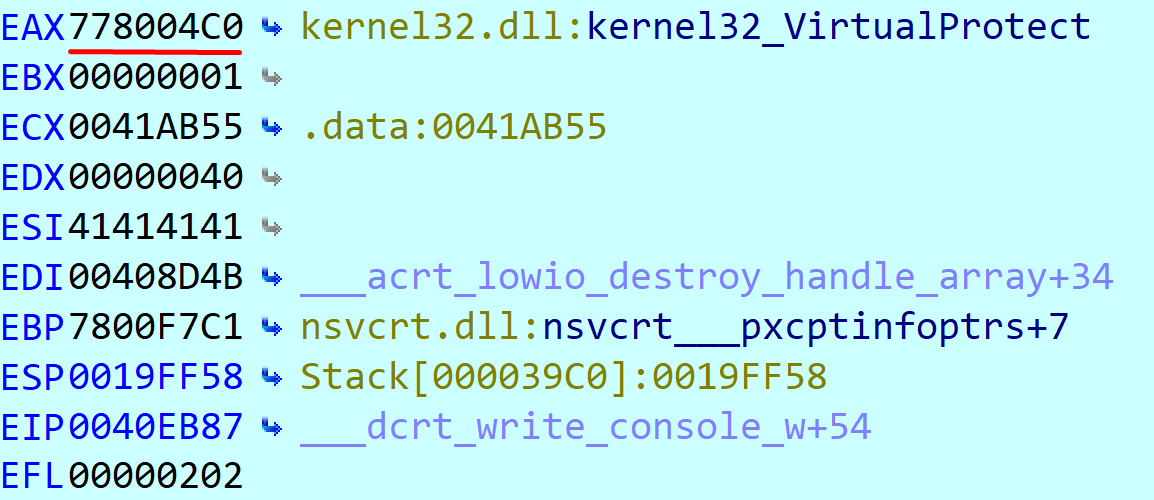


Listo! Ya tenemos la dirección de VirtualProtect, ahora debemos de moverlo a **ESI.**

Este es el único **gadget** que encontré, y la verdad creo que fue al pedo pero bueno,ya quedo asi jeje

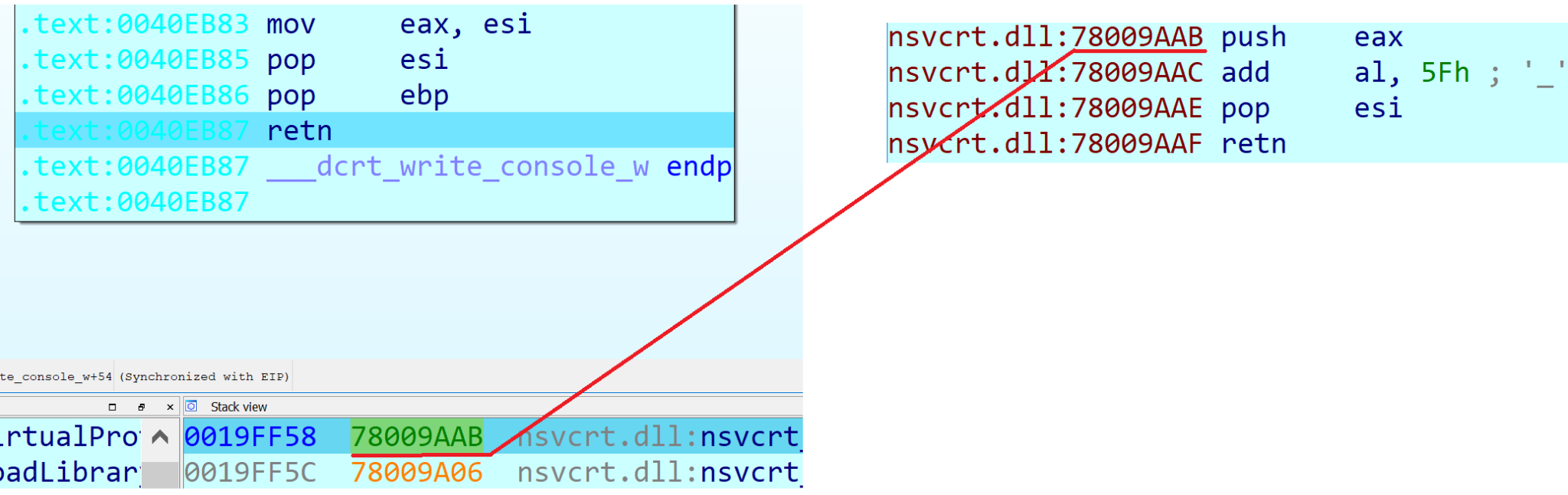


y como vemos finaliza con dos **pop**, machacando nuevamente lo que teniamos en **esi -.-,** y machaca también a **ebp**, pero este valor es constante así que se lo podemos mandar con la fruta, así como vemos en **0019FF54** en el stack.



Bueno volvemos a quedar como antes con EAX apuntando a VirtualProtect, ya dije que fue al pedo el pasito anterior xD

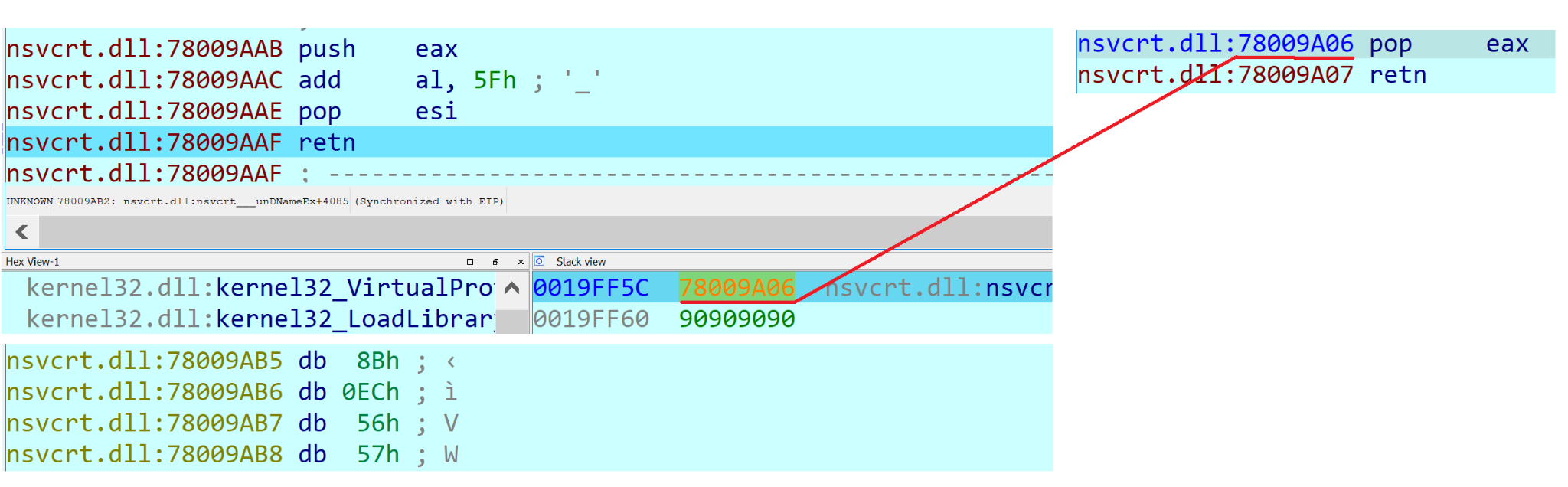
Me di cuenta que podemos hacer un **PUSH EAX / POP ESI**

****

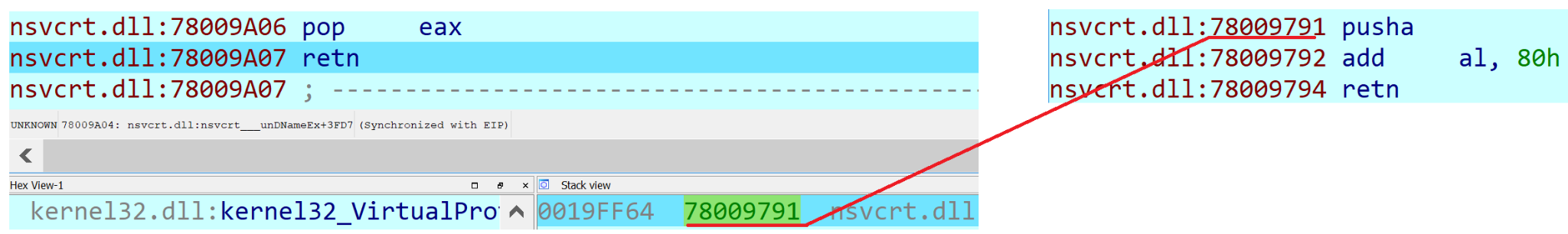
Con esto ya tendremos todos los valores necesarios en los registros!!!

El siguiente paso es buscar un **PUSHAD / RETN,** esto nos pusheara todos los registros en la pila. Pero aquí tenemos un problema! Cuando se ejecute el **JMP ESP** que habíamos seteado la dirección a ese **gadget** en el registro **ebp**, al momento de saltar nos dejará ubicado justo encima de los **bytes que contenga eax.**

Es por eso que es necesario hacer un **pop eax / retn** y que meta en **eax** los bytes **0x90909090** que serán ejecutados y no crashara el **abo.**

****

y ahora si finalmente ejecutamos el **pushad / retn**

****

Si todo está bien, y ejecutamos…

