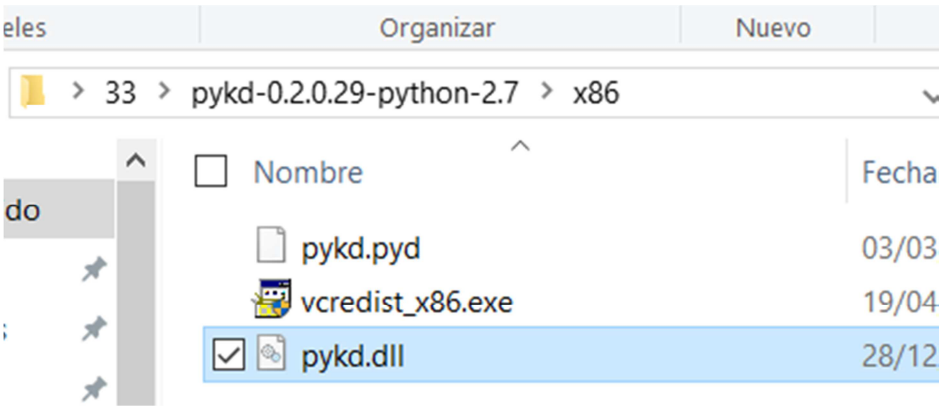
PART 33

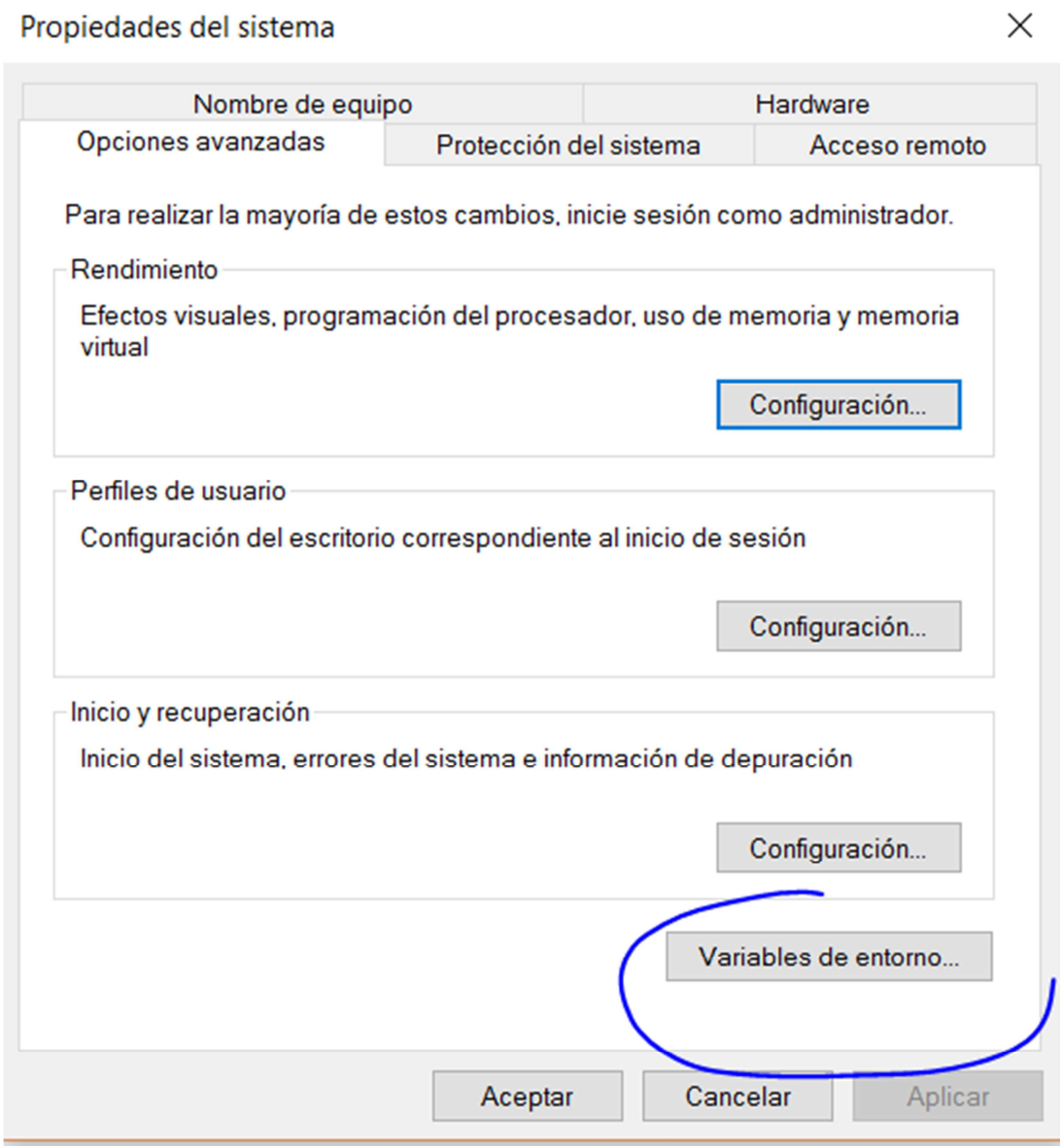
Let's install some plugins for WinDbg that will help us more ahead when we work.

Unfortunately, these plugins only run in WinDbg, but if you run them in WinDbg inside IDA, it crashes because it has conflicts with the Python included in IDA, but we'll use them separately in WinDbg when we need them.

Copy the files that I attached in the **winext** folder located in WinDbg installation folder and install the vcredist_x86.exe runtime.

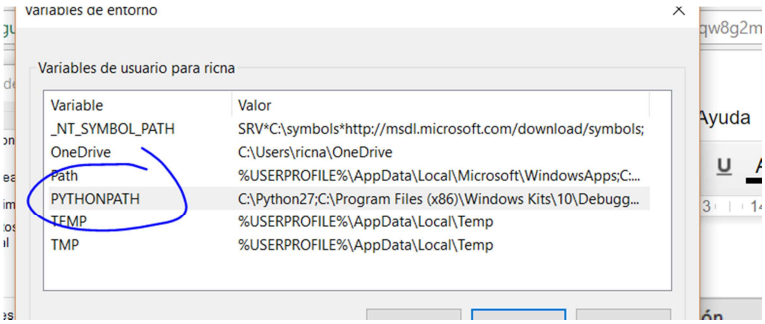


Then, set the system environment variables.

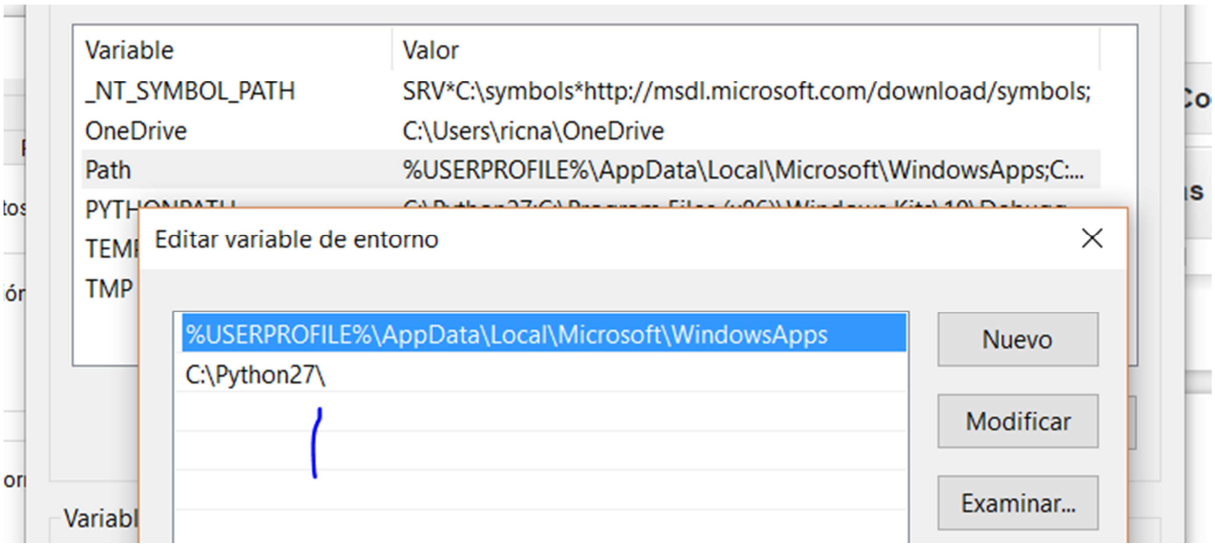


Add the PYTHONPATH variable to which I added the Python path. Then, a semicolon '**;**' and the winext path. In my case, it is like this:

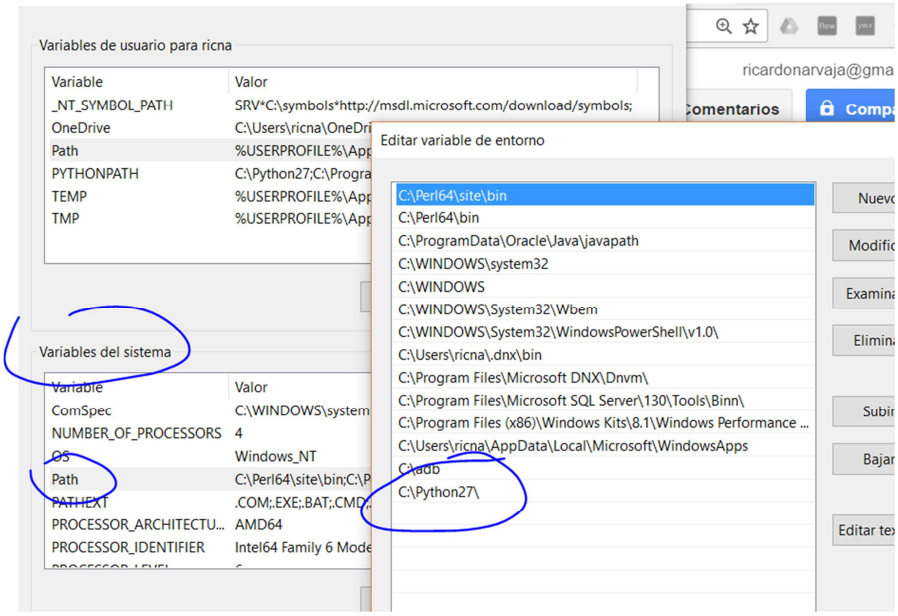
C:\Python27**;**C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\winext



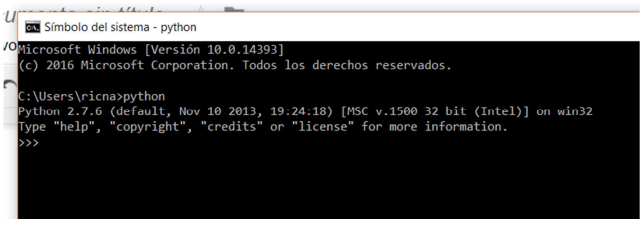
And I add a **semicolon** to the end of the **Path** variable that already existed and then **C:\Python27**



I make sure it has the final bar. I also add it to **System Variables** in **Path**.



After restarting the PC or killing the Windows explorer, we could type **python** in any console and it should accept it.

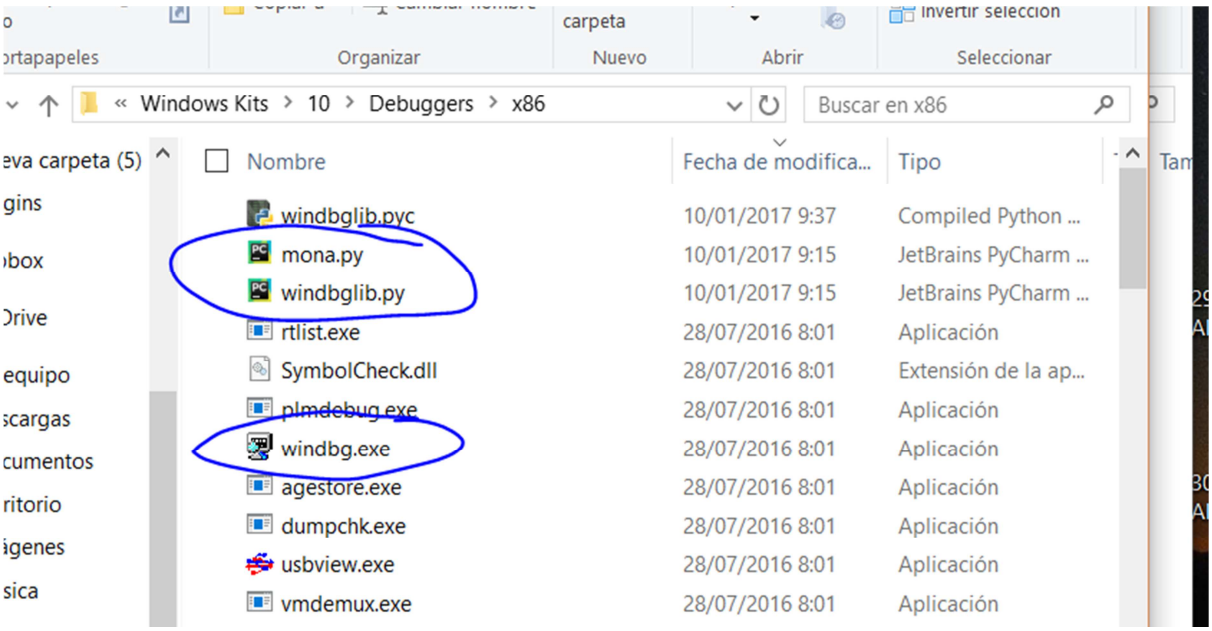


The last thing is to download the latest versions of:

windbglib.py https://github.com/corelan/windbglib/raw/master/windbglib.py

mona.py from https://github.com/corelan/mona/raw/master/mona.py

And copy them in the same folder where windbg.exe is.



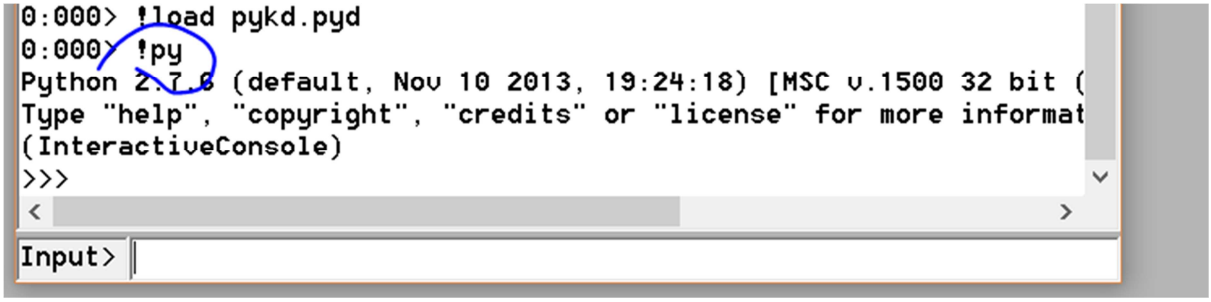
With that, it should work. Let's run WinDbg outside IDA. If it fails, you need some runtime. If not, it should run correctly.

Open any executable with CTRL + E.

When it stops, type:

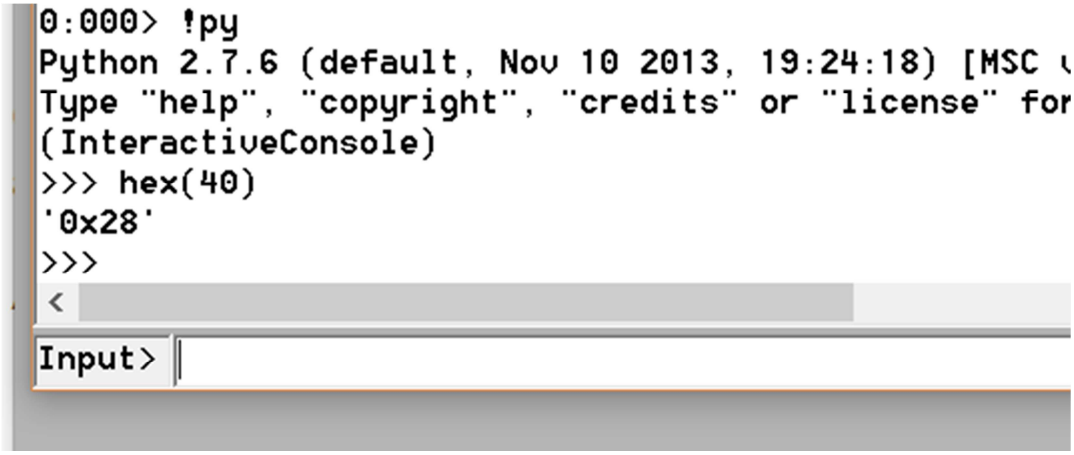
!load pykd.pyd

Nothing would happen, but it must not give any error.



I can execute scripts with **!py** if I enter its path after it, but I opened a Python console at least.

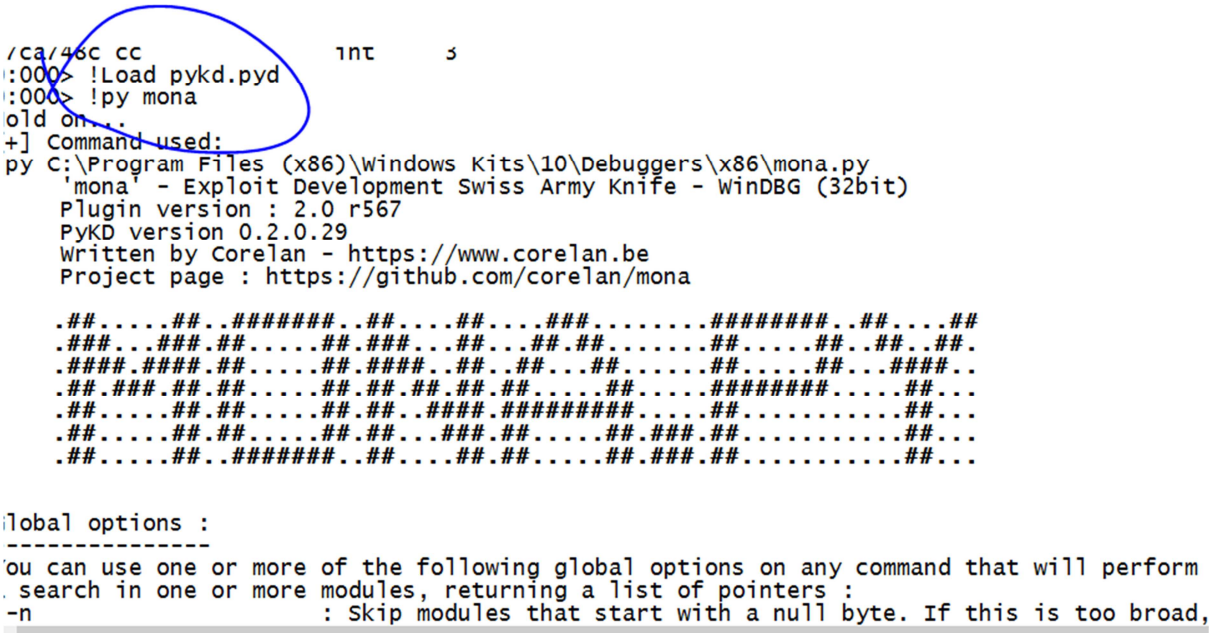
I can also execute Python commands there.



Now, I'll try mona. I go out with exit() and type:

!Load pykd.pyd

!py mona



Let's try some commands.

!py mona modules

```
Module info :
-----------------------------------------------------------------------------------------------------------------------------
Base       | Top        | Size       | Rebase | SafeSEH | ASLR  | NXCompat | OS Dll | Version, Modulename & Path
-----------------------------------------------------------------------------------------------------------------------------
0x74740000 | 0x748e1000 | 0x001a1000 | False  | True    | True  |  False   | True   | 10.0.14393.479 [KERNELBASE.dll] (C:\WINDOWS\Sys
0x77c00000 | 0x77d83000 | 0x00183000 | False  | True    | True  |  False   | True   | 10.0.14393.479 [ntdll.dll] (ntdll.dll)
0x751b0000 | 0x75290000 | 0x000e0000 | False  | True    | True  |  False   | True   | 10.0.14393.0 [KERNEL32.DLL] (C:\WINDOWS\SysWoW6
0x75c70000 | 0x75d2e000 | 0x000be000 | False  | True    | True  |  False   | True   | 7.0.14393.0 [msvcrt.dll] (C:\WINDOWS\SysWoW64\m
0x00400000 | 0x00406000 | 0x00006000 | False  | False   | False |  False   | False  | -1.0- [IDA1.exe] (image00400000)
-----------------------------------------------------------------------------------------------------------------------------
```

We see the running modules. We'll study them later. Now, we are preparing the workplace to have everything ready for the war.

!py mona rop

That would last so much. It will try to see if there is some module where it could generate a ROP (Later, we'll see what it is) and try to create it.

```
Want more info about a given command ?  Run !mona help

0:000> !py mona rop
Hold on...
[+] Command used:
!py C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\mona.py rop

---------- Mona command started on 2017-01-10 11:48:28 (v2.0, rev 567) ----------
[+] Processing arguments and criteria
    - Pointer access level : X
[+] Generating module info table, hang on...
    - Processing modules
    - Done. Let's rock 'n roll.
[+] Preparing output file '_rop_progress_IDA1.exe_8316.log'
    - (Re)setting logfile _rop_progress_IDA1.exe_8316.log
[+] Progress will be written to _rop_progress_IDA1.exe_8316.log
[+] Maximum offset : 40
[+] (Minimum/optional maximum) stackpivot distance : 8
[+] Max nr of instructions : 6
[+] Split output into module rop files ? False
[+] Enumerating 22 endings in 1 module(s)...
    - Querying module IDA1.exe
    - Search complete :
       Ending : RETN, Nr found : 14
       Ending : RETN 0x04, Nr found : 2
    - Filtering and mutating 16 gadgets
      - Progress update : 16 / 16 items processed (Tue 2017/01/10 11:48:30 AM) - (100%)
[+] Creating suggestions list
[+] Processing suggestions
[+] Launching ROP generator
[+] Attempting to produce rop chain for VirtualProtect
```

Sometimes, it could create it, but at least, it is working.

```
def create_rop_chain()

  # rop chain generated with mona.py - www.corelan.be
  rop_gadgets =
  [
    0x00000000,  # [-] Unable to find gadgets to pickup the desired API pointer into esi
    0x75231150,  # ptr to &VirtualAlloc() (skipped module criteria, check if pointer is reli
    0x0040172f,  # POP EBP # RETN [IDA1.exe]
    0x00000000,  # &  [Unable to find ptr to 'JMP ESP']
    0x004013c7,  # POP EBX # POP EBP # RETN [IDA1.exe]
    0x00000001,  # 0x00000001-> ebx
    0x41414141,  # Filler (compensate)
    0x00000000,  # [-] Unable to find gadget to put 00001000 into edx
    0x00000000,  # [-] Unable to find gadget to put 00000040 into ecx
    0x0040152b,  # POP EDI # POP EBP # RETN [IDA1.exe]
    0x00401346,  # RETN (ROP NOP) [IDA1.exe]
    0x41414141,  # Filler (compensate)
    0x00000000,  # [-] Unable to find gadget to put 90909090 into eax
    0x00000000,  # [-] Unable to find pushad gadget
  ].flatten.pack("V*")

  return rop_gadgets

end
```

It returned what it could and as I didn't run WinDbg as administrator, it couldn't write the file with the output, but it prints it anyways.

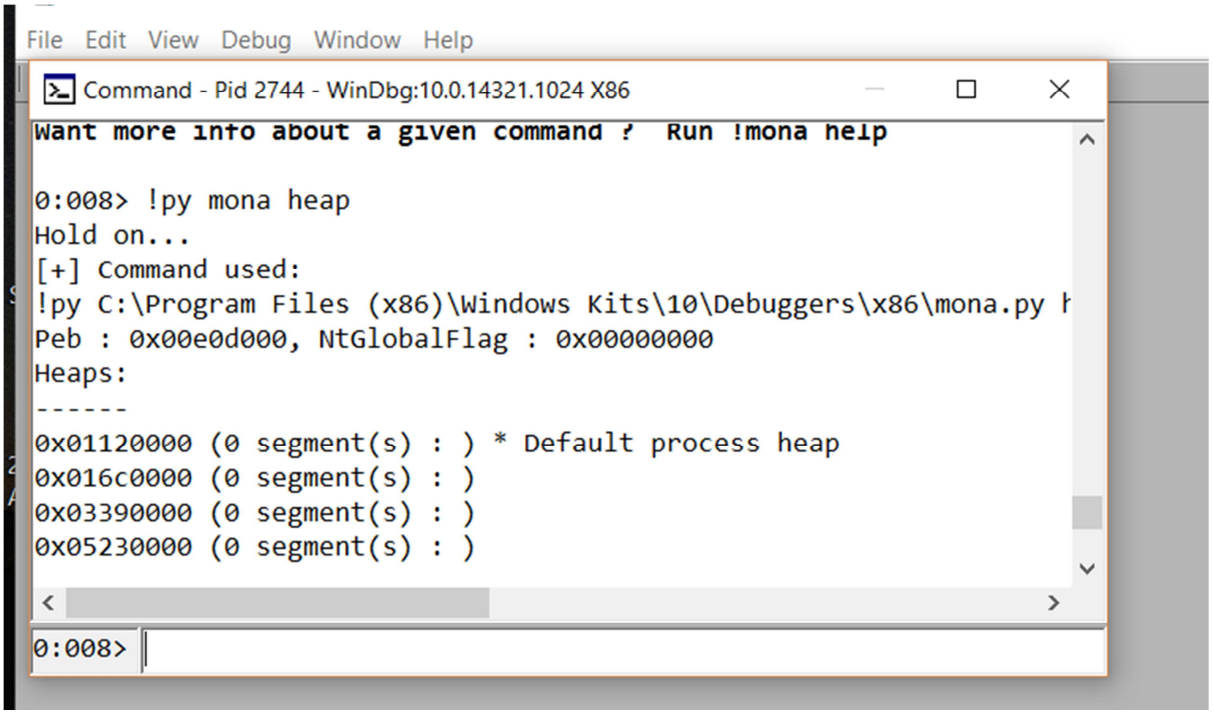
```
   "%u172f%u0040" + // 0x0040172f : ,# POP EBP # RETN [IDA1.exe]
   "%u0000%u0000" + // 0x00000000 : ,# &[Unable to find ptr to 'JMP ESP']
   "%u13c7%u0040" + // 0x004013c7 : ,# POP EBX # POP EBP # RETN [IDA1.exe]
   "%u0001%u0000" + // 0x00000001 : ,# 0x00000001-> ebx
   "%u4141%u4141" + // 0x41414141 : ,# Filler (compensate)
   "%u0000%u0000" + // 0x00000000 : ,# [-] Unable to find gadget to put 00001000 into edx
   "%u0000%u0000" + // 0x00000000 : ,# [-] Unable to find gadget to put 00000040 into ecx
   "%u152b%u0040" + // 0x0040152b : ,# POP EDI # POP EBP # RETN [IDA1.exe]
   "%u1346%u0040" + // 0x00401346 : ,# RETN (ROP NOP) [IDA1.exe]
   "%u4141%u4141" + // 0x41414141 : ,# Filler (compensate)
   "%u0000%u0000" + // 0x00000000 : ,# [-] Unable to find gadget to put 90909090 into eax
   "%u0000%u0000" + // 0x00000000 : ,# [-] Unable to find pushad gadget
   ""); //  :

--------------------------------------------------------------------------------------------------

   ROP generator finished

+] Writing stackpivots to file stackpivot.txt
   Wrote 0 pivots to file
+] Writing suggestions to file rop_suggestions.txt
********************************************************************************
raceback (most recent call last):
 File "C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\mona.py", line 18183, in main
   commands[command].parseProc(opts)
 File "C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\mona.py", line 11341, in procROP
   findROPGADGETS(modulecriteria,criteria,endings,maxoffset,depth,split,thedistance,fast,mode)
 File "C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\mona.py", line 6345, in findROPGADGETS
   with open(thislog, "a") as fh:
OError: [Errno 13] Permission denied: 'rop_suggestions.txt'

********************************************************************************
```

We can check if there is a mona update.

```
0:000> !Load pykd.pyd
0:000> !py mona update
Hold on...
[+] Command used:
!py C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\mona.py update
[+] Version compare :
    Current Version : '2.0', Current Revision : 567
    Latest Version : '2.0', Latest Revision : 567
[+] You are running the latest version
[+] Locating windbglib path
[+] Checking if C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\windbglib.py needs an update...
[+] Version compare :
    Current Version : '1.0', Current Revision : 141
    Latest Version : '1.0', Latest Revision : 141
[+] You are running the latest version

[+] This mona.py action took 0:00:07.741000
```

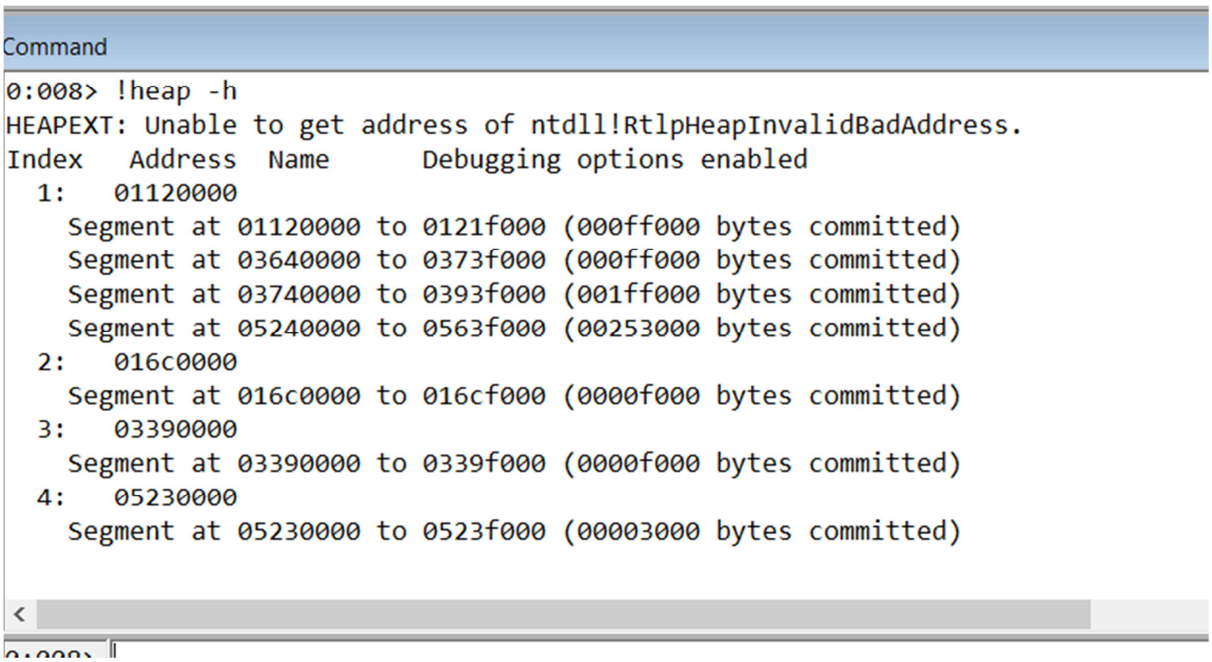
```
+] This mona.py action took 0:00:07.741000
:000> !py mona heap
old on...
+] Command used:
py C:\Program Files (x86)\Windows Kits\10\Debuggers\x86\mona.py heap
eb : 0x0020f000, NtGlobalFlag : 0x00000070
eaps:
-----
x00720000 (0 segment(s) : ) * Default process heap

lease specify a valid searchtype -t
alid values are :
  lal
  lfh
  all
  segments
  chunks
  layout
  fea
  bea

+] This mona.py action took 0:00:00.008000
```

Attaching a running process, in this case, notepad++.



I can see the heap status.

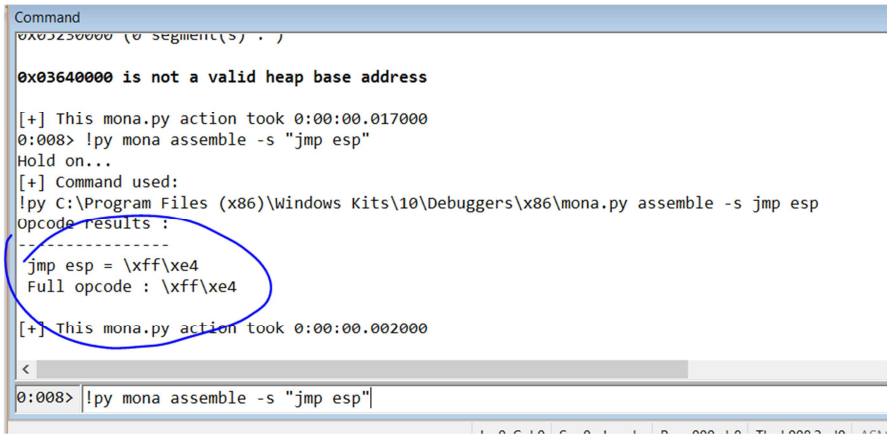
We'll get in deep about this soon. Anyways, WinDbg also has heap commands that we can use inside IDA without using **mona**.



So, we're OK. At least, if we need it, we have many options inside or outside IDA and we have all installed to go ahead.

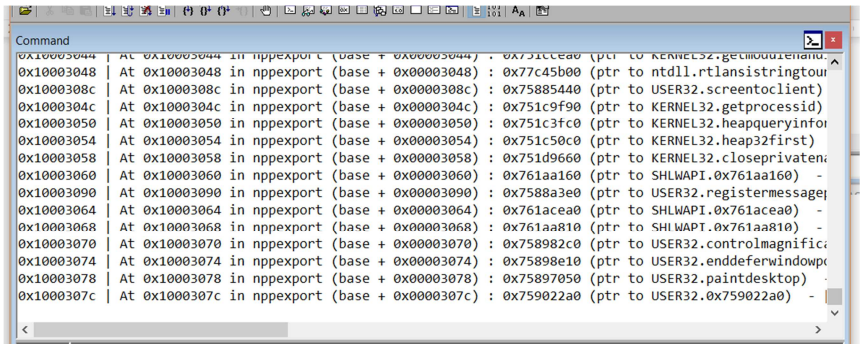
More commands to have fun (it has thousands)

!py mona assemble -s "jmp esp"



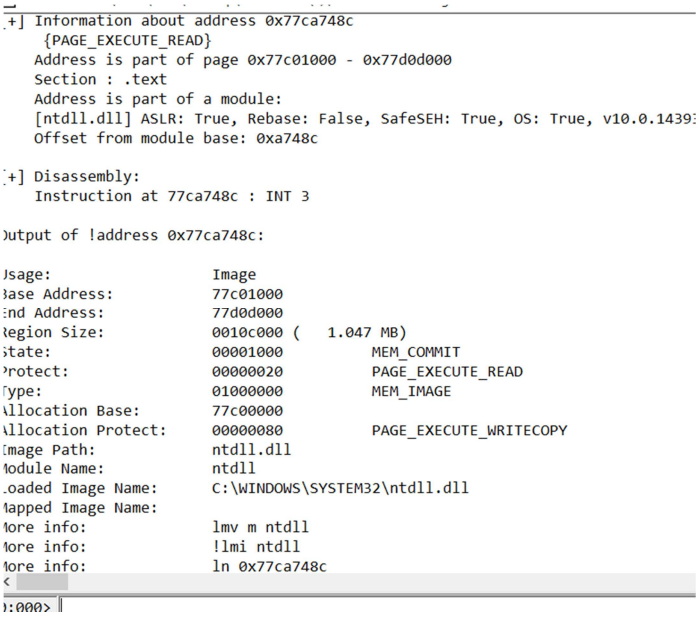
!py mona getiat

It really has many useful commands to see the imported functions, although it lasts too much.



Execute it as administrator to get much info in a file. We can also get info of an address.

!py mona info -a **address**



We had fun for a while and we installed the necessary tools to go ahead.

Ricardo Narvaja

Translated by: @IvinsonCLS