

Win32 Exploit Development with pvefindaddr

+

Project Quebec



who is that guy ?



- Peter “corelanc0d3r” Van Eeckhoutte

- Corelan Team – www.corelan.be



@corelanc0d3r

- I’m **not** a

CISSP,CEH,MCSE,A+,OCSE,CCNA,SSCP,CIW,GIAC,R
SA/CSE,CCSA,CCSE,YMCA,CCSP,TICSA,TICSE,BIS,B
NS,PSP,NSCP,Security+,SCNP,SCNA

- I’m not Lulzsec or Anonymous either

But I am between you and the next 0xc0ff33 break !

time



not enough

flies by

money

universal

stress

deadline

hard to manage

Fact



■ U
V



e TIME

..

Finding a balance is painful

Pain

- W
- u
- M
- B



AA

Multitask/automate

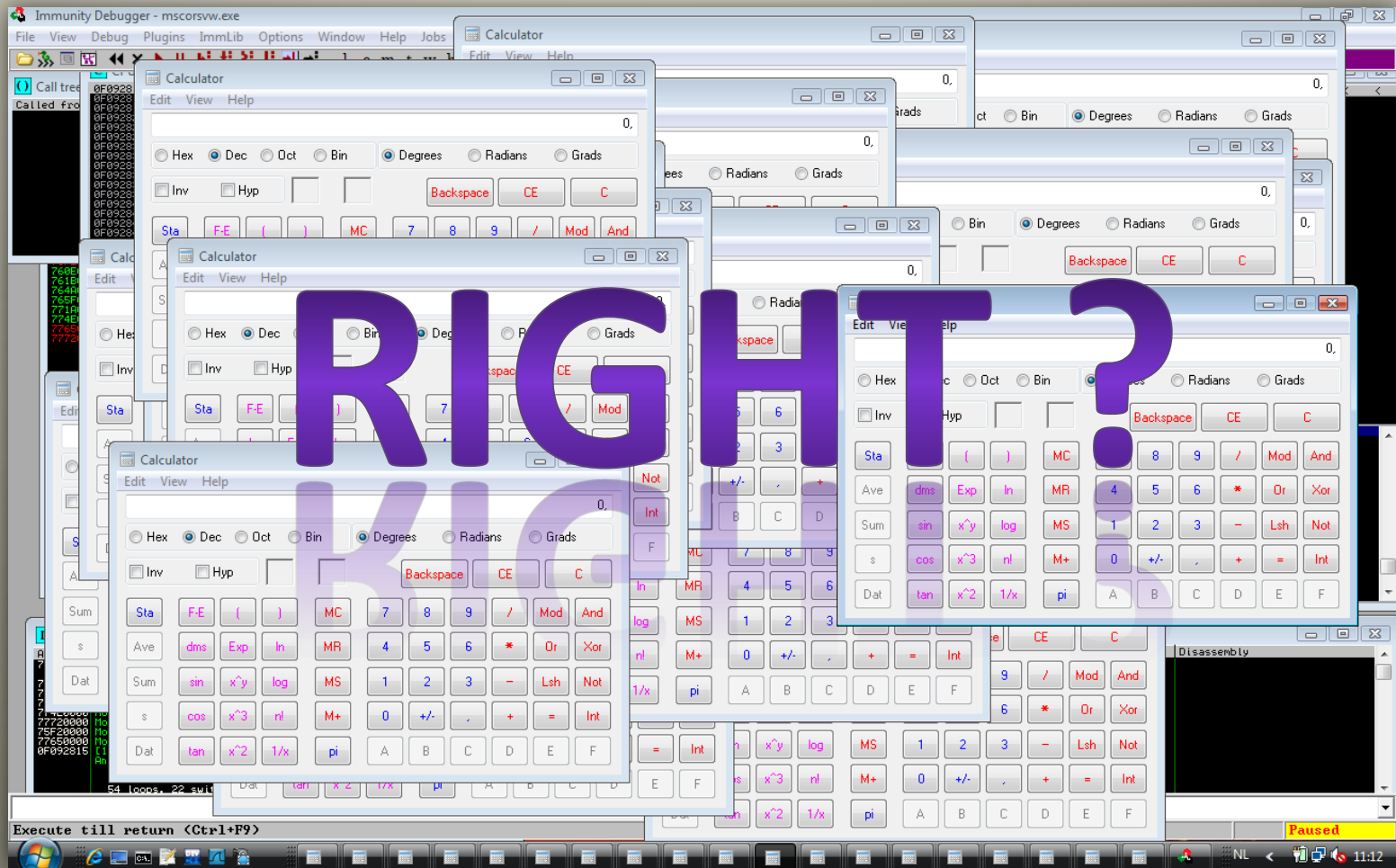


- Before going to work/school
 - Launch your fuzzers
 - *Automated process*
- When the fuzzer finds something
 - A script evaluates the crash
 - We get an email or twitter DM
 - *(We try to automate this)*

when we get home



- Our 1337 script turned the crashes into exploits



...Yeah right...



I wish

**Writing the exploit usually
requires manual work**

Exploit development



- Manual exploit development takes time
- We don't have enough time
 - Pentest => deadline
- Fast, reliable & efficient exploiting

=

*more time for
the harder ones*



Debugger plugins ftw

But...



- ... I was still frustrated
- I wanted something different / better :
 - A single plugin
 - Immunity Debugger
 - “Smart” & reliable

pvefindaddr

Statistics



Statistics



-  Fiction
-  Facts

80% of the statistics is based on fiction, including this one

Pie charts



Pie charts



- Look like a butt
- Don't look like a butt

pvefindaddr



- First version : sept 2009
- PyCommand for Immunity Debugger
- > 5000 lines of code
- Initially written to “find addresses”
- Run when debugger is attached to the application / at crash time
- Don't touch ImmDbg when it runs !
- Usage :

```
!pvefindaddr command [<parameters>]
```

- <http://redmine.corelan.be/projects/pvefindaddr>

pvefindaddr commands



- find
- a
- p / p1 / p2
- xp / xp1 / xp2
- jseh
- j
- jp
- jo
- fa
- fd
- pdep
- depxp
- depwin2k3
- modules
- nosafeseh
- nosafesehaslr
- noaslr
- rop
- jrop
- ropcall
- findmsp
- pattern_create
- pattern_offset
- suggest
- compare
- assemble
- offset
- encode
- info

Enough chitchat



- Seeing = believing
- Saved Return Pointer overwrite
- EIP via function epilog : ESP points at payload
- **“JMP ESP”**
- In general, let's assume we need to find a pointer that jumps to a register

Jmp r32



- Without pvefindaddr
 - Use debugger built-in search
 - Finds one pointer at a time, in the current module
 - Use a command line tool
 - Tell it what module to query
 - If it supports regex, it might actually provide good results
 - Use a plugin that will query one or all modules
 - Lots of results, which one to pick ?
 - Frustration when some/most of the pointers don't work

Jump r32



■ Issues

- We either have to select the modules to query, or we simply can't select them at all
- Why select modules ?
 - ASLR (how to tell ?)
 - Rebase : Often overlooked ! (how to tell ?)
 - OS modules vs application modules
- Pointer properties
 - What if we don't want pointers with null bytes
 - What if we want pointers that are ascii printable ?
- Packed modules vs out-of-debugger scripts
- If you use debugger search, you either are a ninja or you are pushing your luck
- Other plugins are often outdated



Context = key

Jmp r32



■ pvefindaddr

- Will automatically filter out aslr & rebase modules
- Will indicate (or allow you to exclude) pointers that contain null bytes
- Will indicate if a pointer consists of ascii bytes, etc
- Can ignore OS modules if you tell it to
- Writes results to log window & text file for future use (grep) - <http://sourceforge.net/projects/unxutils/>
- Looks for bytes, not instructions
- Searches for

“jmp r32” / “call r32” / “push r32 + ret [offset]”

“mov r32b,r32 + jmp r32b / call r32b / push 32b + r32b”

“push r32 + pop r32b + jmp r32b / call r32b / push r32b”

```
!pvefindaddr j -r esp -n -o
```



Example



- Easy RM to MP3 Converter
- See exploit writing tutorial 1 on www.corelan.be
- Needs “jmp esp”
- Results

	All modules	App modules	App modules not rebased	App modules not rebased, no nulls
Nr of pointers	235	94	5	1



We've got a good pointer...



- Where should we put it ?
- Without pvefindaddr
 - Create a cyclic pattern (metasploit tools)
`./pattern_create.rb 10000 > /tmp/pattern10000.txt`
 - At crash time, find the offset
`./pattern_offset.rb Df2D`
2496

We can do better than that



- Same behaviour with pvefindaddr :
 - !pvefindaddr pattern_create 10000
 - !pvefindaddr pattern_offset Df2D
- *Once you have a crash with a cyclic pattern, there's much more you can do with it !*
- Enumerate primitives before building an exploit !
 - !pvefindaddr findmsp

tip of the day : tell your fuzzer to use a cyclic pattern and always run "findmsp" first at crash time

'findmsp' features



- Finds all cyclic pattern instances in memory
- See if a register is overwritten (+ show offset)
- See if a register points into a cyclic pattern (+ show offset)
- See if a SEH record is overwritten (+ show offset)
- See if there is a pointer into a pattern on the stack
- Indicates if the found pattern is 'normal' or 'unicode'

SEH record overwrites



- Your buffer ends up overwriting an exception handler structure on the stack
- You find a way to trigger an AV
- When the SE Handler kicks in, a pointer to nseh is at ESP+8
- Common exploit technique : overwrite SE Handler with a pointer to p/p/r

SEH record overwrites



- We all know we should avoid using p/p/r from safeseh protected modules
- Similar issues with some of the plugins
 - First find non-safeseh protected modules yourself
 - Query each one of them separately
 - What about aslr & rebase ?
 - What about pointer criteria ? (nulls, ascii, unicode)
 - What about alternative routines ?
 - `add esp+8 / ret <+offset>`
 - `call dword [ebp+offset]`

SEH record overwrites



- **!pvefindaddr p**
 - Search in non-safeseh + non-aslr modules
- **!pvefindaddr p1**
 - Search in non-safeseh + non-aslr + non-rebase modules
- **!pvefindaddr p2**
 - Search in all modules
- **!pvefindaddr a**
 - Search for add esp+8 / ret
- **!pvefindaddr jseh**
 - Search for call dword [ebp+offset] (even outside of loaded modules !)

- Other options :
 - -n : no null pointers
 - -o : no OS modules
 - -m modulename : only search in a given module

If you are lazy/in a hurry



- 3 steps to victory :
 - Trigger a crash with cyclic pattern
 - **!pvefindaddr suggest**
 - pwn





My Documents



mount.bat



My Computer



My Network Places



Recycle Bin



Internet Explorer



msfvenom.txt



Notepad++



Security Configuration...



Command Prompt



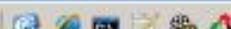
IGSS



Immunity Debugger



igss_poc.py



root@krypt2: /pentest/e...



14:23

Questions ?



Photo : Image: dream designs / FreeDigitalPhotos.net

Bad Characters



- Requirement for reliable exploits
 - Lottery-fu
 - Guess... or
 - Build accurate list (but can be very time consuming)
- Concept:
 - Build array with all bytes ['\x00' -> '\xff']
 - Put array in payload and write it to a separate binary file
 - At crash time, run **!pvefindaddr compare <filename>**
 - Remove bad chars & try again (until array was found unaltered in memory)
- **Bonus** : it will actually locate ALL instances of the array.



- Unicode buffer:
 - Not just inserting null byte, but result of conversion with a given codepage
 - Transforms
 - Transform table well documented by FX (2004)
 - Simply searching for 00xx00yy pointers is not enough
- Haven't seen a lot of scripts that will handle the transforms
- Each pvefindaddr search will indicate unicode AND unicode transforms
- Xion player : <http://www.exploit-db.com/exploits/14517>
 - PoC posted on july 31st 2010, clear SEH overwrite
 - Still no exploit after 2 weeks
 - Wonder why ? 0 unicode pointers
 - pvefindaddr found 3 transforms
 - Example : 0x00470084 -> transformed to 0x0047201e -> p/p/r
 - Exploit (aug 13, 2010) : <http://www.exploit-db.com/exploits/14633/>

Find



- Sure, the debugger has 'find' functionality
- `pvefindaddr find` nicely lists all locations at once
- Hint : looking for eggs ?
 - **`!pvefindaddr find 77303074`**
 - Can help you to tweak start location for hunter & speed up the exploit

What else



■ Some 'quickies' :

- !pvfindaddr assemble "instruction#instruction"
- !pvfindaddr offset <address> <address> (or reg)
 - Will show distance
 - Will generate code to jump the distance
- !pvfindaddr info <address>
- !pvfindaddr modules
- !pvfindaddr noaslr
- !pvfindaddr nosafeseh
- !pvfindaddr noaslr safeseh

Still not impressed ?



- pvefindaddr offers ways to avoid ASLR and safeseh... What about Hardware DEP ?
- pvefindaddr ROP gadgets generator publicly available since mid june 2010 (publication of ROP tutorial).
- Happy Birthday pvefindaddr ROP gadget generator !
- Slow but accurate
- Finds gadgets up to 8 instructions by default (customizable)
- Finds gadgets with custom endings
- Has all the features of other commands (pointer properties, filter ASLR/rebase automatically)
- Performs opcode splitting
 - **EB 58 C3** = JMP SHORT +0x58 / RETN
 - **58 CE** = POP EAX / RETN
- Check timeline of ROP exploits on exploit-db vs publication of tutorial & pvefindaddr rop. Coincidence ?



With pvefindaddr



Questions ?



Project Quebec




What's wrong ?



- pvefindaddr was never designed to do what it does today.
 - Functionality was added over time
 - No real functional design
 - messy code, bad programming
 - space indentation ? (headache++)
 - Not a lot of interaction between the various functions
 - Adding more features/functionality would only make things worse
- Everything works, but it's **very** slow
 - pvefindaddr first searches entire process memory for pointers afterwards
 - Search uses immelib wrapper, which is not optimized
- All output files are written into the current directory
- Hard to exclude certain modules from search
- etc



A photograph of a white cross-shaped grave marker in a cemetery. The cross is the central focus, with the text 'pvefindaddr' printed on its horizontal arm. At the base of the cross is a floral arrangement featuring red and white flowers and greenery. In the background, a row of similar white crosses is visible, receding into the distance on a grassy slope. The lighting is bright, suggesting a sunny day.

pvefindaddr

...welcome...



mona.py

mona



Project Team



	Twitter
ekse	@ekse0x
_sinn3r	@_sinn3r
rick2600	@rick2600
lincoln	
Acidgen	@Acidgen
corelanc0d3r	@corelanc0d3r

mona



- Improvements

- Easier to pronounce
- “help” for each command
- Config file
- Global options
- Performance
- Better interaction between various functions and classes
- Ruby output (Metasploit)
- etc

Usage



■ !mona : show available commands

- | | | |
|-----------------|------------|---------|
| •seh | •find | •header |
| •config | •assemble | •getpc |
| •jmp | •info | •egg |
| •ropfunc | •dump | |
| •rop | •offset | |
| •stackpivot | •compare | |
| •modules | •bp | |
| •filecompare | •findmsp | |
| •pattern_create | •Suggest | |
| •pattern_offset | •bytearray | |

■ !mona help <command> : show detailed info

Config file : mona.ini



- 2 issues

- We needed a better way to store the output of various commands

```
!mona config -set workingfolder c:\logs\%p
```

- We want to exclude certain modules from all searches (shell extensions, VM guest additions, ...)

```
!mona config -set excluded_modules module.dll  
!mona config -add excluded_modules module2.dll
```

Global options : modules



- Options `-n` and `-o` still work
- We need more granularity
 - `-cm <option>=True/False`
 - safeseh
 - aslr
 - os
 - rebase
 - Example : find p/p/r in non-safeseh modules, but don't care about aslr :
 - `!mona seh -cm aslr=true`

Global Options : modules



- Specify list of modules to query

-m “module1.dll,module2.dll,module3.dll”

Wildcards :

*blah.dll | ends with blah.dll

blah* | starts with blah

blah | contains blah

Global Options : Pointers



- Pointers = data !
- Finding one pointer that meets certain criteria might not be too bad
- Encoders usually take care of your shellcode
- ROP makes things harder
- Solution : -cp
- pvefindaddr had “no null bytes” and indicated if a pointer is ascii and/or unicode

Global Options : Pointers



- -cp <option>,<option>
 - nonull
 - unicode (<- improved !)
 - ascii
 - asciiprint
 - upper
 - lower
 - uppernum
 - lowernum
 - numeric
 - alphanum
 - startswithnull

Global Options : Pointers



- Bonus : -cpb <badchars>
- Just like with an encoder, you can specify a list of badchars, this time for pointers
- Example : !mona seh -cpb '\x00\x0a\x0d\xff'



Performance

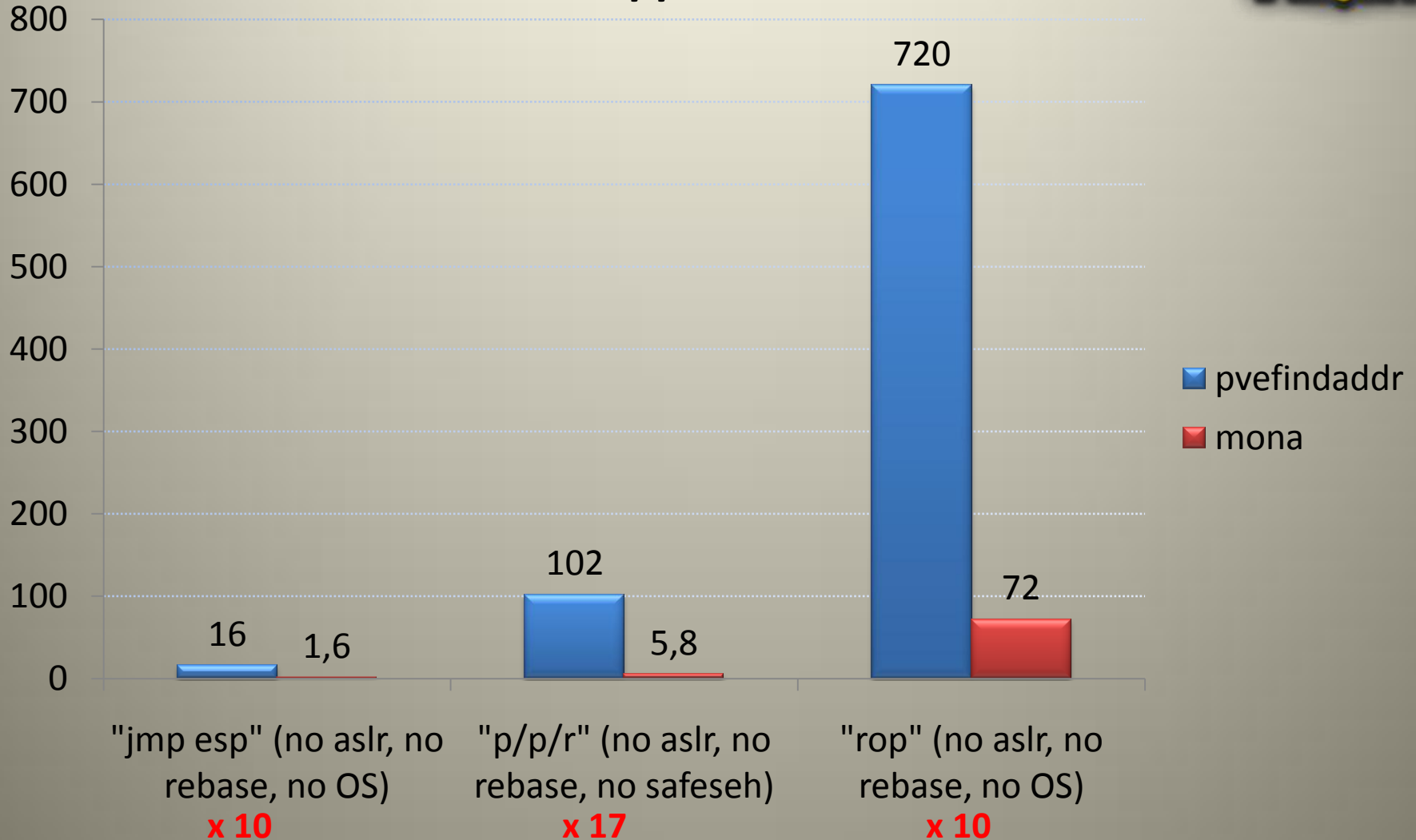


- pvefindaddr was a time saver
- mona : designed to be a lot faster
 - Does not use immelib for searches
 - It will filter during search, not after search
 - Smarter
 - Doesn't search modules that don't start with 00 if you are looking for unicode or 'startwithnull' pointer
 - etc
 - You can specify the number of pointers to return
 - Only need 5 pointers ? Use option -p 5
- Oh, did I mention it searches for more combinations ?
- Access Level
 - -x <level>
 - Levels : R,W,X,RW,RX,WX,RWX,*

Statistics (Facts, not fiction)



Time to generate results, in seconds - App : 7T IGSSDataServer.exe



My Documents mount.bat

My Computer

My Network Places

Recycle Bin

Internet Explorer

msfvenom.bat

Notepad++

Security Configurati...

Command Prompt

IGSS

Immunity Debugger

igss_poc.py

Find



- “p2p”
- Imagine this :
 - You control location referenced by ECX

```
00344338 8BF9 MOV EDI,ECX
0034433A FF7424 04 PUSH DWORD PTR SS:[ESP+4]
0034433E 8B0F MOV ECX,DWORD PTR DS:[EDI]
00344340 8B01 MOV EAX,DWORD PTR DS:[ECX]
00344342 FF50 08 CALL DWORD PTR DS:[EAX+8] ; EIP
```

- Flow :

- ECX -> EDI
- [EDI] -> ECX
- [ECX] -> EAX
- CALL [EAX+8]

- We need ptr -> ptr -> “jmp ecx”

- !mona find -type instr -s “jmp ecx” -p2p -m ntdll.dll

Some other goodies



- bp
- filecompare
- egg
- bytearray
- header

'header' output

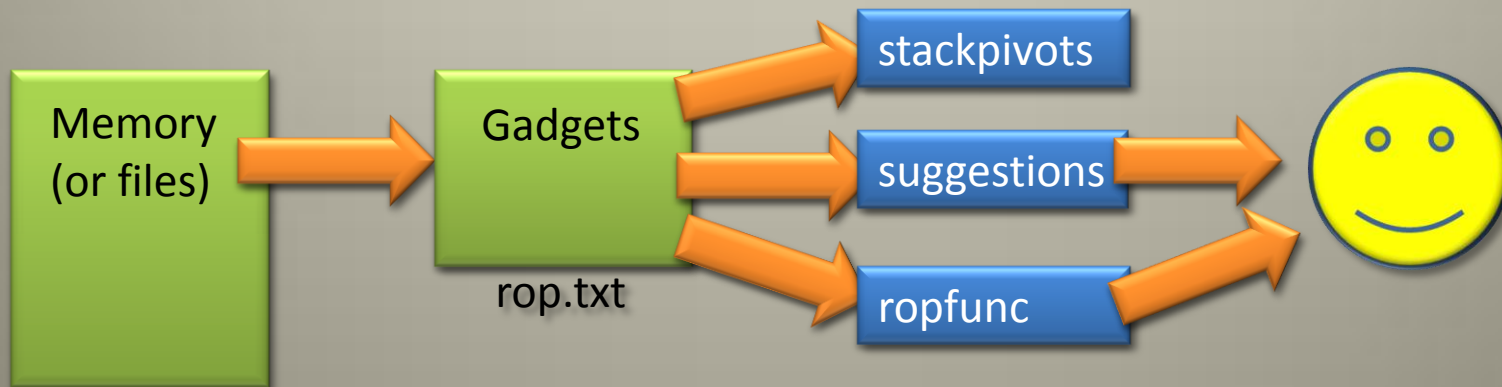


```
header = "BBBBBBBBBAAAAAAAAA"  
header << "\x00 * 12"  
header << "\x1a\x10\x00\x00"  
header << "$"  
header << "\x00\x00\x01\x00"  
header << "\r"  
header << "\x08\x00"  
header << " "  
header << "\x13\x02\x00"  
header << "SCRM@"  
header << "\x06\x94\xb0\x10\xfc"  
header << "\x00 * 11"  
header << "\x08\x01\x09\x02"  
header << "\n"  
header << "\x03\x0b\x04\x0c"  
header << "\xFF * 22"  
header << "\x06\x00\x03\x01\x02\x04\x05\x07\x08\x09\x09"
```

ROP



- More options
- Optional separate stackpivot search (min/max)
- Read from file(s)
- Generation process



What if ?



ROP
the
BUILDER

Can we do it ?

File Edit
root@b

Computer desktop.ini

Network Browser Choice

Control Panel mount.bat

apps

Wireshark Immunity Debugger Notepad++

Recycle Bin

desktop.ini

cmd.exe



Conclusion



- pvefindaddr is dead, long live mona !
- **When can I haz teh mona ?**
 - Expected release : tomorrowz
 - Follow me on twitter or keep an eye on www.corelan.be

Thanks !



Questions ?

IRC : #corelan (freenode)