G DATA | TRUST IN GERMAN SICHERHEIT

FEEDBACK ON WINDBG USAGE

For better or worse (especially worse)

TLP: RAINBOW

ABOUT ME

- Paul Rascagnères
- @r00tbsd
- I'm a malware analyst for the G Data SecurityLabs. I'm specialized in Advanced Persistent Threat (APT) and incident response. I worked on several complex cases such as government linked malware or rootkits analysis. I'm a worldwide speaker at several security events.

- And

  ...

  I don't like WinDBG !!

SURVEY 1

- First:
  Who think it's a good idea to speak about ASM/debugging/WinDBG at 9AM?

SURVEY 2

- Secondly:
  Who think it's a good idea to speak about ASM/debugging/WinDBG?

SURVEY 3

- Thirdly:
  Who think I must wear a costume this year?

ABOUT ME

- TLP: **RAINBOW**

- When should it be used?
  Sources may use TLP: **RAINBOW** when information is cool and useful for the participants. Furthermore, the information must contain funny pictures and probably mistakes or imprecision but shit happens ;)

- How may it be shared?
  Recipients may share TLP: **RAINBOW** information as they want, without restriction, without controls and without copyright: do what you want!!! But do not hesitate to offer a beer to the author.
- Source: wikipedia page I updated 1 hour ago…

WHY THIS TALK?

- Feedback on WinDBG
- How and when to use it
- How to have an acceptable working environment
- Some tricks
- Kind of cheat sheet

- **My personal point of view only!!**
- **I'm not a WinDBG expert…**
- Cause I always said I will never do a talk about WinDBG

WHAT IS WINDBG?

- Free debugger for Windows systems developed by Microsoft
- User-land/**Kernel-land**
- Last cases where I used WinDBG
- **Uroburos**

- **Regin**

WHEN USE WINDBG?

- Only acceptable in kernel-land!!
- For user-land: Immunity Debugger, OllyDBG
- Why?

WHEN USE WINDBG?

- Debugging user-land application with WinDBG is
  - Like have costume in the wrong area… it can hurt you!!

WHEN USE WINDBG?

- Debugging user-land application with WinDBG is
  - Like peel an orange with a circular saw… it can hurt you!!

  - The illustration on Google Image with: “circular+saw+hurt” was so bad than I decided to not show an image

WHEN USE WINDBG?

- Or
  - Like mix coke with Mentos, it can soil you!!

WHEN USE WINDBG?

- Or
  - Like erect a Christmas tree in Paris… Yes it's true!!

  - I know what you think, it looks like…

WHEN USE WINDBG?

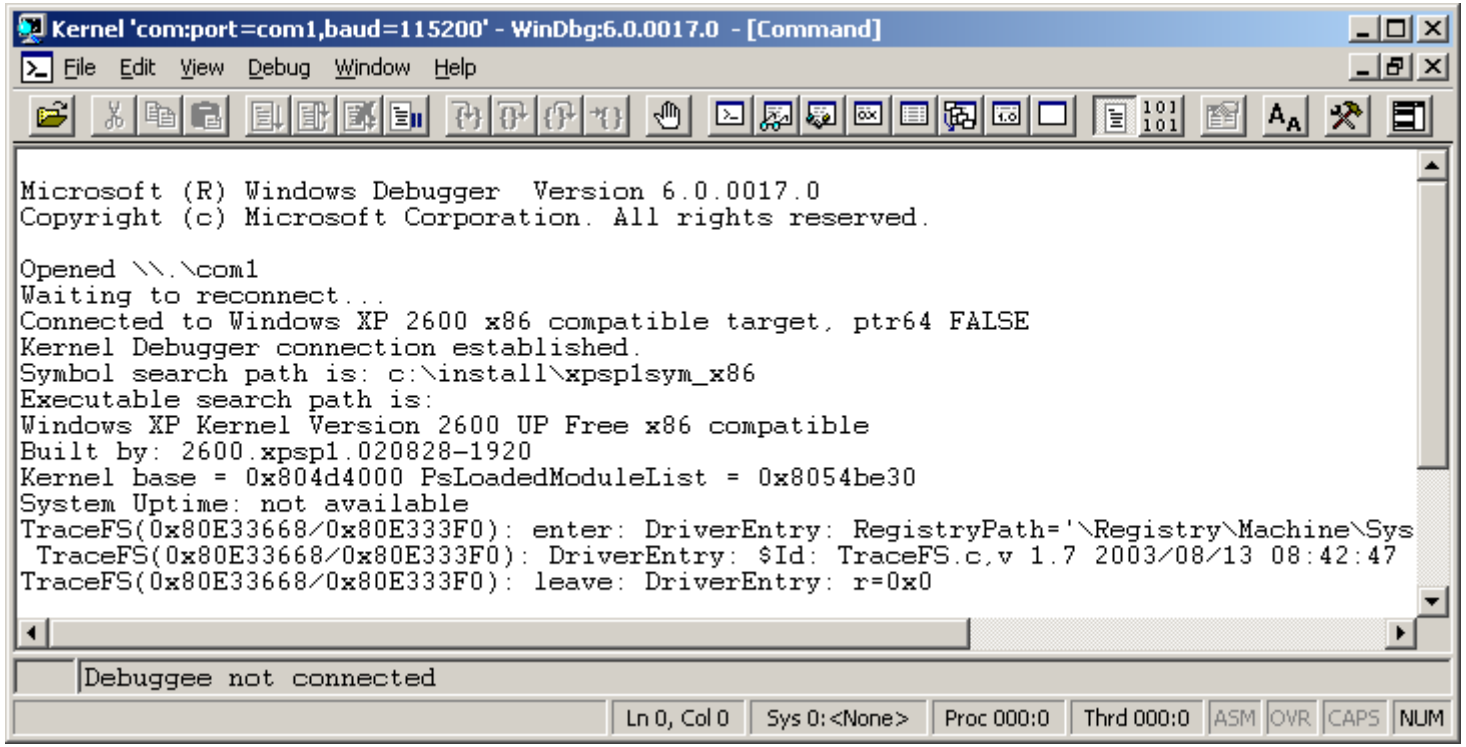
- A top !!!
  - What else…

WHEN USE WINDBG?

- More seriously, never use WinDBG in a plane… it could afraid people around you…
  - True story… “Could you please switch off your laptop? You afraid other people…”

FIRST PROBLEM: THE LAYOUT

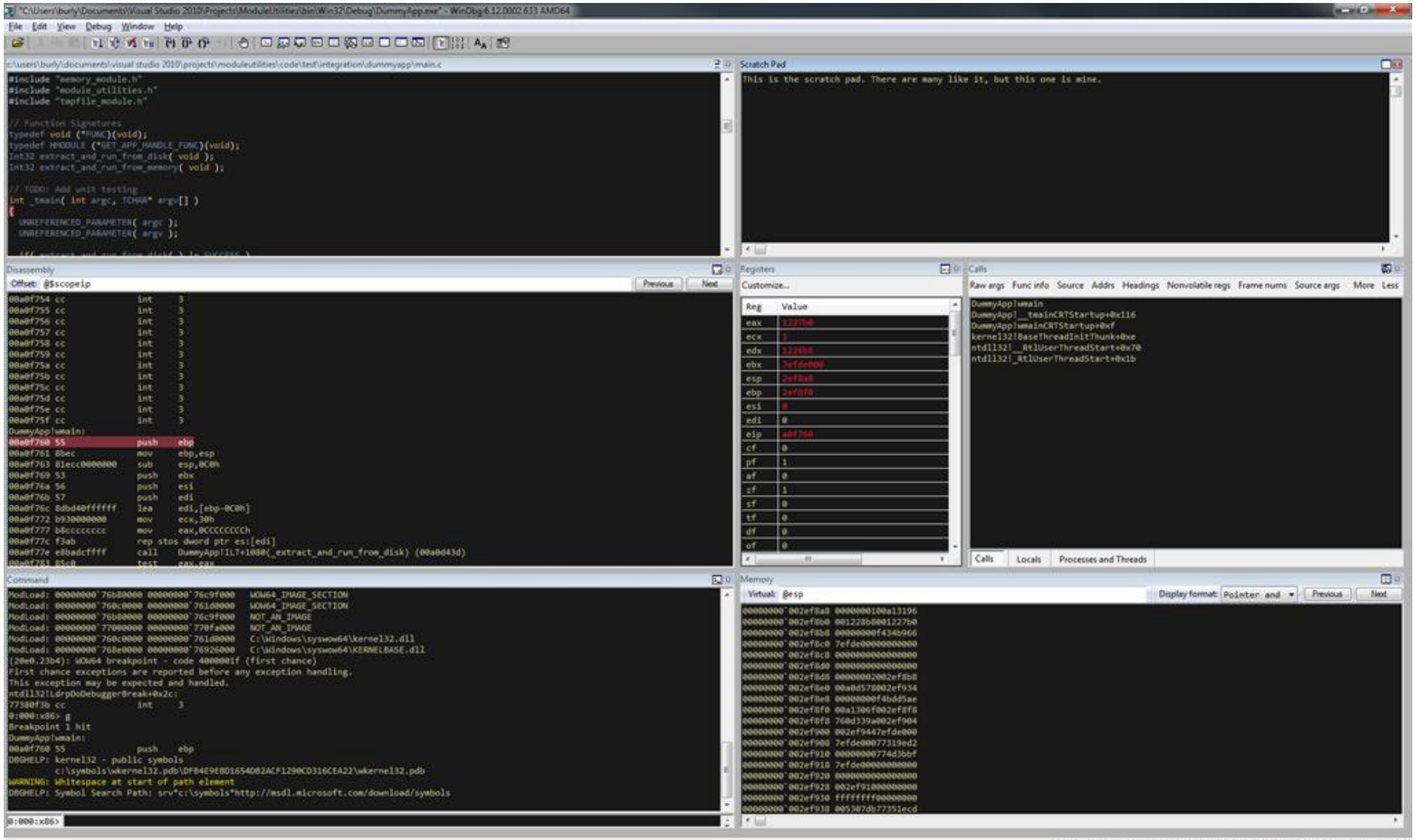
- WinDBG default layout:



FIRST PROBLEM: THE LAYOUT

- Solution: http://www.zachburlingame.com/2011/12/customizing-your-windbg-workspace-and-color-scheme/

FIRST PROBLEM: THE LAYOUT



SECOND PROBLEM: THE SCRIPTING LANGUAGE

- Here is an example of WinDBG script:
  - `0:003> r? $t0 = (ntdll!_peb *) @$peb; eb (@$t0+0x02) 0; ?? @$t0->BeingDebugged`
- So instinctive…

SECOND PROBLEM: THE SCRIPTING LANGUAGE

- Solution: https://pykd.codeplex.com/
- **PyKd** - Python extension to access Debug Engine
- Example:
  -
```
#!/usr/bin/python
import pykd
import socket
#pykd script to modify inet_addr calls to a supplied IP address

def getAddress(localAddr):
    res = pykd.dbgCommand("x " + localAddr)
    if res.count("\n") > 1:
        print "[-] Warning, more than one result for", localAddr
    return res.split()[0]

class handle_inet(pykd.eventHandler):
    def __init__(self):
        #pykd.eventHandler.__init__(self)
        self.localAddr = socket.gethostbyname(socket.gethostname())
        print "[+] Using ip address: " + self.localAddr
[…]
```

2 WAYS TO DEBUG

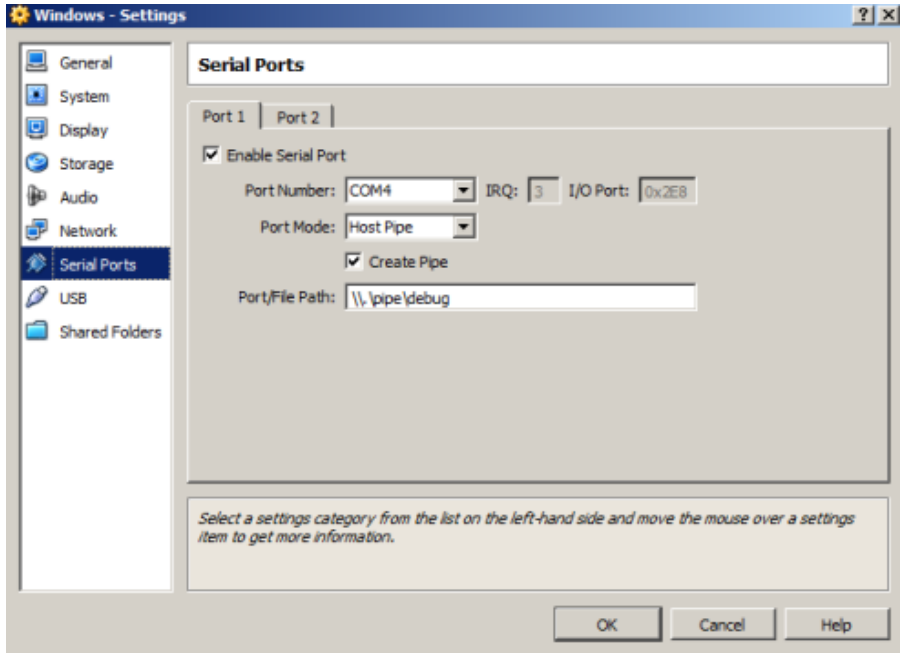
- Live debugging
- Windows crash dump debugging

LIVE DEBUGGING

- Dynamic
- Debug a virtual machine remotely

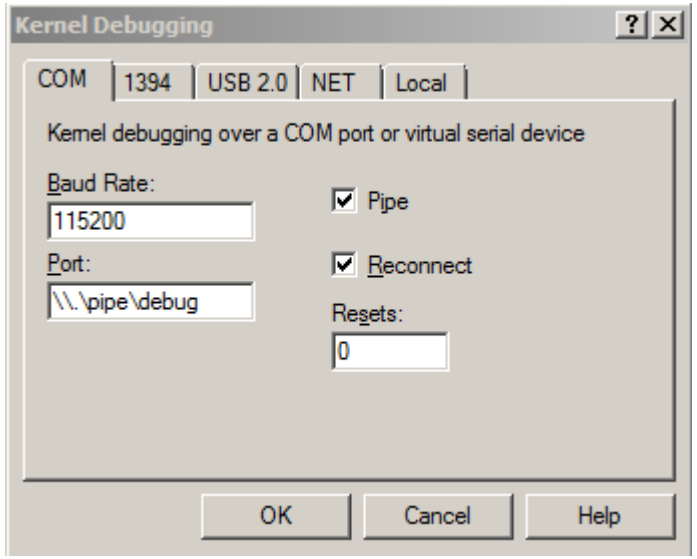
LIVE DEBUGGING

- Example of setup with VirtualBox
  - On the virtual machine:
    - bcdedit /copy {current} /d “Windows with with serial debugging”
    - bcdedit /set {guid} debug on
    - bcdedit /set {guid} debugport 4
    - bcdedit /set {guid} baudrate 115200
  - On VirtualBox:

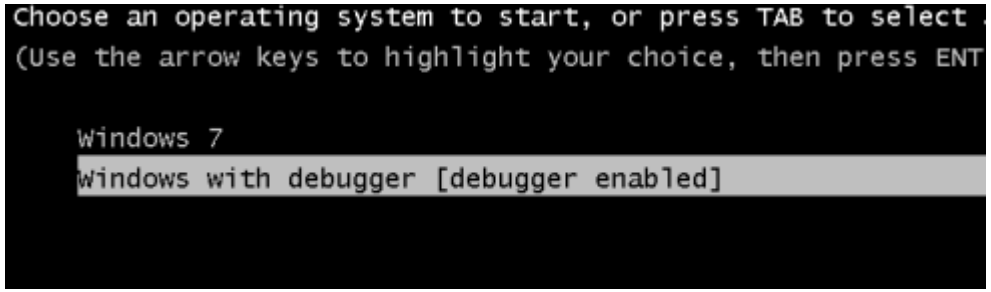


LIVE DEBUGGING

- Example of setup with VirtualBox
  - On WinDBG:



  - During the boot of the virtual machine:



LIVE DEBUGGING

- Problem with this configuration: very slow
- Alternative: http://virtualkd.sysprogs.org/
- **VirtualKD** is a tool that improves your kernel debugging performance with VMWare and VirtualBox

CRASH DUMP DEBUGGING

- Static
- Only a view of the system…
- So no step by step debugging ;)

- Crash dump can be realized by Moonsols tools for example or by crashing your system!

SYMBOLS

- Symbols: Microsoft function name, structure, global variable…
- Remotely: use a Microsoft server
  - **.sympath SRV * f:\localsymbols* http://msdl.microsoft.com/download/symbols**
- Locally: download the symbol archive and install it on your system (useful in the train for example)

AND NOW…

- Now we can start to work !!!!
- First command to execute:
  - `0:000> .logopen /t c:\logs\mylogfile.txt`
    `Opened log file 'c:\logs\mylogfile_02BC_2005-02-28_09-05-50-935.txt'`

DRIVER

- Useful command for driver analysis
  - lm: list module
  - kd> **lm**

```
start    end        module name
00400000 0041a000   win32dd_400000   (deferred)
737f0000 73815000   POWRPROF   (deferred)
74ff0000 75017000   CFGMGR32   (deferred)
75020000 7506b000   KERNELBASE   (deferred)
75070000 75082000   DEVOBJ     (deferred)
751c0000 7520e000   GDI32      (deferred)
```

DRIVER

- kd> **lm v**

```
start    end        module name
00400000 0041a000   win32dd_400000   (deferred)
    Image path: C:\Users\paul.rascagneres\Desktop\win32dd.exe
    Image name: win32dd.exe
    Timestamp:        Sat Apr 17 17:44:49 2010 (4BC9D771)
    CheckSum:         0001AFE6
    ImageSize:        0001A000
    […]
    Timestamp:        Tue Jul 14 03:10:36 2009 (4A5BDB0C)
    CheckSum:         00026DF1
    ImageSize:        00025000
    File version:     6.1.7600.16385
    Product version:  6.1.7600.16385
    File flags:       0 (Mask 3F)
    File OS:          40004 NT Win32
    File type:        2.0 Dll
  Translations:     0409.04b0
    CompanyName:      Microsoft Corporation
    ProductName:      Microsoft® Windows® Operating System
    InternalName:     POWRPROF
    OriginalFilename: POWRPROF.DLL
    ProductVersion:   6.1.7600.16385
    FileVersion:      6.1.7600.16385 (win7_rtm.090713-1255)
    FileDescription:  Power Profile Helper DLL
    LegalCopyright:   © Microsoft Corporation. All rights reserved.
```

DRIVER

- kd> **lm f**

```
start    end        module name
00400000 0041a000   win32dd_400000
C:\Users\paul.rascagneres\Desktop\win32dd.exe
737f0000 73815000   POWRPROF C:\Windows\system32\POWRPROF.dll
74ff0000 75017000   CFGMGR32 C:\Windows\system32\CFGMGR32.dll
75020000 7506b000   KERNELBASE C:\Windows\system32\KERNELBASE.dll
75070000 75082000   DEVOBJ   C:\Windows\system32\DEVOBJ.dll
889cf000 889f1000   VBoxVideo \SystemRoot\system32\DRIVERS\VBoxVideo.sys
```

POOL

- List the pool tag. A pool tag is a four-byte character that is associated with a dynamically allocated chunk of pool memory. The tag is specified by a driver when it allocates the memory.
  -
```
kd> !pool 85985e60
Pool page 85985e60 region is Nonpaged pool
*85980000 : large page allocation, Tag is NtFs, size is 0x92000 bytes
        Pooltag NtFs : StrucSup.c, Binary : ntfs.sys
```

OBJECT

- List driver object
- kd> **!object \driver**

```
Object: 8985ea70  Type: (84841e90) Directory
    ObjectHeader: 8985ea58 (new version)
    HandleCount: 0  PointerCount: 92
    Directory Object: 89805e28  Name: Driver

    Hash Address  Type            Name
    ---- -------  ----            ----
     00  85ae0530 Driver          rdpbus
         8576a1d8 Driver          Beep
         855b74b0 Driver          NDIS
         85598c88 Driver          KSecDD
         8576a3f8 Driver          Null
```

OBJECT

- kd> **!drvobj \Driver\Null**
```
Driver object (8576a3f8) is for:
 \Driver\Null
Driver Extension List: (id , addr)

Device Object list:
864473e0  862531e0  86253748  8576a2d0
```

- kd> **!devobj 864473e0**
```
Device object (864473e0) is for:
 FWPMCALLOUT \Driver\Null DriverObject 8576a3f8
Current Irp 00000000 RefCount 0 Type 00000000 Flags 000000c0
Dacl 8985aaf0 DevExt 00000000 DevObjExt 86447498
ExtensionFlags (0x00000800)  DOE_DEFAULT_SD_PRESENT
Characteristics (0000000000)
Device queue is not busy.
```

INTEGRITY CONTROL

- WinDBG provides a really interesting command to check the code integrity: !chkimg
- Detect corrupt image by comparing the code in memory and the symbol store
- Example to check the kernel integrity (useful to detect inline hook for example):
  - `kd> !chkimg nt -d`

MORE COMMAND

- To disassemble the code at a specific address:
  - `kd> u 859e84f0  L0x16`

```
859e84f0 90              nop
859e84f1 90              nop
859e84f2 90              nop
859e84f3 90              nop
859e84f4 90              nop
[…]
859e84fd 90              nop
859e84fe 90              nop
859e84ff 90              nop
859e8500 6a08            push    8
859e8502 6808859e85      push    859E8508h
859e8507 cb              retf
859e8508 fb              sti
859e8509 50              push    eax
859e850a 51              push    ecx
```

MORE COMMAND

- Display options `d[a| u| b| w| W| d| c| q| f| D] a = ascii chars; u = Unicode chars; b = byte + ascii; w = word (2b); W = word (2b) + ascii; d = dword (4b); c = dword (4b) + ascii ; q = qword (8b) ; f = floating point (single precision - 4b) ; D = floating point (double precision - 8b) ; b = binary + byte ; d = binary + dword` :
  - `kd> db 85980000 L0x100`

```
85980000  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
85980010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
85980020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
85980030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
85980040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
85980050  00 00 00 00 00 00 00 00-61 6d 20 63 61 6e 6e 6f  ........am canno
85980060  74 20 62 65 20 72 75 6e-20 69 6e 20 44 4f 53 20  t be run in DOS
85980070  6d 6f 64 65 2e 0d 0d 0a-24 00 00 00 00 00 00 00  mode....$.......
85980080  00 f8 30 25 44 99 5e 76-44 99 5e 76 44 99 5e 76  ..0%D.^vD.^vD.^v
85980090  44 99 5f 76 94 99 5e 76-1d ba 4d 76 4d 99 5e 76  D._v..^v..MvM.^v
859800a0  63 5f 23 76 46 99 5e 76-63 5f 2f 76 31 99 5e 76  c_#vF.^vc_/v1.^v
859800b0  63 5f 26 76 45 99 5e 76-52 69 63 68 44 99 5e 76  c_&vE.^vRichD.^v
859800c0  00 00 00 00 00 00 00 00-50 45 00 00 4c 01 04 00  ........PE..L...
859800d0  95 41 04 4e 00 00 00 00-00 00 00 00 e0 00 02 21  .A.N...........!
859800e0  0b 01 08 00 00 8e 06 00-00 62 02 00 00 00 00 00  .........b......
859800f0  e0 d2 00 00 00 10 00 00-00 90 06 00 00 00 01 00  ................
```

MORE COMMAND

- Display type: dt
  - kd> **dt ntdll!_PEB @$peb**

```
ntdll!_PEB
+0x000 InheritedAddressSpace : 0 ''
+0x001 ReadImageFileExecOptions : 0 ''
+0x002 BeingDebugged : 0x1 ''
+0x003 BitField : 0x8 ''
+0x003 ImageUsesLargePages : 0y0
+0x003 IsProtectedProcess : 0y0
+0x003 IsLegacyProcess : 0y0
+0x003 IsImageDynamicallyRelocated : 0y1
+0x003 SkipPatchingUser32Forwarders : 0y0
+0x003 SpareBits : 0y000
+0x008 Mutant : 0xffffffff`ffffffff Void
+0x010 ImageBaseAddress : 0x00000000`ff800000 Void
+0x018 Ldr : 0x00000000`77422640 _PEB_LDR_DATA
```

Do

MORE COMMAND

- List exported symbols: x
  - kd> **x USER32!***
  - kd> **x USER32!Create***

MORE COMMAND

- Check if the driver's signing enforcement is enabled:
  - kd> **dq nt!g_cienabled**
    fffff800`02e45eb8 00000001
- List process:
  - kd> **.tlist**

```
      0n0 System Process
      0n4 System
    0n268 smss.exe
    0n344 csrss.exe
    0n380 csrss.exe
    0n388 wininit.exe
    0n428 winlogon.exe
    0n476 services.exe
    0n484 lsass.exe
    0n492 lsm.exe
    0n600 svchost.exe
    0n656 VBoxService.exe
    0n708 svchost.exe
    0n808 svchost.exe
    0n848 svchost.exe
    0n888 svchost.exe
```

MORE COMMAND

- More info about process:
  - kd> **!process iexplore.exe**

```
PROCESS 864d7030  SessionId: 1  Cid: 0b1c    Peb: 7ffdf000  ParentCid:
0ae0
    DirBase: 7ec9b3a0  ObjectTable: 9a117600  HandleCount:  49.
    Image: win32dd.exe
    VadRoot 8649b180 Vads 44 Clone 0 Private 125. Modified 0. Locked 0.
    DeviceMap 94926f10
    Token                             94e67030
    ElapsedTime                       00:00:02.653
    UserTime                          00:00:00.000
    KernelTime                        00:00:00.000
    QuotaPoolUsage[PagedPool]         59636
    QuotaPoolUsage[NonPagedPool]      3744
    Working Set Sizes (now,min,max)  (635, 50, 345) (2540KB, 200KB,
1380KB)
    PeakWorkingSetSize                643
    VirtualSize                       29 Mb
    PeakVirtualSize                   32 Mb
    PageFaultCount                    655
```

MORE COMMAND

- List notepad.exe's loaded libs:
  - `0:000> lm`

```
start end module name
00007ff6`32820000 00007ff6`3285a000 notepad (pdb symbols)
C:\...\notepad.pdb
00007ffc`ab7e0000 00007ffc`ab85b000 WINSPOOL (deferred)
00007ffc`aba10000 00007ffc`abc6a000 COMCTL32 (deferred)
00007ffc`adea0000 00007ffc`adf3f000 SHCORE (deferred)
00007ffc`af490000 00007ffc`af59f000 KERNELBASE (deferred)
00007ffc`af7d0000 00007ffc`af877000 msvcrt (deferred)
00007ffc`af880000 00007ffc`b0c96000 SHELL32 (deferred)
00007ffc`b0e40000 00007ffc`b0ef7000 OLEAUT32 (deferred)
00007ffc`b0f00000 00007ffc`b0f57000 sechost (deferred)
00007ffc`b0f60000 00007ffc`b1005000 ADVAPI32 (deferred)
00007ffc`b1010000 00007ffc`b1155000 GDI32 (deferred)
00007ffc`b1160000 00007ffc`b1296000 RPCRT4 (deferred)
00007ffc`b12a0000 00007ffc`b1411000 USER32 (deferred)
00007ffc`b1420000 00007ffc`b15f6000 combase (deferred)
00007ffc`b16c0000 00007ffc`b17f9000 MSCTF (deferred)
00007ffc`b1800000 00007ffc`b189a000 COMDLG32 (deferred)
```

DEBUGGING COMMAND

- Attach to a process: **`.attach PID`**
- Detach from a process: **`.detach`**
- Run: **`g`** or F5
- Step into: **`t`** or F11
- Step over: **`p`** or F10
- Step to the next return: **`tt`**
- List breakpoint: **`bl`**
- Disable breakpoint: **`bd`**
- Add breakpoint: **`bp`** (for example `bp kernel32!CreateEventW` or `bp 0040108c` or `bp main+5c`)

DEBUGGING COMMAND

- Callstack display
  - `0:000> kb`

```
ChildEBP RetAddr  Args to Child
0008eb38 76eacf0e 0008eb48 76f10718 003a0043 kernel32!LoadLibraryW
0008ed54 76eab8b3 0008f4dc 0008f420 00000001 USER32!User32InitializeImmEntryTable+0xffa
0008f40c 776f9950 76e90000 00000001 0008f714 USER32!UserClientDllInitialize+0x1c6
0008f42c 776fd8c9 76eab6ed 76e90000 00000001 ntdll!RtlQueryEnvironmentVariable+0x241
0008f520 7770681c 0008f714 7efdd000 7efde000 ntdll!LdrResSearchResource+0xb4d
0008f6a0 777052d6 0008f714 776c0000 77efe3a9 ntdll!RtlGetNtVersionNumbers+0x9b
0008f6f0 776f9e79 0008f714 776c0000 00000000 ntdll!RtlSetUnhandledExceptionFilter+0x50
0008f700 00000000 0008f714 776c0000 00000000 ntdll!LdrInitializeThunk+0x10
```

PYKD

- Example of python script used on the Uroburos analysis

```
import pykd

output = pykd.dbgCommand("x nt!*").split("\n")
for i in output:
  if i != "":
    addr=i.split()[0]
    name=i.split()[1]
    opcode=pykd.dbgCommand("db %(addr)s+2 L2" % {'addr': addr}).split()
    if (opcode[1] == "cd") and (opcode[2] == "c3"):
      print "Hook: "+name
```

PYKD

- Example of python script used on the Uroburos analysis

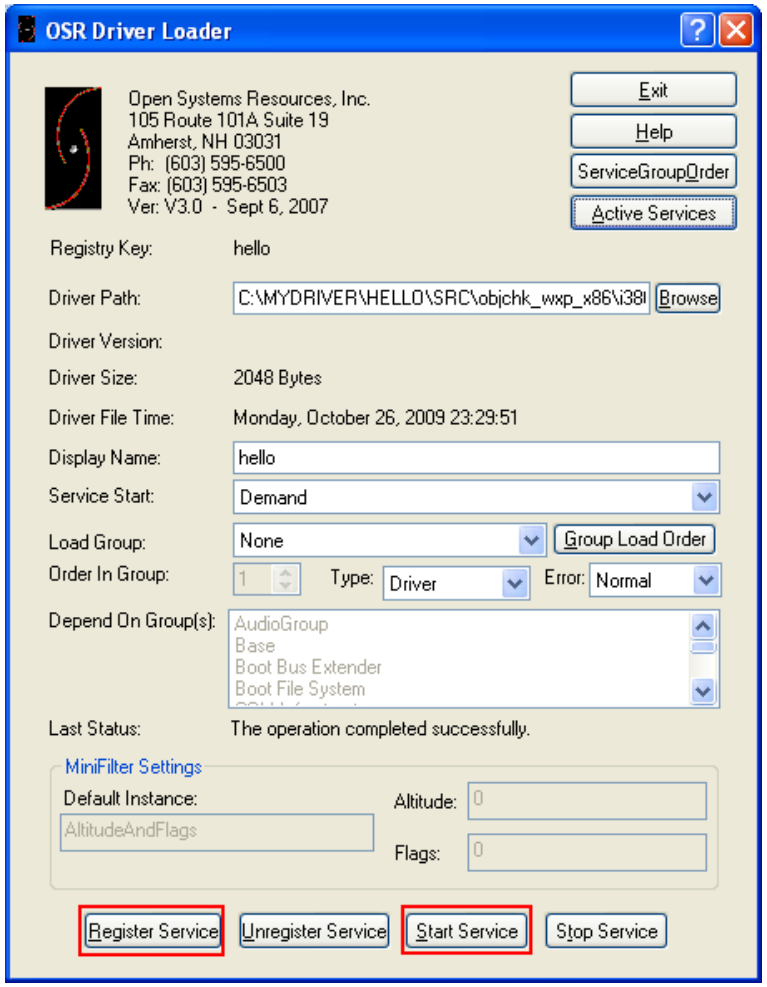
```
kd> !py c:\hook.py
Hook:  nt!NtCreateKey
Hook:  nt!NtQueryInformationProcess
Hook:  nt!NtQuerySystemInformation
Hook:  nt!ObOpenObjectByName
Hook:  nt!NtClose
Hook:  nt!IoCreateDevice
Hook:  nt!NtEnumerateKey
Hook:  nt!NtShutdownSystem
Hook:  nt!NtTerminateProcess
Hook:  nt!IofCallDriver
Hook:  nt!NtQueryKey
Hook:  nt!NtCreateUserProcess
Hook:  nt!NtCreateThread
Hook:  nt!NtSaveKey
Hook:  nt!NtReadFile
```

CASE STUDY

- **Case study: Uroburos unpack**

UROBUROS PACKED DRIVER

- How to unpack the Uroburos packed driver?
- How to load the driver (.sys)?
- Solution: OSR Driver Loader

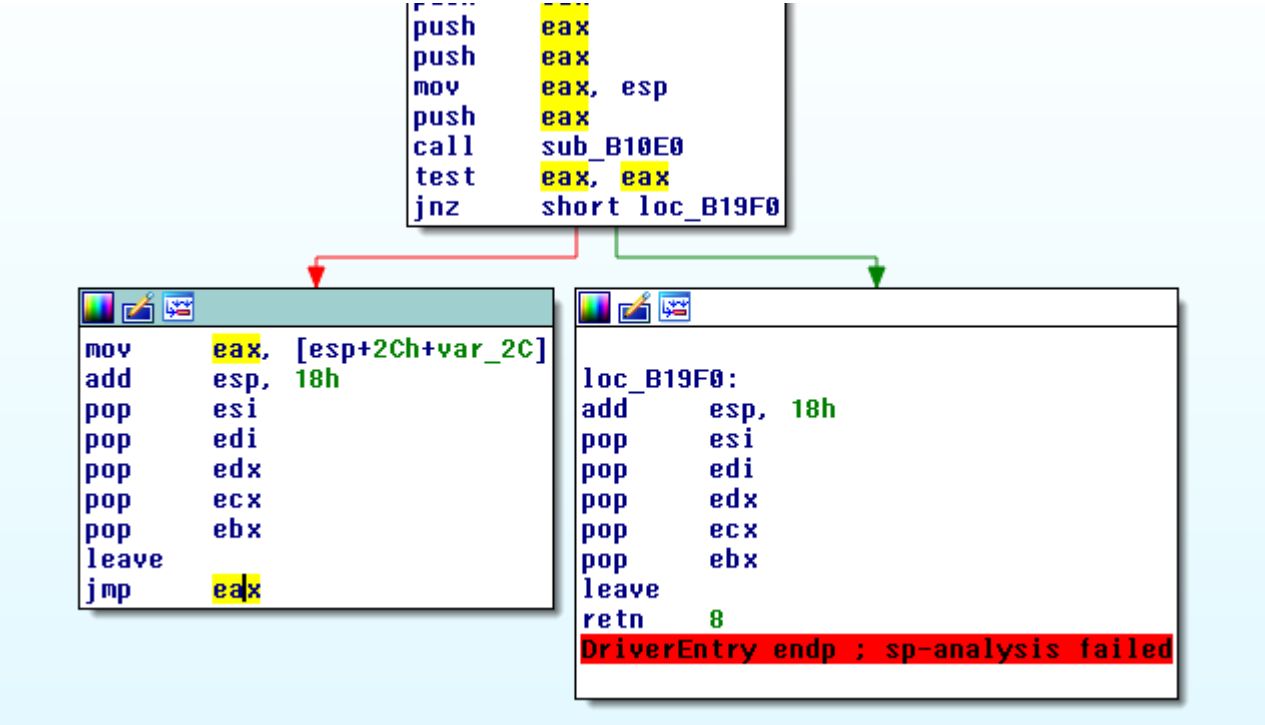


UROBUROS PACKED DRIVER

- Breakpoint on the entry of point of the driver
  - `kd> bu mybaddriver!DriverEntry`
    #if it does not work (only on Windows 7): **`nt!IopLoadDriver+0x66a`**
  - `kd> g`
- Output
  -
```
kd> g
Breakpoint 0 hit
nt!IopLoadDriver+0x66a:
80576988 ff572c          call    dword ptr [edi+2Ch]
kd> t
f4f809b0 8b0424          mov     eax,dword ptr [esp]
```
- `0xf4f809b0` is the address of `DriverEntry`

UROBUROS PACKED DRIVER

- Next step breakpoint at test eax, eax



- kd> **bp f4f809b0+0x2E**
  kd> **g**
  Breakpoint 2 hit
  f4f809de 85c0 test eax,eax

UROBUROS PACKED DRIVER

- Registry values
  - kd> **r edi**
    edi=f4f85000
    kd> **da edi**
    f4f85000 "MZ."

    kd> **!dh edi**
    File Type: DLL
    FILE HEADER VALUES
         14C machine (i386)
           5 number of sections
    5192150F time date stamp Tue May 14 12:42:23 2013
           0 file pointer to symbol table
           0 number of symbols
          E0 size of optional header
        2102 characteristics
                Executable
                32 bit word machine
                DLL
    […]

UROBUROS PACKED DRIVER

- Dump the unpack driver
  - `kd> .writemem C:\\unpacked.sys edi edi+9D000-1`
    `Writing 9D000 bytes.....................................`
- 0x9d000 is the value of SizeOfImage in the PE header
  - `paul@gdata$ strings -a unpacked.sys | grep -i ur0`
    `Ur0bUr()sGoTyOu#`

CONCLUSION

- Thanks for your attention.
- Questions or awkward silence?