

# Herramientas de Análisis

## Introducción

*Revisión de conceptos*  
*Laboratorios de malware*

Course

**Análisis y Detección de Malware**

Instructor

**Acosta Bermejo Raúl**

Lecture notes

**2024-B**  
7 de noviembre del 2024  
Última actualización



# Table of contents (outline)

## Tabla de contenido

1. Introducción
2. Archivos ejecutables
3. Herramientas de análisis
4. Des-ensamblado
5. Volcado de memoria
6. Conclusiones





# Introduction

## Introducción

Motivación





# Introduction

## Introducción

### Definiciones (algunas)

- Des-ensamblar / Disassembly, Disassemblers
- De-compilar / Decompilers
- URL
  - Lista de herramientas

[https://en.wikibooks.org/wiki/X86\\_Disassembly/Disassemblers\\_and\\_Decompile](https://en.wikibooks.org/wiki/X86_Disassembly/Disassemblers_and_Decompile)





# Estructura de Ejecutables

## PE, ELF, Mach-O

Conceptos, herramientas.





# Herramientas de análisis

## Introducción

### Ejemplos

- Web  
<https://elfy.io/>
- GUI
- CLI: readelf (linux)
- API

### Librerías

C++, <http://elfio.sourceforge.net/>  
C#, <http://elfsharp.hellsgate.pl>

### ELF parser

<http://www.elfparser.com>

The screenshot shows two windows side-by-side. The left window is a simple application titled 'Score' with a large digital-style display showing '120'. Below it are four buttons: 'Open', 'Reset', 'About', and 'Close'. The right window is titled 'ELF Parser' and contains a detailed analysis of an ELF file. It has tabs for Overview, ELF Header, SHeaders, PHeaders, Symbols, Capabilities, and Scoring. The Scoring tab is active, showing a table with categories like Random Functions, Process Manipulation, Pipe Functions, and Network Functions, each with a list of found functions such as accept(), bind(), connect(), gethostbyname(), gethostbyname\_r(), inet\_addr(), listen(), recv(), sendto(), and socket(). Other tabs like ELF Header show binary data hex dump.



# Herramientas de análisis

## Introducción

### Ejemplos

- Web

<https://malyzer.org/>

- GUI

Peviwer, PEStudio

- CLI

<https://github.com/erocarrera/pefile>

<http://pev.sourceforge.net/>

- API

Librerías

– <https://pypi.org/project/pefile/>

The screenshot shows two windows side-by-side. The top window is 'PeStudio 8.28 - Windows Executable Scoring - www.winitor.com'. It displays a tree view of file indicators under 'c:\documents and settings\admin' and a large list of descriptive text on the right. The bottom window is 'PEview - C:\Documents and Settings\Administrator\Desktop\0170.dll'. It shows a navigation toolbar at the top, followed by a tree view of section headers like IMAGE\_SECTION\_HEADER, SECTION .text, and SECTION .rdata. Below the tree is a table with columns: pFile, Data, Description, and Value. The table lists several function RVA entries.

pFile	Data	Description	Value
0000C168	0000378E	Function RVA	0001
0000C16C	0000378F	Function RVA	0002
0000C170	00003C6B	Function RVA	0003
0000C174	00004EA9	Function RVA	0004
0000C178	00004BE9	Function RVA	0005
0000C17C	00004FF8	Function RVA	0006
0000C180	00004EAA	Function RVA	0007



# Herramientas de análisis

## Introducción

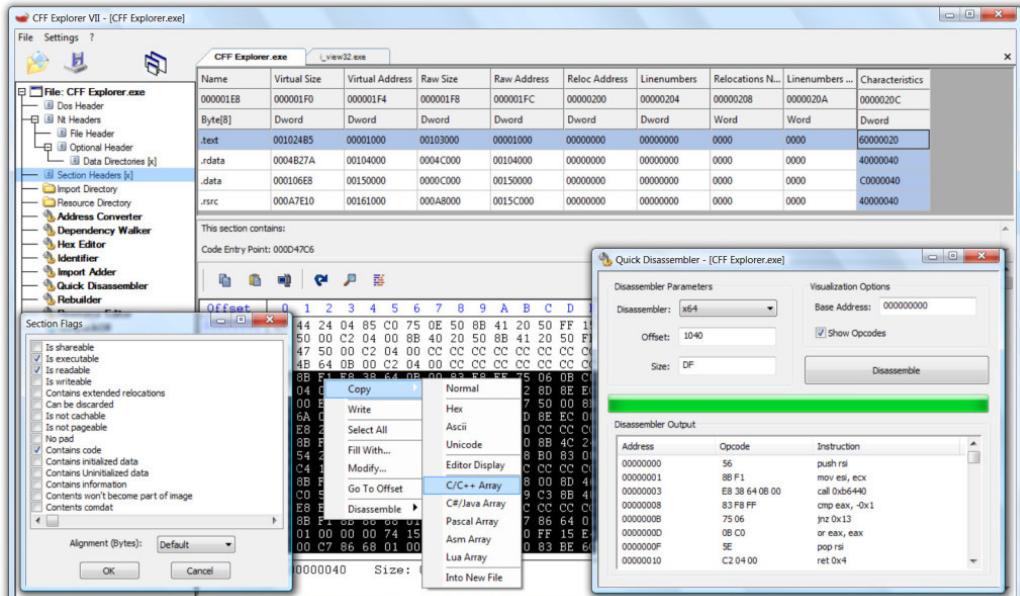
### Ejemplos

- **GUI**

Varias herramientas para realizar el análisis de archivos en Windows.

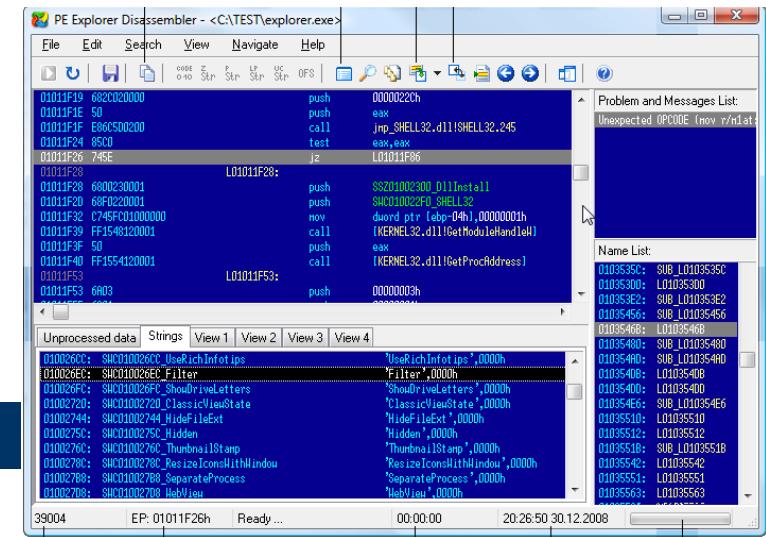
Explorer Suite, CFF  
Freeware

- Permite editar el PE
  - Encabezado o su contenido.
- Link
  - <https://ntcore.com/>



# Des-ensamblador

## PE Explorer's disassembler



- It tried to achieve most of the power of IDA Pro, while requiring less skill or knowledge, by automating more of the disassembly process.
- The disassembler supports the most common Intel x86 instruction sets and extensions (MMX, 3D Now!, SSE, SSE2 and SSE3).
- The PE Explorer disassembler assumes that some manual editing of the reproduced code will be needed. To facilitate additional hand coding, however, the disassembler utilizes a qualitative algorithm designed to reconstruct the assembly language source code of target binary win32 PE files (EXE, DLL, OCX) with the highest degree of accuracy possible.
- It runs on: Windows 98/ME/NT/2000/XP/Vista/7/8.

## Links

- [http://www.heaventools.com/PE\\_Explorer\\_Disassembler.htm](http://www.heaventools.com/PE_Explorer_Disassembler.htm).





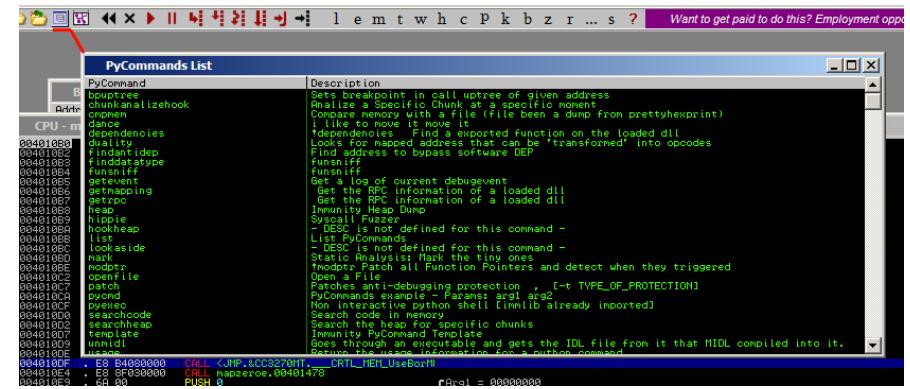
# Des-ensamblador

## Immunity Debugger

- Immunity Debugger is a branch of OllyDbg v1.10, with built-in support for Python scripting and much more.
- Immunity Debugger is a powerful new way to write exploits, analyze malware, and reverse engineer binary files. It builds on a solid user interface with function graphing, the industry's first heap analysis tool built specifically for heap creation, and a large and well supported Python API for easy extensibility.

### Links

- <http://www.immunityinc.com/products/debugger/>.



# Des-ensamblador

## Hiew

The screenshot shows the Hiew debugger interface. The main pane displays assembly code for a 64-bit x86 processor. The registers pane shows the current state of CPU registers. The stack pane shows the stack contents. The memory dump pane shows memory dump information. A status bar at the bottom provides various file and command-related details.

- It views and edits files of any length in text, hex, and decode modes(x86-64 disassembler & assembler (AVX instructions include))
- Physical & logical drive view & edit
- Support for NE, LE, LX, PE/PE32+, ELF/ELF64(little-endian), Mach-O(little-endian) executable format
- Support for Netware Loadable Modules like NLM, DSK, LAN,...
- Following direct call/jmp instructions in any executable file with one touch
- Pattern search in disassembler
- Built-in simple 64bit decrypt/crypt system.
- Unicode support.
- ArmV6 disassembler.

### Links

- [http://www.hiew.ru/.](http://www.hiew.ru/)





# Análisis de Ejecutables

## Varios elementos

Conceptos, herramientas.





# Introduction

## Herramientas

Son programas especializados en el análisis de archivos binaries, por ejemplo, de archivos ejecutables, librerías, imágenes, etc.

Para malware la mayoría se centra en archivos ejecutables y su manipulación (ej, depuración):

- |    |                               |     |
|----|-------------------------------|-----|
| 1. | IDA pro, comercial y gratuito |     |
| 2. | Radare2, gratuito             | GUI |
| 3. | x64dbg                        | CLI |
| 4. | OllyDbg                       | API |
| 5. | ODA                           |     |
| 6. | Hooper, sólo para MacOSX      |     |



# Herramientas de análisis

## Introducción

### Ejemplos de CLI

- Windows
  - Dumpbin
  - <https://docs.microsoft.com/en-us/cpp/build/reference/dumpbin-reference?view=msvc-170>
- Linux
  - Readelf
  - <https://man7.org/linux/man-pages/man1/readelf.1.html>





# YASCA (Yet Another Source Code Analyzer)

## Descripción

- It is an open source program which looks for security vulnerabilities, code-quality, performance, and conformance to best practices in program source code.
- It leverages external open source programs, such as FindBugs, PMD, JLint, JavaScript Lint, PHPLint, Cppcheck, ClamAV, Pixy, and RATS to scan specific file types, and also contains many custom scanners developed for Yasca.
- It is a command-line tool that generates reports in HTML, CSV, XML, MySQL, SQLite, and other formats.
- It is listed as an inactive project since 2010.

### Link

- <https://en.wikipedia.org/wiki/Yasca>
- <http://www.scovetta.com/yasca.html>



# Des-ensamblador

## IDA

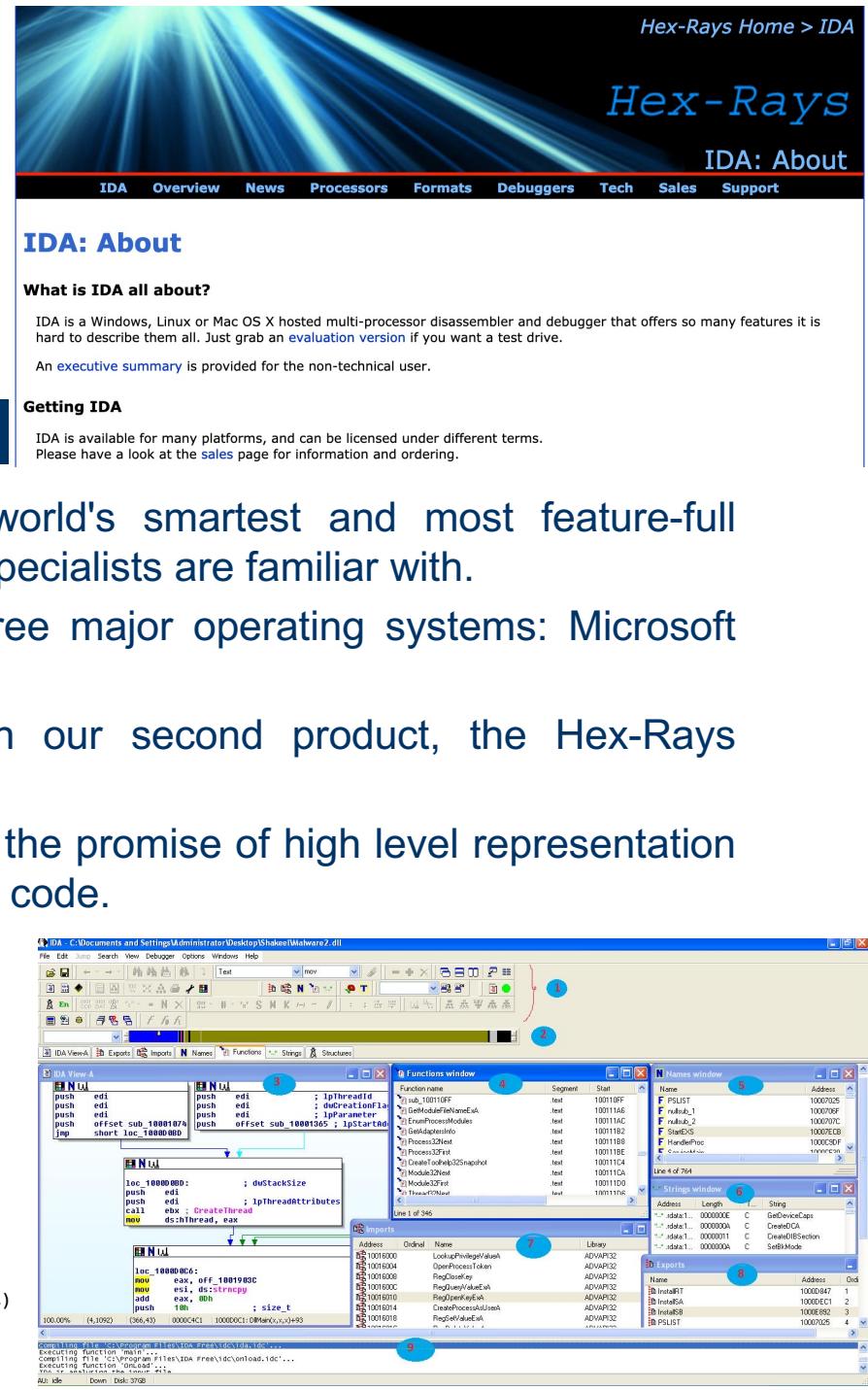
- IDA is the Interactive DisAssembler: the world's smartest and most feature-full disassembler, which many software security specialists are familiar with.
- Written entirely in C++, IDA runs on the three major operating systems: Microsoft Windows, Mac OS X, and Linux.
- IDA is also the solid foundation on which our second product, the Hex-Rays decompiler, is built.
- The unique Hex-Rays decompiler delivers on the promise of high level representation of binary executables. It can handle real world code.

### Links

– <https://www.hex-rays.com/index.shtml>

En noviembre del 2019 se adquirió:

LICENSE	PRODUCT	NSEATS	SHA-1 DIGEST
48-B675-7A24-0C	IDAPROFL	1-user	511904fd95f43babb367a75d6aa316025cda741b (x64)





# Des-ensamblador

## IDA

### Scripts

- IDC
  - <https://hex-rays.com/products/ida/support/tutorials/idc/>
- IDApython
  - <https://github.com/tmr232/idapython>
- CLI
  - <https://www.hex-rays.com/products/ida/support/idadoc/417.shtml>
- Tutoriales
  - <https://programmerclick.com/article/64621916607/>





# Des-ensamblador

# x64dbg

- x64dbg is an **open source** that can debug both **x64** and **x32** applications for **Windows**.

## Links

- <http://x64dbg.com/>
  - <https://sourceforge.net/projects/x64dbg/files/snapshots/.>

The screenshot shows the Immunity Debugger interface with the assembly, registers, and stack panes visible. The assembly pane displays assembly code for the x32\_dbg module, specifically the dummycrackme.exe thread. A red box highlights the initial instructions, and a red arrow points from the assembly to the registers pane. The registers pane shows CPU register values, and the stack pane shows memory starting at 0x001FFC.



# Des-ensamblador

# OllyDbg

- It is a 32-bit assembler level analysing debugger for Microsoft® Windows®.
  - Emphasis on binary code analysis makes it particularly useful in cases where source is unavailable.
  - OllyDbg is a shareware, but you can download and use it for free.
  - Version 2.01 (27/sept/2013)

## Links

– <http://www.ollydbg.de/>.



# Des-ensamblador

## Radare2

- Radare2 is an open source tool to disassemble, debug, analyze and manipulate binary files.
- It actually supports many architectures:  
x86{16,32,64}, Dalvik, avr, ARM, java, PowerPC, Sparc, MIPS.
- Several binary formats:  
pe{32,64}, [fat]mach0{32,64}, ELF{32,64}, dex and Java classes
- Apart from support for filesystem images and many more features.

### Links

- <http://radare.org/r/>.
- **Book:** <https://book.rada.re/>
- **Wiki:** <https://r2wiki.readthedocs.io/en/latest/>





# Des-ensamblador

## Radare2

### Radare2 GUI

- Cutter
- PyGTK

Screenshot of the Radare2 GUI showing the Cutter interface. The left pane displays a list of functions and symbols, with 'sym.Aeropause' selected. The main pane shows the assembly code for the 'Aeropause' function. A call graph for 'sym.Aeropause' is shown in the center, with specific nodes highlighted in red. The bottom pane contains a console window with the message: '[0x8884924c] ?E Welcom to Cutter!'.

21

Screenshot of the Radare2 GUI showing the Hexdump and Decompiler interfaces. The top pane shows the assembly code for the 'Aeropause' function. The middle pane shows the decompiled C code for the same function. The bottom pane shows the hex dump of the memory at address 0x88849200. A red box highlights a section of assembly code: 'mov eax, dword [Bright]; mov eax, dword [eax]; mov eax, dword [eax + 4]; mov eax, dword [Bright]; mov eax, dword [eax + 4]; mov eax, dword [eax + 4];'. A green arrow points from this code to the corresponding decompiled C code: 'if (Bright->ambassador == AMBASSADOR\_REASON)'.



# Des-ensamblador

## ODA

- A lightweight, online service that explores executables by dissecting its sections, strings, symbols, raw hex and machine level instructions.
- It supports over 60 machine architectures, including x86, ARM, PowerPC, MIPS and many more.
- All major file formats supported, including Windows PE, ELF, COFF, SREC, Mach-O, and more.
- Graph View visually shows the control flow of the currently selected function.

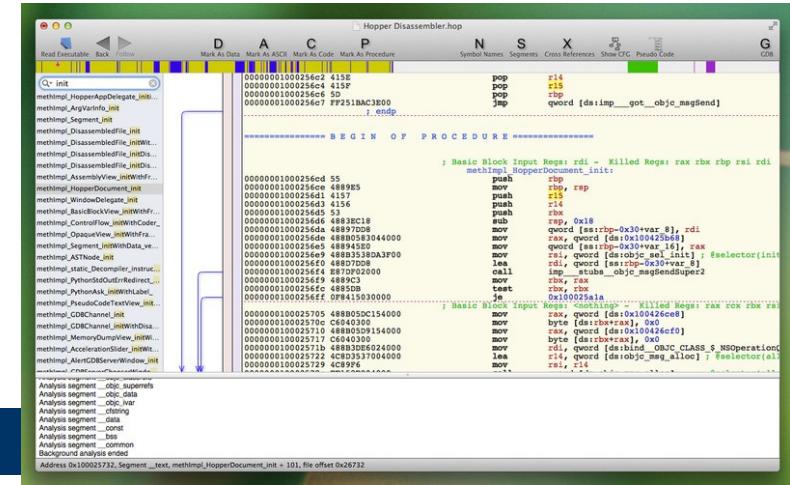
### Links

– <https://www.onlinedisassembler.com/>.

The screenshot shows the Onlinedisassembler (ODA) web application. At the top, there's a navigation bar with links for 'Share', 'File', 'Edit', 'Examples', 'Help', 'Blog', and 'Contact Us'. On the right side of the header, there are 'Sign In' and 'Sign Up' buttons. Below the header, there are tabs for 'Symbols', 'Disassembly' (which is selected), 'Graph', 'Hex', 'Sections', and 'File Info'. The main area contains two panes. The left pane is a 'Symbols' table with columns for 'Address', 'Type', and 'Name'. It lists several symbols such as '\_init', '\_ptt', 'func\_00401170', 'func\_00401180', 'func\_00401190', 'func\_004011a0', 'func\_004011b0', 'func\_004011c0', 'func\_004011d0', 'func\_004011e0', 'func\_004011f0', 'func\_00401200', 'func\_00401210', 'func\_00401220', 'func\_00401230', 'func\_00401240', 'func\_00401250', 'func\_00401260', and 'func\_00401270'. The right pane is the 'Disassembly' view, which shows assembly code for a function. The code includes instructions like push rbx, mov rbp, mov QWORD PTR [rbp-0x18], and various jumps and comparisons. To the right of the assembly code, there is a detailed 'FUNCTION' section with information about code references and local variables. A vertical graph on the right side visualizes the control flow between the listed symbols.

# Des-ensamblador

## Hopper



- Hopper is a reverse engineering tool for OS X and Linux, that lets you disassemble, and decompile your 32/64bits Intel Mac, Linux, Windows and iOS executables.
- Release 3 (13/ago/2016), Commercial and Demo with limitations.
- The idea behind Hopper is to transform a set of bytes (the binary you want to analyze) into something that could be read by a human. To do so, Hopper will try to **associate a type to each byte of the file**. Because it would be too much expensive to do it manually, Hopper proceeds to an automatic analysis as soon as you loaded a file. The various types that can be used in hopper are:
  - **Data**: an area is set to the *data* type when Hopper thinks it is an area that represents a constant, like an array of *int* for instance.
  - **ASCII**: a NULL terminated C string.
  - **Code**: an instruction
  - **Procedure**: a byte receives this type once it has been determinate that it is part of a method that has been successfully reconstructed by Hopper.
  - **Undefined**: this is an area that has not yet been explored by Hopper.

## Links

- <https://www.hopperapp.com/>





# Des-ensamblador

## Capstone

The current version is 4.0.2, which was released on May 8, 2020

- Capstone is a lightweight multi-platform, multi-architecture disassembly framework. Our target is to make Capstone the ultimate disassembly engine for binary analysis and reversing in the security community.
- Highlight features
  - Multi-architectures: Arm, Arm64 (Armv8), BPF, Ethereum Virtual Machine, M68K, M680X, Mips, MOS65XX, PowerPC, RISCV, Sparc, SystemZ, TMS320C64X, Web Assembly, XCore & X86 (include X86\_64).
  - Clean/simple/lightweight/intuitive architecture-neutral API.
  - Provide details on disassembled instruction (called “decomposer” by some others).
  - Provide some semantics of the disassembled instruction, such as list of implicit registers read & written.
  - Implemented in pure C language, with bindings for D, Clojure, F#, Common Lisp, Visual Basic, PHP, PowerShell, Haskell, Perl, Python, Ruby, C#, NodeJS, Java, GO, C++, OCaml, Lua, Rust, Delphi, Free Pascal & Vala available.
  - Native support for Windows & \*nix (with Mac OSX, iOS, Android, Linux, \*BSD & Solaris confirmed).
  - Thread-safe by design.
  - High performance & suitable for malware analysis (capable of handling various X86 malware tricks).
  - Distributed under the open source [BSD license](#).

24

### Links

- <https://www.capstone-engine.org/>





# Des-ensamblador

## Comando

### Uso de CLIs

- Linux
  - Comando objdump.
  - \$ objdump -d /bin/ls
- Windows
  - Comando que forma parte del SDK
  - \$ dumpbin /disasm





# Des-ensamblador

## API

### Uso de APIs

- IDA pro

- Tiene varios plugins y uno permite hacer el desensamblado con Python.
- Requiere que IDA pro esté corriendo todo el tiempo.
- Links
  - [https://www.hex-rays.com/products/ida/support/idapython\\_docs/](https://www.hex-rays.com/products/ida/support/idapython_docs/)
  - <https://sites.utexas.edu/iso/2016/03/24/reverse-engineering-necurs-part-4-ida-pros-python-api/>
  - **Lista de plugins:** <https://github.com/onethawt/idaplugins-list>



# Des-ensamblador

## API

### Uso de APIs

- Capstone
  - The next generation disassembly framework!
  - Usa el toolchain LLVM.
  - Links
    - <https://www.capstone-engine.org/>.
    - Desensamblado: <https://www.capstone-engine.org/BHUSA2014-capstone.pdf>

Hay varios  
problemas  
en el  
desensamblado

Features	Distorm3	BeaEngine	Udis86	Libopcode
X86 Arm	✓ X	✓ X	✓ X	✓ ✓ <sup>1</sup>
Linux Windows	✓ ✓	✓ ✓	✓ ✓	✓ X
Python Ruby bindings	✓ X <sup>2</sup>	✓ X	✓ X	✓ X
Update	X	?	X	X
License	GPL	LGPL3	BSD	GPL



# Des-ensamblador

## API

### Capstone

- 28 Ejemplos
  - ❖ <https://www.programcreek.com/python/example/102929/capstone.Cs>
- Ejemplos completos
  - ❖ <https://isleem.medium.com/create-your-own-disassembler-in-python-pefile-capstone-754f863b2e1c>





# Des-ensamblador

## API

### Otras herramientas

- Una lista larga en:
  - <http://www.capstone-engine.org/showcase.html>
  - <https://reverseengineering.stackexchange.com/questions/1817/is-there-any-disassembler-to-rival-ida-pro>
- Pywe
  - <http://joxeankoret.com/blog/2010/02/08/pyew-a-python-tool-to-analyze-malware/>
  - <https://github.com/joxeankoret/pyew>
- ImmunityDbg, similar a IDA pro usando python
  - <http://www.immunityinc.com/products/debugger/>





# Des-ensamblador

## API

Cada semestre ...

Se encuentran nuevas herramientas, por ejemplo:

2024-B

- Standalone App
- <https://gchq.github.io/CyberChef/>

The screenshot shows the CyberChef web application interface. On the left, there is a sidebar with a list of operations categorized under 'Operations'. The visible categories include 'Favourites' (with a star icon), 'Data format', 'Encryption / Encoding', 'Public Key', 'Arithmetic / Logic', 'Networking', and 'Language'. Under 'Language', the 'Python' option is selected. The main workspace is divided into 'Input' and 'Output' panes. The 'Input' pane contains a single line of text: 'sec 0 | F 1'. The 'Output' pane also contains a single line of text: 'sec 0 | F 1'. At the bottom center is a green button labeled 'BAKE!' with a chef icon. To its right is a checked checkbox labeled 'Auto Bake'. The top of the page has a navigation bar with links for 'Download CyberChef', 'Last build: 15 days ago - Version 10 is here! Read about the new features here', 'Options', 'About / Support', and other icons.





# Volcado de memoria

## Toda o por proceso

Conceptos, herramientas.





# Volcado de memoria

## Herramientas



### Volatility ver. 2.6, 3 v1.0.0 (Python 3 Rewrite)

- It is a command line memory analysis and forensics tool for extracting artifacts from memory dumps.
- It has significant in-depth research into OS internals, applications, malicious code, and suspect activities. In general, has many forensics capabilities.
- Operating System Support: 19 types
  - 10 for windows: Win Server, Win 10, ..., WinXP.
  - 1 for Linux: 32- and 64-bit Linux kernels from 2.6.11 to 4.2.3
  - 8 for MacOS
- Memory Format Support: 10 types
  - Virtualbox Core Dumps
  - VMware Saved State (.vmss) and Snapshot (.vmsn)
  - QEMU memory dumps.

### Links

- [www.volatilityfoundation.org](http://www.volatilityfoundation.org)





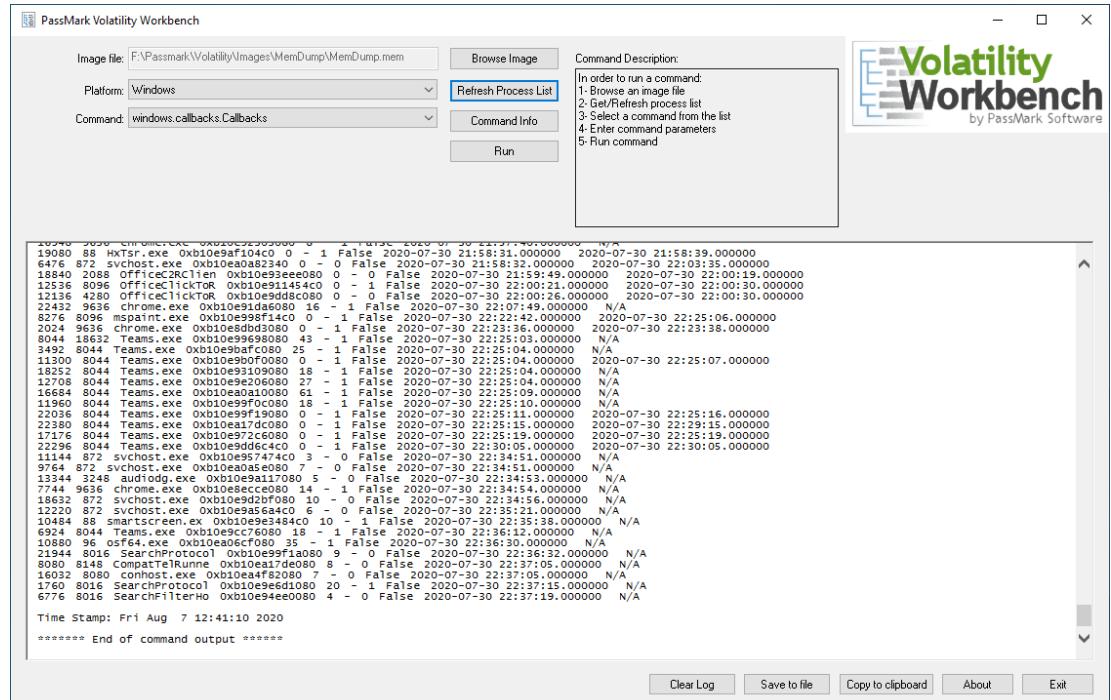
# Volcado de memoria

## Herramientas

### Volatility Workbench

- It is a GUI for the Volatility tool.
- It is free, open source and runs in Windows.
- It provides a number of advantages over the command line version including: ...

Ver sitio.



### Links

- [www.osforensics.com/tools/volatility-workbench.html](http://www.osforensics.com/tools/volatility-workbench.html)



# Volcado de memoria

## Herramientas



**Rekall**  
Forensics

### Rekall

- It is an advanced forensic and incident response framework.
- **Rekall Agent** is a complete endpoint incident response and forensic tool. The Rekall Agent extends Rekall's advanced capabilities to a scalable, distributed environment. The Rekall Agent is easy to deploy and scale, based on modern cloud technologies.
- Characteristics:
  - It locates significant kernel data structures.
  - It maintains the largest public profile repository for many operating system versions.
  - While other tools rely on heuristics and signatures.

### Links

- <http://www.rekall-forensic.com/>



# The end

## Contacto

Raúl Acosta Bermejo

<http://www.cic.ipn.mx>

<http://www.ciseg.cic.ipn.mx/>

[racostab@ipn.mx](mailto:racostab@ipn.mx)

[racosta@cic.ipn.mx](mailto:racosta@cic.ipn.mx)

57-29-60-00

Ext. 56652





# Práctica de Análisis

## Archivos binarios

Por equipo





# Práctica

## Archivos binarios

### Descripción

- Obtener la información de varios archivos binaries (6)
  - Obtener información general:
  - Desensamblar con 3 herramientas: gui, cli, api.
- Cada alumno debe analizar:
  - Archivo(s) malware (dentro de cuckoo) y archivo(s) benigno.
- Analizar archivos de tipo: librería, ejecutable del sistema
- Realizar un **reporte** por equipo
  - Pueden determiner con que compilador se creo?
  - Cuando se compiló? Que librerias usa?
  - Que diferencias hay entre los ensambladores?
  - Que procesador require y cuántos bits?, Etc.