XIII JORNADAS STIC CCN-CERT

ccn-cert
centro criptológico nacional

Taller de reversing:
"Introducción a Ghidra"

COMUNIDAD Y CONFIANZA,
BASES DE NUESTRA CIBERSEGURIDAD

#XIIIJORNADASCCNCERT

Ricardo J. Rodríguez

- **Líneas de investigación:**
  - Análisis binario de aplicaciones
  - Análisis forense de memoria
  - Seguridad en RFID/NFC

Universidad Zaragoza
1542

e: rjrodriguez@unizar.es
w: https://webdiis.unizar.es/~ricardo
@RicardoJRdez

Índice

1. Introducción

2. Conociendo la herramienta Ghidra

3. Ejemplos de uso

4. Conclusiones

```
ricardo@freyja:~/ghidra$ find . -name "*.java" | xargs grep -r "@version"
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/lang/UnknownInstructionException.java: * @version 2000-02-15
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/lang/InsufficientBytesException.java: * @version 2000-02-15
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/lang/UnknownContextException.java: * @version 2000-02-15
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/lang/UnknownDataException.java: * @version 2000-02-15
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/address/AddressIterator.java: * @version 2000-02-16
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/address/AddressOutOfBoundsException.java: * @version 1999-03-31
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/address/AddressRangeIterator.java: * @version 2000-02-16
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/scalar/ScalarOverflowException.java: * @version 1999-03-31
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/scalar/ScalarFormat.java: * @version 1999/02/04
./Ghidra/Framework/SoftwareModeling/src/main/java/ghidra/program/model/mem/MemoryAccessException.java: * @version 1999-03-31
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/LongArrayArrayTest.java: * @version
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/StringArrayArrayTest.java: * @version
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/IntArrayArrayTest.java: * @version
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/ArrayTest.java: * @version
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/ByteArrayArrayTest.java: * @version
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/DataTableTest.java: * @version
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/StringArrayTest.java: * @version
./Ghidra/Framework/Generic/src/test/java/ghidra/util/datastruct/ShortArrayArrayTest.java: * @version
./Ghidra/Framework/Generic/src/main/java/ghidra/util/exception/NotYetImplementedException.java: * @version 1999/02/05
./Ghidra/Features/Base/src/test/java/ghidra/framework/options/SaveStateTest.java: * @version 1.0
./Ghidra/Features/Base/src/test/java/ghidra/program/model/address/AddressObjectMapTest.java: * @version
```

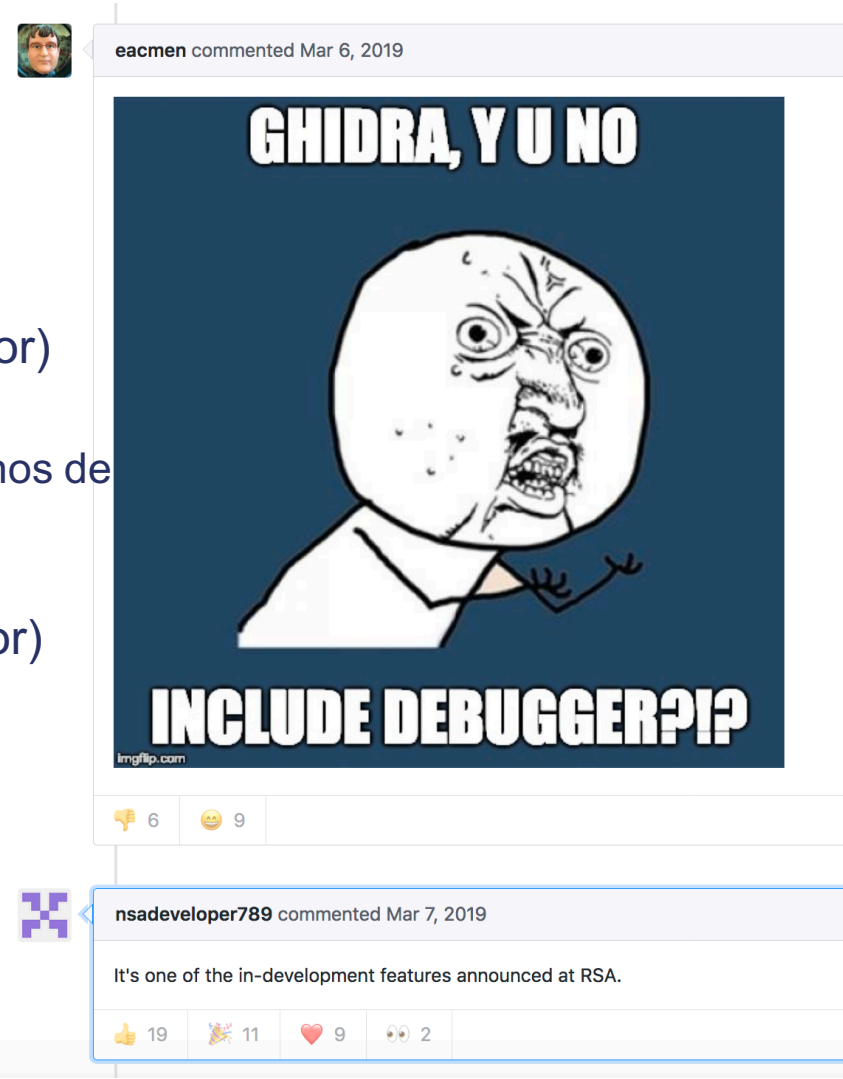
- **Código abierto**, liberado en conferencia RSA 2019
  - Desarrollado por la NSA
  - Licencia Apache 2.0
  - +1.2M SLOC
  - La historia de la herramienta no se conoce, pero comentarios del código la datan en febrero 1999

memeguy.com

1. Introducción

Características de Ghidra

- *Aproximaciones en el análisis de un programa binario*
  - **Análisis estático**: código muerto / análisis frío (desensamblador)
    - El código binario no se ejecuta
    - Se analiza la estructura del código, con todos los posibles caminos de ejecución.
    - **PROBLEMA**: **Explosión de estados**
  - **Análisis dinámico**: código vivo / análisis en caliente (depurador)
    - Código binario en ejecución
    - Sólo se analiza un único camino (el que se está ejecutando)
    - **PROBLEMA**: **Completitud del análisis**



- Actualmente, **Ghidra sólo soporta análisis estático**
  - La posibilidad de depuración está en desarrollo

1. Introducción

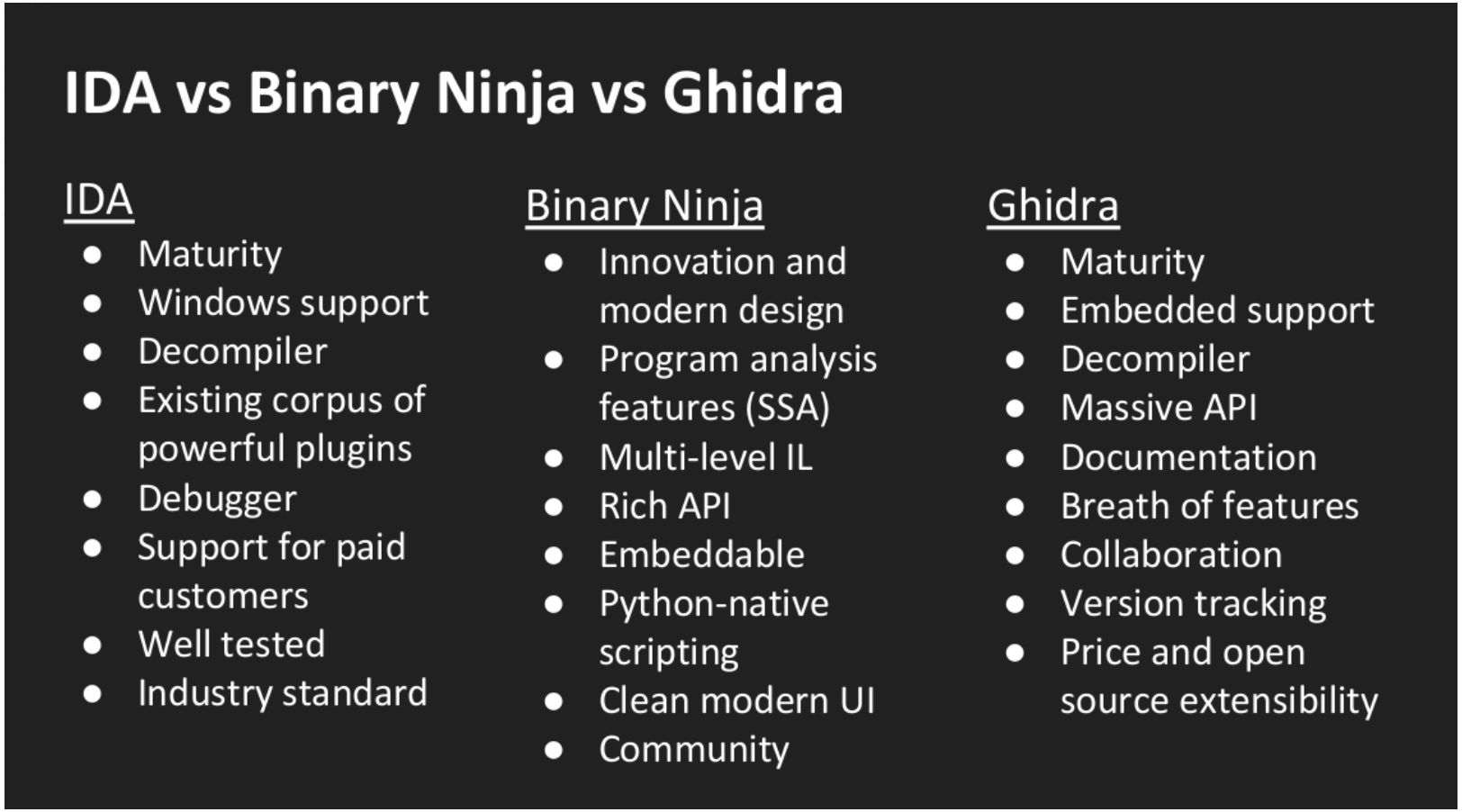
Ghidra comparada con otras SREs

Ghidra / También se buscó

- Faltaría BINARY NINJA en esta imagen (y sobran otros, claro…)

1. Introducción

Ghidra comparada con otras SREs



Créditos: "Three Heads Are Better Than One: Mastering NSA's Ghidra Reverse Engineering Tool", Alexei Bulazel, Jeremy Blackthorne, INFILTRATE Con 2019

1. Introducción

Ghidra comparada con otras SREs

Decompiler - IDA Hex-Rays vs Ghidra

IDA Hex-Rays

- Optional add-on for IDA for IDA
- Microcode-based
- Supports limited architectures
- Better built-in support for Windows
- Variables, data, and functions can be xrefed from decompiler
- Variables can be mapped
- Variable representation can be changed in the decompiler (decimal, hex, char immediate, etc)
- Click to highlight

Ghidra Decompiler Decompiler

- Deeply integrated with Ghidra
- P-code based
- Supports all architectures
- No way to xref from decompiler
- Produces fewer `goto` statements and seemingly more idiomatic C
- Built in program analysis features, e.g., slicing and data flow
- Variables cannot be mapped
- Variable representation cannot be changed in the decompiler
- *Middle click* to highlight

Créditos: "Three Heads Are Better Than One: Mastering NSA's Ghidra Reverse Engineering Tool", Alexei Bulazel, Jeremy Blackthorne, INFILTRATE Con 2019

1. Introducción

Obteniendo Ghidra

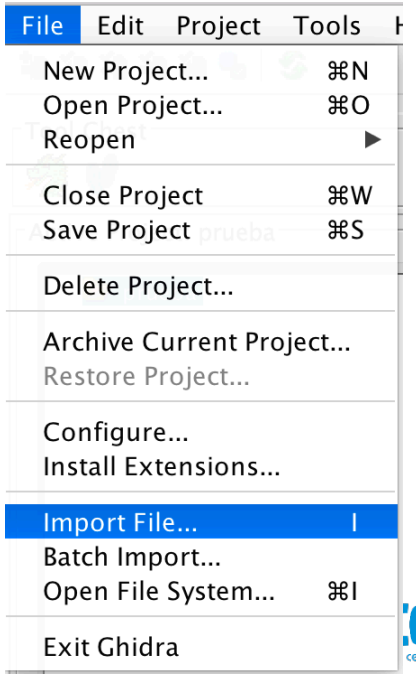
- **Página web oficial**: https://ghidra-sre.org/
  - Código fuente en GitHub: https://github.com/NationalSecurityAgency/ghidra

- Versión actual: 9.1

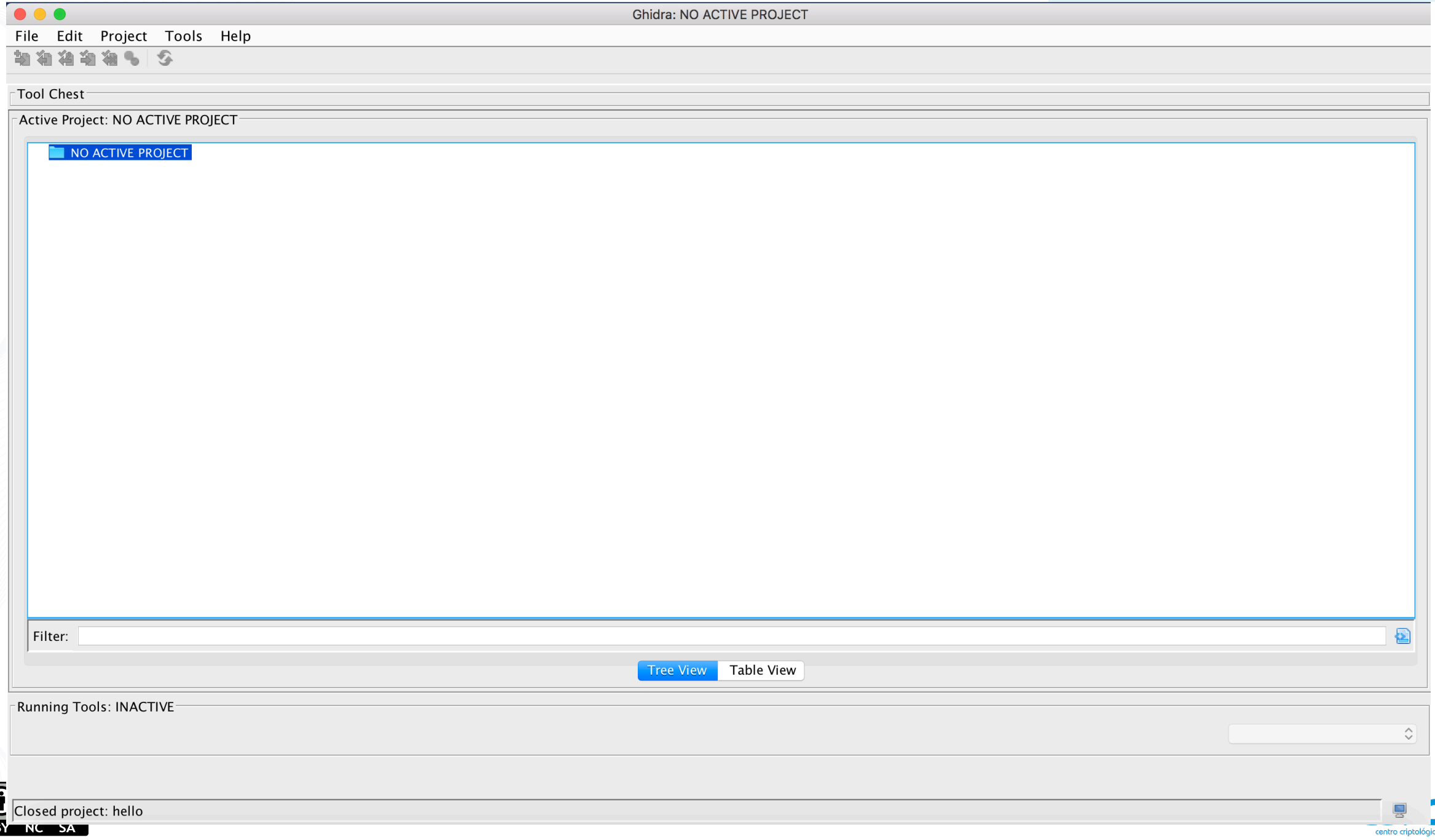
- **Instalación sencilla**:
  1. Descargar ZIP
  2. Descomprimir
  3. Ejecutar desde consola "./ghidraRun" (MacOS X / Linux)
     - **RECUERDA**: Java VM >= 11

2. Conociendo la herramienta Ghidra

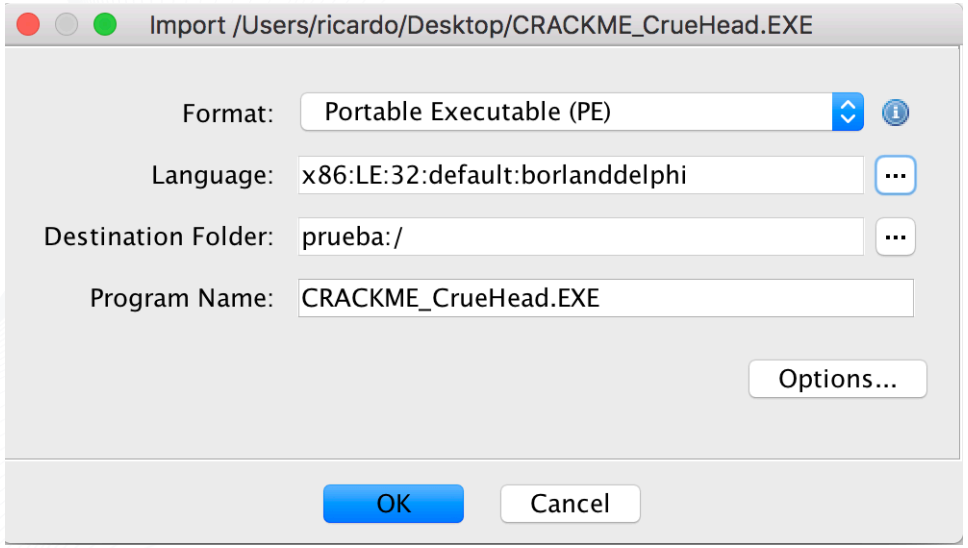
- Gestión de aplicaciones a analizar mediante proyectos
  - Se pueden añadir a un proyecto TODOS los ficheros que correspondan al mismo análisis
  - Permite tener proyectos compartidos (“collaborative reversing”)
    - Similar al plugin DArling del IDAPro
    - Activo por defecto, se ha de configurar
    - Lectura recomendada: https://medium.com/@jannis.kirschner/ghidra-collaborative-reversing-1-2-how-to-setup-a-ghidra-server-711f4212912e
- **Ficheros**: se pueden añadir a cada proyecto uno a uno, o en serie

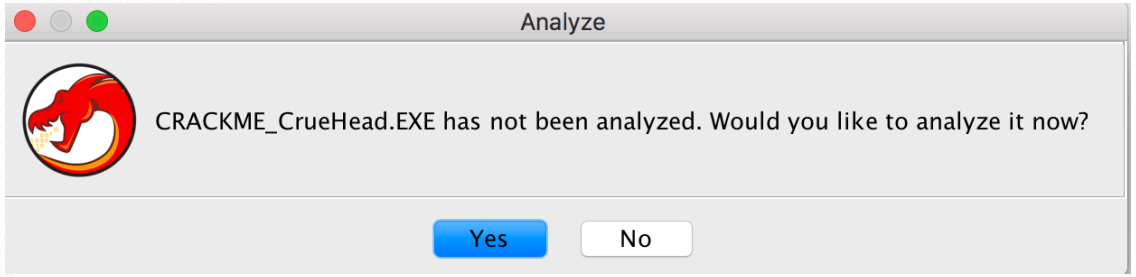


Veamos un ejemplo de la herramienta con un crackME sencillito…

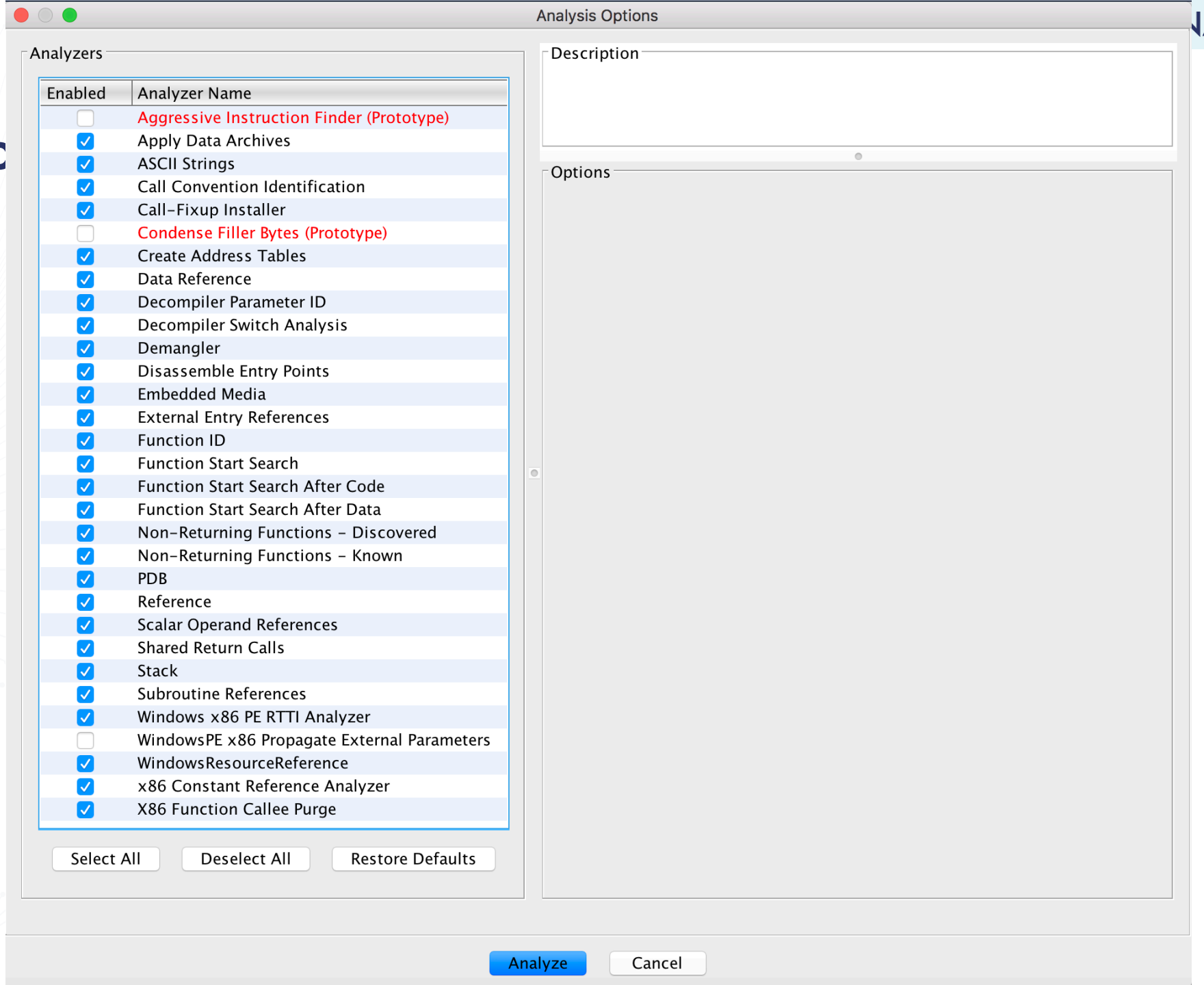

Ghidra: NO ACTIVE PROJECT
File
Edit
Project
Tools
Help
Tool Chest
Active Project: NO ACTIVE PROJECT
NO ACTIVE PROJECT
Filter:
Tree View
Table View
Running Tools: INACTIVE
Closed project: hello

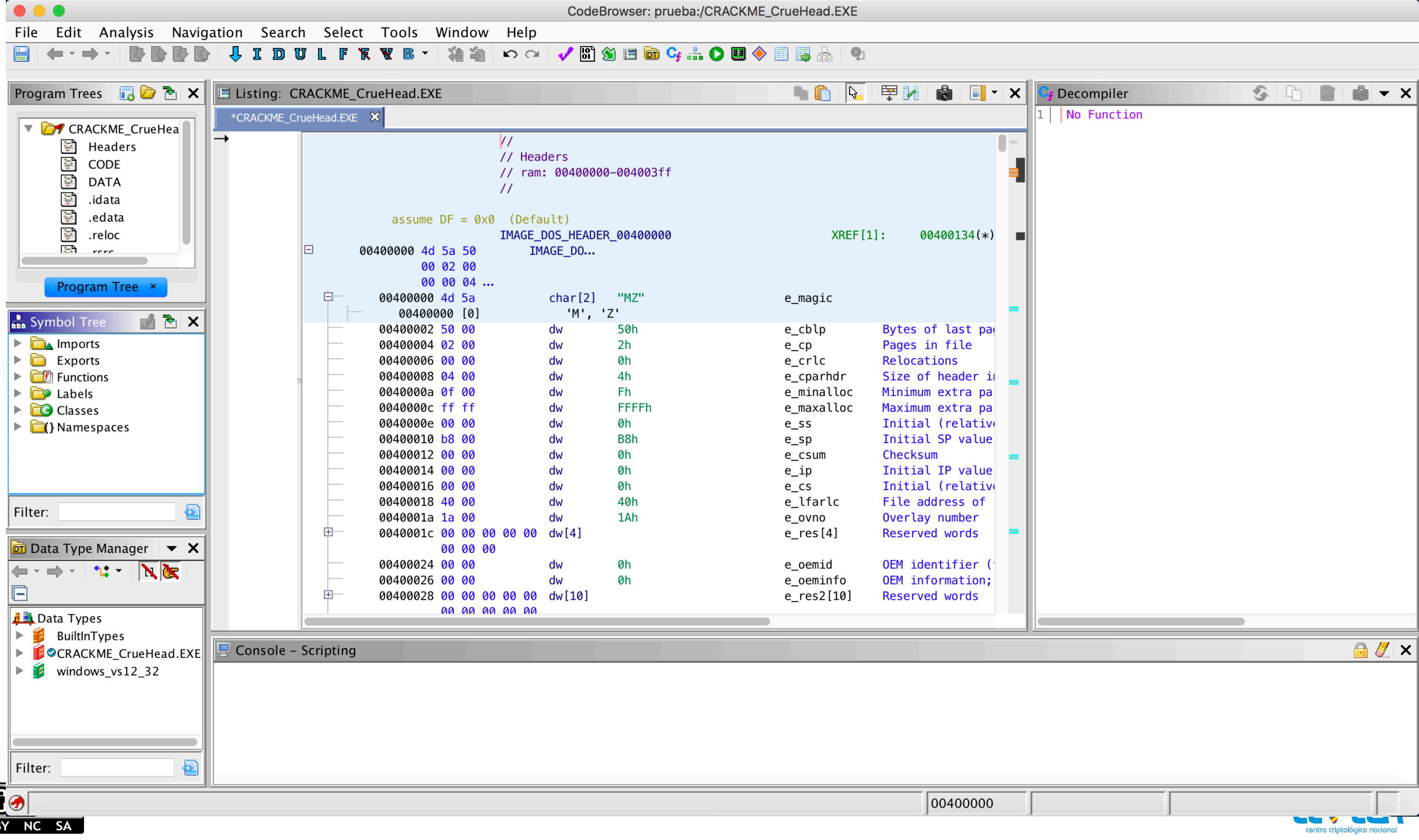
2. Conociendo la herramienta G





2. Conc



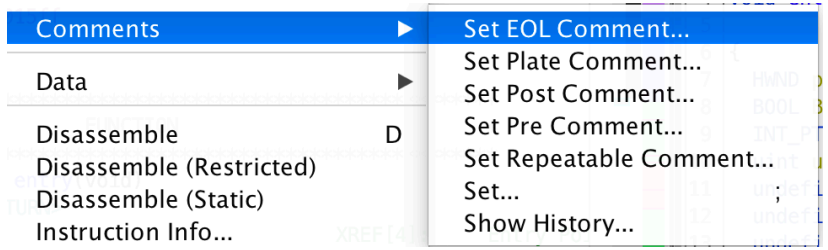

CodeBrowser: prueba:/CRACKME_CrueHead.EXE
File Edit Analysis Navigation Search Select Tools Window Help
Program Trees
CRACKME_CrueHea
Headers
CODE
DATA
.idata
.edata
.reloc
Program Tree
Symbol Tree
Imports
Exports
Functions
Labels
Classes
Namespaces
Filter:
Data Type Manager
Data Types
BuiltInTypes
CRACKME_CrueHead.EXE
windows_vs12_32
Filter:
Listing: CRACKME_CrueHead.EXE
*CRACKME_CrueHead.EXE
//
// Headers
// ram: 00400000-004003ff
//
assume DF = 0x0 (Default)
IMAGE_DOS_HEADER_00400000 XREF[1]: 00400134(*)
00400000 4d 5a 50 IMAGE_DO...
00 02 00
00 00 04 ...
00400000 4d 5a char[2] "MZ" e_magic
00400000 [0] 'M', 'Z'
00400002 50 00 dw 50h e_cblp Bytes of last pa
00400004 02 00 dw 2h e_cp Pages in file
00400006 00 00 dw 0h e_crlc Relocations
00400008 04 00 dw 4h e_cparhdr Size of header i
0040000a 0f 00 dw Fh e_minalloc Minimum extra pa
0040000c ff ff dw FFFFh e_maxalloc Maximum extra pa
0040000e 00 00 dw 0h e_ss Initial (relativ
00400010 b8 00 dw B8h e_sp Initial SP value
00400012 00 00 dw 0h e_csum Checksum
00400014 00 00 dw 0h e_ip Initial IP value
00400016 00 00 dw 0h e_cs Initial (relativ
00400018 40 00 dw 40h e_lfarlc File address of
0040001a 1a 00 dw 1Ah e_ovno Overlay number
0040001c 00 00 00 00 00 dw[4] e_res[4] Reserved words
00 00 00
00400024 00 00 dw 0h e_oemid OEM identifier (
00400026 00 00 dw 0h e_oeminfo OEM information;
00400028 00 00 00 00 00 dw[10] e_res2[10] Reserved words
Decompiler
1 No Function
Console - Scripting
00400000

2. Conociendo la herramienta Ghidra

- **Ventana principal dividida en 2 partes**
- **Parte lateral**:
  - "Program trees": **permite organizar las secciones del código desensamblado de diferentes modos**. Haciendo click derecho, en "Modularize By" aparecen las opciones ("Subroutine", "Complexity Depth" o "Dominance")
  - "Symbol trees": muestra los **funciones de importación, exportación, propias, etiquetas, clases y namespaces de la aplicación**. Con click derecho sobre una función de importación y "Show references to" nos mostrará todas las referencias a esa función
  - "Data Type Manager" muestra todos los **tipos definidos, incluyendo los propios del compilador, así como los específicos de la aplicación y otros de Ghidra** (los de Windows en "windows_vs12_32"). Con click derecho sobre uno de ellos, se pueden encontrar las ocurrencias con "Find uses of"
- **Parte principal**:
  - "Listing": **ventana de desensamblado**. Altamente configurable (botón "Edit the Listing Fields")
  - "Decompile": **ventana de decompilado**. Interpretación en pseudo-C del lenguaje de bajo nivel (bastante bueno; menos GOTO que IDA Pro)
    - *Lectura recomendada*: "No More Gotos: Decompilation Using Pattern-Independent Control-Flow Structuring and Semantics-Preserving Transformations" (NDSS'15)
  - Mantiene sincronía entre ambas ventanas
- **Manual de ayuda**: selecciona la parte que quieres consultar y presiona tecla "F1"

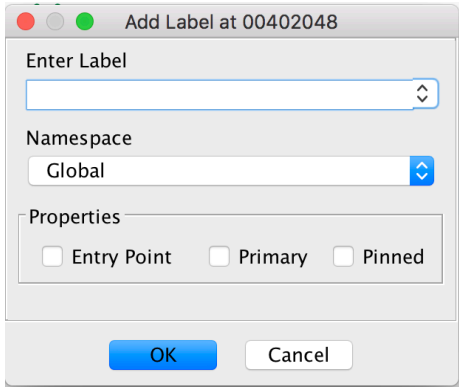
2. Conociendo la herramienta Ghidra

Opciones del panel de desensamblado



- **Adición de comentarios**: diferentes tipos de comentarios

- **Navegación en llamadas**
  - Hacia delante (Alt/Opt + flecha derecha) y hacia atrás (Alt/Opt + fecha izquierda) [iconos de flecha]



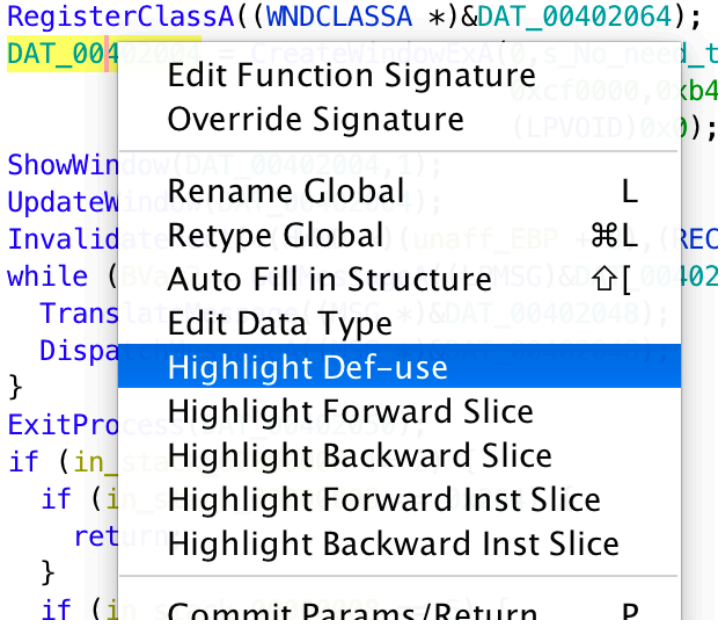
- **Etiquetas** en direcciones para facilitar análisis

- Tanto las etiquetas como los comentarios **se mantienen entre vistas**
  - **Ghidra mantiene la coherencia entre las diferentes partes de la herramienta**

2. Conociendo la herramienta Ghidra

Opciones del panel de decompilado

- **Análisis de flujo de datos** (Data Flow Analysis) automático (auto-análisis)
- Conceptos mínimos sobre DFA:
  - **Permite calcular el efecto de un bloque básico, mediante la composición de los efectos de cada expresión**
  - **Ejemplo**: considera la expresión "a = b + c"
    - Está usando "b" y "c"
    - "Mata" una definición anterior (valor anterior de "a")
    - Proporciona una nueva definición ("a")
  - **Útil para diferentes análisis**:
    - *Available expressions*
    - *Liveness*
    - *Very busy expressions*
    - *Reaching definitions*: definiciones def-use
      - Una definición de una variable "x" es una expresión que asigna un valor a "x"
      - Una definición "d" alcanza un punto de programa "p" si ∃ un camino desde el punto que sigue inmediatamente a "d" hasta "p" tal que "d" no es eliminado en el camino

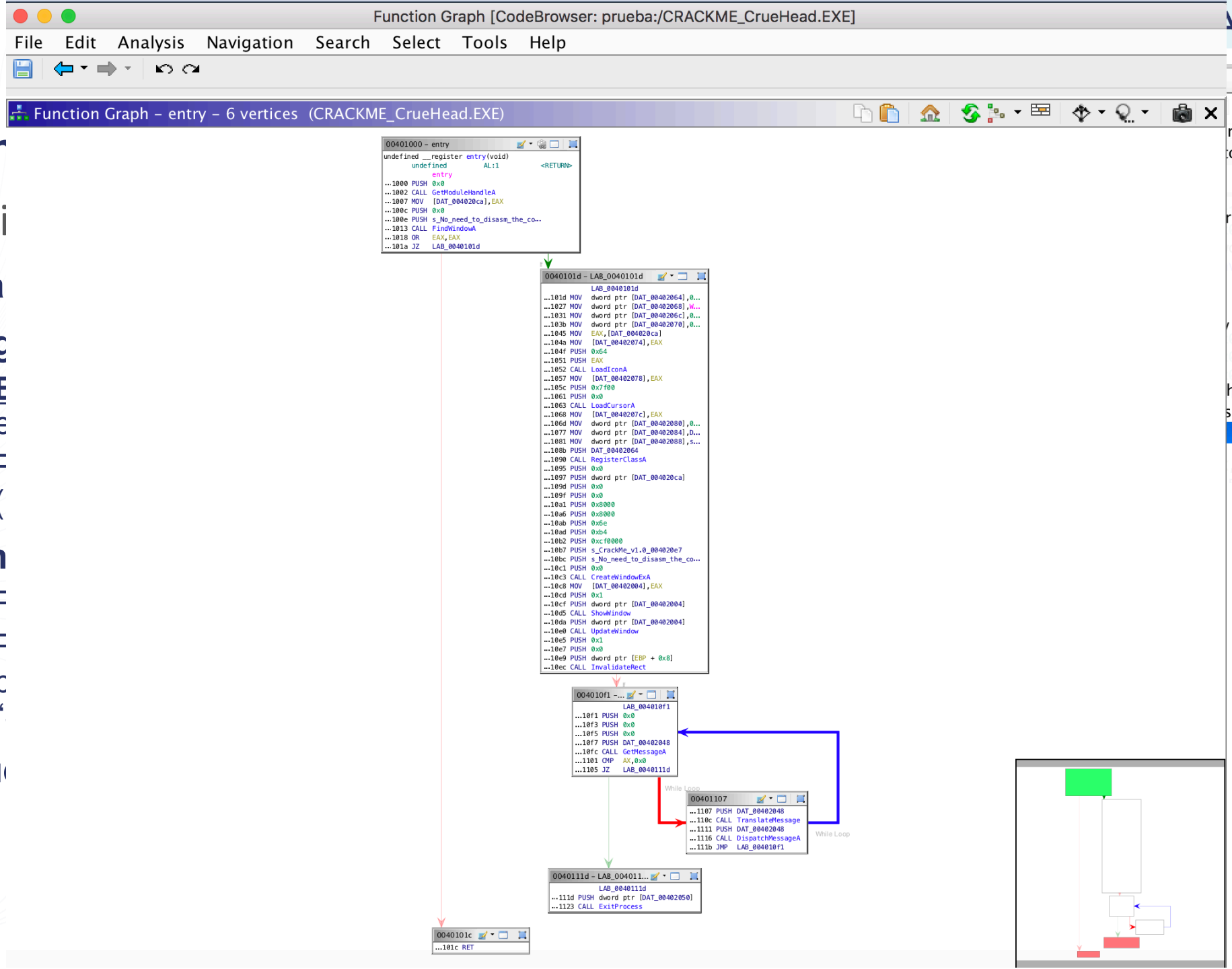


- Click derecho para ver cadenas def-use y forward/backward slices; flujos desde menú "Select"

2. Cor

Otras vi

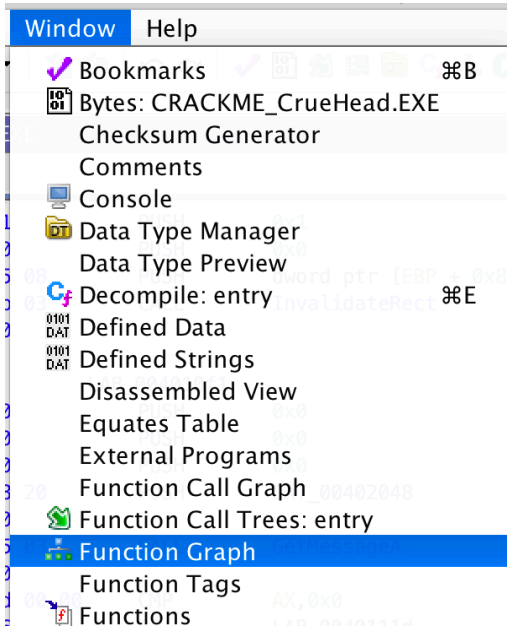
- Típica
- **Naveg**
  - **E**
    e
  - F
    (
- **Altam**
  - F
  - F
    p
    “
- De nu

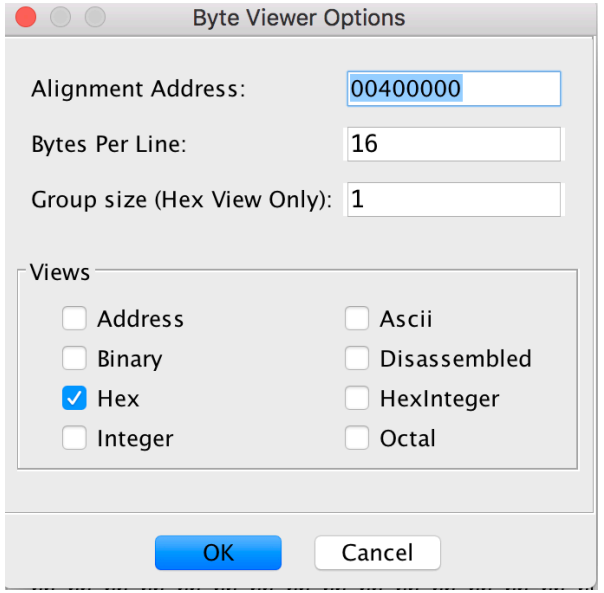


2. Conociendo la herramienta Ghidra

Otras vistas de interés

- **"Function Call Graph"**
  - **ICFG** (interprocedural CFG)
  - Navegación (delante/atrás) como en la ventana de desensamblado
- **"Disassembled View"**
  - Ventana con **código desensamblado, sin florituras**
- **"Defined strings"**
  - **Listado de cadenas del binario**
  - Desde la ventana del desensamblado, click derecho "References > Show References to Address"
- **"Bytes: …"**
  - **Visor hexadecimal** (permite edición)
  - Opciones de visión configurables
- **Terminal de Python integrada**





2. Conociendo la herramienta Ghidra

```
MOV    RAX,RSI
         RAX = COPY RSI
```

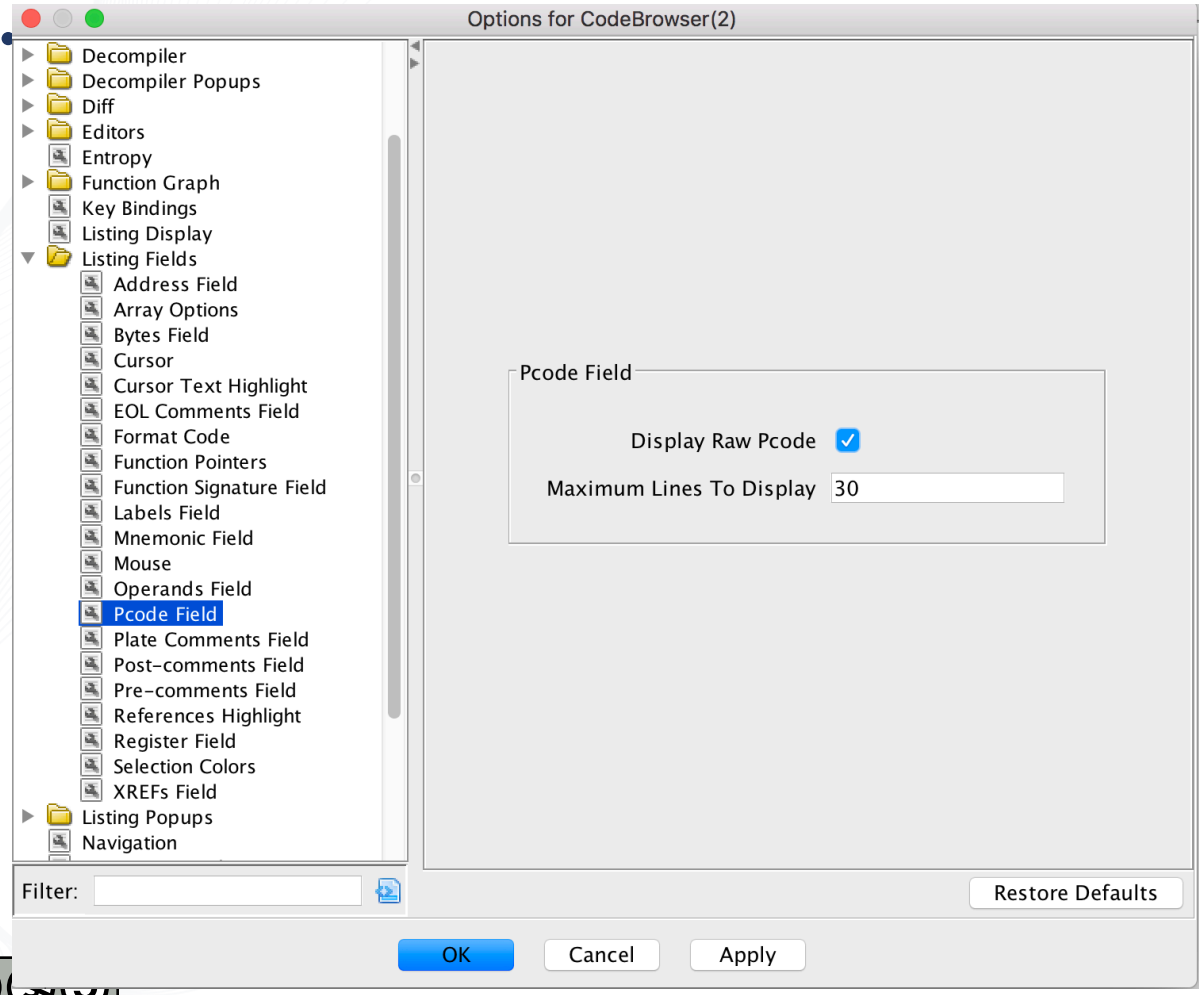
¿P-Code?

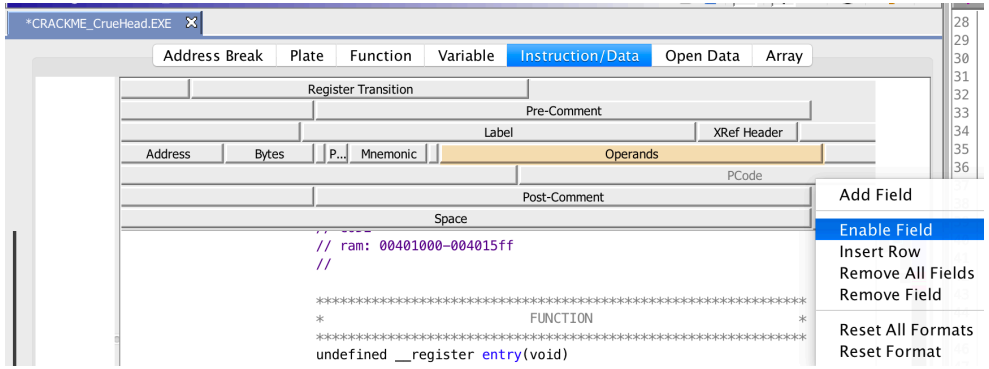
- **NO CONFUNDIR con VB P-code**
  - https://web.archive.org/web/20010222035711/http://msdn.microsoft.com/library/backgrnd/html/msdn_c7pcode2.htm
  - https://web.archive.org/web/20151222171103/http://www.woodmann.com/crackz/Tutorials/Vbpcode.htm

- **Lenguaje intermedio para instrucciones de ensamblador**
  - Permite hacer análisis más fácil
  - **Abstrae la complejidad de cada ISA, proveyendo un conjunto de instrucciones común y simplificado, e independiente de la arquitectura**
- **Generado con SLEIGH**
  - Lenguaje específico de Ghidra que proporciona información de desensamblado (bytes 89 d8 es MOV EAX, EBX) e información semántica (MOV EAX, EBX es EAX = COPY EBX)
- **Representación Single Static Assignment** (SSA)
  - A cada variable se le asigna un valor una única vez
- Documentación adicional en docs/languages/html/pcoderef.html
  - Accesible en https://ghidra-decompiler-docs.netlify.com/index.html

```
SHR    RAX,0x3f
         $Ub7c0:4 = INT_AND 63:4, 63:4
         $Ub7d0:8 = COPY RAX
         RAX = INT_RIGHT RAX, $Ub7c0
         $U33e0:1 = INT_NOTEQUAL $Ub7c0, 0:4
         $U33f0:4 = INT_SUB $Ub7c0, 1:4
         $U3400:8 = INT_RIGHT $Ub7d0, $U33f0
         $U3410:8 = INT_AND $U3400, 1:8
         $U3430:1 = INT_NOTEQUAL $U3410, 0:8
         $U3440:1 = BOOL_NEGATE $U33e0
         $U3450:1 = INT_AND $U3440, CF
         $U3460:1 = INT_AND $U33e0, $U3430
         CF = INT_OR $U3450, $U3460
         $U3490:1 = INT_EQUAL $Ub7c0, 1:4
         $U34b0:1 = INT_SLESS $Ub7d0, 0:8
         $U34c0:1 = BOOL_NEGATE $U3490
         $U34d0:1 = INT_AND $U34c0, OF
         $U34e0:1 = INT_AND $U3490, $U34b0
         OF = INT_OR $U34d0, $U34e0
         $U2e00:1 = INT_NOTEQUAL $Ub7c0, 0:4
         $U2e20:1 = INT_SLESS RAX, 0:8
         $U2e30:1 = BOOL_NEGATE $U2e00
         $U2e40:1 = INT_AND $U2e30, SF
         $U2e50:1 = INT_AND $U2e00, $U2e20
         SF = INT_OR $U2e40, $U2e50
         $U2e80:1 = INT_EQUAL RAX, 0:8
         $U2e90:1 = BOOL_NEGATE $U2e00
         $U2ea0:1 = INT_AND $U2e90, ZF
         $U2eb0:1 = INT_AND $U2e00, $U2e80
         ZF = INT_OR $U2ea0, $U2eb0
```

2. Conociendo la herramienta Ghidra





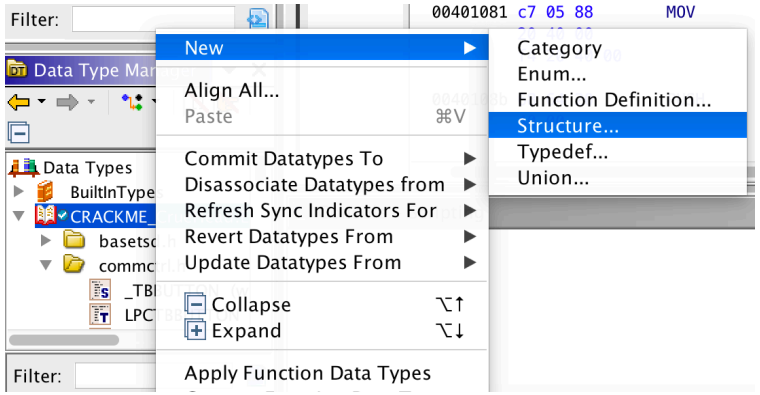

```
undefined          AL:1          <RETURN>
                   entry                                  XREF[4]:

00401000 6a 00     PUSH          0x0
                                 $U6730:4 = COPY 0:4
                                 ESP = INT_SUB ESP, 4:4
                                 STORE ram(ESP), $U6730
00401002 e8 ff 04  CALL          GetModuleHandleA
         00 00
                                 ESP = INT_SUB ESP, 4:4
                                 STORE ram(ESP), 0x401007:4
                                 CALL *[ram]0x401506:4
00401007 a3 ca 20  MOV           [DAT_004020ca],EAX
         40 00
                                 *[ram]0x4020ca:4 = COPY EAX
0040100c 6a 00     PUSH          0x0
                                 $U6730:4 = COPY 0:4
                                 ESP = INT_SUB ESP, 4:4
                                 STORE ram(ESP), $U6730
0040100e 68 f4 20  PUSH          s_No_need_to_disasm_the_code!_004020f4
```

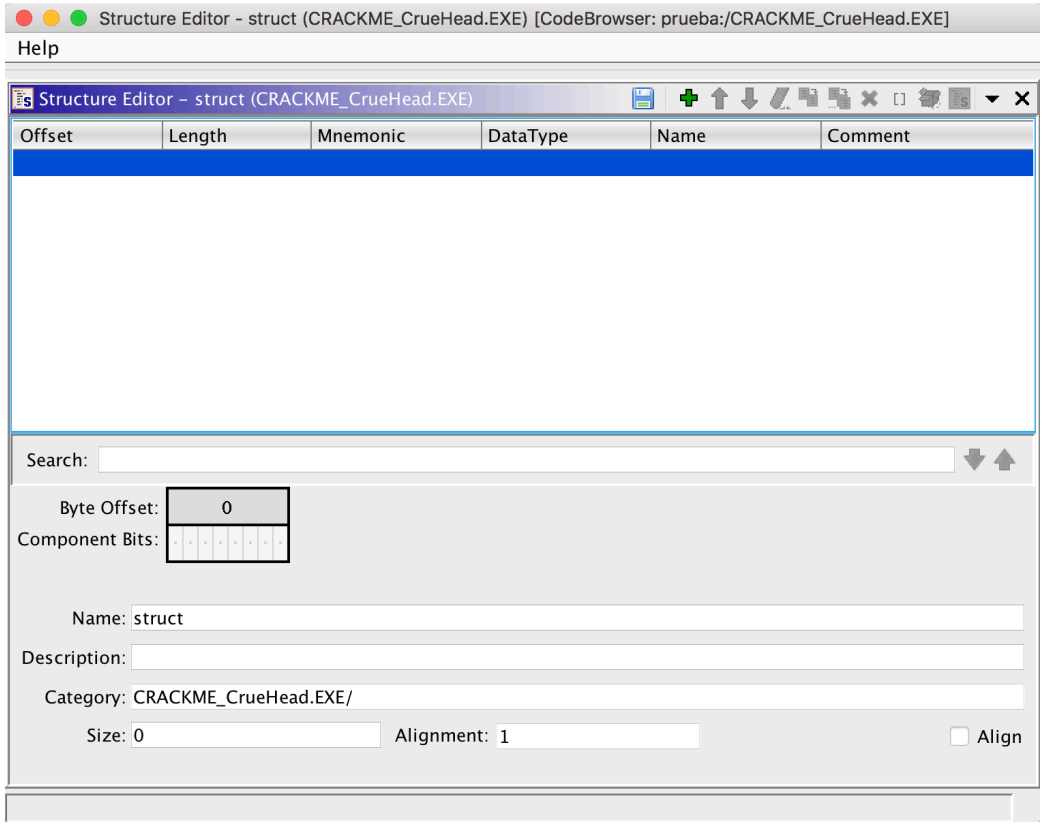
2. Conociendo la herramienta Ghidra

Otras características

- Soporte de **decompilación de diferentes arquitecturas**
  - X86, x64, SPARC, PPC, …
  - *Extensible a otras arquitecturas*

- "Tools > Program Differences"
  - Útil para detectar cambios en el mismo binario
- **Editor de estructuras**
  - Accesible desde ventana "Data Type Manager"





Version Tracking: VT__WallaceSrc.exe__WallaceVersion2.exe

File Edit Window Help

Version Tracking Matches - [Session: VT__WallaceSrc.exe__WallaceVersion2.exe] -354 matches

Tag	Sess...	St...	Type	Score	Confide...	Votes	# Co...	Mul...	Source Name...	Source Label	Source Ad...	Mul...	Dest Namesp...	Dest Label	Dest Address	Source...	Dest L...	Algorithm
	3		Function	1.000	1.000	1	0		Global	__FindPESection	004132d0		Global	__FindPESection	004132b0	117	117	Exact Function Instructi...
	2		Data	1.000	1.000	0	0			<No Symbol>	004194ca			<No Symbol>	004194ba	15	15	Exact Data Match
	4		Function	1.000	1.000	6	0		Global	FUN_00411da0	00411da0		Global	FUN_00411d80	00411d80	290	290	Exact Function Mnemon...
	2		Data	1.000	1.000	0	0			<No Symbol>	0041a028			<No Symbol>	0041a028	8	8	Exact Data Match
	2		Data	1.000	1.000	0	0			<No Symbol>	004193fe			<No Symbol>	004193ee	11	11	Exact Data Match
	2		Data	1.000	1.000	0	0		Global	s_Stack_memory_c...	00416a14		Global	s_Stack_memory_co...	00416a14	24	24	Exact Data Match
	2		Data	1.000	1.000	1	0		Global	u__controlfp_s(((vo...	004171a8		Global	u__controlfp_s(((voi...	004171a8	100	100	Exact Data Match
	4		Function	1.000	1.000	0	0		Global	_pre_cpp_init	00411e70		Global	_pre_cpp_init	00411e50	90	90	Exact Function Mnemon...
	2		Data	1.000	1.000	0	0		Global	s_Stack_memory_a...	00416ac0		Global	s_Stack_memory_ar...	00416ac0	44	44	Exact Data Match
	2		Data	1.000	1.000	1	0		Global	s_Lady_Tottington...	0041688c		Global	s_Lady_Tottington_...	0041688c	16	16	Exact Data Match
	2		Data	1.000	1.000	0	0			<No Symbol>	0041956c			<No Symbol>	0041955c	19	19	Exact Data Match
	2		Data	1.000	1.000	0	0			<No Symbol>	004194f4			<No Symbol>	004194e4	6	6	Exact Data Match
	2		Data	1.000	1.000	0	0			<No Symbol>	004195c2			<No Symbol>	004195b2	15	15	Exact Data Match
	3		Function	1.000	1.000	1	0		Global	_NtCurrentTeb	00412200		Global	_NtCurrentTeb	004121e0	13	13	Exact Function Instructi...
	2		Data	1.000	1.000	1	0		Global	s_Were_Rabbit_00...	0041687c		Global	s_Were_Rabbit_004...	0041687c	12	12	Exact Data Match
	2		Data	1.000	1.000	0	0			<No Symbol>	00419646			<No Symbol>	00419636	28	28	Exact Data Match
	2		Data	1.000	1.000	1	0		Global	s_%s_%s_deployed...	00416830		Global	s_%s_%s_deployed_...	00416830	22	22	Exact Data Match

Filter: | Score Filter: 0.000 to 1.000 | Confidence Filter: -9.999 to 9.999 | Length Filter: 0

Version Tracking Markup Items - [Session: VT__WallaceSrc.exe__WallaceVersion2.exe] -3 markup items

Status	Source Address	Dest Address	Markup Type	Source Value	Current Dest Value	Original Dest Value
✓✓	00412200	004121e0	Plate Comment	Library Function - Single Match Nam...	Library Function - Single Match Nam...	Library Function - Single Match Nam...
✓✓	00412200	004121e0	Function Name	**_NtCurrentTeb**	**_NtCurrentTeb**	**_NtCurrentTeb**
✓✓	00412200	004121e0	Function Signature	_TEB * _NtCurrentTeb(void)	_TEB * _NtCurrentTeb(void)	_TEB * _NtCurrentTeb(void)

Filter:

Decompile View | Listing View

Source: **_NtCurrentTeb()** in /WallaceSrc.exe

```
                    ****************************************************************
                    * Library Function - Single Match                              *
                    * Name: _NtCurrentTeb                                          *
                    * Library: Visual Studio 2010 Debug                            *
                    ****************************************************************
                    _TEB * _NtCurrentTeb(void)
        _TEB *        EAX:4          <RETURN>
                    _NtCurrentTeb                                XREF[1]:     _NtCurrentTeb:004111
                                                                              _NtCurrentTeb:004111
→       00412200 8b ff          MOV         EDI,EDI
```

Destination: **_NtCurrentTeb()** in /WallaceVersion2.exe

```
                    ****************************************************************
                    * Library Function - Single Match                              *
                    * Name: _NtCurrentTeb                                          *
                    * Library: Visual Studio 2010 Debug                            *
                    ****************************************************************
                    _TEB * _NtCurrentTeb(void)
        _TEB *        EAX:4          <RETURN>
                    _NtCurrentTeb                                XREF[1]:     _NtCurrentTeb:004111
                                                                              _NtCurrentTeb:004111
→       004121e0 8b ff          MOV         EDI,EDI
```

2. Conociendo la herramienta Ghidra

Scripting

- **Acepta scripts** tanto en Java como en Python (con Jython)
- **Ejecución desde la GUI o modo headless** (vía línea de comandos, sin interfaz)
- Para usar tus propios scripts, **primero debes de importarlos**:
  - "Display Script Manager > Script Directories > +"
- **Ventana "Console"**
  - Muestra salida del script

```
//Escribe el nombre de cada función en la consola
//@category    Demo Jornadas STIC

import ghidra.app.script.GhidraScript;
import ghidra.program.model.listing.*;

public class HelloWorld extends GhidraScript{

    @Override
    public void run() throws Exception{
        println("Hola, CCN CERT!");

        Function current = getFirstFunction();
        while(current != null){
            println(current.getName());
            current = getFunctionAfter(current);
        }
    }
}
```

- **Ejemplo**: cálculo de complejidad ciclomática (de todas las funcs)
  - Medida de Ingeniería del Software
  - Calcula caminos únicos desde de una función
  - Permite identificar funciones complejas, parsers, o máquinas de estados
  - ***Créditos**: presentación de INFILTRATECON 2019*

2. Conociendo la herramienta Ghidra

Scripting

```
ricardo@freyja:~/ghidra_9.1_PUBLIC$ ./support/analyzeHeadless ~/ prueba -scriptPath ~/Desktop/demo-scripts -postscript ComputeCyclomaticComplexityForAllFunctions.java -process -recursive
java version "11.0.5" 2019-10-15 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.5+10-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.5+10-LTS, mixed mode)
INFO  Using log config file: jar:file:/Users/ricardo/ghidra_9.1_PUBLIC/Ghidra/Framework/Generic/lib/Generic.jar!/generic.log4j.xml (LoggingInitialization)
INFO  Using log file: /Users/ricardo/.ghidra/.ghidra_9.1_PUBLIC/application.log (LoggingInitialization)
INFO  Loading user preferences: /Users/ricardo/.ghidra/.ghidra_9.1_PUBLIC/preferences (Preferences)
INFO  Class search complete (987 ms) (ClassSearcher)
INFO  Initializing SSL Context (SSLContextInitializer)
INFO  Initializing Random Number Generator... (SecureRandomFactory)
INFO  Random Number Generator initialization complete: NativePRNGNonBlocking (SecureRandomFactory)
INFO  Trust manager disabled, cacerts have not been set (ApplicationTrustManagerFactory)
INFO  HEADLESS Script Paths:
    /Users/ricardo/Desktop/demo-scripts
    /Users/ricardo/ghidra_scripts
```

```
-------------------------------------------------
 (AutoAnalysisManager)
INFO  REPORT: Analysis succeeded for file: /CRACKME_CrueHead.EXE (HeadlessAnalyzer)
INFO  SCRIPT: /Users/ricardo/Desktop/demo-scripts/ComputeCyclomaticComplexityForAllFunctions.java (HeadlessAnalyzer)
INFO  entry complexity: 3 (GhidraScript)
INFO  WndProc complexity: 13 (GhidraScript)
INFO  FUN_00401257 complexity: 8 (GhidraScript)
INFO  FUN_0040130e complexity: 4 (GhidraScript)
INFO  FUN_0040134d complexity: 1 (GhidraScript)
INFO  FUN_00401362 complexity: 1 (GhidraScript)
INFO  FUN_0040137e complexity: 4 (GhidraScript)
INFO  FUN_004013c2 complexity: 2 (GhidraScript)
INFO  FUN_004013d2 complexity: 1 (GhidraScript)
INFO  FUN_004013d8 complexity: 2 (GhidraScript)
INFO  LoadCursorA complexity: 0 (GhidraScript)
INFO  MessageBeep complexity: 0 (GhidraScript)
INFO  LoadIconA complexity: 0 (GhidraScript)
INFO  SetFocus complexity: 0 (GhidraScript)
INFO  MessageBoxA complexity: 0 (GhidraScript)
INFO  PostQuitMessage complexity: 0 (GhidraScript)
INFO  InvalidateRect complexity: 0 (GhidraScript)
INFO  TranslateMessage complexity: 0 (GhidraScript)
INFO  ShowWindow complexity: 0 (GhidraScript)
INFO  UpdateWindow complexity: 0 (GhidraScript)
INFO  RegisterClassA complexity: 0 (GhidraScript)
INFO  CreateWindowExA complexity: 0 (GhidraScript)
INFO  DefWindowProcA complexity: 0 (GhidraScript)
INFO  DialogBoxParamA complexity: 0 (GhidraScript)
INFO  DispatchMessageA complexity: 0 (GhidraScript)
INFO  EndDialog complexity: 0 (GhidraScript)
INFO  FindWindowA complexity: 0 (GhidraScript)
INFO  GetDlgItemTextA complexity: 0 (GhidraScript)
INFO  GetMessageA complexity: 0 (GhidraScript)
INFO  GetModuleHandleA complexity: 0 (GhidraScript)
INFO  ExitProcess complexity: 0 (GhidraScript)
INFO  ANALYZING changes made by post scripts: /CRACKME_CrueHead.EXE (HeadlessAnalyzer)
INFO  REPORT: Post-analysis succeeded for file: /CRACKME_CrueHead.EXE (HeadlessAnalyzer)
INFO  REPORT: Save succeeded for processed file: /CRACKME_CrueHead.EXE (HeadlessAnalyzer)
```

3. Ejemplos de uso

- **Metodología de ingeniería inversa**
  - Si se puede, ejecutarlo para comprender funcionamiento (flujo de ventanas, información, etc.)
  - **Análisis de cadenas**
    - Búsqueda de cadenas de interés y análisis del código cercano
  - **Comprensión del código desensamblado / decompilado**
  - **¿Aplicar scripts?**
    - Cálculo de complejidad ciclomática
    - Búsqueda de constantes típicas de criptografía
    - …

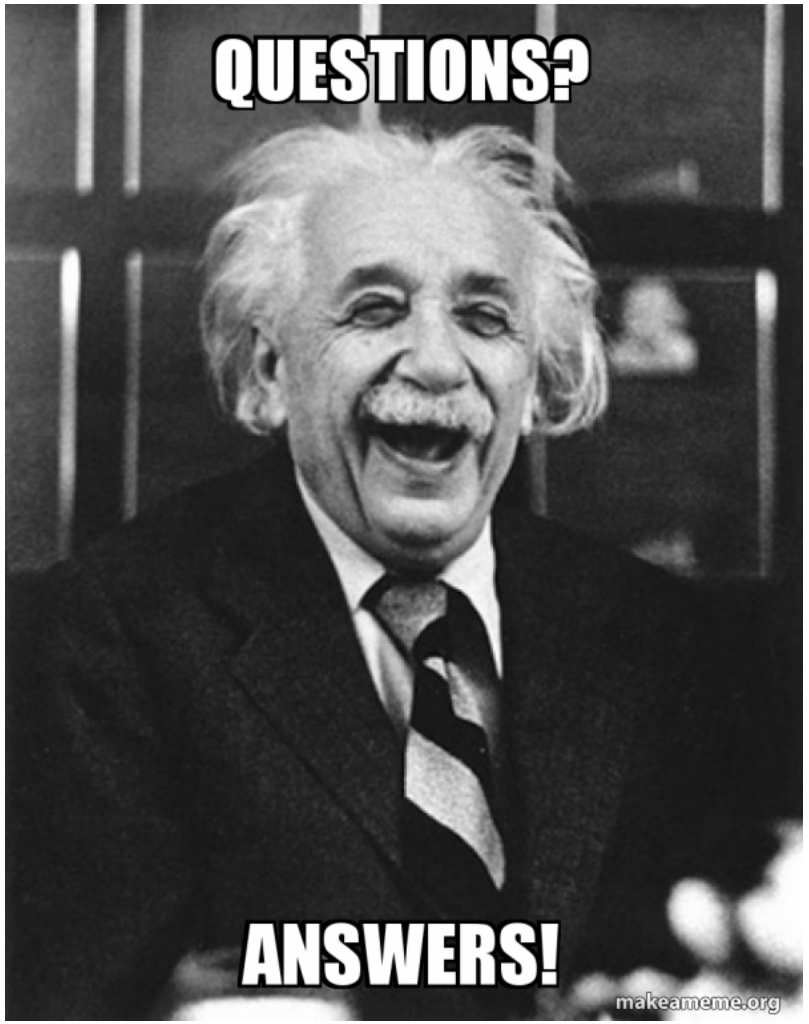
3. Ejemplos de uso

4. Conclusiones

- **Retos de análisis estático**:
  - Ofuscación
  - Software packing
- **Herramienta SRE bastante completa**
  - Desarrollada por la NSA
  - *For reverse engineers, by reverse engineers*
  - **Pros**:
    - Código abierto
    - Técnicas de análisis estático incorporadas (CFA, DFA, symbolic execution?)
    - Útil como alternativa a IDA Pro para talleres o clases prácticas
    - Modo GUI y modo headless
    - Scripting
  - **Contras**:
    - Le falta capacidad de depuración
    - Interfaz demasiado compleja
    - Java
- **Una herramienta útil para nuestra tool chain (pero no única!)**

- **Recursos online**: https://ghidra.re/online-courses/

Q&A



XIII JORNADAS STIC CCN-CERT

COMUNIDAD Y CONFIANZA,
BASES DE NUESTRA CIBERSEGURIDAD

#XIIIJORNADASCCNCERT

OC.CCN.CNI.ES
WWW.CCN.CNI.ES
WWW.CCN-CERT.CNI.ES