



**EDUCACIÓN**  
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO  
NACIONAL DE MÉXICO®



## **Instituto Tecnológico de San Juan del Río**



### **Tópicos de Ciberseguridad**

**P R E S E N T A:**

Maqueda Durazno Alexis

Mendoza Garcia Daniela

Rengel Olvera Paloma

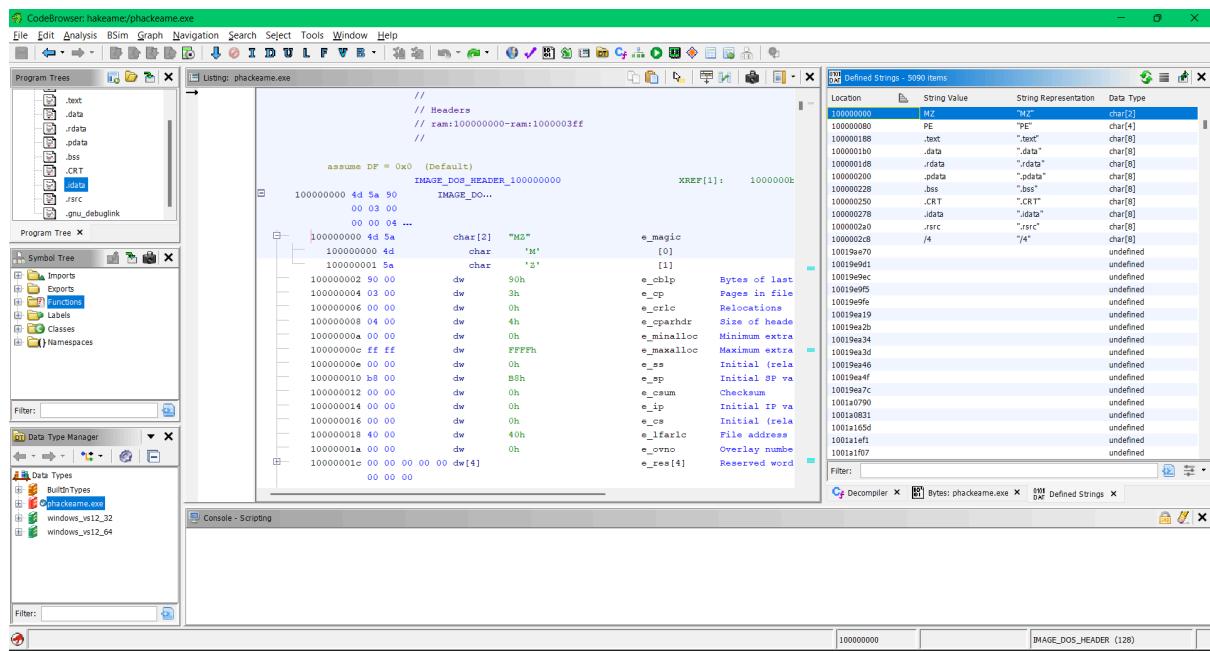
Ing. Sistemas Computacionales





Video de Youtube: <https://youtu.be/E9dX-bS26Ww>

1- Para comenzar, se requiere iniciar la herramienta de ingeniería inversa Ghidra. El primer paso es crear un nuevo proyecto dentro de Ghidra. Una vez creado el espacio de trabajo, se debe cargar el archivo ejecutable (.exe) que se desea analizar. Ghidra importará el binario y comenzará el proceso de análisis inicial, identificando funciones, cadenas de texto y referencias.



2- Con el ejecutable cargado y analizado, el siguiente paso es la búsqueda de cadenas de texto específicas. Se debe utilizar la ventana o el panel de "Defined Strings". En este panel, se aplicará el buscador (filtro) para localizar cualquier cadena que contenga la subcadena "USR". La localización de estas cadenas a menudo apunta a secciones del código que



manejan la entrada o validación de credenciales de usuario.

The screenshot shows the CodeBrowser interface with the following details:

- Program Tree:** Shows the file structure of `phackame.exe`, including Headers, .text, .data, .rdata, .bss, .CRT, and .idata.
- Symbol Tree:** Shows imports, exports, functions, labels, classes, and namespaces.
- Listing:** Displays assembly code for `phackame.exe`. A specific line of code is highlighted:

```
s_["USB"];"_1001c2dd8
```
- Data Type Manager:** Shows built-in types and user-defined types for `phackame.exe`.
- Defined Strings:** A table showing defined strings with their locations, string values, representations, and data types. For example, `1001c2d88` contains the string `"\USB\""`.
- Console - Scripting:** An empty console window.

3- Una vez localizadas las referencias a las cadenas de texto en el código ensamblador, se procede a obtener una vista de alto nivel del código. Estando posicionado en las líneas de código relevantes, se accede al apartado del menú "Window" y luego se selecciona "Decompile" (Descompilar).

The screenshot shows the CodeBrowser interface with the following details:

- Program Tree:** Shows the file structure of `phackame.exe`.
- Symbol Tree:** Shows imports, exports, functions, labels, classes, and namespaces.
- Listing:** Displays assembly code for `phackame.exe`. A specific line of code is highlighted:

```
s_["USB"];"_1001c2dd8
```
- Data Type Manager:** Shows built-in types and user-defined types for `phackame.exe`.
- Defined Data:** Shows defined data items.
- Defined Strings:** Shows defined strings items.
- Decompile:** A context menu option is selected over the assembly line, showing options like "Decompile: FUN\_100184d10 Ctrl+E".
- Data View:** A table showing defined strings with their locations, string values, representations, and data types, matching the one shown in the previous screenshot.

4- La acción anterior resultará en la apertura de la ventana del descompilador, la cual arrojará una aproximación del código fuente base (en un pseudocódigo similar a C/C++) a partir del código ensamblador del .exe. Aunque no es el código fuente original exacto, esta



aproximación es invaluable para entender la lógica del programa.

The screenshot displays the Immunity Debugger interface. The assembly pane at the bottom shows the following assembly code:

```

10002ff8 48 89 c1    MOV    RCX,RAX
10002ffb 45 31 c0    XOR    RSD,RSR
10002ffe 48 8d 15    LEA    RDX,[DAT_1001c2eb8]
b3 2e 19 00          = 52h

10003005 a8 d6 a2    CALL   FUN_1000fa2e0
03 00

1000300a 89 45 cc    MOV    dword ptr [RBP + local_3c],EAX
1000300d 48 9b 4d d8  MOV    RCX,qword ptr [RBP + local_30]

10003001 a8 ea 9b    CALL   FUN_100139e00
10 00
10003016 81 7d ec    CMP    dword ptr [RBP + local_3c],0xc8
e8 00 00 00

10003001d 75 21     JNZ   LAB_100030040
10003001e 48 8b 45 f0  MOV    RAX,qword ptr [RBP + local_18]
100030023 48 9b 88    MOV    RCX,qword ptr [RAX + 0x7f0]
f0 07 00 00

10003002a b2 01     MOV    DL,offset s_._PASS:"_1001c2e01"
10003002c 48 8b 45 f0  MOV    RAX,qword ptr [RBP + local_18]
100030030 48 8b 80    MOV    RAX,qword ptr [RAX + 0x7f0]
f0 07 00 00

100030037 48 8b 00    MOV    RAX,qword ptr [RAX]
10003003a ff 90 50    CALL   qword ptr [RAX + 0x450]
04 00 00

```

The decompiled C pane at the top shows the following function:

```

Decompile: FUN_10002fe90 - (phckame.exe)
37     FUN_10012ba50(*(undefined8 *) (local_18 + 0x7e0),&local_78)
38     local_68 = local_78;
39     local_60 = &DAT_1001c2e00;
40     FUN_10012ba50(*(undefined8 *) (local_18 + 2000),&local_80);
41     local_58 = local_80;
42     local_50 = &DAT_1001c2e28;
43     FUN_1000093b0((local_28,&local_70,4,0);
44     FUN_1000300d0((Local_20,&local_30,"http://45.76.173.114:800
45     if (local_30 == 0) {
46         FUN_100139e00("No se pudo conectar al servidor");
47     }
48     else {
49         FUN_1000097b0((local_88,local_30,0xfde9));
50         uVar1 = FUN_100065840((local_88,1));
51         local_38 = FUN_1000e0f0((DAT_1001d6e58,uVar1));
52         local_3c = FUN_10006a2e0((local_38,&DAT_1001c2eb8,0));
53         FUN_100139e00((local_30));
54         if (local_3c == 200) {
55             (**(code **)((longlong *) (local_18 + 0x7f0)) + 0x450
56             (*((undefined8 *) (local_18 + 0x7e0),1);
57         }
58     }
59 }
60 }
61 FUN_10002fe40((stack0xfffffffffffffff8));
62 return;
63 }
```

5- Localizamos el If que valida si accedemos o nos deniega la entrada (su posición exacta, hexadecimal, etc)

The screenshot shows the OllyDbg debugger interface with two windows open. The left window displays assembly code for the function `phackame.exe!FUN_10002fe90`. The right window shows the decompiled code for the same function, which is mostly empty except for a few lines at the end.

**Assembly View (Left):**

Address	OpCode	Registers	Comments
10002fff8	48 89 c1	MOV RCX, RAX	
10002ffeb	45 31 e0	XOR R8D, R8D	
10002ffe4	48 8d 15	LEA RDX, [DAT_1001c2eb8]	= 52h
100030005	b3 2e 19 00		
100030005	e8 d6 a2	CALL FUN_10006a2e0	undef
	03 00		
10003000a	89 45 cc	MOV dword ptr [RBP + local_3c], EAX	
10003000d	48 8b 4d d8	MOV RCX, qword ptr [RBP + local_30]	
100030011	e8 ea 9b	CALL FUN_100139c00	undef
	10 00		
100030016	81 7d cc	CMP dword ptr [RBP + local_3c], 0xc8	
	c8 00 00 00		
10003001d	75 21	JNZ LAB_100030040	
10003001f	48 8b 45 f0	MOV RAX, qword ptr [RBP + local_18]	
100030023	48 8b 88	MOV RCX, qword ptr [RAX + 0x7f0]	
	f0 07 00 00		
10003002a	b2 01	MOV DL, offset _, "PASS:" _1001c2e01	= ",\n
10003002c	48 8b 45 f0	MOV RAX, qword ptr [RBP + local_18]	
100030030	48 8b 80	MOV RCX, qword ptr [RAX + 0x7f0]	
	f0 07 00 00		
100030037	48 8b 00	MOV RAX, qword ptr [RAX]	
10003003a	ff 90 50	CALL qword ptr [RAX + 0x450]	
	04 00 00		
	LAB_100030040		
	XREF[4]: 10002fee		
		10002ffc	
100030040	90	NOP	
100030041	48 89 e9	MOV RCX, RBP	

**Decompiler View (Right):**

```
Decompile: FUN_10002fe90 - (phackame.exe) RD
```

```
37     FUN_10012ba50(*(_undefined8 *)(local_18 + 0x7e0),&local_78);
38     local_68 = local_78;
39     local_60 = &DAT_1001c2e00;
40     FUN_10012ba50(*(_undefined8 *)(local_18 + 2000),&local_80);
41     local_58 = local_80;
42     local_50 = &DAT_1001c2e28;
43     FUN_1000093b0(&local_28,&local_70,4,0);
44     FUN_1000300d0(local_20,&local_30,"http://45.76.173.114:801
45     if (local_30 == 0) {
46         FUN_100139c00("No se pudo conectar al servidor");
47     }
48     else {
49         FUN_1000097b0(&local_88,local_30,0xfde9);
50         uVar1 = FUN_1000658d0(local_88,1);
51         local_38 = FUN_10000e0f0(&DAT_1001d6e658,uVar1);
52         local_3c = FUN_10006a2e0(local_38,&DAT_1001c2eb8,0);
53         FUN_100139c00(local_30);
54         if (local_3c == 200) {
55             (**(code **)(**longlong **)(local_18 + 0x7f0) + 0x450
56             (*(_undefined8 *)(local_18 + 0x7f0),1);
57         }
58     }
59 }
60 }
```

```
61 FUN_10002fe40(&stack0xfffffffffffffff8);
62 return;
```

```
C:\Decompile: FUN_10002fe90 x Defined Strings x
```

5- entramos a x64dbg y cargamos el .exe, Buscamos la dirección exacta (10003001d) que nos dio Ghidra para ver el comportamiento del procesador en tiempo real.

The screenshot shows the x64dbg debugger interface with the title bar "phackeame.exe - PID: 53E4 - Module: phackeame.exe - Thread: Main Thread 6248 - x64dbg". The menu bar includes File, View, Debug, Trace, Plugins, Favourites, Options, Help, and the date "Aug 2 2020". Below the menu is a toolbar with various icons. The main window displays assembly code in the CPU tab. The assembly code is as follows:

```
jne phackeame.100030040
mov rax,qword ptr ss:[rbp-10]
mov rcx,qword ptr ds:[rax+7F0]
mov dl,1
mov rax,qword ptr ss:[rbp-10]
mov rax,qword ptr ds:[rax+7F0]
mov rax,qword ptr ds:[rax]
call qword ptr ds:[rax+450]
nop
mov rcx,rbp
call phackeame.10002FE40
nop
lea rsp,qword ptr ss:[rbp]
```



```
0000000100030016 817D CC C8000000 cmp dword ptr ss:[rbp-34],C8
000000010003001D 75 21 jne phackeame.100030040
000000010003001F 48:8B45 F0 mov rax,qword ptr ss:[rbp-10]
0000000100030023 48:8B88 F0070000 mov rcx,qword ptr ds:[rax+7F0]

Assemble at 000000010003001D
jne 0x0000000100030040
Keep Size Fill with NOP's XEDParse asmjit OK Cancel
Instruction encoded successfully!
```

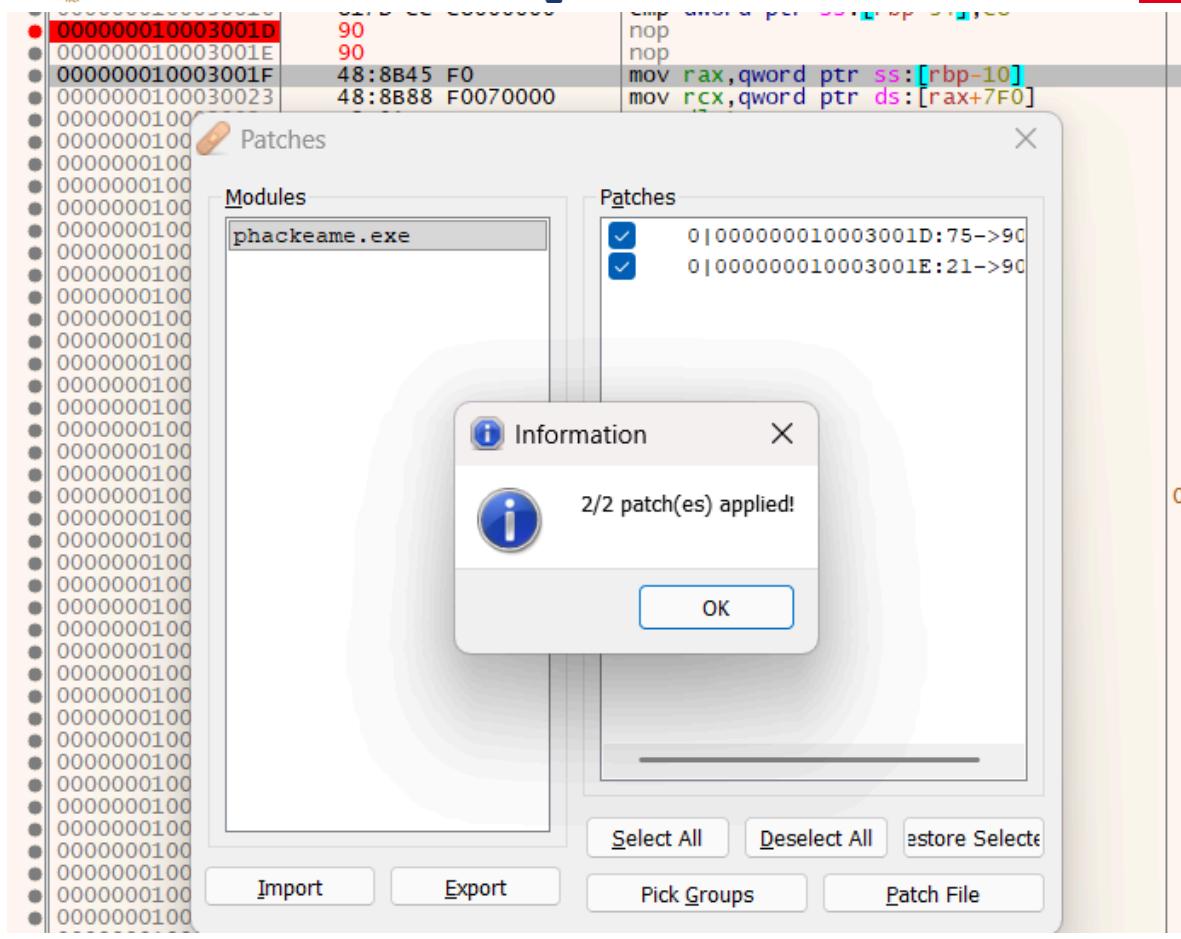
6- Seleccionamos la instrucción de salto (jne) y la reemplazamos por instrucciones nop. Al poner nop, el procesador simplemente "pasa de largo". Ya no importa qué contraseña pongas; el programa nunca saltará al error y siempre te dará acceso.

```
0000000100030016 817D CC C8000000 cmp dword ptr ss:[rbp-34],C8
000000010003001D 75 21 jne phackeame.100030040
000000010003001F 48:8B45 F0 mov rax,qword ptr ss:[rbp-10]
0000000100030023 48:8B88 F0070000 mov rcx,qword ptr ds:[rax+7F0]

Assemble at 000000010003001D
nop
Keep Size Fill with NOP's XEDParse asmjit OK Cancel
Instruction encoded successfully!
```

7- Aplicamos los parches (2/2 patches applied) para que los cambios sean permanentes.

```
Patches
Modules: phackeame.exe
Patches:
0100000010003001D:75->90
0100000010003001E:21->90
Select All Deselect All Restore Selection
Import Export Pick Groups Patch File
```



8- Al ejecutar phackeame\_patched.exe, cualquier usuario y contraseña activa el mensaje de éxito: "¡Lo lograste!".

