

PEF

ANÁLISIS AVANZADO DE MALWARE

Montesinos Guzmán Wilma Alejandra

Tabla de contenido

1.	RESUMEN EJECUTIVO	1
2.	DETALLES GENERALES	1
2.1.	ESTÁTICO	1
2.2.	DINÁMICO	26
2.3.	ESTÁTICO AVANZADO	33
3.	MITIGACIÓN	36
4.	REGLA DE DETECCIÓN YARA	37
5.	CONCLUSIÓN	38
6.	REFERENCIAS	39

1. RESUMEN EJECUTIVO

En este informe analiza un archivo malicioso diseñado para sistemas Windows, el cual aparenta ser un programa normal, pero en realidad es un virus que puede ejecutar código oculto. A través de diversas herramientas, se descubrió que este archivo incluye partes de lenguaje Python y está diseñado para evadir algunos antivirus, aunque Windows Defender logra detectarlo. El malware tiene la capacidad de ocultarse usando procesos legítimos del sistema, lo que le permite pasar desapercibido y realizar acciones como ejecutar comandos, manipular configuraciones e incluso desactivar herramientas de seguridad. También contiene archivos comprimidos en su interior y utiliza técnicas avanzadas para evitar ser detectado fácilmente. El análisis identificó comportamientos sospechosos, como ejecutar scripts maliciosos y aprovechar funciones del sistema para obtener mayor control. Finalmente, se proponen medidas para prevenir este tipo de ataques, incluyendo la actualización del sistema y del antivirus, restricciones en la ejecución de programas desconocidos y la concienciación del usuario.

2. DETALLES GENERALES

2.1. ESTÁTICO

La muestra analizada corresponde a un troyano diseñado para sistemas operativos Windows, específicamente compilado como un binario de consola PE32+ para arquitecturas x86-64. A continuación, se presentan las firmas hash de la muestra, empleadas para su identificación y verificación:

Hash	Valor
MD5	83f7625b243f0594484bc87b6ba4bf2b
SHA-1	428c2012fc69a5072b5163f7ff53f33bdac8e673
SHA-256	fd92ba4c1d0390761a5afad41d2be3ac98463c1eb1d0d7b2e8dff54505341cbe
SSDEEP	196608:ow5PB5F5682StHSRT13nho2WJTm6F9qB4:ow5kD3S913GtVm6F9L

El sample fue remitido, por primera vez, a la plataforma VirusTotal el 12 de abril de 2025 y presenta un tamaño de 8.15 MB (8.543.952 bytes). El archivo fue compilado el 22 de julio de 2024, haciendo uso del compilador Microsoft Visual C/C++ 2019. Además, el análisis revela que el malware contiene código escrito en Python.

El binario ha sido clasificado como malicioso por un total de 6 motores antivirus de los 72 disponibles en la plataforma. La detección limitada podría ser atribuida al uso de técnicas de empaquetado.

6 / 72 security vendors flagged this file as malicious

fd92ba4c1d0390761a5afad41d2be3ac98463c1eb1d0d7b2e8dff54505341cbe

Size: 8.15 MB | Last Analysis Date: 56 minutes ago

peexe 64bits overlay detect-debug-environment

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Security vendors' analysis

Bkav Pro	W64.AIDetectMalware	Deeplinstinct	MALICIOUS
Elastic	Malicious (moderate Confidence)	Jiangmin	Trojan.Python.fb
McAfee Scanner	TiFD92BA4C1D03	SecureAge	Malicious
Acronis (Static ML)	Undetected	AhnLab-V3	Undetected
Alibaba	Undetected	AliCloud	Undetected

Basic properties

MD5: 83f7625b243f0594484bc87b6ba4bf2b

SHA-1: 428c2012fc69a5072b51637f53f33bdac8e673

SHA-256: fd92ba4c1d0390761a5afad41d2be3ac98463c1eb1d0d7b2e8dff54505341cbe

Vhash: 086066555d1515651a299hz13z1dz41

Authentichash: 07b5419bae900aca37da2d1dd0619448afc795b7843a5b41c2c4144f084de389

Imphash: b588036b5202b7426cd84298c3decc62

Rich PE header hash: d0bfec8871e08ab2c5c777595635797c

SSDEEP: 196608:ow5PBV5F5682SHSRT13nho2wJTm6F9qB4:ow5K03S913GtVm6F9L

TLSH: T15C86DF22A26401A4D9B79477C5278647CBF18C542F20D34F41A4BAE92F73BA317EE706

File type: Win32 EXE | executable | windows | win32 | pe | peexe

Magic: PE32+ executable (console) x86-64, for MS Windows

TrID: Win32 EXE PECompact compressed (generic) (45.7%) | Microsoft Visual C++ compiled executable (generic) (18.2%) | Win64 Executable (generic) (11.5%) | Win32 Dynam...

DetectItEasy: PE64 | Compiler: Microsoft Visual C/C++ (19.29.30154) [LTCG/C] | Linker: Microsoft Linker (14.29.30154) | Tool: Visual Studio (2019 version 16.11) | Archive: Zip (2.0) [42....

Magika: PEBIN

File size: 8.15 MB (8543952 bytes)

History

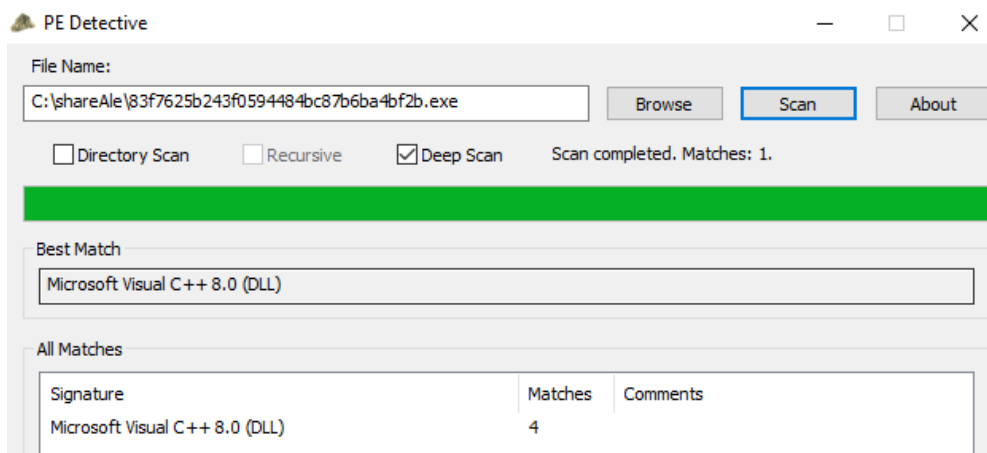
Creation Time: 2024-07-22 19:42:30 UTC

First Submission: 2025-04-12 10:43:42 UTC

Last Submission: 2025-04-12 10:43:42 UTC

Last Analysis: 2025-04-12 10:43:42 UTC

En la herramienta PE detective podemos observar que efectivamente está compilado en Microsoft Visual C/C++ sin embargo esta herramienta lo detecta como un fichero portable ejecutable (PE) DLL en lugar de un EXE.



Es común en malware, aunque inicialmente el archivo se muestra como un .exe en VirusTotal, en realidad se comporta como un DLL (Dynamic Link Library). Esto puede observarse al analizar el archivo con herramientas como PE detective, que lo identifican como un DLL lo cual tiene bastante sentido ya que exporta múltiples funciones. Esta característica es más típica de software malicioso, ya que los programas legítimos rara vez exportan tantas funciones. Este patrón es común en muestras de malware con estructura modular o que actúan como "droppers" para distribuir otras cargas útiles.

Esto es algo que podemos observar en el análisis de VirusTotal.

Exports		
PyArg_ParseTuple	PyLong_FromLong	
PyBool_FromLong	PyLong_FromVoidPtr	
PyBytes_AsString	PyMapping_HasKeyString	
PyCMethod_New	PyModuleDef_Init	
PyConfig_Clear	PyModule_Create2	
PyConfig_InitIsolatedConfig	PyModule_ExecDef	
PyConfig_SetArgv	PyModule_FromDefAndSpec2	
PyConfig_SetString	PyModule_GetDef	
PyErr_Clear	PyModule_GetDict	
PyErr_Occurred	PyModule_GetState	
PyErr_Print	PyModule_New	PyUnicode_FromFormat
PyErr_SetImportError	PyObject_CallObject	PyUnicode_FromString
PyErr_SetObject	PyObject_SetAttrString	PyUnicode_FromWideChar
PyErr_SetString	PyRun_InteractiveLoopFlags	PyWideStringList_Append
PyEval_EvalCode	PyRun_SimpleStringFlags	Py_ExitStatusException
PyExc_ImportError	PySequence_GetItem	Py_FdisInteractive
PyExc_RuntimeError	PySequence_Size	Py_Finalize
PyExc_SystemError	PyStatus_Exception	Py_GetPath
PyGILState_Ensure	PyStatus_IsExit	Py_Initialize
PyGILState_Release	PySys_SetArgvEx	Py_InitializeFromConfig
PyImport_AddModule	PySys_SetObject	Py_IsInitialized
PyImport_AppendInittab	PyTuple_New	Py_SetPath
PyImport_GetModuleDict	PyTuple_SetItem	Py_SetProgramName
	PyType_IsSubtype	_PyImport_FindExtensionObject
		_PyImport_FixupExtensionObject
		_Py_Dealloc

Estas funciones aparecen típicamente en muestras de malware que integran un intérprete de Python dentro del DLL, para ejecutar código Python embebido o cargado dinámicamente.

Podemos contrastarlo con la muestra que tenemos mediante CFF explorer, donde podemos ver las funciones que son exportadas por este fichero.

Utilizar funciones como PyEval_EvalCode, PyErr_SetImportError, PyConfig_SetArgv, entre otras se puede integrar un intérprete de Python dentro del programa en C++. Con esto, queremos decir que el atacante ha compilado un binario que incluye el runtime de Python para ejecutar código Python internamente, sin necesidad de que Python esté instalado en el sistema de la víctima.

CFF Explorer VIII - [83f7625b243f0594484bc87b6ba4bf2b.exe]

File Settings ?

83f7625b243f0594484bc87b6ba4b

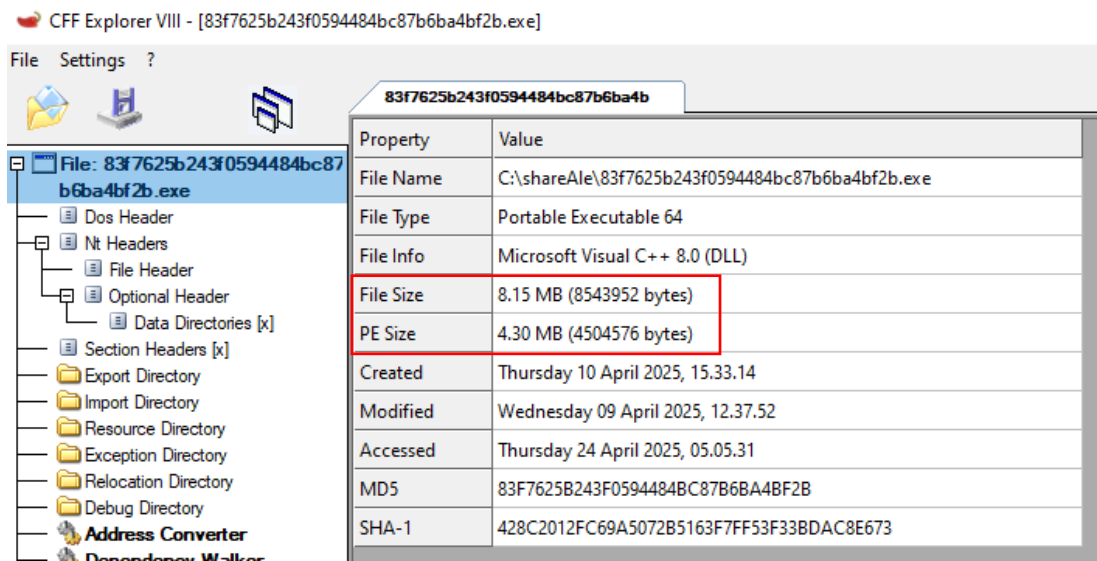
File: 83f7625b243f0594484bc87b6ba4bf2b.exe

- Dos Header
- Nt Headers
 - File Header
 - Optional Header
 - Data Directories [x]
- Section Headers [x]
- Export Directory
- Import Directory
- Resource Directory
- Exception Directory
- Relocation Directory
- Debug Directory
- Address Converter
- Dependency Walker
- Hex Editor
- Identifier
- Import Adder
- Quick Disassembler
- Rebuilder
- Resource Editor

Member	Offset	Size	Value
Characteristics	00006CE0	Dword	00000000
TimeDateStamp	00006CE4	Dword	FFFFFFFF
MajorVersion	00006CE8	Word	0000
MinorVersion	00006CEA	Word	0000
Name	00006CEC	Dword	00007B92
Base	00006CF0	Dword	00000001
NumberOfFunctions	00006CF4	Dword	00000041
NumberOfNames	00006CF8	Dword	00000041
AddressOfFunctions	00006CFC	Dword	00007908

Ordinal	Function RVA	Name Ordinal	Name RVA	Name
(nFunctions)	Dword	Word	Dword	szAnsi
00000001	00004810	0000	00007BAB	PyArg_ParseTuple
00000002	00005110	0001	00007BBC	PyBool_FromLong
00000003	00004A50	0002	00007BCC	PyBytes_AsString
00000004	000051B0	0003	00007BDD	PyCMethod_New
00000005	00004F60	0004	00007BEB	PyConfig_Clear
00000006	00004D30	0005	00007BFA	PyConfig_InitIsolatedConfig
00000007	00004D70	0006	00007C16	PyConfig_SetArgv
00000008	00004DF0	0007	00007C27	PyConfig_SetString
00000009	00004330	0008	00007C3A	PyErr_Clear
0000000A	00004370	0009	00007C46	PyErr_Occurred
0000000B	000043B0	000A	00007C55	PyErr_Print
0000000C	00004000	000B	00007C61	PyErr_SetImportError
0000000D	000050C0	000C	00007C76	PyErr_SetObject
0000000E	00004060	000D	00007C86	PyErr_SetString
0000000F	000045A0	000E	00007C96	PyEval_EvalCode
00000010	000099D8	000F	00007CA6	PyExc_ImportError
00000011	000099D0	0010	00007CB8	PyExc_RuntimeError

Técnicamente, tanto los .exe como los .dll comparten similitudes porque ambos utilizan el formato PE (Portable Executable) en sistemas Windows. Esto incluye estructuras comunes como la cabecera DOS, la cabecera NT y las secciones estándar como .text y .data. Sin embargo, los archivos .exe están diseñados para ejecutarse como aplicaciones independientes y poseen un punto de entrada que inicia la ejecución del programa (main()). En contraste, las DLLs están pensadas como bibliotecas compartidas, con una tabla de exportación que contiene funciones específicas para ser usadas por otros programas. A diferencia de los archivos .exe, las DLLs no se ejecutan por sí solas; su propósito es proporcionar funcionalidades específicas que las aplicaciones pueden invocar. Aquí podemos analizar otro punto importante, para infectar la máquina de los usuarios al malware le conviene que sea un ejecutable (EXE) que el usuario haga clic en lugar de un DLL que no podría ejecutar tan intuitivamente. Además, una DLL maliciosa puede exportar funciones que se infiltran en procesos legítimos del sistema.



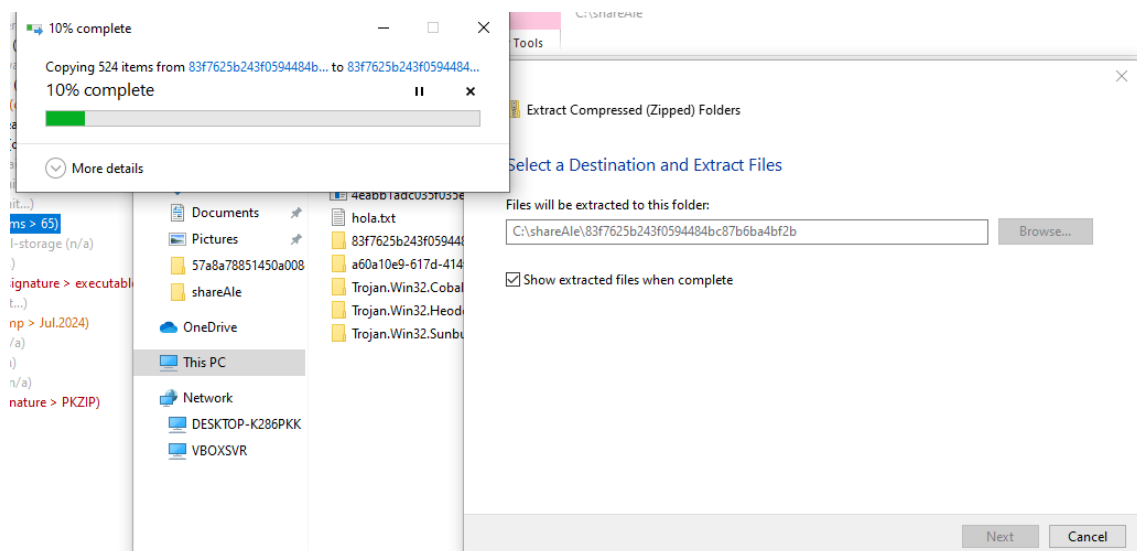
File Size representa el tamaño total del archivo tal como está almacenado en disco, este valor incluye no solo los datos y secciones necesarias para que el archivo funcione, sino también posibles rellenos, encabezados o incluso información adicional que no se utiliza directamente en la ejecución del programa. Por otro lado, PE Size hace referencia al tamaño en memoria del archivo una vez cargado por el sistema operativo.

En este caso, el hecho de que el PE Size sea la mitad del File Size puede evidenciar características propias de un malware. Una posibilidad es que el archivo contenga datos adicionales ocultos. También podría tratarse de un archivo empaquetado, donde solo ciertas partes necesarias se cargan en memoria mientras que el resto permanece en el archivo para evitar ser detectado.

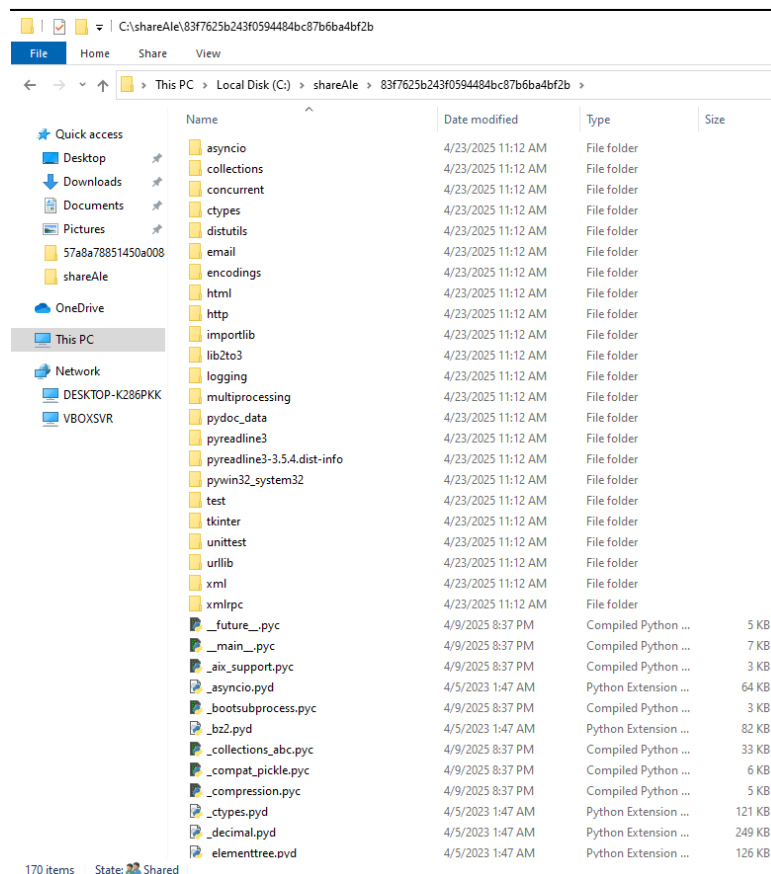
Al plantear esta posibilidad, le cambiamos la extensión del archivo a la muestra de malware a .zip para verificar si se encuentra comprimida y si es que nos solicita alguna contraseña.

Name	Date modified	Type	Size
83f7625b243f0594484bc87b6ba4bf2b.exe	4/9/2025 12:37 PM	Application	8,344 KB
83f7625b243f0594484bc87b6ba4bf2b.zip	4/9/2025 12:37 PM	Compressed (zipp...	8,344 KB

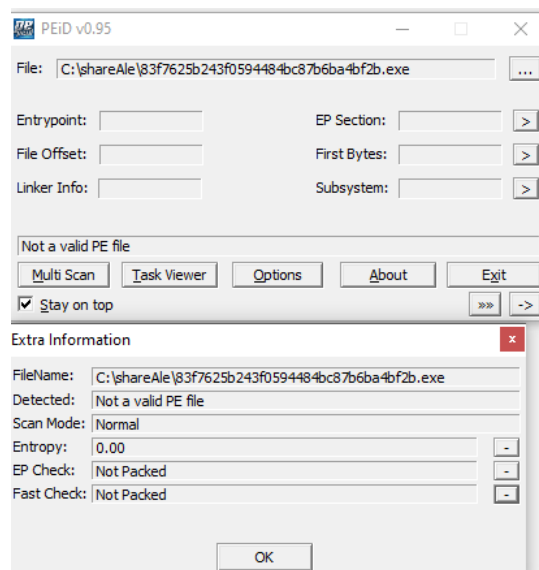
Efectivamente se encontraba comprimido y no solicita ninguna contraseña cuando lo descomprimimos.



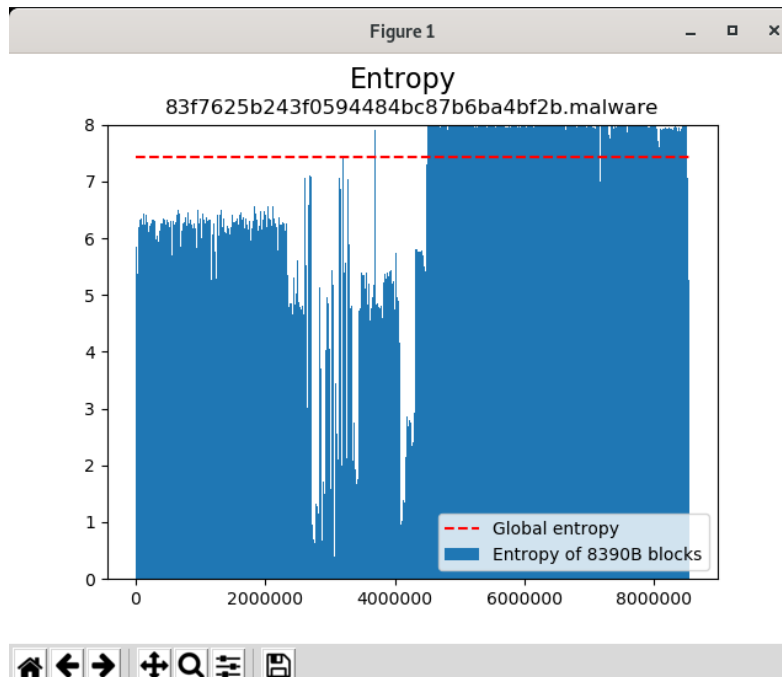
El resultado de la descompresión se observa en la siguiente imagen, donde se observan 170 ficheros.



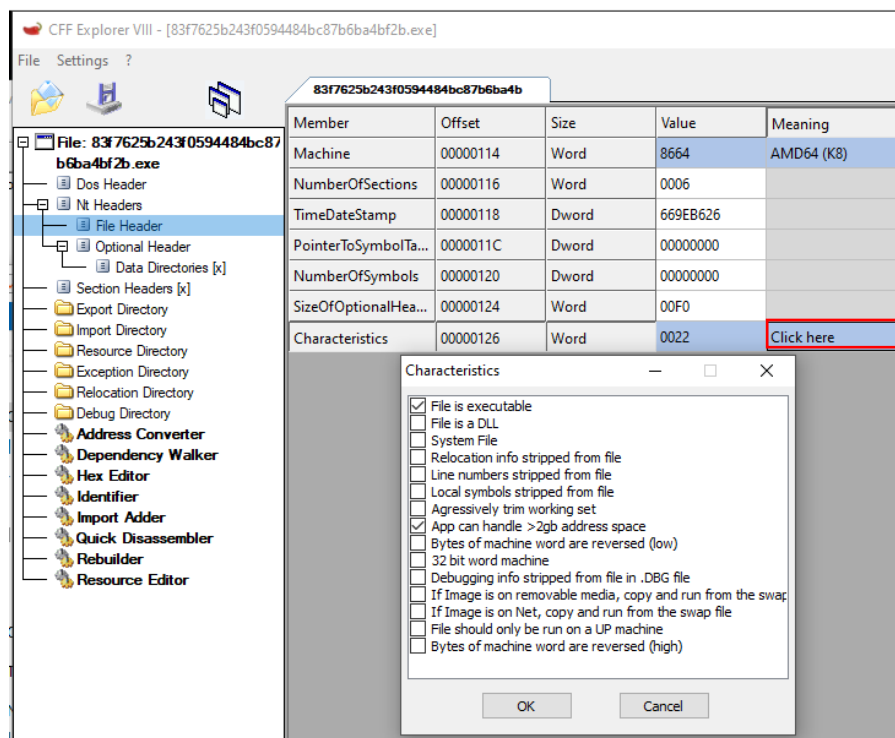
Otra forma que tenemos para verificar si un tipo de fichero contiene varios tipos de ficheros internamente es por la entropía. Para esto, la primera prueba la realizamos con PEiD, pero este programa no lo detecta como PE. Por lo tanto, tampoco nos indica la entropía ni si está empaquetado o no, lo cual podría deberse a que ha sido obfusado o modificado para evadir herramientas de detección.



Por último, para el tema de la entropía utilizamos la herramienta Entroper. Cuanto más cercano esté un archivo al código máquina, menor es su entropía, ya que su estructura es más ordenada y predecible. En este caso, podemos observar varios cambios y picos en los niveles de entropía, lo que sugiere que el archivo contiene distintos tipos de ficheros en su interior.



En la siguiente imagen podemos observar que la muestra de malware analizada posee dos características clave: está diseñada para ejecutarse como un programa independiente (File is executable), lo que le permite realizar acciones autónomas sin depender de otros programas y la segunda es que puede gestionar un espacio de direcciones superior a 2 gigabytes (App can handle >2gb address space). Esta capacidad sugiere que el malware está optimizado para manejar grandes volúmenes de datos o realizar operaciones complejas en sistemas con abundante memoria disponible.



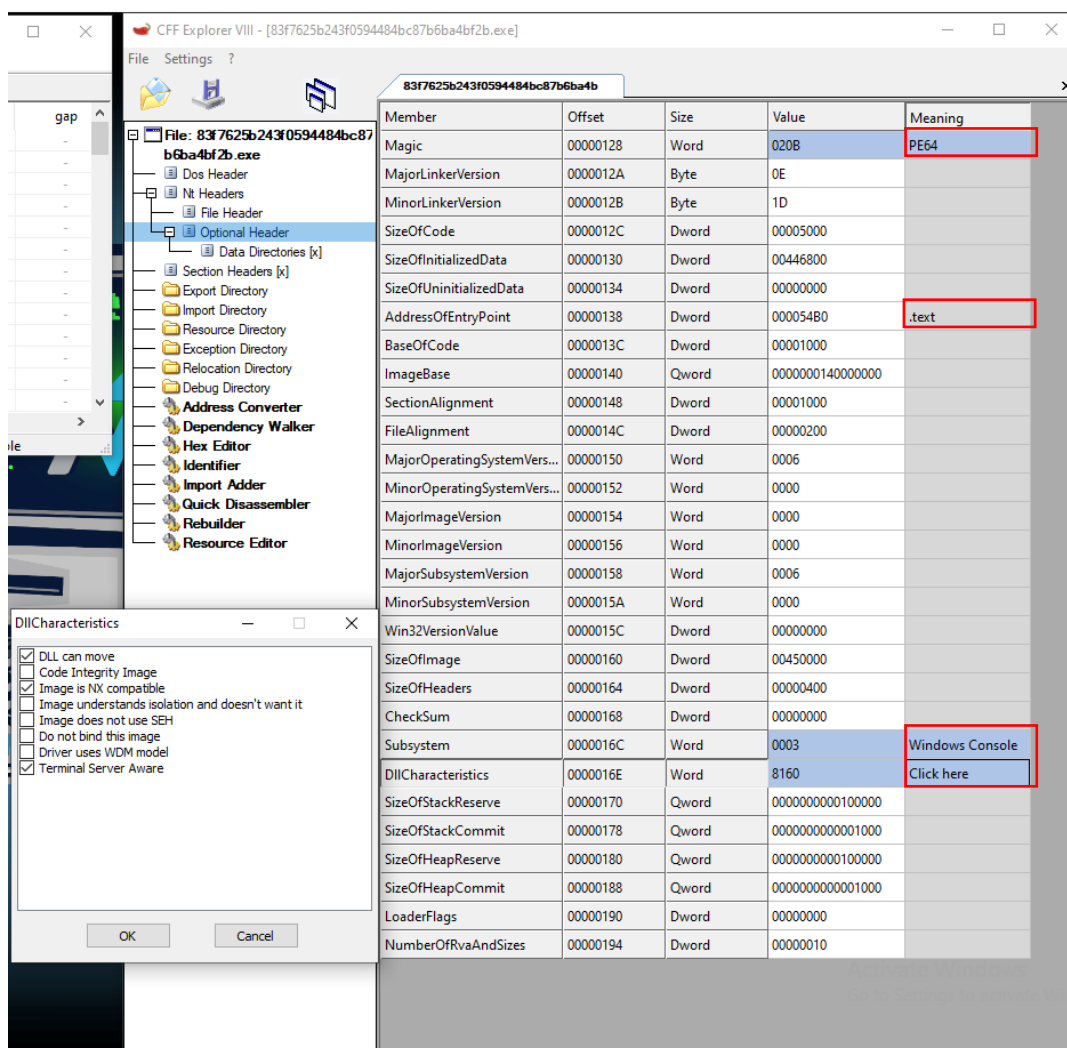
El análisis del encabezado opcional del binario confirma que se trata de un ejecutable de 64 bits, además podemos destacar que:

- **AddressOfEntryPoint:** El punto de entrada está ubicado en la sección .text offset 0x54B0, lo cual nos indica donde comienza la ejecución del código.
- **ImageBase:** Tiene un valor de 00000000140000000 es una dirección base habitual en sistemas de 64 bits, este valor define la dirección virtual base en memoria donde se cargará al momento de su ejecución.
- **Subsystem:** Nos muestra que está configurado como Windows Console, lo cual refuerza la hipótesis de que se trata de un payload ejecutado sin interfaz gráfica, es muy común que los troyanos esten diseñados para operar en segundo plano.

En el campo DllCharacteristics tiene habilitado los siguientes flags:

- **DLL can move:** Es la capacidad de un DLL (Dynamic-Link Library) de ser reubicado en memoria durante su carga, adaptándose a entornos con protecciones como ASLR (Address Space Layout Randomization).

- **NX compatible:** Esto significa que un archivo es compatible con DEP (Data Execution Prevention) para proteger regiones de memoria marcadas como no ejecutables. Un malware podría habilitar esta cabecera para camuflarse como software legítimo y operar en sistemas con protección DEP. Al declararse compatible, evita activar alertas o restricciones iniciales, aumentando sus posibilidades de infiltración.
- **Terminal Server Aware:** Indica que está diseñado para funcionar en entornos de escritorio remoto RDP (Remote Desktop Protocol). En este caso, un vector de ataque frecuente en escenarios de intrusión corporativa y movimientos laterales post-explotación.

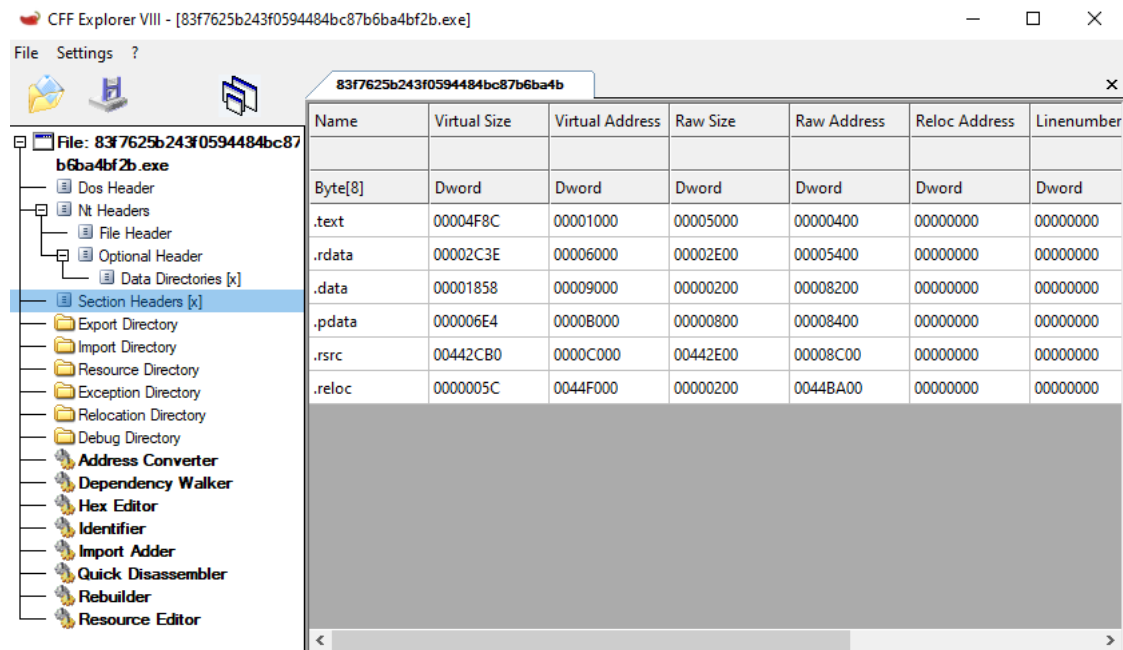


En la pestaña cabeceras de sección tenemos que el fichero se divide en 6 secciones de las cuales vale la pena comparar Virtual Size y Raw Size porque reflejan cómo se

mapean las secciones en memoria y disco. Discrepancias entre ambos pueden indicar técnicas maliciosas como cargas útiles ocultas.

- **.text:** Contiene código ejecutable del programa. El tamaño virtual es ligeramente menor que el tamaño en disco, lo que refleja la alineación de las secciones.
 - **Virtual Size:** 00004F8C
 - **Raw Size:** 00005000
- **.rdata:** Contiene datos de solo lectura, como cadenas constantes. El tamaño virtual es ligeramente menor que el tamaño en disco, lo que refleja la alineación de las secciones.
 - **Virtual Size:** 00002C3E
 - **Raw Size:** 00002E00
- **.data:** Incluye datos inicializados que se usan durante la ejecución. El tamaño virtual es ligeramente menor que el tamaño en disco, lo que refleja la alineación de las secciones.
 - **Virtual Size:** 00001858
 - **Raw Size:** 00002000
- **.pdata:** Contiene tablas para manejar excepciones o procedimientos. El tamaño virtual es ligeramente menor que el tamaño en disco, lo que refleja la alineación de las secciones.
 - **Virtual Size:** 000006E4
 - **Raw Size:** 00000800
- **.rsrc:** Contiene recursos incrustados en el archivo, como iconos o configuraciones. El tamaño virtual es ligeramente menor que el tamaño en disco, lo que refleja la alineación de las secciones.

- **Virtual Size:** 00001A00
- **Raw Size:** 00001C00
- **.reloc:** Contiene información para relocalizar el archivo en memoria, esencial para adaptarse a diferentes ubicaciones al cargar, especialmente en sistemas que emplean protecciones como ASLR. El tamaño virtual es significativamente menor que el tamaño en disco, lo que refleja que solo una fracción de los datos en disco se mapea efectivamente en memoria.
- **Virtual Size:** 0000005C
- **Raw Size:** 00000200

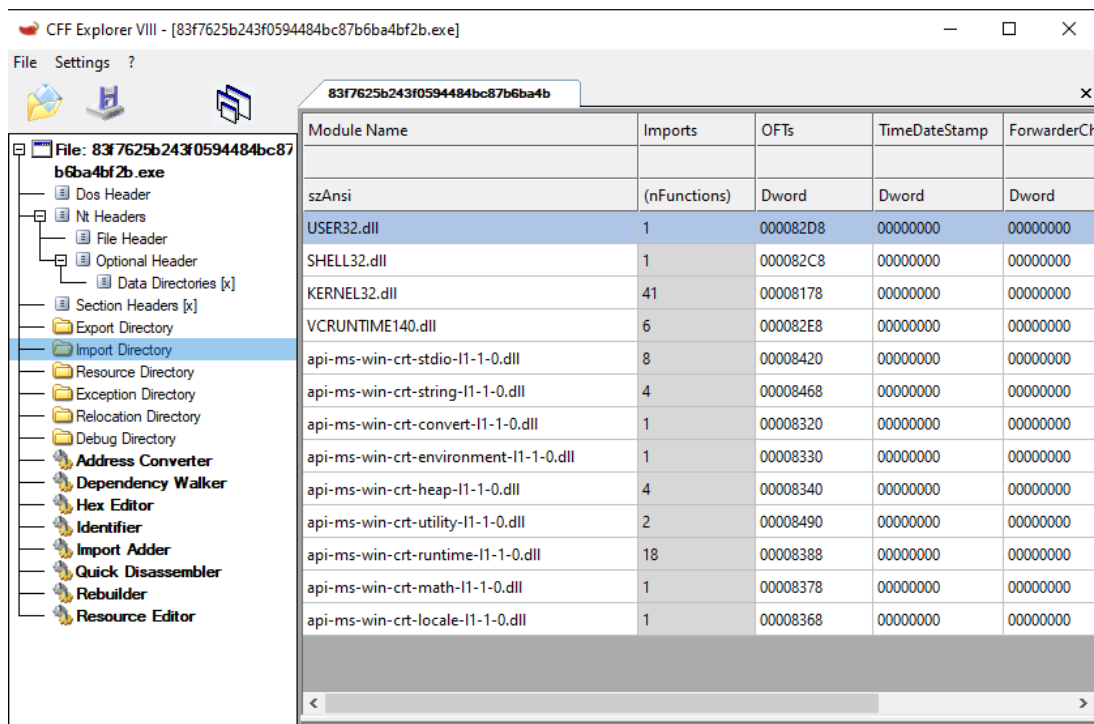


La siguiente imagen muestra el análisis de la pestaña importaciones, donde se destacan las bibliotecas dinámicas (DLL) que el malware requiere para su funcionamiento. A continuación, se describe cada DLL y su posible propósito en el contexto malicioso:

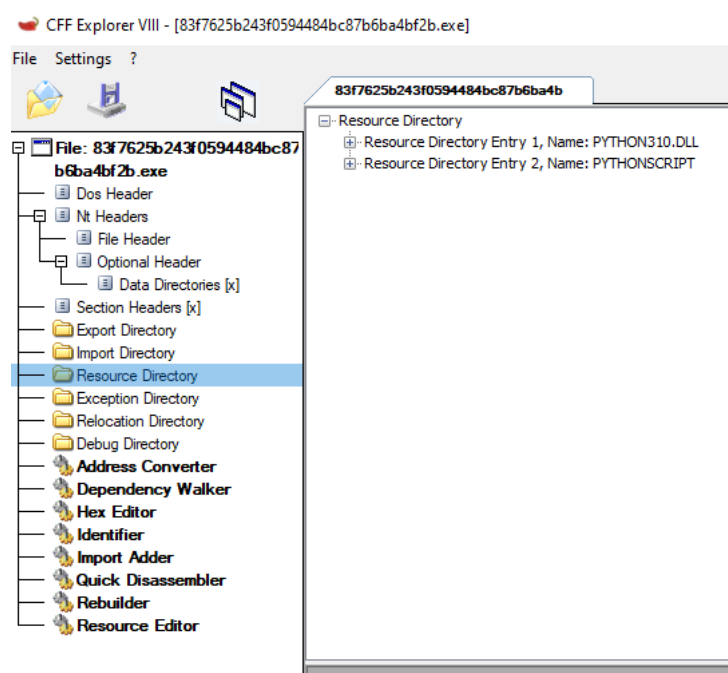
- **USER32.dll:** Permite la manipulación de ventanas y la interacción con el usuario. El malware podría utilizarla para capturar entradas de teclado, gestionar eventos de ratón o mostrar mensajes en pantalla.

- **SHELL32.dll:** Facilita la interacción con el sistema de archivos y la ejecución de comandos del sistema operativo, podría ser útil para abrir, copiar, mover archivos o lanzar procesos maliciosos.
- **KERNEL32.dll:** Proporciona funciones básicas de manejo de memoria, creación de procesos e hilos.
- **VCRUNTIME140.dll:** Contiene rutinas de soporte para aplicaciones compiladas en C++. El malware podría depender de esta biblioteca para ejecutar su lógica interna.
- **api-ms-win-crt-stdio-l1-1-0.dll:** Ofrece funciones de entrada/salida que son necesarias para leer o escribir datos.
- **api-ms-win-crt-string-l1-1-0.dll:** Permite la manipulación de cadenas de texto.
- **api-ms-win-crt-convert-l1-1-0.dll:** Gestiona conversiones de tipos de datos.
- **api-ms-win-crt-environment-l1-1-0.dll:** Proporciona acceso a variables y configuraciones del entorno del sistema operativo, lo que podría ser aprovechado para recolectar información del sistema.
- **api-ms-win-crt-heap-l1-1-0.dll:** Ofrece funciones de gestión dinámica de memoria, permitiendo al malware asignar, liberar y manipular memoria durante su ejecución.
- **api-ms-win-crt-runtime-l1-1-0.dll:** Implementa funciones de tiempo de ejecución, utilizado para el control de flujo del programa y manejo de excepciones.
- **api-ms-win-crt-math-l1-1-0.dll:** Contiene operaciones matemáticas básicas y avanzadas, que podrían ser utilizadas en cálculos relacionados con ofuscación o cifrado.

- **api-ms-win-crt-locale-l1-1-0.dll:** Administra configuraciones de localización e idioma.



En el análisis del directorio de recursos del binario observamos la presencia de los elementos PYTHON310.DLL y PYTHONSCRIPT. La inclusión explícita de la biblioteca PYTHON310.DLL confirma que el ejecutable incorpora un intérprete de Python 3.10, permitiendo la ejecución de scripts en tiempo de ejecución sin necesidad de dependencias externas. Por otra parte, el recurso PYTHONSCRIPT sugiere que la carga útil o lógica principal del malware se encuentra embebida en un script de Python. Además, mediante el editor de recursos podemos extraerlos para un posterior análisis.



PYTHON310.DLL

Hash	Valor
MD5	63a1fa9259a35eaeac04174cecb90048
SHA-1	0dc0c91bcd6f69b80dcdd7e4020365dd7853885a
SHA-256	14b06796f288bc6599e458fb23a944ab0c843e9868058f02a91d4606533505ed

PYTHONSCRIPT

Hash	Valor
MD5	e5b8a4998728123027a7b2560ee3c407
SHA-1	3255329d1cfc1fa17775f3651ab4206c320a57c7
SHA-256	bce0717ce4ec0a42ac4ed6f340049d42809469a7577f256075c05d66c80e8170

El análisis del directorio de excepciones muestra la presencia de múltiples entradas de manejo estructurado de excepciones, consistentes con binarios compilados en entornos Visual Studio modernos. Podemos denotar la implementación intensiva de rutinas "unwind" (flag 0x04) que sugiere un manejo detallado de errores, podemos considerar que esto es bastante común en software legítimo, pero también es utilizado en muestras de malware avanzado para mejorar la estabilidad del código malicioso.

CFF Explorer VIII - [83f7625b243f0594484bc87b6ba4bf2b.exe]

File Settings ?

83f7625b243f0594484bc87b6ba4b

File: 83f7625b243f0594484bc87b6ba4bf2b.exe

- File Header
- Optional Header
- Data Directories [x]
- Section Headers [x]
- Export Directory
- Import Directory
- Resource Directory
- Exception Directory
- Relocation Directory
- Debug Directory
- Address Converter
- Dependency Walker
- Hex Editor
- Identifier
- Import Adder
- Quick Disassembler
- Rebuilder
- Resource Editor

BeginAddress	EndAddress	UnwindData	Flags
Dword	Dword	Dword	Byte: 5
00001010	00001055	000073D8	00
00001060	00001128	000073E4	03
00001130	0000114E	000073FC	00
0000114E	00001179	00007404	04
00001179	00001193	00007418	04
00001193	0000127A	0000742C	04
0000127A	00001329	00007440	04
00001329	0000134C	00007450	04
0000134C	00001351	00007460	04
00001360	000013BC	00007470	00
000013C0	0000140B	00007480	00
0000140B	00001499	0000748C	04
00001499	000014D6	000074A8	04
000014D6	00001684	000074C0	04
00001684	0000168E	000074D4	04
0000168E	000016A4	000074E4	04
000016A4	000016A8	000074F4	04
000016A8	000016AC	00007518	04
000016AC	000016BA	00007534	04
000016BA	000016C3	00007548	04
000016D0	00001AD2	00007558	00
00001AE0	00001DFE	00007560	03
00001DFE	00001F20	0000757C	04
00001F20	000024CC	00007594	04

Al analizar el fichero con PESTudio, se identificaron varios indicadores relevantes.

- **file > embedded > signature:** Identifica la presencia de datos comprimidos en el overlay del ejecutable, en específico el formato PKZIP. Esto es algo que verificamos cuando descomprimos el archivo.
- **file > embedded > signature:** El archivo contiene un recurso incrustado dentro de la sección .rsrc.
- **overlay > file-ratio:** El overlay representa el 47,28% del tamaño total del fichero, lo cual es un porcentaje elevado.
- **overlay > entropy:** El valor de entropía del overlay alcanza 7,997, lo que indica la presencia de compresión o cifrado.

- **string > size > suspicious:** Se detectan dos grupos de cadenas de texto de tamaños anómalos (1409 y 5596 bytes).
- **file > embedded > Python-script:** En la sección .rsrc se encuentra un script Python de 8.894 bytes embebido.
- **string > URL:** Se observan referencias a URLs pertenecientes al dominio ocsp.digicert.com.
- **file > technique:** Se asocian al fichero varias técnicas del marco MITRE ATT&CK.
 - **T1082 - System Information Discovery:** Recopilación de información del sistema huésped.
 - **T1106 - Execution via API:** Uso directo de APIs para ejecución de código.
 - **T1055 - Process Injection:** Posible inyección de código en procesos legítimos.
 - **T1089 - Disabling Security Tools:** Desactivación de herramientas de seguridad locales.
 - **T1204 - User Execution:** Activación tras interacción del usuario.
 - **T1110 - Brute Force:** Posibles intentos de fuerza bruta para obtener credenciales.
- **file > entropy:** El análisis global de entropía del fichero en general es de 7.428.
- **security > protection:** El ejecutable tiene habilitadas las protecciones DEP (Data Execution Prevention) y ASLR (Address Space Layout Randomization), mientras que las protecciones CFG (Control Flow Guard) y CI (Code Integrity) se encuentran desactivadas, lo que podría facilitar técnicas de alteración del flujo de ejecución o la carga de código no firmado durante el proceso de explotación.

pestudio 9.55 - Malware Initial Assessment - www.winitor.com - [c:\shareale\83f7625b243f0594484bc87b6ba4bf2b.exe]

file settings about

indicator (34)	detail	level
file > embedded	signature: PKZIP, location: overlay, offset: 0x0044BC00, size: 4039376 (by...	1
file > embedded	signature: executable, location: .rsrc, offset: 0x00008CD8, size: 4458776 (...)	1
file > extension > count	29	1
imports > flag > count	12	1
overlay > file-ratio	47.28%	2
overlay > size	4039376 bytes	2
overlay > entropy	7.997	2
string > size > suspicious	1409 bytes	2
string > size > suspicious	5596 bytes	2
compiler > stamp	Mon Jul 22 19:42:30 2024	2
directory > stamp	Mon Jul 22 19:42:30 2024	2
file > embedded	signature: Python-script, location: .rsrc, offset: 0x004495F0, size: 8894 (b...	2
debug > stamp	Mon Jul 22 19:42:30 2024	2
file > checksum	0x00000000	2
groups > API	reconnaissance, dynamic-library, synchronization, memory, diagnostic, ...	2
string > URL	http://ocsp.digicert.com0\	2
string > URL	http://ocsp.digicert.com0X	2
string > URL	www.digicert.com1S0"	2
string > URL	http://ocsp.digicert.com0C	2
mitre > technique	T1082, T1106, T1055, T1057, T1124, T1222, T1089, T1105, T1059, T1045, T11...	2
file > entropy	7.428	3
file > footprint	FD92BA4C1D0390761A5AFAD41D2BE3AC98463C1E81D0D7B2E8DFF5450...	3
file > size	8543952 bytes	3
rich-header > checksum	0x681BCF9B	3
rich-header > offset	0x00000080	3
rich-header > footprint	A79EC81D5825D5FA9230A8809EFD470CEB728285F3F5AB30D165C27E791...	3
file > tooling	Visual Studio 2008	3
security > protection	data-execution-prevention (DEP) > ON	3
security > protection	control-flow-guard (CFG) > OFF	3
security > protection	address-space-layout-randomization (ASLR) > ON	3
file > subsystem	console	3
security > protection	code-integrity (CI) > OFF	3
imphash > md5	B588036B5202B7426CD84298C3DECC62	3
file-name > exports	run-py3.10-win-amd64.exe	3

Al observar la pestaña de importaciones vemos que hay algunas que PESTudio las marca como observadas y además nos dice con que técnica de mitre la asocia, de aquí podemos destacar que:

- Se identificaron múltiples llamadas a funciones como VirtualAlloc, VirtualFree, HeapAlloc, HeapFree, y VirtualProtect, pertenecientes a la API de Windows para gestión de memoria, comportamiento que podría indicar que el malware descarga o descifra payloads cuando se ejecuta.
- El uso de funciones como VirtualProtect y VirtualAlloc, en combinación con CreateThread y GetCurrentThread, sugiere que la muestra podría implementar técnicas de inyección de código en procesos, asociadas a la técnica T1055 (Process Injection) del marco MITRE ATT&CK.
- Se observan llamadas a APIs como GetSystemInfo, GetNativeSystemInfo y GetCurrentProcessId, las cuales permiten al malware identificar las características del sistema operativo y de los procesos. Esto está relacionado con las técnicas T1082 (System Information Discovery) y T1057 (Process Discovery).

- La presencia de funciones como SetUnhandledExceptionFilter y GetExceptionCode indica que el binario podría utilizar manejo avanzado de errores para interceptar fallos.
- El empleo de funciones como LoadLibraryA/W y GetProcAddress revela que el binario es capaz de cargar librerías o resolver funciones de manera dinámica en tiempo de ejecución, técnica asociada a la evasión de controles estáticos y clasificada dentro de T1106 (Execution through API).

pestudio 9.55 - Malware Initial Assessment - www.winitor.com - [c:\shareale\83f7625b243f0594484bc87b6ba4bf2b.exe]									
file	settings	about							
indicators (file > embedded)	imports (89)	flag (12)	first-thunk-original (INT)	first-thunk (IAT)	hint	group (17)	technique (5)	type (1)	
footprints (count > 14) *	InitialListHead	-	0x00000000000007DC	0x00000000000007DC	906 (0x036A)	synchronization	-	-	implicit
virustotal (error)	SizeResource	-	0x0000000000000506	0x0000000000000506	1459 (0x05B3)	resource	-	-	implicit
dos-header (size > 64 bytes)	FindResourceA	-	0x000000000000052E	0x000000000000052E	429 (0x01AD)	resource	-	-	implicit
dos-stub (size > 208 bytes)	LockResource	-	0x000000000000053E	0x000000000000053E	1022 (0x03FE)	resource	-	-	implicit
rich-header (tooling > Visual Studio)	LoadResource	-	0x000000000000056E	0x000000000000056E	1002 (0x03EA)	resource	-	-	implicit
file-header (compiler-stamp > Jul.2024)	SHGetSpecialFolderPathW	-	0x000000000000084C2	0x000000000000084C2	366 (0x016E)	reconnaissance	-	-	implicit
optional-header (subsystem > console)	IsDebuggerPresent	-	0x000000000000087E2	0x000000000000087E2	928 (0x03A0)	reconnaissance	T1082 System Information Discovery	-	implicit
directories (count > 8)	GetNativeSystemInfo	✖	0x0000000000000856A	0x0000000000000856A	674 (0x02A2)	reconnaissance	-	-	implicit
sections (files > 3)	IsProcessorFeaturePresent	-	0x00000000000008780	0x00000000000008780	936 (0x03A8)	reconnaissance	-	-	implicit
libraries (count > 13) *	QueryPerformanceCounter	-	0x000000000000087C0	0x000000000000087C0	1136 (0x0470)	reconnaissance	-	-	implicit
imports (flag > 89)	GetCurrentProcessId	✖	0x00000000000008796	0x00000000000008796	563 (0x0233)	reconnaissance	T1057 Process Discovery	-	implicit
exports (items > 65)	LocalFree	-	0x000000000000094F8	0x000000000000094F8	1010 (0x03F2)	memory	-	-	implicit
thread-local-storage (n/a)	VirtualProtect	✖	0x0000000000000939E	0x0000000000000939E	1541 (0x0605)	memory	T1055 Process Injection	-	implicit
.NET (n/a)	HeapFree	-	0x000000000000095B0	0x000000000000095B0	880 (0x0370)	memory	-	-	implicit
resources (signature > executable)	VirtualFree	-	0x000000000000095CC	0x000000000000095CC	1538 (0x0602)	memory	T1055 Process Injection	-	implicit
strings (count > 229121)	VirtualAlloc	✖	0x000000000000095DA	0x000000000000095DA	1535 (0x05FF)	memory	T1055 Process Injection	-	implicit
debug (stamp > Jul.2024)	HeapAlloc	-	0x00000000000009600	0x00000000000009600	876 (0x036C)	memory	-	-	implicit
manifest (n/a)	GetProcessHeap	-	0x0000000000000981E	0x0000000000000981E	724 (0x02D4)	memory	-	-	implicit
version (n/a)	RtlVirtualUnwind	-	0x0000000000000986A	0x0000000000000986A	1284 (0x0504)	memory	-	-	implicit
certificate (n/a)	memset	-	0x0000000000000986A	0x0000000000000986A	62 (0x003E)	memory	-	-	implicit
overlay (signature > PKZIP)	memcpy	-	0x00000000000009C34	0x00000000000009C34	60 (0x003C)	memory	-	-	implicit
	malloc	-	0x00000000000009938	0x00000000000009938	25 (0x0019)	memory	-	-	implicit
	GetSystemTimeAsFileTime	-	0x000000000000087C2	0x000000000000087C2	778 (0x030A)	file	T1124 System Time Discovery	-	implicit
	GetCurrentThreadId	✖	0x000000000000087AC	0x000000000000087AC	567 (0x0237)	execution	T1057 Process Discovery	-	implicit
	RtlCaptureContext	-	0x000000000000086BC	0x000000000000086BC	1269 (0x04F5)	execution	-	-	implicit
	RtlLookupFunctionEntry	✖	0x000000000000086D0	0x000000000000086D0	1277 (0x04FD)	execution	-	-	implicit
	GetCurrentProcess	✖	0x00000000000008738	0x00000000000008738	562 (0x0232)	execution	T1057 Process Discovery	-	implicit
	TerminateProcess	✖	0x0000000000000874C	0x0000000000000874C	1476 (0x05C4)	execution	-	-	implicit
	UnhandledExceptionFilter	-	0x000000000000086FE	0x000000000000086FE	1510 (0x05E6)	exception	-	-	implicit
	SetUnhandledExceptionFilter	-	0x0000000000000871A	0x0000000000000871A	1444 (0x05A4)	exception	-	-	implicit
	GetProcAddress	-	0x0000000000000860C	0x0000000000000860C	717 (0x02CD)	dynamic-library	-	-	implicit
	GetModuleFileNameW	-	0x00000000000008518	0x00000000000008518	657 (0x0291)	dynamic-library	-	-	implicit
	LoadLibraryA	-	0x0000000000000854E	0x0000000000000854E	996 (0x03E4)	dynamic-library	T1106 Execution through API	-	implicit
	FreeLibrary	-	0x0000000000000857E	0x0000000000000857E	453 (0x01C5)	dynamic-library	-	-	implicit
	LoadLibraryExW	-	0x0000000000000858C	0x0000000000000858C	998 (0x03E6)	dynamic-library	T1106 Execution through API	-	implicit
	GetModuleHandleExW	✖	0x00000000000008640	0x00000000000008640	660 (0x0294)	dynamic-library	-	-	implicit
	GetModuleHandleA	-	0x00000000000008656	0x00000000000008656	658 (0x0292)	dynamic-library	-	-	implicit
	SetDllDirectoryA	✖	0x0000000000000866A	0x0000000000000866A	1343 (0x053F)	dynamic-library	T1055 Process Injection	-	implicit
	SetDllDirectoryW	✖	0x0000000000000867E	0x0000000000000867E	1344 (0x0540)	dynamic-library	T1055 Process Injection	-	implicit
	GetModuleHandleW	-	0x000000000000086A8	0x000000000000086A8	661 (0x0295)	dynamic-library	-	-	implicit
	FormatMessageA	-	0x000000000000084F4	0x000000000000084F4	448 (0x01C0)	diagnostic	-	-	implicit

Entre los recursos se encuentran Python310.dll ubicado en .rsrc0x0008CD8, que constituye el 52,19% del tamaño total del archivo y PythonScript localizado en .rsrc0x0004495F0 que constituye el 0,10%. Cabe la posibilidad es que PythonScript sea un fichero incrustado que utiliza las funciones de Python310.dll.

pestudio 9.55 - Malware Initial Assessment - www.winitor.com - [c:\shareale\83f7625b243f0594484bc87b6ba4bf2b.exe]

file settings about

	name	instance	signature	location	size (4467670 byt...	file-ratio (52.29%)	footprint (sha256)	entropy	language
indicators (file > embedded)									
footprints (count > 14)									
indicators (count)									
dos-header (size > 64 bytes)									
dos-stub (size > 208 bytes)									
rich-header (tooling > Visual Studio)									
file-header (compiler-stamp > Jul.2024)									
optional-header (subsystem > console)									
directories (count > 8)									
sections (files > 3)									
libraries (count > 13)									
imports (flag > 89)									
exports (items > 63)									
thread-local-storage (n/a)									
JNET (n/a)									
resources (signature > executable)									
strings (count > 229121)									
debug (stamp > Jul.2024)									
manifest (n/a)									
version (n/a)									
certificate (n/a)									
overlay (signature > PKZIP)									

Al observar la pestaña strings en PESTudio, se identifican varias llamadas y comandos que son asociados a técnicas del marco MITRE ATT&CK. De esta información podemos destacar que:

- Se detectan llamadas a funciones que manipulan el registro de Windows, como RegDeleteValue, RegDeleteKey, RegSetValue, y RegCreateKey, asociadas a las técnicas T1485 (Data Destruction) y T1112 (Modify Registry). Estas acciones podrían ser utilizadas para destruir información, alterar configuraciones del sistema o instalar persistencia maliciosa.
- La presencia de funciones de manipulación de privilegios como OpenProcessToken, LookupPrivilegeValue, y AdjustTokenPrivileges, sugiere que el ejecutable intenta escalar privilegios locales, relacionado con la técnica T1134 (Access Token Manipulation).
- Se identificaron funciones de ejecución de comandos, como CreateProcess y ShellExecute, las cuales permiten la ejecución de código adicional durante la ejecución, esto es asociado a la técnica T1106 (Execution through API).
- Se observan llamadas que permiten cargar dinámicamente librerías, como LoadLibraryA, LoadLibraryExW y GetProcAddress, comportamiento habitual en malware que busca evadir detecciones estáticas. Esto se clasifica también bajo T1106 (Execution through API).
- Se detectan indicadores de manipulación y ejecución de archivos y/o directorios, como FindFirstFile, FindNextFile, MoveFileEx. Estas funciones

están relacionadas con T1005 (Data from Local System) y T1083 (File and Directory Discovery).

- Existen referencias a técnicas de evasión y descubrimiento de entorno, como `IsDebuggerPresent`, `ExpandEnvironmentStrings`, y `GetCurrentProcessId`, lo que indica posibles mecanismos para detectar que el programa está siendo ejecutado en un entorno de depuración, vinculados a T1082 (System Information Discovery) y T1057 (Process Discovery).
- Se evidencia la intención de desactivar mecanismos de seguridad, mediante comandos y cadenas que intentan "matar" procesos o eliminar información, asociados a T1089 (Disabling Security Tools).

pestudio 9.55 - Malware Initial Assessment - www.winitor.com - [c:\shareale\83f7625b243f0594484bc87b6ba4bf2b.exe]								
	encoding (2)	size (bytes)	location	flag (50)	label (1727)	group (17)	technique (19)	value (229121)
indicators (file > embedded)	ASCII	14	JSPC	X	-	registry	T1485 Data Destruction	RegDeleteValue
footprints (count > 14) *	ASCII	12	JSPC	X	-	registry	T1485 Data Destruction	RegDeleteValue
virtualcall (error)	ASCII	14	JSPC	-	-	registry	T1485 Data Destruction	RegDeleteKeyEx
dos-header (size > 64 bytes)	ASCII	10	JSPC	X	-	file	T1485 Data Destruction	DeleteFile
rich-header (size > 208 bytes)	ASCII	5	JSPC	-	utility	-	T1222 File Permissions Modification	chmod
rich-header (tooling > Visual Studio)	ASCII	18	JSPC	-	library	security	T1134 Access Token Manipulation	SeRestorePrivilege
file-header (compiler-stamp > Jul.2024)	ASCII	16	JSPC	X	-	security	T1134 Access Token Manipulation	OpenProcessToken
optional-header (subsystem > console)	ASCII	20	JSPC	X	-	security	T1134 Access Token Manipulation	LookupPrivilegeValue
directories (count > 8)	ASCII	21	JSPC	X	-	security	T1134 Access Token Manipulation	AdjustTokenPrivileges
sections (files > 3)	ASCII	23	JSPC	-	import	file	T1134 System Time Discovery	GetSystemTimeofFileTime
libraries (count > 13) *	ASCII	23	JSPC	-	import	file	T1134 System Time Discovery	GetSystemTimeofFileTime
imports (flag > 89) *	ASCII	11	JSPC	X	-	registry	T1112 Modify Registry	RegSetValue
exports (items > 65)	ASCII	12	JSPC	X	-	registry	T1112 Modify Registry	RegCreateKey
thread-local-storage (n/a)	ASCII	13	JSPC	X	-	registry	T1112 Modify Registry	RegSetValueEx
.NET (n/a)	ASCII	10	JSPC	X	-	registry	T1112 Modify Registry	RegSaveKey
resources (signature > executable)	ASCII	14	JSPC	X	-	registry	T1112 Modify Registry	RegCreateKeyEx
strings (count > 229121)	ASCII	11	JSPC	X	-	registry	T1112 Modify Registry	RegFlushKey
debug (stamp > Jul.2024)	ASCII	13	JSPC	X	-	execution	T1106 Execution through API	CreateProcess
manifest (n/a)	ASCII	12	JSPC	X	-	execution	T1106 Execution through API	ShellExecute
version (n/a)	ASCII	13	JSPC	X	-	execution	T1106 Execution through API	CreateProcess
certificate (n/a)	ASCII	11	JSPC	-	import	dynamic-library	T1106 Execution through API	LoadLibrary
overlay (signature > PKZIP)	ASCII	13	JSPC	-	import	dynamic-library	T1106 Execution through API	LoadLibraryEx
	ASCII	11	JSPC	-	import	dynamic-library	T1106 Execution through API	LoadLibrary
	ASCII	13	JSPC	-	import	dynamic-library	T1106 Execution through API	LoadLibraryEx
	ASCII	10	JSPC	X	-	file	T1105 Remote File Copy	MoveFileEx
	ASCII	24	JSPC	-	utility	compression	T1105 Remote File Copy	Expand environment vars.
	ASCII	6	JSPC	-	utility	compression	T1105 Remote File Copy	expand
	ASCII	29	JSPC	-	utility	-	T1089 Disabling Security Tools	Kill a process with a signal.
	ASCII	4	JSPC	-	utility	-	T1089 Disabling Security Tools	kill
	ASCII	13	JSPC	X	-	file	T1083 File and Directory Discovery	FindFirstFile
	ASCII	12	JSPC	X	-	file	T1083 File and Directory Discovery	FindNextFile
	ASCII	17	JSPC	-	import	reconnaissance	T1082 System Information Discovery	IsDebuggerPresent
	ASCII	17	JSPC	-	import	reconnaissance	T1082 System Information Discovery	IsDebuggerPresent
	ASCII	24	JSPC	-	-	-	T1082 System Information Discovery	ExpandEnvironmentStrings
	ASCII	24	JSPC	-	-	-	T1082 System Information Discovery	ExpandEnvironmentStrings
	ASCII	3	JSPC	-	utility	-	T1059 Command-Line Interface	cmd
	ASCII	8	JSPC	-	utility	-	T1059 Command-Line Interface	cmd /c *
	ASCII	3	overlay	-	utility	-	T1059 Command-Line Interface	cMd
	ASCII	3	overlay	-	utility	-	T1059 Command-Line Interface	Cmd
	ASCII	19	JSPC	X	import	reconnaissance	T1057 Process Discovery	GetCurrentProcessId
	ASCII	19	JSPC	X	import	reconnaissance	T1057 Process Discovery	GetCurrentProcessId

Analizamos los recursos encontrados utilizando el comando `file`, obteniendo que PYTHON310.DLL corresponde a un archivo PE32+ ejecutable (DLL) diseñado para sistemas Windows de 64 bits, mientras que PYTHONSCRIPT se identifica como un archivo de tipo data, lo cual sugiere que puede contener datos personalizados o potencialmente cifrados.

```
remnux@remnux:~/Documents$ file PYTHON310-DLL
PYTHON310-DLL: PE32+ executable (DLL) (GUI) x86-64, for MS Windows
remnux@remnux:~/Documents$ file PYTHONSCRIPT
PYTHONSCRIPT: data
```

Al realizar el análisis con ClamScan sobre la muestra principal y los dos recursos asociados, la herramienta no logró identificar estos archivos como maliciosos.

```
remnux@remnux:~/theJungle/malware$ clamscan 83f7625b243f0594484bc87b6ba4bf2b.malware
/home/remnux/theJungle/malware/83f7625b243f0594484bc87b6ba4bf2b.malware: OK

----- SCAN SUMMARY -----
Known viruses: 8705951
Engine version: 0.103.12
Scanned directories: 0
Scanned files: 1
Infected files: 0
Data scanned: 30.93 MB
Data read: 8.14 MB (ratio 3.80:1)
Time: 69.311 sec (1 m 9 s)
Start Date: 2025:04:12 07:38:33
End Date: 2025:04:12 07:39:43
```

Se identificó un archivo embebido denominado cookiejar.pyc, correspondiente al módulo http.cookiejar de la librería estándar de Python. En el contexto de esta muestra, su inclusión sugiere que el malware podría estar utilizando funciones de gestión de cookies para los siguientes escenarios:

- Interceptar o almacenar cookies de sesión HTTP
- Mantener sesiones persistentes con servidores de comando y control (C2)
- Simular tráfico web legítimo o autenticado

Asimismo, al tratarse de un archivo .pyc embebido y potencialmente modificado, no se descarta que contenga código adicional malicioso orientado a la exfiltración de credenciales, persistencia o comunicación encubierta.

```
remnux@remnux:~/theJungle/malware$ rabin2 -zzz 83f7625b243f0594484bc87b6ba4bf2b.malware | grep cookiejar.pyc
105973 0x0066baff 0x14066eeff 21 23 () utf8 http://cookiejar.pycuy|
141846 0x0082364a 0x140826a4a 20 21 () ascii http://cookiejar.pycPK
remnux@remnux:~/theJungle/malware$
```

Entre las cadenas analizadas con la herramienta rabin2, se observó una referencia al fichero cmd.pyc, un script de Python compilado que podría estar relacionado con actividades maliciosas. También se identificaron otras cadenas asociadas al procesamiento de comandos, como cmd /c, config_parse_cmdlin y core_read_precmdline. El uso de cmd /c sugiere que el malware podría ejecutar comandos del sistema mediante el intérprete de comandos de Windows, permitiendo realizar acciones arbitrarias en el sistema comprometido. Por otro lado, config_parse_cmdline parece estar diseñado para interpretar los argumentos de línea de comandos, mientras que core_read_precmdline se utiliza para preparar

configuraciones iniciales antes de procesar dichos parámetros. Finalmente, es bastante particular la aparición de la cadena C:\\Windows\\explorer.exez\\t\\10.0.1.1z\\bcmd /c, que imita la ruta de un ejecutable legítimo del sistema (explorer.exe) pero con una extensión modificada (.exe), e incluye una dirección IP malformada 10.0.1.1z.

```
remnux@remnux:~/theJungle/malware$ rabin2 -zzz 83f7625b243f0594484bc87b6ba4bf2b.malware | grep -Ei 'cmd'
39232 0x0023f488 0x140242888 4 10 (.rsrc) utf16le .cmd
42615 0x0026e8b8 0x140271cb8 1315 1316 (.rsrc) ascii startfile($module, /, filepath, operation=<unrepresentable>,\n arguments=<unrepresentable>, cwd=
None, show_cmd=1)\n-\n\nstart a file with its associated application.\n\nWhen "operation" is not specified or "open", this acts like\ndouble-clicking the fil
e in Explorer, or giving the file name as an\argument to the DOS "start" command: the file is opened with whatever\napplication (if any) its extension is ass
ociated.\nWhen another "operation" is given, it specifies what should be done with\nthe file. A typical operation is "print".\n\n\narguments" is passed to the
application, but should be omitted if the\nfile is a document.\n\n\n"cwd" is the working directory for the operation. If "filepath" is\nrelative, it will be re
solved against this directory. This argument\nshould usually be an absolute path.\n\n\nshow_cmd" can be used to override the recommended visibility option.\n5se
e the Windows ShellExecute documentation for values.\n\n\nstartfile returns as soon as the associated application is launched.\n\nThere is no option to wait for t
he application to close, and no way\nto retrieve the application's exit status.\n\n\nThe filepath is relative to the current directory. If you want to use\nan
absolute path, make sure the first character is not a slash ("/");\n\nthe underlying Win32 ShellExecute function doesn't work if it is.
54829 0x00321cdd 0x140325d0d 15 16 (.rsrc) ascii k%zmcmdmvm\rmam
54815 0x00383ad8 0x140386ed8 591 592 (.rsrc) ascii Options and arguments (and corresponding environment variables):\n-b : issue warnings about str(bytes i
nstance), str(bytearray instance)\n and comparing bytes/bytearray with str. (-bb: issue errors)\n-B : don't write .pyc files on import; also PYTHO
NDONTWRITEBYTECODE=x\n-c cmd : program passed in as string (terminates option list)\n-d : turn on parser debugging output (for experts only, only works on
\n debug builds); also PYTHONDEBUG=x\n-E : ignore PYTHON* environment variables (such as PYTHONPATH)\n-h : print this help message and exit (a
iso -? or --help)\n
54816 0x00383d28 0x140387128 63 64 (.rsrc) ascii usage: %s [option] ... [-c cmd | -m mod | file | -] [arg] ... \n
54821 0x00384818 0x140387c18 3007 3008 (.rsrc) ascii -u : force the stdout and stderr streams to be unbuffered;\n this option has no effect on std
in; also PYTHONUNBUFFERED=x\n-v : verbose (trace import statements); also PYTHONVERBOSE=x\n can be supplied multiple times to increase verbosity\n
-V : print the Python version number and exit (also --version)\n when given twice, print more information about the build\n-n-w arg : warning contro
l; arg is action:message:category:module:lineno\n also PYTHONWARNINGS=arg\n-x : skip first line of source, allowing use of non-Unix forms of #!cmd
\n-X opt : set implementation-specific option. The following options are available:\n -X faulthandler: enable faulthandler\n -X showrefcount:
output the total reference count and number of used\n memory blocks when the program finishes or after each statement in the\n interac
tive interpreter. This only works on debug builds\n -X tracemalloc: start tracing Python memory allocations using the\n tracemalloc module
. By default, only the most recent frame is stored in a\n traceback of a trace. Use -X tracemalloc=NFRAME to start tracing with a\n tr
aceback limit of NFRAME frames\n -X importtime: show how long each import takes. It shows module name,\n cumulative time (including nested
imports) and self time (excluding\n nested imports). Note that its output may be broken in multi-threaded\n application. Typical usag
e is python3 -X importtime -c 'import asyncio'\n -X dev: enable CPython's "development mode", introducing additional runtime\n checks whic
h are too expensive to be enabled by default. Effect of the\n developer mode:\n * Add default warning filter, as -W default\n
* Install debug hooks on memory allocators: see the PyMem_SetupDebugHooks()\n C function\n * Enable the faulthandler
module to dump the Python traceback on a crash\n * Enable asyncio debug mode\n * Set the dev_mode attribute of sys.flags to Tru
e\n * io.IOBase destructor logs close() exceptions\n -X utf8: enable UTF-8 mode for operating system interfaces, overriding the default
\n locate-aware mode. -X utf8=0 explicitly disables UTF-8 mode (even when it would\n otherwise activate automatically)\n -X py
cache.prefix=PATH: enable writing .pyc files to a parallel tree rooted at the\n given directory instead of to the code tree\n -X warn_defa
ult.encoding: enable opt-in EncodingWarning for 'encoding=None'\n -X int_max_str_digits=number: limit the size of int->str conversions.\n
This helps avoid denial of service attacks when parsing untrusted data.\n The default is sys.int_info.default_max_str_digits. 0 disables.\n\n-c
heck-hash-based-pycs always[default]never:\n control how Python invalidates hash-based .pyc files\n
55476 0x0038cc88 0x1403900b8 8 9 (.rsrc) ascii show cmd
57297 0x0039d6e0 0x1403a0ae0 20 21 (.rsrc) ascii config parse cmdline
57303 0x0039d788 0x1403a0b88 20 21 (.rsrc) ascii core_read_pre cmdline
61441 0x0044b771 0x14044eb71 44 45 (.rsrc) ascii C:\\Windows\\explorer.exez\\t\\10.0.1.1z\\bcmd /c "
72371 0x004cf214 0x1404d2014 7 8 () ascii cmd.pyc
91499 0x005bb041 0x1405be441 4 5 () ascii @cmd
```

Se identificaron referencias a las funciones OpenProcessToken y AdjustTokenPrivileges. OpenProcessToken permite al malware obtener el token de acceso asociado a un proceso, lo que le proporciona información sobre los permisos y privilegios del mismo. Por otro lado, AdjustTokenPrivileges se utiliza para modificar los privilegios de un token. Esto puede indicar que el malware intenta elevar sus privilegios y manipular procesos.

```

remnux@remnux:~/theJungle/malware$ rabin2 -zzz 83f7625b243f0594484bc87b6ba4bf2b.malware | grep -Ei 'token'
41861 0x00256088 0x140259488 29 30 (.rsrc) ascii tokenizer beginning of buffer
41931 0x00256d78 0x14025a178 15 16 (.rsrc) ascii get_cache_token
41943 0x00256d78 0x14025a178 26 27 (.rsrc) ascii get_cache_token(smodule, /)/\n-\nReturns the current ABC cache token.\n\nThe token is an opaque object (s
upporting equality testing) identifying the\ncurrent version of the ABC cache for virtual subclasses. The token changes\nevery call to register() on any
ABC.
50989 0x00352bc8 0x140355fc8 293 294 (.rsrc) ascii set($self, value, /)/\n-\n\ncall to set a new value for the context variable in the current context.\n\nThe
required value argument is the new value for the context variable.\n\nReturns a Token object that can be used to restore the variable to its previous\value
via the 'ContextVar.reset()' method.
50991 0x00352e78 0x140356278 160 161 (.rsrc) ascii reset($self, token, /)/\n-\n\ncall to reset the context variable.\n\nThe variable is reset to the value it had befo
re the 'ContextVar.set()' that\ncreated the token was used.
51297 0x00356c22 0x14035a022 11 12 (.rsrc) ascii \btokenizer[
54263 0x00379148 0x14037c548 18 19 (.rsrc) ascii contextvars.Token
54264 0x00379160 0x14037c560 13 14 (.rsrc) ascii Token.MISSING
54470 0x00379c08 0x14037d008 8 9 (.rsrc) ascii tokenize
54471 0x00379c14 0x14037d014 5 6 (.rsrc) ascii token
56718 0x00397f88 0x14039b388 12 13 (.rsrc) ascii <ERRORTOKEN>
56722 0x00397fc0 0x14039b3c0 10 11 (.rsrc) ascii <N TOKENS>
56744 0x00398190 0x14039b590 13 14 (.rsrc) ascii invalid token
57131 0x0039be78 0x14039f278 33 34 (.rsrc) ascii an instance of Token was expected
57142 0x0039bff0 0x14039f3f0 37 38 (.rsrc) ascii expected an instance of Token, got %R
57143 0x0039c018 0x14039f418 41 42 (.rsrc) ascii Tokens can only be created by ContextVars
57144 0x0039c044 0x14039f444 6 7 (.rsrc) ascii <token
57147 0x0039c060 0x14039f460 15 16 (.rsrc) ascii <Token.MISSING>
59165 0x003d8c1f 0x1403dc01f 19 20 (.rsrc) ascii PyContextToken_Type
59845 0x003dc095 0x1403df495 15 16 (.rsrc) ascii PyToken_OneChar
59846 0x003dc0a5 0x1403df4a5 18 19 (.rsrc) ascii PyToken_ThreeChars
59847 0x003dc0b8 0x1403df4b8 16 17 (.rsrc) ascii PyToken_TwoChars
60377 0x003dee7b 0x1403e227b 20 21 (.rsrc) ascii PyParserTokenNames
60668 0x003e13be 0x1403e47be 16 17 (.rsrc) ascii OpenProcessToken
60672 0x003e1408 0x1403e4808 21 22 (.rsrc) ascii AdjustTokenPrivileges
61012 0x00403e78 0x140441b78 1349 1350 (.rsrc) ascii <?xml version="1.0" encoding="UTF-8" standalone="yes"?>\r\n<assembly xmlns="urn:schemas-microsoft-com:asm
.v1" manifestVersion="1.0">\r\n <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">\r\n <security>\r\n <requestedPrivileges>\r\n <requestedEx
ecutionLevel level="asInvoker" uiAccess="false"/>\r\n </requestedPrivileges>\r\n </trustInfo>\r\n <compatibility xmlns="urn:schemas-
microsoft-com:compatibility.v1">\r\n <application>\r\n <supportedOS Id="{e2011457-1546-43c5-a5fe-0080deee3d3f}">\r\n <supportedOS Id="{35138b9a-
5d96-4fbd-b2d2-a244025f93a}">\r\n <supportedOS Id="{4a2f28e3-53b9-4441-ba9c-d69d4a4a4e38}">\r\n <supportedOS Id="{1f676c76-80e1-4239-95bb-83d0f6d
0da78}">\r\n <supportedOS Id="{8e0f7a12-bf63-4fe8-b9a5-48fd50a15a9a}">\r\n </application>\r\n </compatibility>\r\n <application xmlns="urn:schemas-
microsoft-com:asm.v3">\r\n <windowsSettings>\r\n <longPathAware xmlns="http://schemas.microsoft.com/SMI/2016/WindowsSettings">true</longPathAware>\r\n
n </windowsSettings>\r\n </application>\r\n <dependency>\r\n <dependentAssembly>\r\n <assemblyIdentity type="win32" name="Microsoft.Windows.Commo
n-Controls">\r\n
version="6.0.0.0" processorArchitecture="" publicKeyToken="6595b64144ccf1df" language="" />\r\n </dependentAssembl

```

Durante el análisis con rabin2, se identificaron referencias al protocolo XML-RPC, el cual es utilizado para realizar llamadas a procedimientos remotos mediante el intercambio de datos estructurados en formato XML sobre HTTP. Esto puede indicar que este está diseñado para comunicarse con un servidor de comando y control.

```

remnux@remnux:~/theJungle/malware$ rabin2 -zzz 83f7625b243f0594484bc87b6ba4bf2b.malware | grep -Ei 'xml'
41295 0x00249f08 0x14024db38 17 18 (.rsrc) ascii xmlcharrefreplace
50025 0x003259f8 0x140328df8 398 399 (.rsrc) ascii encode($self, /, input, errors=None)\n-\n\nReturn an encoded string version of 'input'.\n\n'errors' may be
given to set a different error handling scheme. Default is\n'strict' meaning that encoding errors raise a UnicodeEncodeError. Other possible\nvalues are 'ign
ore', 'replace' and 'xmlcharrefreplace' as well as any other name\nregistered with codecs.register error that can handle UnicodeEncodeErrors.
50847 0x003484f8 0x14034b8f8 510 511 (.rsrc) ascii encode($self, /, encodings='utf-8', errors='strict')\n-\n\nEncode the string using the codec registered for
encoding.\n\n encoding\n The encoding in which to encode the string.\n errors\n The error handling scheme to use for encoding errors.\n The default
t is 'strict' meaning that encoding errors raise a\n UnicodeEncodeError. Other possible values are 'ignore', 'replace' and\n 'xmlcharrefreplace' as wel
l as any other name registered with\n codecs.register error that can handle UnicodeEncodeErrors.
54252 0x00378f00 0x14037c300 24 25 (.rsrc) ascii xmlcharrefreplace errors
54253 0x00378f28 0x14037c328 136 137 (.rsrc) ascii Implements the 'xmlcharrefreplace' error handling, which replaces an unencodable character with the appropr
iate XML character reference.
54487 0x00379c08 0x14037d0b8 6 7 (.rsrc) ascii xmlrpc
56522 0x00396618 0x140399a18 17 36 (.rsrc) utf16le xmlcharrefreplace
59147 0x003d8a80 0x1403dbe80 31 32 (.rsrc) ascii PyCodec.XMLCharRefReplaceErrors
61012 0x00403e78 0x140441b78 1349 1350 (.rsrc) ascii <?xml version="1.0" encoding="UTF-8" standalone="yes"?>\r\n<assembly xmlns="urn:schemas-microsoft-com:asm
.v1" manifestVersion="1.0">\r\n <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">\r\n <security>\r\n <requestedPrivileges>\r\n <requestedEx
ecutionLevel level="asInvoker" uiAccess="false"/>\r\n </requestedPrivileges>\r\n </trustInfo>\r\n <compatibility xmlns="urn:schemas-
microsoft-com:compatibility.v1">\r\n <applications>\r\n <supportedOS Id="{e2011457-1546-43c5-a5fe-0080deee3d3f}">\r\n <supportedOS Id="{35138b9a-
5d96-4fbd-b2d2-a244025f93a}">\r\n <supportedOS Id="{4a2f28e3-53b9-4441-ba9c-d69d4a4a4e38}">\r\n <supportedOS Id="{1f676c76-80e1-4239-95bb-83d0f6d
0da78}">\r\n <supportedOS Id="{8e0f7a12-bf63-4fe8-b9a5-48fd50a15a9a}">\r\n </application>\r\n </compatibility>\r\n <application xmlns="urn:schemas-
microsoft-com:asm.v3">\r\n <windowsSettings>\r\n <longPathAware xmlns="http://schemas.microsoft.com/SMI/2016/WindowsSettings">true</longPathAware>\r\n
n </windowsSettings>\r\n </application>\r\n <dependency>\r\n <dependentAssembly>\r\n <assemblyIdentity type="win32" name="Microsoft.Windows.Commo
n-Controls">\r\n
version="6.0.0.0" processorArchitecture="" publicKeyToken="6595b64144ccf1df" language="" />\r\n </dependentAssembl
y>\r\n </dependency>\r\n</assembly>\r\n
69025 0x004a71de 0x1404aa5de 17 18 () ascii xmlrpc/client.pyc
70269 0x004b5bff 0x1404b8fff 25 26 () ascii xml/etree/ElementPath.pyc
75298 0x004f1ac8 0x1404f4ec8 25 26 () ascii xml/etree/ElementTree.pyc
75768 0x004f7504 0x1404fa9b4 22 23 () ascii xml/etree/_init_.pyc
76447 0x004ff744 0x140502b44 26 27 () ascii lib2to3/btn_utils.py[XML]e
76706 0x005026c3 0x140505ac3 19 20 () ascii xmlrpc/_init_.pyc
85736 0x00572d5a 0x14057615a 4 5 () ascii tXML
94745 0x005e35d3 0x1405e69d3 4 5 () ascii XMLI
107647 0x0067f821 0x140682c21 17 18 () ascii xml/_init_.pyc]
107656 0x0067f9f9 0x140682df9 25 26 () ascii xml/parsers/expat.py[pycPMK
108258 0x006867d1 0x140689bd1 25 26 () ascii xml/parsers/_init_.pyc]
123030 0x0073ac3b 0x14073e03b 8 9 () ascii \nPyXMLSm
123036 0x0073ac11 0x14073e011 7 8 () ascii xml/ml"
130741 0x00799787 0x14079cbb7 4 5 () ascii XMLVe
141542 0x0081f1c4 0x1408225c4 19 20 () ascii xmlrpc/client.pycPK
141554 0x0081f4a5 0x1408228a5 27 28 () ascii xml/etree/ElementPath.pycPK
141591 0x0081fcf5 0x1408230f5 27 28 () ascii xml/etree/ElementTree.pycPK
141593 0x0081fd7d 0x14082317d 24 25 () ascii xml/etree/_init_.pycPK

```

Durante el análisis se identificaron múltiples URLs relacionadas principalmente con dependencias legítimas utilizadas por el malware para su funcionamiento. Estas incluyen dominios como digicert.com y microsoft.com, que podrían estar vinculados a la validación de certificados o a bibliotecas necesarias para ejecutar su lógica. Además, dominios como beopen.com o intertools.com pueden estar

relacionados con herramientas y módulos de desarrollo, como bibliotecas de Python.

```
remnux@remnux:~/theJungle/malware$ RABINZ_DEBASE64=0 rabin2 -zzz 83f7625b243f0594484bc87b6ba4bf2b.malware | grep -Eo '([a-zA-Z0-9-]|\.|\.)+(com|org|net|edu|gov|info|io|biz|edu)'
```

re.com
re.com
Be0pen.com
itertools.com
itertools.com
itertools.com
python.org
schemas.microsoft.com
Be0pen.com
www.digicert.com
www.digicert.com
www.digicert.com
ocsp.digicert.com
cacerts.digicert.com
crl3.digicert.com
crl3.digicert.com
crl4.digicert.com
www.digicert.com
ocsp.digicert.com
cacerts.digicert.com
crl3.digicert.com
ocsp.digicert.com
cacerts.digicert.com
www.digicert.com
ocsp.digicert.com
cacerts.digicert.com
crl3.digicert.com
www.digicert.com
www.digicert.com
ocsp.digicert.com
cacerts.digicert.com
crl3.digicert.com

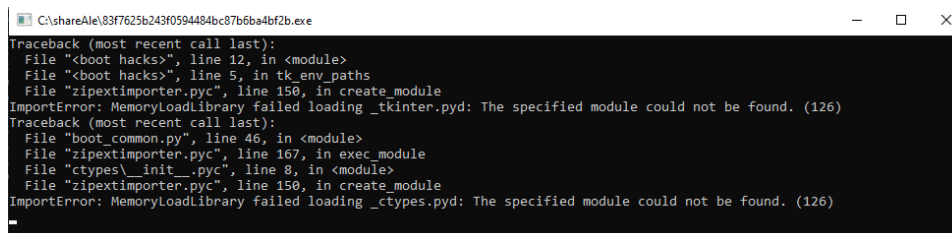
En la próxima imagen podemos evidenciar el hallazgo de múltiples cadenas que comienzan con `__map_` seguido de nombres de codificaciones de caracteres como `gb2312`, `gbcommon`, `jisx0212`, `jisxcommon`, `jisx0208`, `ksx1001`, `cp949`, `jisx0213_pair`, `jisx0213_2_emp`, `jisx0213_1_emp`, `jisx0213_2_bmp`, `jisx0213_1_bmp`, `jisx0213_bmp` y `big5`, esto sugiere que el malware tiene la capacidad de trabajar con texto en diversas codificaciones de caracteres, incluyendo las utilizadas para representar chino simplificado (GB2312), japonés (JIS), coreano (KSX1001, CP949) y chino tradicional (Big5). Esta funcionalidad podría ser utilizada para procesar o analizar datos en diferentes idiomas, como también para comunicarse con sistemas que utilizan estas codificaciones.

```
remnux@remnux:~/theJungle/malware$ rabin2 -zzz 83f7625b243f0594484bc87b6ba4bf2b.malware | grep '__map'
```

40003	0x00243530	0x140246930	22	23	(.rsrc)	ascii	multibytecodec.__map_*
40007	0x00243578	0x140246978	12	13	(.rsrc)	ascii	__map_gb2312
40008	0x00243588	0x140246988	14	15	(.rsrc)	ascii	__map_gbcommon
40009	0x00243598	0x140246998	14	15	(.rsrc)	ascii	__map_jisx0212
40010	0x002435a8	0x1402469a8	16	17	(.rsrc)	ascii	__map_jisxcommon
40011	0x002435c0	0x1402469c0	14	15	(.rsrc)	ascii	__map_jisx0208
40012	0x002435d0	0x1402469d0	13	14	(.rsrc)	ascii	__map_ksx1001
40013	0x002435e0	0x1402469e0	11	12	(.rsrc)	ascii	__map_cp949
40014	0x002435f0	0x1402469f0	19	20	(.rsrc)	ascii	__map_jisx0213_pair
40015	0x00243608	0x140246a08	20	21	(.rsrc)	ascii	__map_jisx0213_2_emp
40016	0x00243620	0x140246a20	20	21	(.rsrc)	ascii	__map_jisx0213_1_emp
40017	0x00243638	0x140246a38	18	19	(.rsrc)	ascii	__map_jisx0213_emp
40018	0x00243650	0x140246a50	20	21	(.rsrc)	ascii	__map_jisx0213_2_bmp
40019	0x00243668	0x140246a68	20	21	(.rsrc)	ascii	__map_jisx0213_1_bmp
40020	0x00243680	0x140246a80	18	19	(.rsrc)	ascii	__map_jisx0213_bmp
41742	0x0024f8c8	0x140252cc8	10	11	(.rsrc)	ascii	__map_big5
41912	0x00256aee	0x140259eee	8	9	(.rsrc)	ascii	Y@__map_

2.2. DINÁMICO

Para el análisis dinámico hacemos doble clic a la muestra 83f7625b243f0594484bc87b6a4bf2b.exe, se despliega la siguiente pantalla en la consola inmediatamente.



```
C:\shareAle\83f7625b243f0594484bc87b6a4bf2b.exe
Traceback (most recent call last):
  File "<boot hacks>", line 12, in <module>
  File "<boot hacks>", line 5, in tk_env_paths
  File "zipextimporter.pyc", line 150, in create_module
ImportError: MemoryLoadLibrary failed loading _tkinter.pyd: The specified module could not be found. (126)
Traceback (most recent call last):
  File "boot_common.py", line 46, in <module>
  File "zipextimporter.pyc", line 167, in exec_module
  File "ctypes\__init__.pyc", line 8, in <module>
  File "zipextimporter.pyc", line 150, in create_module
ImportError: MemoryLoadLibrary failed loading _ctypes.pyd: The specified module could not be found. (126)
```

La siguiente imagen de Process Explorer muestra una jerarquía de procesos donde el proceso legítimo svchost.exe (Host Process for Windows Services) aparece como padre de varios subprocesos:

- **taskhostw.exe:** Responsable de gestionar tareas en segundo plano.
- **MicrosoftEdgeUpdate.exe:** Proceso que actualiza el navegador Microsoft Edge.
- **CompatTelRunner.exe:** Esta relacionado con la telemetría de Microsoft.
- **powershell.exe:** Herramienta utilizada comúnmente para automatización, pero en este caso está siendo utilizada por el malware.
- **conhost.exe:** Proporciona una interfaz de consola, pero como hijo de svchost.exe, podría estar facilitando la ejecución de scripts maliciosos en el sistema.

Específicamente que los procesos MicrosoftEdgeUpdate.exe, powershell.exe y conhost.exe que aparezcan como hijos de svchost.exe no es comportamiento habitual. El malware está utilizando svchost.exe, un proceso confiable, para ocultar sus actividades maliciosas y evitar ser detectado.

Process Explorer - Sysinternals: www.sysinternals.com [DESKTOP-K286PKK\flare] (Administrator)

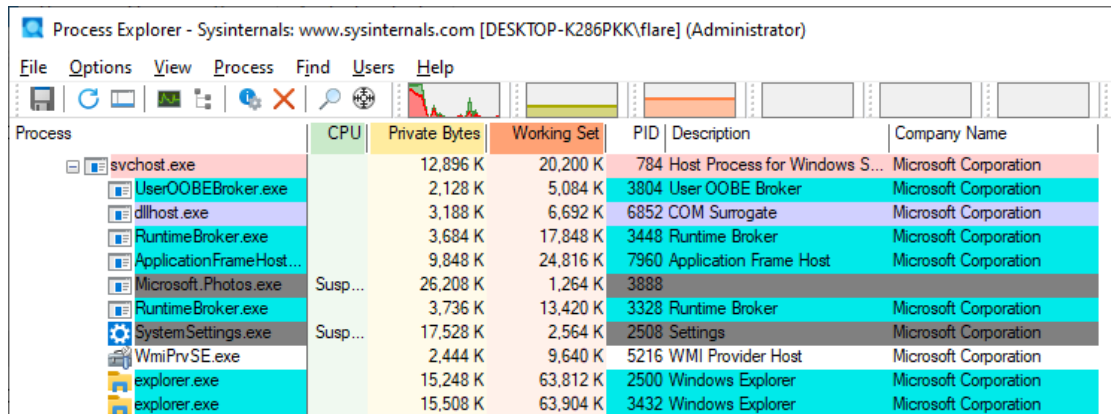
File Options View Process Find Users Help

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
SystemSettings.exe	Susp...	17,528 K	2,564 K	2508	Settings	Microsoft Corporation
WmiPrvSE.exe		2,612 K	9,676 K	5216	WMI Provider Host	Microsoft Corporation
explorer.exe		15,140 K	63,756 K	2500	Windows Explorer	Microsoft Corporation
explorer.exe		15,296 K	63,828 K	3432	Windows Explorer	Microsoft Corporation
smartscreen.exe		2,448 K	9,368 K	8284	Windows Defender SmartScr...	Microsoft Corporation
WmiPrvSE.exe		4,500 K	13,924 K	6080	WMI Provider Host	Microsoft Corporation
svchost.exe		8,440 K	13,236 K	888	Host Process for Windows S...	Microsoft Corporation
svchost.exe		2,408 K	3,836 K	936	Host Process for Windows S...	Microsoft Corporation
svchost.exe		6,540 K	10,056 K	696	Host Process for Windows S...	Microsoft Corporation
taskhostw.exe		8,812 K	14,416 K	2120	Host Process for Windows T...	Microsoft Corporation
MicrosoftEdgeUpdate...		1,952 K	3,584 K	8364	Microsoft Edge Update	Microsoft Corporation
CompatTelRunner.exe		1,208 K	2,208 K	4420	Microsoft Compatibility Telem...	Microsoft Corporation
conhost.exe	< 0.01	6,656 K	1,088 K	4784	Console Window Host	Microsoft Corporation
CompatTelRunner...	11.77	57,496 K	29,776 K	6584	Microsoft Compatibility Telem...	Microsoft Corporation
powershell.exe	1.47	32,248 K	30,472 K	7480	Windows PowerShell	Microsoft Corporation
conhost.exe		6,292 K	9,772 K	6624	Console Window Host	Microsoft Corporation
powershell.exe	1.47	32,308 K	30,496 K	7056	Windows PowerShell	Microsoft Corporation
conhost.exe		6,288 K	10,052 K	2312	Console Window Host	Microsoft Corporation
svchost.exe		2,136 K	4,760 K	848	Host Process for Windows S...	Microsoft Corporation
svchost.exe		3,228 K	6,020 K	1036	Host Process for Windows S...	Microsoft Corporation
svchost.exe		2,520 K	5,848 K	1112	Host Process for Windows S...	Microsoft Corporation
svchost.exe		3,300 K	5,516 K	1132	Host Process for Windows S...	Microsoft Corporation
VBoxService.exe	< 0.01	2,900 K	4,724 K	1280	VirtualBox Guest Additions S...	Oracle and/or its affiliates

Seguimos analizando los procesos de svchost.exe, en este caso tenemos los siguientes subprocessos:

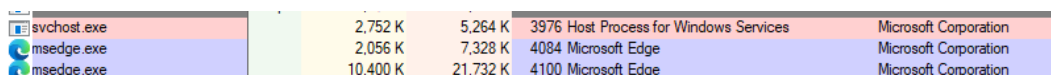
- **UserOOBEBroker.exe:** Responsable de facilitar el proceso inicial de configuración del sistema.
- **dllhost.exe:** Encargado de manejar extensiones COM (Component Object Model).
- **RuntimeBroker.exe:** Garantiza que las aplicaciones tengan los permisos adecuados y protege el sistema de accesos no autorizados.
- **ApplicationFrameHost.exe:** Gestiona las ventanas de las aplicaciones de la Plataforma Universal de Windows (UWP).
- **Microsoft.Photos.exe:** Proceso relacionado con la aplicación Fotos de Microsoft para gestionar imágenes.
- **SystemSettings.exe:** Responsable de gestionar la interfaz gráfica para ajustar configuraciones del sistema.
- **WmiPrvSE.exe:** Proceso del Servicio de Instrumentación de Administración de Windows (WMI), utilizado para monitorear y gestionar el sistema.
- **explorer.exe:** Representa el explorador de Windows, manejando la interfaz gráfica, carpetas y escritorio.

Aunque estos procesos suelen ser legítimos y esenciales, es importante destacar que Microsoft.Photos.exe y explorer.exe normalmente no se presentan como hijos de svchost.exe.



Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
svchost.exe		12,896 K	20,200 K	784	Host Process for Windows S...	Microsoft Corporation
UserOOBEBroker.exe		2,128 K	5,084 K	3804	User OOBE Broker	Microsoft Corporation
dllhost.exe		3,188 K	6,692 K	6852	COM Surrogate	Microsoft Corporation
RuntimeBroker.exe		3,684 K	17,848 K	3448	Runtime Broker	Microsoft Corporation
ApplicationFrameHost...		9,848 K	24,816 K	7960	Application Frame Host	Microsoft Corporation
Microsoft.Photos.exe	Susp...	26,208 K	1,264 K	3888		
RuntimeBroker.exe		3,736 K	13,420 K	3328	Runtime Broker	Microsoft Corporation
SystemSettings.exe	Susp...	17,528 K	2,564 K	2508	Settings	Microsoft Corporation
WmiPrvSE.exe		2,444 K	9,640 K	5216	WMI Provider Host	Microsoft Corporation
explorer.exe		15,248 K	63,812 K	2500	Windows Explorer	Microsoft Corporation
explorer.exe		15,508 K	63,904 K	3432	Windows Explorer	Microsoft Corporation

Finalmente, el último proceso que esta como hijo de svchost es msedge.exe lo cual tampoco es un comportamiento normal.



Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
svchost.exe		2,752 K	5,264 K	3976	Host Process for Windows Services	Microsoft Corporation
msedge.exe		2,056 K	7,328 K	4084	Microsoft Edge	Microsoft Corporation
msedge.exe		10,400 K	21,732 K	4100	Microsoft Edge	Microsoft Corporation

En Process Monitor observamos que la muestra de malware realizó varias acciones desde el evento "Process Start" con PID 9024 y padre 5116 "Explorador de Windows", de las cuales podemos destacar:

Aquí está una ampliación detallada con más puntos clave observados en el registro de actividades:

1. CreateFile (C:\Windows\System32\ntdll.dll) – "SUCCESS".

- Acceso exitoso al archivo ntdll.dll, una librería dinámica esencial que contiene funciones relacionadas con el núcleo de Windows.
- Esto podría ser un intento del malware de interceptar o modificar funciones críticas del sistema operativo.

2. CreateFile

(C:\Windows\Prefetch\3F7652B2543F0595448BC786B4ABC29E15EF.pf)

– "NAME NOT FOUND".

- Intento de acceder a un archivo de prefetch que no existe.
- Es posible que el malware esté investigando rastros de ejecución previos de aplicaciones.

3. RegOpenKey (HKLM\System\CurrentControlSet\Control\Session Manager) – “REPARSE”.

- Intenta acceder a la clave principal del registro relacionada con la gestión de sesiones del sistema operativo.
- El resultado "REPARSE" indica que el acceso fue redirigido internamente dentro del sistema.

4. RegOpenKey (HKLM\System\CurrentControlSet\Control\Session Manager) – “SUCCESS”.

- Accede exitosamente a la clave del registro que controla las sesiones del sistema.
- Esto sugiere que el malware podría estar evaluando configuraciones críticas relacionadas con la gestión de sesiones en el sistema operativo.

5. RegOpenKey (HKLM\System\CurrentControlSet\Control\Session Manager\RaiseExceptionOnPossibleDeadlock) – “NAME NOT FOUND”.

- Intenta acceder a una subclave que determinar si el sistema debe generar una excepción cuando se detecta un posible deadlock (varios procesos o hilos quieren el mismo recurso).
- El resultado "NAME NOT FOUND" indica que esta clave no existe en el sistema.

Después de realizar esto, el malware carga kernel32.dll, que es una librería esencial para las operaciones del núcleo de Windows, como gestión de memoria y manejo de excepciones. También carga KernelBase.dll, que sirve como base para muchas APIs de Windows y facilita operaciones básicas del sistema, como la

administración de archivos y procesos. Posteriormente tenemos los siguientes procesos:

6. CreateFile (C:\Windows\System32\conhost.exe) – “SUCCESS”.

- El proceso accede exitosamente a conhost.exe, que proporciona una interfaz de consola para procesos de línea de comandos.
- Esto sugiere que el malware podría estar utilizando comandos maliciosos para interactuar con el sistema.

7. QueryEaFile (C:\Windows\System32\conhost.exe) – “SUCCESS”.

- Consulta atributos extendidos de conhost.exe, posiblemente para obtener más información sobre el proceso.

8. FileSystemControl (C:\Windows\System32\conhost.exe) – “INVALID DEVICE REQUEST”.

- Realiza una solicitud de control del sistema de archivos en conhost.exe, pero la operación no es válida.

9. CreateFileMapping (C:\Windows\System32\conhost.exe) – “FILE LOCKED WITH ONLY READERS”.

- Intenta mapear el archivo conhost.exe, pero este está bloqueado para escritura.
- A pesar de que el archivo está bloqueado para escritura, el proceso posterior muestra un estado “SUCCESS”. Esto indica que, aunque no puede modificarse directamente, el mapeo en memoria fue exitoso.

- El malware podría estar intentando manipular esta biblioteca para interceptar o alterar funciones de seguridad relacionadas con el cifrado.

Después de esto, la imagen muestra una serie de operaciones realizadas sobre el archivo C:\Windows\System32\cryptbase.dll por el proceso con ID “9024”. Las operaciones incluyen CreateFileMapping, Load Image, CloseFile, CreateFile, y QuerySecurityFile. Aunque la mayoría de las operaciones tienen un resultado “SUCCESS”, hay un detalle importante:

13. QuerySecurityFile - BUFFER OVERFLOW: El proceso intenta consultar información de seguridad del archivo cryptbase.dll, pero la operación produce un error de “BUFFER OVERFLOW” que es una vulnerabilidad que permite a un atacante escribir más datos en un búfer de los que este puede manejar.

- Esto ocurre cuando el tamaño del búfer proporcionado para almacenar la información es insuficiente para contener todos los datos requeridos.
- Es un comportamiento que puede ser utilizado por el malware para probar los límites del sistema o forzar errores que puedan ser explotados posteriormente.

El evento de BUFFER OVERFLOW fue detectado en múltiples registros, indicando intentos reiterados por parte del proceso de realizar consultas o acceder a claves específicas del sistema.


```

Decompile: FUN_140005334 - (83f7625b243f0594484bc87b6ba4bf2b-1.exe)
4
5
6
7 bool bVar1;
8 int iVar2;
9 undefined8 uVar3;
10 undefined8 uVar4;
11 code **ppcVar5;
12 longlong *p1Var6;
13 undefined8 *puVar7;
14 uint *puVar8;
15 undefined8 uVar9;
16 longlong uVar10;
17 undefined8 unaff_RBX;
18 undefined8 in_R9;
19 undefined8 uVar11;
20
21 iVar2 = (int)unaff_RBX;
22 uVar3 = __srt_initialize_crt(1);
23 if ((char)uVar3 == '\0') {
24     FUN_1400059b0(7);
25 }
26 else {
27     bVar1 = false;
28     uVar11 = 0;
29     uVar4 = __srt_acquire_startup_lock();
30     iVar2 = (int)CONCAT71((int7)((ulonglong)unaff_RBX >> 8), (char)uVar4);
31     if (DAT_140009710 != 1) {
32         if (DAT_140009710 == 0) {
33             DAT_140009710 = 1;
34             iVar2 = _initterm_e(&DAT_140006370, &DAT_140006388);
35             if (iVar2 != 0) {
36                 return 0xff;
37             }
38         }
39     }
40 }
41
42 ppcVar5 = (code **)FUN_140005998();
43 if ((*ppcVar5 != (code *)0x0) &&
44     (uVar4 = __srt_is_nonwritable_in_current_image((longlong)ppcVar5), (char)uVar4 !=
45     (**ppcVar5)(0,2,0,in_R9,uVar11));
46 )
47 {
48     p1Var6 = (longlong *)FUN_1400059a0();
49     if ((*p1Var6 != 0) &&
50         (uVar4 = __srt_is_nonwritable_in_current_image((longlong)p1Var6), (char)uVar4 !=
51         __register_thread_local_exe_atexit_callback(*p1Var6);
52         )
53     )
54     {
55         __get_initial_wide_environment();
56         puVar7 = (undefined8 *)__p_wargv();
57         uVar3 = *puVar7;
58         puVar8 = (uint *)__p_argc();
59         uVar10 = (ulonglong)*puVar8;
60         uVar4 = FUN_140001130(uVar10,uVar3);
61         iVar2 = (int)(uVar4 & 0xffffffff);
62         uVar9 = __srt_is_managed_app();
63         if ((char)uVar9 != '\0') {
64             if (!bVar1) {
65                 __cexit();
66             }
67             __srt_uninitialize_crt(CONCAT71((int7)(uVar10 >> 8),1),'\0');
68             return uVar4 & 0xffffffff;
69         }
70         goto LAB_1400054a0;
71     }
72 }
73
74 FUN_1400059b0(7);
75 LAB_1400054a0:
76     /* WARNING: Subroutine does not return */
77     exit(iVar2);
78 }
79

```

En la función FUN_140001130 se han identificado llamadas a PyImport_ImportModule, PyMethod_New, PyObject_SetAttrString y PyObject_CallObject. Estas funciones confirman que el ejecutable incluye un intérprete de Python embebido, lo cual le permite ejecutar scripts dinámicamente.

También se observaron llamadas a funciones como SHGetSpecialFolderPath, indicando que el ejecutable puede estar accediendo a carpetas del sistema. Este acceso es potencialmente utilizado para escribir archivos maliciosos en rutas comunes u ocultas, facilitando la persistencia del malware en el sistema y la carga de payloads secundarios.

```

Decompile: FUN_140001130 - (83f7625b243f0594484bc87b6ba4bf2b-1.exe)
1
2
3
4
5
6
7
8
9
10
11
12
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
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
24
```

<pre> 140005997 c3 RET ;***** ; FUNCTION ;***** undefined * __fastcall FUN_140005998(void) assume GS_OFFSET = 0xffff00000000 RAX:8 <RETURN> undefined * FUN_140005998 XREF[1]: FUN_140005334;1400053d0(c) 140005998 48 8d 05 LEA RAX,[DAT_1400099c8] 29 40 00 00 14000599f c3 RET </pre>	<pre> 2 undefined * FUN_140005998(void) 3 4 5 return &DAT_1400099c8; 6 7 </pre>
--	---

```

1 4000059a 00
2 00000000 00000000
3
4 ***** FUNCTION *****
5
6 void FUN_1400059a8(void)
7
8 {
9     _DAT_140005978 = 0;
10    return;
11
12 }
13
14 undefined * __fastcall FUN_1400059a0(void)
15     assume GS_OFFSET = 0xf000000000
16     RAX:8 <RETURN>
17
18 FUN_1400059a0 XREF[1]: FUN_140005334;1400053fc(c)
19
20 1400059a0 48 8d 05 LEA RAX,[DAT_1400059c0]
21
22 19 40 00 00
23
24 1400059a7 c3 RET

```

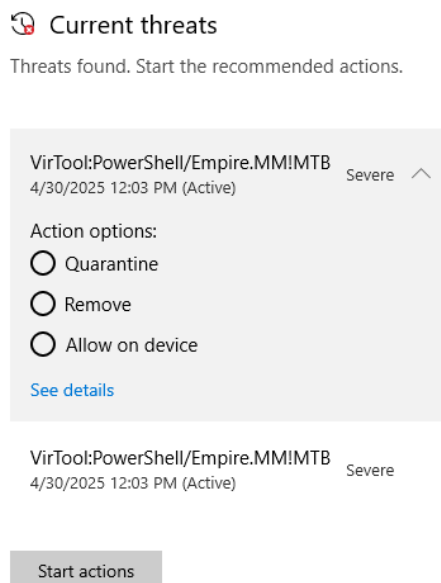
1400099c6	??	??
1400099c7	??	??
1400099c8	DAT_1400099c8	XREF[1]: FUN_14000599e:14000599e(*)
1400099c9	??	??
1400099ca	??	??
1400099cb	??	??
1400099cc	??	??
1400099cd	??	??
1400099ce	??	??
1400099cf	??	??

1
2 undefined * FUN_14000599e(void)
3
4
5 return &DAT_1400099c8;
6
7

35

3. MITIGACIÓN

Como medidas de mitigación ante esta amenaza, se recomienda mantener actualizado tanto el sistema operativo como Windows Defender, asegurando la activación de funcionalidades como la protección en tiempo real y las reglas de reducción de superficie de ataque (ASR). Asimismo, debe restringirse la ejecución de binarios desconocidos mediante políticas como AppLocker o WDAC y limitar el uso de herramientas como PowerShell y cmd para usuarios sin privilegios administrativos. Es fundamental aplicar el principio de mínimo privilegio, monitorear eventos sospechosos mediante soluciones EDR y SIEM, además de bloquear la carga de DLL no autorizadas o comportamientos anómalos de procesos legítimos. Adicionalmente, se debe fomentar la concienciación del usuario para prevenir la ejecución de archivos maliciosos distribuidos mediante técnicas de ingeniería social.



VirTool:PowerShell/Empire.MM!MTB

Alert level: Severe
Status: Active
Date: 4/30/2025 12:02 PM
Category: Tool
Details: This program is used to create viruses, worms or other malware.

[Learn more](#)

Affected items:

containerfile: C:\Users\Public\launcher.lnk

file: C:\Users\Public\launcher.lnk->[EmbeddedEnc]->(Base64)->(UTF-16LE)

OK

4. REGLA DE DETECCIÓN YARA

```
remnux@remnux:~/Documents$ yara yaraPEF.yar 83f7625b243f0594484bc87b6ba4bf2b.exe
UAH_PEF_20250430 83f7625b243f0594484bc87b6ba4bf2b.exe
```

```
rule UAH_PEF_20250430 {
```

```
  meta:
```

```
    author = "Wilma Alejandra Montesinos Guzmán"
```

```
    date = "2025-04-30"
```

```
    sha256 =
```

```
    "fd92ba4c1d0390761a5afad41d2be3ac98463c1eb1d0d7b2e8dff54505341cbe"
```

```
    description = "Regla YARA para detección de malware con Python embebido"
```

```
  strings:
```

```
    $s1 = "PYTHON310.DLL" wide ascii
```

```
    $s2 = "PYTHONSCRIPT" wide ascii
```

```
    $s3 = "cookiejar.pyc" wide ascii
```

```
    $s4 = "cmd.pyc" wide ascii
```

```
    $s5 = "cmd /c" ascii
```

```
    $api1 = "VirtualAlloc" ascii
```

```
    $api2 = "VirtualProtect" ascii
```

```
    $api3 = "LoadLibraryA" ascii
```

```
    $api4 = "GetProcAddress" ascii
```

```
$api5 = "AdjustTokenPrivileges" ascii  
condition:  
    filesize > 8000000 and  
    all of ($s*) and any of ($api*)  
}
```

5. CONCLUSIÓN

El análisis realizado permitió identificar una muestra de malware caracterizada por el uso de técnicas de evasión, empaquetado y ejecución de código embebido en Python. A pesar de ser detectado por pocos motores antivirus en plataformas como VirusTotal debido a que es una muestra bastante reciente, Windows Defender logró clasificarla correctamente como maliciosa, lo que demuestra la eficacia de los mecanismos de protección integrados en sistemas actualizados. El malware fue diseñado como un binario PE de 64 bits que se comporta como una DLL debido a que exporta muchas funciones, pero se presenta como un ejecutable para facilitar su ejecución por parte del usuario. Se observaron múltiples indicadores que evidencian comportamiento malicioso, como el uso de APIs para inyección de código, escalada de privilegios y manipulación del registro. Asimismo, incorpora recursos incrustados, como bibliotecas y scripts en Python, que le otorgan flexibilidad para ejecutar acciones dinámicas y potencialmente modificar su comportamiento en tiempo de ejecución. El análisis estático y dinámico reveló también el intento de ocultar su actividad dentro de procesos legítimos del sistema, lo que complica su detección mediante técnicas tradicionales. Por tanto, se concluye que este malware representa una amenaza real para sistemas Windows, especialmente si no se cuenta con medidas de mitigación adecuadas, como políticas de restricción de ejecución, actualizaciones constantes y monitoreo proactivo de comportamiento.

6. REFERENCIAS

- VirusTotal. (2025). *File analysis details: fd92ba4c1d0390761a5afad41d2be3ac98463c1eb1d0d7b2e8dff54505341cbe*. Recuperado el 9 de abril de 2025, de <https://www.virustotal.com/gui/file/fd92ba4c1d0390761a5afad41d2be3ac98463c1eb1d0d7b2e8dff54505341cbe/details>.