Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería Telemática

Trabajo Fin de Grado

Estudio del malware WannaCry utilizando técnicas de reversing

Autor: Adrián Cortés Arranz

Tutor: Susel Fernández Melián

2021

Universidad de Alcalá

Ingeniería telemática

Trabajo Fin de Grado

Estudio del malware WannaCry utilizando técnicas de reversing

Autor: Adrián Cortés Arranz

Tutor: Susel Fernández Melián

TRIBUNAL:

Presidente: Luis de la Cruz Piris

Vocal 1º: Bernardo Alarcos Alcázar

Vocal 2º: Susel Fernández Melián

Resumen en español del trabajo

WannaCry fue un tipo de malware que tuvo especial repercusión en el mundo entero, debido a que consiguió acaparar los noticiarios el día 12 de mayo de 2017 por infectar a más de 230.000 equipos en un solo día [1]. Este ataque consiguió propagarse a través de la red, infectando a numerosos dispositivos, al aprovecharse de una vulnerabilidad de Windows llamada EthernalBlue, que fue desarrollada por la Agencia de Seguridad Nacional (NSA) y filtrado por el grupo de hackers *Shadow Brokers* el 14 de abril de 2017 [2].

Este Trabajo de Fin de Grado (TFG), consiste en el análisis de una muestra de este famoso software malicioso, WannaCry, a través de técnicas de reversing, proceso mediante el cual se estudia el código de un programa con el fin de entender su funcionamiento y comportamiento para mejorarlo, erradicarlo, prevenirlo o neutralizarlo [3].

El desarrollo de este trabajo se divide en dos etapas principales, análisis estático y análisis dinámico, donde se describe el funcionamiento del programa estudiando su código fuente y ejecutándolo en un ambiente controlado.

Resumen en inglés del trabajo

WannaCry was a type of malware that had a special impact around the world, because it managed to monopolize the news on May 12, 2017 by infecting more than 230,000 computers in a single day [1]. This attack managed to spread through the network, infecting so many devices, by exploiting the Window´s vulnerability EthernalBlue, developed by the National Security Agency (NSA) and leaked by the hacker group *Shadow Brokers* on April 14, 2017 [2].

This Final Degree Project (TFG) consists of the analysis of a sample of this famous malicious software, WannaCry, through reversing techniques, a process by which the code of a program is studied in order to understand how it works. and behavior to improve, eradicate, prevent or neutralize it [3].

The development of this work is divided into two main stages, static analysis and dynamic analysis, where the operation of the program is described by studying its source code and executing it in a controlled environment.

Glosario de acrónimos y abreviaturas

MS17-010	Nombre del parche de Windows
NSA	Agencia de Seguridad Nacional
SMB	Server Message Block
DNS	Domain Name System
PE	Portable Executable
DOS	Sistema operativo MS-DOS
HEX	Hexadecimal
ASSII	Código de caracteres basado en el alfabeto latino
DACL	Listas de control de acceso discrecional
VT	Virus Total
PC	Ordenador, equipo informatico
HTTP	Hypertext Transfer Protocol

Índice

Índice de Ilustraciones

1. Introducción

En esta última década, se ha podido apreciar que, tanto el uso de dispositivos informáticos como de las redes e internet, ha crecido exponencialmente. Una de las razones de esta crecida, podría residir en la variedad de intereses, proyectos, desempeños o habilidades que tienen los usuarios. Empero, a todos ellos les afecta un problema muy significativo, la seguridad.

De todos los perfiles que hacen uso de dispositivos informáticos, hay uno que lo obra malintencionadamente con el fin de hacerse con datos de otros usuarios o empresas, contraseñas, o simplemente por diversión. Este tipo de actos, se conoce como ciberdelito, donde su motivación principal es conseguir un beneficio económico, aunque en otros casos, puede darse por motivos políticos o personales [4].

Uno de los métodos utilizados por los ciberdelincuentes o *hackers* es el malware, que procede de la palabra *malicius software*, también puede recibir otros nombres como *badware*, software malicioso o dañino. Estos programas son diseñados para infiltrarse en un dispositivo sin el consentimiento del usuario. Hay diferentes tipos de malware, cada uno de ellos tiene un cometido especial, sin embargo, todos ellos tienen un rasgo en común, actúan en contra de los intereses del usuario [5].

Existen diferentes tipos de malware. Los *troyanos*, que son los más extendidos y que, a menudo se camuflan como programas legítimos, engañando al usuario mediante ingeniería social para que una vez activado, ocasionen daños o recopilen datos confidenciales [6]. Los *backdoors,* que abren una puerta trasera para evitar los procedimientos de autenticación habituales teniendo acceso remoto. Los *stealers,* se emplean para obtener información confidencial del usuario [7]. Los *ransomware,* son programas que infectan el equipo, restringiendo el acceso a ciertas partes o archivos, exigiendo un rescate económico a cambio de retirar todas las restricciones ocasionadas durante la infección. En muchos casos, se cifran los archivos del sistema operativo impidiendo la utilización de este [8].

La técnica más usada para la detección de este tipo de software es mediante antivirus, que detectan el malware de dos formas. El primer método es de forma reactiva, es decir, basado en firmas donde se tiene una base de datos de firmas y se compara con el programa, si hay una coincidencia se trata de malware. Por otra parte, este método tiene un problema, las muestras tienen que ser previamente identificadas para introducirlas en la base de datos [9]. El segundo método, es de forma proactiva basada en heurística, en otras palabras, su algoritmo genérico compara el comportamiento de un archivo frente a otro ya conocido, dando verdadera importancia al análisis de malware, que tiene como objetivo determinar y comprender el funcionamiento de una muestra de software malicioso [10].

El reversing es uno de los métodos que se emplea para luchar contra este tipo de aplicaciones sin código fuente, que forma parte de la ingeniería inversa, por la cual se analiza el funcionamiento y comportamiento del malware para neutralizarlo, prevenirlo o erradicarlo [11].

En este trabajo nos centramos en el análisis de WannaCry mediante técnicas de reversing. Se trata del *ransomware* más conocido, que apareció el 12 de mayo de 2017 siendo el causante de la infección de más de 230.000 ordenadores (PC) ese día en todo el mundo, afectando a importantes empresas, incluidas españolas como Telefónica, Gas Natural o Iberdrola. Este malware, pedía una cantidad de dinero en un plazo máximo y en caso de que este no se cumpliera, no se recuperaba el acceso a los datos [12].

2. Descripción del trabajo desarrollado

El objetivo que se persigue en este trabajo es analizar el *ransomware* WannaCry, observando su comportamiento mediante la técnica del reversing. Para ello se desensambla el programa para entender su funcionamiento, examinar qué vulnerabilidades puede explotar de nuestro sistema operativo y, evaluar los vectores de ataque que utiliza. Este proceso se divide en dos partes.

La primera parte, consiste en hacerse con una de las versiones del *ransomware* WannaCry. Posteriormente, se realiza un estudio exhaustivo de las diferentes técnicas empleadas en reversing y, de los programas existentes para

después, poder analizar detenidamente la muestra obtenida, así como, conocer los acontecimientos de la historia de WannaCry.

La segunda parte, se divide en dos etapas fundamentales: La primera etapa, llamada análisis estático, consiste en identificar el archivo y el hash MD5 correspondiente al malware, el tamaño en disco y en memoria virtual, así como todos los datos asociados (cabeceras, strings y librerías que utiliza) realizando así un análisis estático básico. Para la realización de un análisis estático más avanzado, se procede a abrir el archivo con un programa de desensamblado, con el fin de entender el funcionamiento y naturaleza del mismo. Los puntos más relevantes de este análisis son los siguientes:

- Propagación
- Tipo de compresión
- Analizar mapa de memoria
- Encontrar punto de inicio y breakpoint
- Creación de servicios
- Conexiones a través de internet mediante direcciones IP o dominios

Tras finalizar el análisis estático, se conoce dónde comienza el programa, por lo que se puede ejecutar de manera controlada. De esta forma se pueden desarrollar medidas de protección [13].

En la segunda etapa, conocida como análisis dinámico, se analiza el software en un ambiente controlado. En este punto, la búsqueda principal es ver la capacidad que tiene el programa malicioso para:

- Abrir conexiones con el mundo exterior, es decir, internet.
- Realizar alteraciones sobre los registros del sistema.
- Modificar el sistema de archivos y procesos de ejecución.

3. Marco teórico

En este apartado se recoge la historia, características y funcionamiento teórico de WannaCry obtenido mediante una búsqueda exhaustiva de documentación teórica. Una vez que se conoce la historia que implica a este *ransomware*, se expone qué es la ingeniería inversa y cuáles son los métodos y programas que se utilizan para llevarla a cabo.

3.1. WannaCry

El *ransomware* es un malware que infecta dispositivos informáticos y los cifra, solicitando un beneficio económico para así descifrar y recuperar la información secuestrada [14]. Dicho en otras palabras, el cometido de este malware es introducirse en los dispositivos informáticos, infectándolos y bloqueándolos de manera parcial o total, hasta que se abone la cantidad de dinero exigida.

WannaCry es el nombre del mayor ataque de tipo *ransomware* de la historia. Se produjo el 12 de mayo de 2017 entre las 8:00 y 17:00, consiguiendo acaparar las portadas de toda la prensa de ese día [12]. Fue descrito como un ataque sin “precedentes” infectando a más de 230 mil computadoras en 150 países diferentes bloqueando el acceso a los sistemas informáticos de instituciones estatales y empresas a nivel mundial con el nombre de Ransom:Win32.WannaCrypt [15]. El país más afectado fue Rusia, donde quedaron inutilizados los sistemas informáticos del Ministerio de Interior, ferrocarriles, bancos y operadoras de telefonía. Sin embargo, no fue el único país afectado ya que India y Gran Bretaña también vieron comprometido su servicio nacional de salud. En España se vio afectada la operadora Telefónica y, Alemania tuvo su principal ataque en la empresa ferroviaria Deutsche Bahn AG que, confirmó que solo se vio atacada en las pantallas informativas de llegadas y salidas de los trenes, pero no en la circulación de los mismos [16].

Todas estas empresas tenían algo en común. En todas las pantallas se mostraba la misma imagen que se expone en la Ilustración 1, en la que aparece un mensaje explicativo sobre lo ocurrido en el equipo, una cuenta atrás en la parte izquierda de la imagen y las instrucciones de cómo realizar el pago solicitado y recuperar los datos.

Ilustración 1: Interfaz Gráfica de WannaCry

3.1.1. Vulnerabilidad explotada

El *ransomware* WannaCry se propagó usando el *exploit* de Windows *EternalBlue* [17]. *EternalBlue,* es un *exploit* creado por la NSA de Estados Unidos (EE.UU) que, aprovecha una vulnerabilidad en la implementación del protocolo Server Message Block (SMB) de Microsoft, denotada como CVE-2017-0144, que acepta paquetes específicos de atacantes remotos, permitiendo ejecutar código en el ordenador remotamente [2].

Este *exploit,* fue robado a la NSA un mes después de que Windows emitiera el parche MS17-010 [18] para solucionar esta vulnerabilidad. El robo, que se produjo en el año 2017 por un grupo que se hacía llamar Shadow Hackers, filtró su botín en las redes y dio uso libre a los cibercriminales para penetrar en sistemas basados en Microsoft Windows [2] [17]. La gran mayoría de computadoras por aquel entonces no se habían actualizado, por lo que WannaCry pudo infectar a todos los ordenadores que no habían instalado el parche MS17-010.

3.1.2. Propagación

WannaCry se comporta como un gusano, lo que significa que se propaga a través de las redes, usando un protocolo de uso compartido de archivos llamado SMBv1 que, permite a los PC comunicarse con otros dispositivos conectados a la misma red. Una vez que se instala en un equipo, WannaCry escanea la red buscando direcciones IP de manera aleatoria encontrando así más dispositivos vulnerables. Se introduce en ellos mediante el código *EternalBlue* y utiliza la herramienta DoublePulsar [19] para instalarse y ejecutarse a través de una puerta trasera. De esta forma, el malware es capaz de autopropagarse entre los distintos equipos sin intervención de otros programas o archivos [20] .

3.1.3. Detección del *ransomware*

El ataque no se siguió expandiendo gracias a Marcus Hutchins [21], conocido como Malware Tech. Este hombre, con tan solo 22 años, encontró “un botón de apagado” en el código del *badware,* analizándolo a través técnicas de reversing [15].

Tras analizar el malware, descubrió que intentaba conectarse a un dominio de internet concreto que no estaba registrado. Al no estar creado, permitía la actuación del *ransomware*. Al registrar dicho dominio, se creó un sumidero Domnain Name System (DNS) que frenó su propagación por completo. A este proceso se le llamó “ Kill switch” [22].

3.2. Ingeniería inversa

La ingeniería inversa en un proceso en el que se intenta recrear los pasos llevados a cabo por el creador del programa, pero a la inversa con el fin de identificar qué vulnerabilidades explota, los vectores de ataque utilizados, el nivel de infección y finalmente las medidas de protección [23].

Cuando un desarrollador escribe un programa, este se codifica mediante un lenguaje de alto nivel. Cuando ya se ha terminado de escribir todo el código fuente del programa, este se compila creando un ejecutable en código máquina binario. Una vez creado el ejecutable nadie podría ver, leer o editar el código fuente del programa.

Es aquí donde entra la ingeniería inversa, una técnica de *cracking* que consiste, por un lado, en la decompilación o compilación inversa del programa con el fin de generar a partir del código máquina binario un lenguaje de programación ya sea de alto o bajo nivel, y por otro lado, analizar el código paso a paso para poder ver su funcionamiento [24].

3.2.1. Herramientas utilizadas en ingeniería inversa

Para poder realizar la decompilación del análisis paso a paso, se hace uso de programas destinados a esta labor. Existen tres clases de programas, desensambladores, decompiladores y depuradores. Dentro de cada una de estas clases, existen diferentes programas destinados a realizar este trabajo [22].

Desensamblador: Un desensamblador convierte el código de lenguaje máquina a lenguaje ensamblador, por lo que es más legible para nosotros. Los más utilizados en Windows son PE Explorer [25], programa que permite inspeccionar el funcionamiento interno de un software junto con las bibliotecas de terceros de Windows. W32DASM [26], esta herramienta, además de desmontar el ejecutable gestiona el código con todas las funciones. IDA 6.6, IDA Pro Freeware 5.0 [27] e IDA Pro [28], estos tres últimos programas, en diferentes versiones son desensambladores capaces de crear mapas de ejecución mostrándolo en un formato grafico parecido a ilustraciones. BORG disassembler [29], es de las herramientas más ligeras de esta familia, su principal propósito es desensablar archivos WIN32 PE [30] en código o datos binarios. Por último, HT Editor, que te permite editar, ver o analizar el ejecutable.

Decompiladores o compiladores inversos: Un decompilador recrea un código en lenguaje de alto nivel a partir del lenguaje máquina binario. Los más utilizados en Windows son DCC Decompiler [31] que, sólo permite decompilar ejecutables DOS y lenguaje C. Boomerang Decompiler Project [32] que, decompila programas en C. Finalmente, Reverse Engineering Compiler [33] que, decompila código ensamblador a una escritura del código semejante a C.

Depuradores: Un depurador es un programa que permite el control de otro programa, dando opción a analizarlo paso a paso para poder ir viendo el estado de las variables, del uso de la memoria y afectación del programa en el equipo.

Los más utilizados en Windows son OllyDbg [34] que, es un depurador de código de 32 bits también conocido como 32dbg. WinDBG [35] que, puede usarse para realizar una depuración local o remota en modo Kernel. VisualDuxDebugger [36] que, es un desensamblador y depurador de 64 bits. Para concluir, SoftICE [37] que, es un depurador en modo kernel y es capaz de suspender todas las operaciones en Windows cuando se desee.

4. Análisis WannaCry mediante técnicas de reversing

Para realizar un correcto análisis de WannaCry, se divide el proceso en cuatro pasos. El primer paso, es hacerse con una muestra del *ransomware*. A través de una búsqueda en internet se encuentran varias versiones. Seleccionamos la muestra con nombre muestra1.bin.exe, obtenida del siguiente enlace https://app.any.run/tasks/83fa5ab4-a712-4b4f-91c3-e5508deec36e/. El segundo paso, es montar un escenario de pruebas, donde la muestra de *ransomware* se pueda ejecutar controladamente. El Tercer paso, consiste en la realización del análisis estático del malware. Este proceso se divide en dos etapas, siendo la primera el análisis estático básico, donde se procede a analizar la muestra, pero sin decompilarla ni desensamblarla y, la segunda, el análisis estático avanzado, donde se desensambla la muestra con el fin de entender el código. El cuarto y último paso es realizar el análisis dinámico. Este consiste en depurarlo paso a paso, terminando de afianzar lo que realiza el código.

4.1. Escenario de pruebas

Para realizar el análisis, se necesita un entorno de pruebas, donde la muestra de *ransomware* no pueda infectar al equipo o en caso contrario, lo ejecute de manera controlada. Para este propósito, se ha construido un escenario de pruebas que se compone de una distribución de Linux, sistema operativo donde no afecta este *ransomware*. Se ha instalado la versión Ubuntu 20.4 en un ordenador HP con las siguientes características:

- Procesador Intel Core i7-3770 de 3,40 GHz.
- 16 Gb de RAM.
- Tarjeta gráfica Nvidia gt 640 DDR3 con 3 Gb dedicada.
- Disco duro de 2 TB HDD.

Dentro de esta distribución de Linux Ubuntu 20.4, se ha instalado el programa VirtualBox con el objetivo de crear una máquina virtual que, pueda ser infectada por el software malicioso logrando analizarlo de manera controlada. En VirtualBox se ha importado una máquina virtual de Windows 7 donde se ha procedido a la instalación de los programas IDA Pro e IDA Pro debugging [28]. Ambos programas han sido elegidos debido a las capacidades de creación de mapas gráficos de ejecución de los programas (*Graph overview*) y de depuración sobre ellos. HxD [38], seleccionado para poder decodificar en hexadecimal (HEX) la muestra y poder editarla. X32dbg [39], electo con el fin de depurar y ejecutar paso a paso la muestra de *ransomware*. Procexp [40], se trata de un programa para monitorizar el sistema y conocer que identificadores y procesos se han abierto o cargado en el sistema. Finalmente, Procmon [41], escogido con el fin de poder visualizar las alteraciones de registros y servicios del sistema.

En la ilustración 2, se muestra un esquema del escenario de pruebas, creado para realizar el análisis de WannaCry mediante técnicas de reversing. En el centro de la ilustración, se observa la maquina anfitrión Ubuntu, en la que está instalado Virtualbox con una máquina virtual de W7. En la parte superior de la misma, se encuentran representados los programas instalados dentro de la máquina virtual de W7 y, necesarios para el análisis. En la parte inferior, se representa la muestra del virus que va a infectar la máquina virtual de W7. Es muy importante destacar que, una vez creado el escenario de pruebas y, antes de infectarlo, es recomendable proceder a tomar una captura de la máquina virtual para que, en caso de infectarla por completo y quedar inutilizada, se pueda volver al punto en el que se tomó la captura y donde la maquina se encontraba “limpia”, es decir, volver al instante en el que el *ransomware* no ha sido ejecutado en la máquina virtual manteniendo todos los programas, configuración y ajustes realizados hasta el momento. Este método es muy útil ya que, en caso de infectar la maquina totalmente, no se tendría que volver a construir el entorno de pruebas desde el principio.

Ilustración 2: Esquema del escenario de pruebas

4.2. Análisis estático básico de muestra1.bin.exe

En este apartado procedemos a analizar la muestra, pero sin decompilarla ni desensamblarla. Su objetivo es recopilar datos del malware que servirán para realizar el análisis estático avanzado. Se hace uso de la Terminal de Linux y se estudia el código binario con el fin de averiguar lo siguiente:

- Hashes de la muestra: se utilizan como identificadores para reconocer el tipo de muestra. Debido a que el hash de un archivo es único, sirve para diferenciar o reconocer muestras modificadas o alteradas. El hash se calcula en formato MD5, SHA1 o SHA256.
- Tipo de archivo: indica el tipo de archivo que corresponde con la muestra.

- Cabeceras: son las extensiones que utiliza el programa para llevar a cabo su función.
- Secciones: indica las secciones que contiene el programa, estas pueden ser .text, .rdata, .data, .rsrc.
- Strings: es la obtención de cadenas que aparecen dentro del programa.
- Direcciones IP: se intentan encontrar direcciones IP a las que el programa conecta o intenta conectar.
- Características del binario: se obtienen las características que aparecen al abrir el binario con un programa como HxD.

Con la finalidad de conseguir todos los datos, el análisis se enumera más abajo en los pasos de ejecución llevados a cabo. Adicionalmente, se utilizan análisis con inteligencia externa con el fin de recopilar más datos del malware.

1) Obtención de los hashes. Mediante los comandos md5sum, sha1sum y sha256sum, se calculan los hashes MD5, SHA1 y SHA256 correspondientes a la muestra.

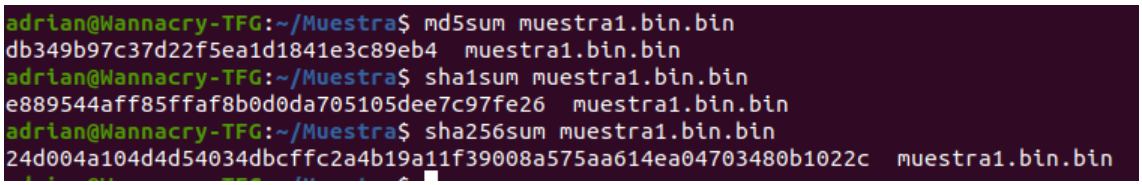


Ilustración 3: Obtención de hashes de muestra1.bin.exe

2) Analizar el tipo de archivo a tratar. Para ello se hace uso del comando File de Linux que, como se puede ver en la ilustración 4, se trata de un ejecutable de tipo Portable Executable (PE) para sistemas MS Windows y que utiliza Intel 80386 que se trata de un procesador de 32 bits, por lo tanto, el programa es de 32 bits igualmente.

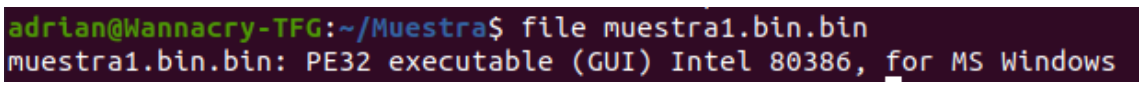


Ilustración 4: Tipo de archivo muestra1.bin.exe

3) Tamaño que ocupa la muestra. Se utiliza el comando du de Linux para ver el tamaño del programa. Representado en la ilustración 5 se puede observar que se utiliza la variable -b y -h para mostrar la salida en Bytes y en formato leíble [42]. El tamaño de la muestra es de 3,6M.

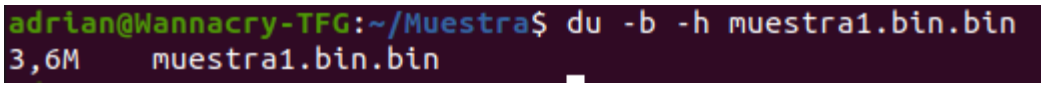


Ilustración 5: Tamaño de muestra1.bin.exe

Con los procesos realizados hasta el momento, ya se conocen los datos principales de la muestra. A continuación, en la tabla 1, se especifican los datos recogidos por ahora.

Nombre	Muestra1.bin.exe
Hash	MD5: DB349B97C37D22F5EA1D1841E3C89EB4
	SHA1: e889544aff85ffaf8b0d0da705105dee7c97fe26
	SHa256: 24d004a104d4d54034dbcffc2a4b19a11f39008a575aa614ea04703480b1022c
Archivo	PE
Sistema	MS Windows
Arquitectura	x32
Tamaño	3,6M

Tabla 1: Datos del ransomware WannaCry

4) Obtención de las cadenas que aparecen dentro del texto del programa malicioso. Este punto se lleva a cabo mediante el comando strings de Linux, como se observa en la ilustración 6. La salida por pantalla de este comando se guardará en un .txt con el fin de poder visualizar todos los datos. Se mostrarán los más relevantes.

```
adrian@Wannacry-TFG:~/Muestra$ strings muestra1.bin.bin > strings.txt
adrian@Wannacry-TFG:~/Muestra$ ls
muestra1.bin.bin  strings.txt
adrian@Wannacry-TFG:~/Muestra$ vim strings.txt
```

Ilustración 6: Obtención strings de muestra1.bin.exe

Al comienzo se puede observar que se muestra un mensaje indicando, que no se puede ejecutar el programa en modo DOS, *"This program cannot run in DOS mode"* y, a continuación, se visualizan las secciones del programa tal y como se contempla en la Ilustración 7.

```
!This program cannot be run in DOS mode.
Rich
.text
`.rdata
@.data
.rsrc
_^][
SUVW
_^][
WUhx
^][Y
```

Ilustración 7: Secciones de programa de muestra1.bin.exe

Secciones de programa que se visualizan en la Ilustración 7:

- **.text:** contiene el código del programa.
- **.rdata:** almacena los datos del programa de solo lectura y las tablas de importaciones y exportaciones, ya que no muestra .idata ni .edata.
- **.data:** almacena los datos del programa con permisos de lectura y escritura.
- **.rsrc:** contiene los recursos del archivo (cursores, sonidos, menús...).

En la ilustración 8 (a), se puede examinar el nombre de algunas de las funciones que tiene el programa, y de las que hace uso para infectar y cifrar el

equipo. Igualmente, en la ilustración 8 (b) se muestran algunos archivos que debe generar o leer el *ransomware,* para poder realizar su cometido. En la ilustración 9, se muestra el Sistema Operativo compatible con sus respectivas versiones, siendo estas Windows Vista, Windows 8 , Windows 8.1 y Windows 10.

(a) (b)

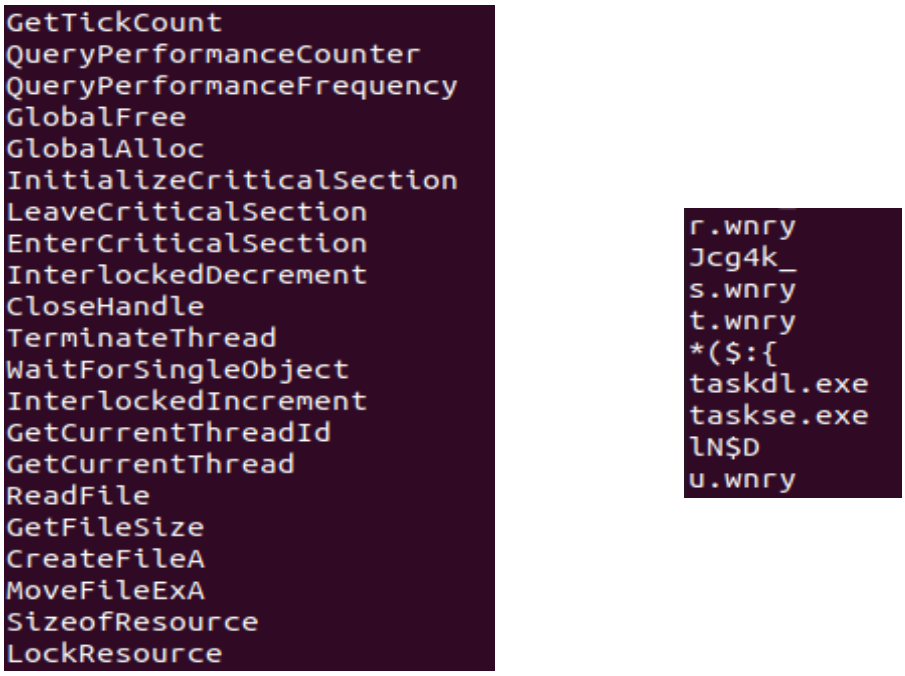


Ilustración 8: Funciones de programa de muestra1.bin.exe (a) y archivos de programa de muestra1.bin.exe (b)

```
    </dependentAssembly>
  </dependency>
  <compatibility xmlns="urn:schemas-microsoft-com:compatibility.v1">
    <application>
      <!-- Windows 10 -->
      <supportedOS Id="{8e0f7a12-bfb3-4fe8-b9a5-48fd50a15a9a}"/>
      <!-- Windows 8.1 -->
      <supportedOS Id="{1f676c76-80e1-4239-95bb-83d0f6d0da78}"/>
      <!-- Windows Vista -->
      <supportedOS Id="{e2011457-1546-43c5-a5fe-008deee3d3f0}"/>
      <!-- Windows 7 -->
      <supportedOS Id="{35138b9a-5d96-4fbd-8e2d-a2440225f93a}"/>
      <!-- Windows 8 -->
      <supportedOS Id="{4a2f28e3-53b9-4441-ba9c-d69d4a4a6e38}"/>
    </application>
```

Ilustración 9: Sistemas operativos soportados por programa muestra1.bin.exe

5) Tamaño de cada sección. Se necesita conocer el tamaño de cada una de las secciones para poder saber más adelante, en el análisis estático avanzado, la dirección de comienzo de cada una de ellas. En la ilustración 10 se representan estos tamaños, mediante el comando size con formato SysV, que aporta las direcciones de cada sección.

```
adrian@Wannacry-TFG:~/Muestra$ size --format=SysV muestra1.bin.bin
muestra1.bin.bin  :
section      size      addr
.text       35786   4198400
.rdata       2456   4235264
.data      159744   4239360
.rsrc     3515476   7405568
Total     3713462
```

Ilustración 10: Tamaño de las secciones de programa de muestra1.bin.exe

6) Se procede a examinar el archivo HEX y ANSII de la muestra ya que, facilita la confirmación de los datos extraídos anteriormente. Para esta función se utiliza el programa HxD que se trata, de un analizador de binarios y editor hexadecimal. Todos los archivos, como . exe, .DDL, .ZIP, tienen un patrón de bytes de cabecera identificativo, por lo que este patrón, permite identificar el tipo de archivo que se está tratando [43]. Se carga el archivo en el programa, visualizándose inicialmente el código Hex 4D 5A 90 00, MZ en ANSII que, queda representado en la ilustración 11. Estos números, corresponden con el patrón de bytes de cabecera de un archivo DLL.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..........ÿÿ..
```

Ilustración 11: Cabecera de archivo .DDL

A continuación, representado en la parte derecha de la ilustración 12, se visualiza otra vez la frase anterior *"This program cannot run in DOS mode".* Esta frase pertenece al encabezado básico que tienen todos los ejecutables de Windows de formato PE

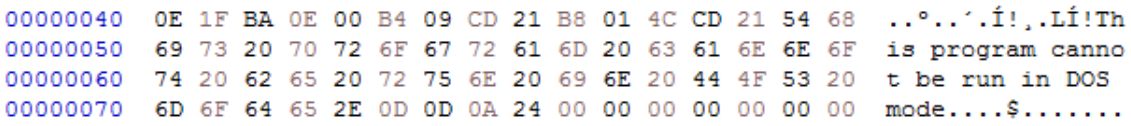


Illustration 12: Encabezado basico de ejecutables formato PE

Posteriormente, se aprecia en la ilustración 13 la cabecera del formato PE. En este instante se sabe que el binario comienza a partir de los bytes 50 45. Seguido viene la letra L en ASCII, confirmando que se trata de un PE de 32 bits [44].

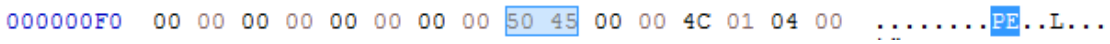


Ilustración 13: Cabecera de archivo PE

Por último, se contemplan las secciones del programa tal y como se muestra en la Ilustración 14. Las secciones mostradas coinciden con las representadas anteriormente en la ilustración 7, esto es debido a que se trata del mismo programa analizado de dos modos diferentes.

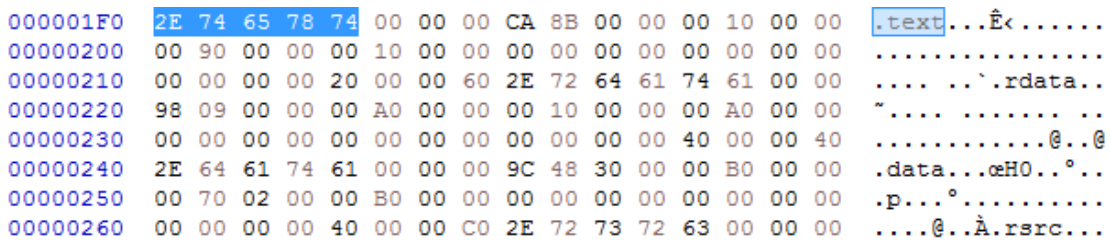


Ilustración 14: Secciones de programa de muestra1.bin.exe

En este punto, se utilizan analizadores virtuales como virus total (VT) y any.run, web donde en este caso se toma la muestra analizada, con el fin de extraer más datos antes de pasar a un análisis estático más avanzado. Existen infinidad de analizadores virtuales, pero estos dos son los más comunes y famosos debido a que tienen una gran base de datos con la que comparar

4.2.1. Análisis con Virus Total

VT [45] es una herramienta de análisis virtual muy potente, contando con una versión gratuita y otra de pago. En este proyecto se utiliza la versión gratuita.

Nada más entrar, se introduce el hash del malware anteriormente calculado en el análisis estático básico de muestra1.bin.exe en el paso 1), iniciando el análisis y detectando 66 de 69 amenazas comparando con diferentes antivirus, que como puede observarse en la ilustración 15 son bastante conocidos.

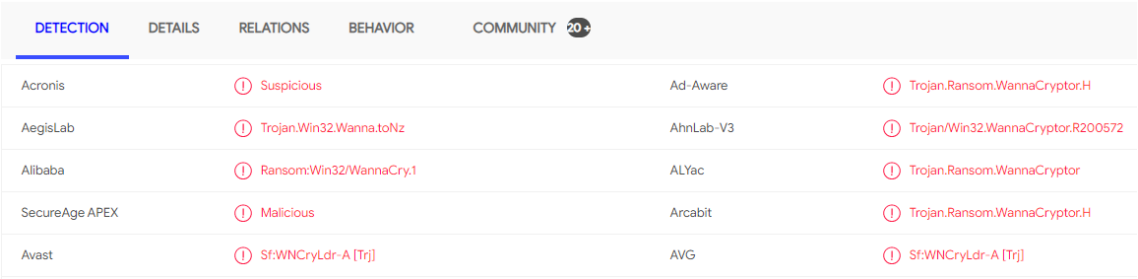


Ilustración 15: Escaneo de antivirus mediante Virus Total

Navegando a las pestañas relations y behavior, se notifica que conecta con la URL www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com y, desplazándose a la pestaña details encontramos las propiedades básicas, fechas de creación, los diferentes nombres registrados que ha tenido, las secciones y más datos anteriormente analizados. Entre estos datos se encuentran las DDLs que importa este archivo representadas en la ilustración 16.

Ilustración 16: DDLs importadas por el programa muestra1.bin.exe

4.2.2. Análisis con Any.run

Any.run es una herramienta de análisis virtual que además dispone de un *sandbox* de análisis interactivo [46]. En este se observan los mismos datos extraídos de VT, pero además permite realizar un pequeño análisis gratuito, en el cual se percibe que la muestra realiza una conexión HTTP a la página web www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com. Como anteriormente mencionado, WannaCry, intentaba conectar con un dominio que al ser comprado actuó como sumidero DNS cortando su propagación y su cifrado. En la ilustración 17 se representa este dominio con una conexión HTTP favorable.

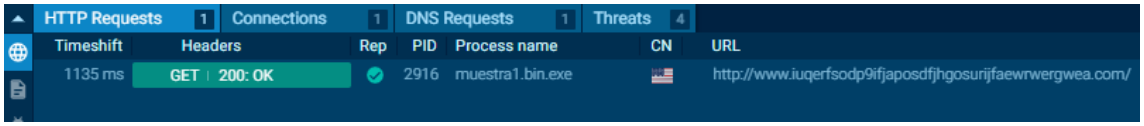


Ilustración 17: Conexión HTTP favorable en Any.run

Ya se tienen los datos más importantes de la muestra como el hash que lo identifica o las conexiones que realiza con el mundo exterior, finalizando así el análisis estático básico de la muestra. Con el fin de seguir avanzando, el siguiente paso es realizar el análisis estático avanzado.

4.3. Análisis estático avanzado de muestra1.bin.exe

En este apartado, se procede a desensamblar la muestra con el programa IDA Pro. Este programa es idóneo para realizar este tipo de análisis puesto que, al desensamblar la muestra, crea mapas de ejecución, visualizando gráficamente las llamadas a las funciones y los saltos de código, este proceso se conoce como *Graph overview.*

Se procede a cargar la muestra en el programa, que nada más abrirlo se tiene que configurar para cumplir con nuestras preferencias. Nos desplazamos a la pestaña "*options*" → *"general"* y ajustamos a 5 bytes el campo *"number of opcode"*, para que nos muestre 5 bytes de código HEX al lado de cada línea de código en ensamblador. Según se carga la muestra, el programa IDA pro se sitúa en una función que reconoce por nombre WinMain(xxx), justo donde comienza el programa.

Seguidamente, se procede a analizar el código desensamblado. En la ilustración 18, se puede ver en la dirección de memoria 0040814A cómo se mueve el contenido de la variable aHttpWww_iuqerf que contiene el dominio www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com al registro ESI [47].

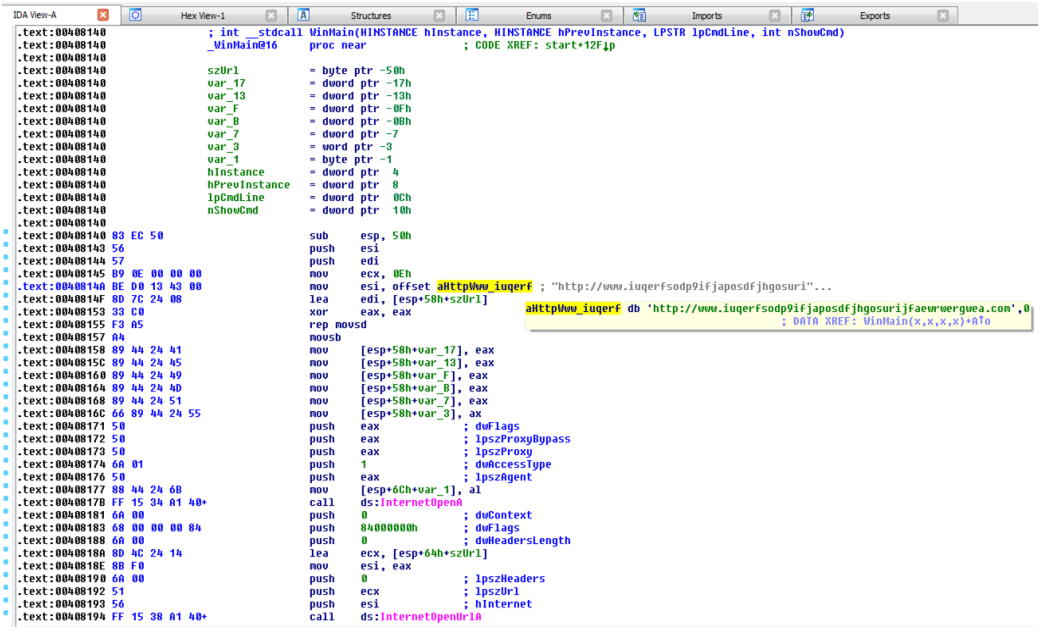


Ilustración 18: Dominio al que conecta el programa muestra1.bin.exe

Después, en la ilustración 19 se muestra el *Graph overview*, así como, en la ilustración 20 el pseudocódigo que genera IDA pulsando la tecla F5. En ambas ilustraciones se visualiza como el programa llama a la función InternetOpenA y se le pasa el registro ESI, para que esta función compruebe si hay conexión con el dominio, finalizando con el programa o, de lo contrario, si no hay conexión, continuando con el mismo.

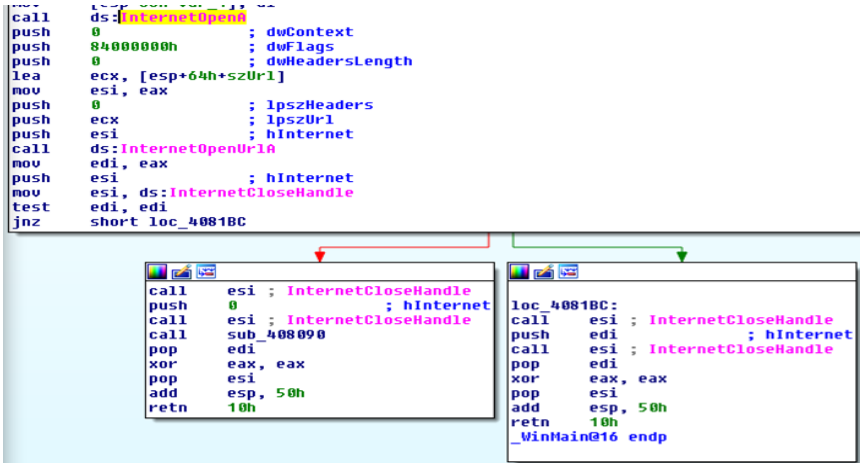


Ilustración 19: Comprobación de conexión con dominio kill switch graph overview

```
qmemcpy(&szUrl, aHttpWww_iuqerf, 0x39u);
v8 = 0;
v9 = 0;
v10 = 0;
v11 = 0;
v12 = 0;
v13 = 0;
v14 = 0;
v4 = InternetOpenA(0, 1u, 0, 0, 0);
v5 = InternetOpenUrlA(v4, &szUrl, 0, 0, 0x84000000, 0);
if ( v5 )
{
  InternetCloseHandle(v4);
  InternetCloseHandle(v5);
  result = 0;
}
else
{
  InternetCloseHandle(v4);
  InternetCloseHandle(0);
  sub_408090();
  result = 0;
}
return result;
}
```

Ilustración 20: Conexión con dominio kill switch pseudocódigo

Tras comprobar que no hay conexión, el programa llama a la subrutina sub_408090, que a su vez llama a sub_407F20 y, esta a su vez llama a sub_407C40. Esta última subrutina, hace que el malware se autorregistre como servicio en el equipo con las características representadas en la tabla 2. En la imagen 21, se muestra el código mediante el cual el software se registra como servicio.

Nombre	mssecsvc2.0
Comando	%s -m security
Descripción	Microsoft Security Center (2.0) Service

Tabla 2: Servicio mssecsvc2.0 registrado en el equipo

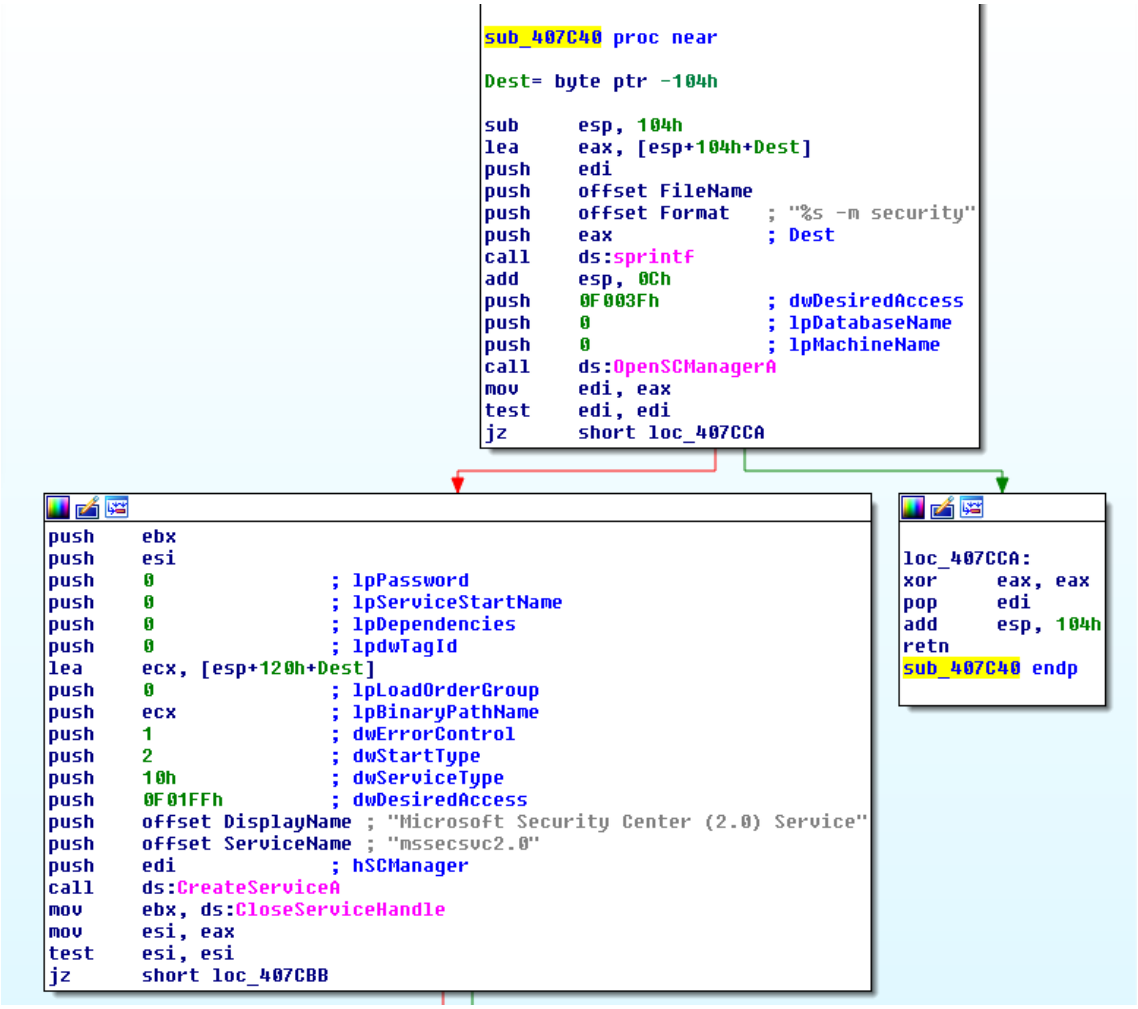


Ilustración 21: Registro de mssecsv2.0 como servicio en el sistema

Cuando la Subrutina sub_407C40 finaliza, se vuelve a sub_407F20 llamando a sub_407CE0. Esta subrutina, resuelve dinámicamente una API y recoge mediante la función GetProcAddress los campos necesarios para la resolución de dicha API tal y como se muestra en la figura 22 (a) Posteriormente, el programa extrae un recurso R, representado en la ilustración 22 (b).

(a) (b)

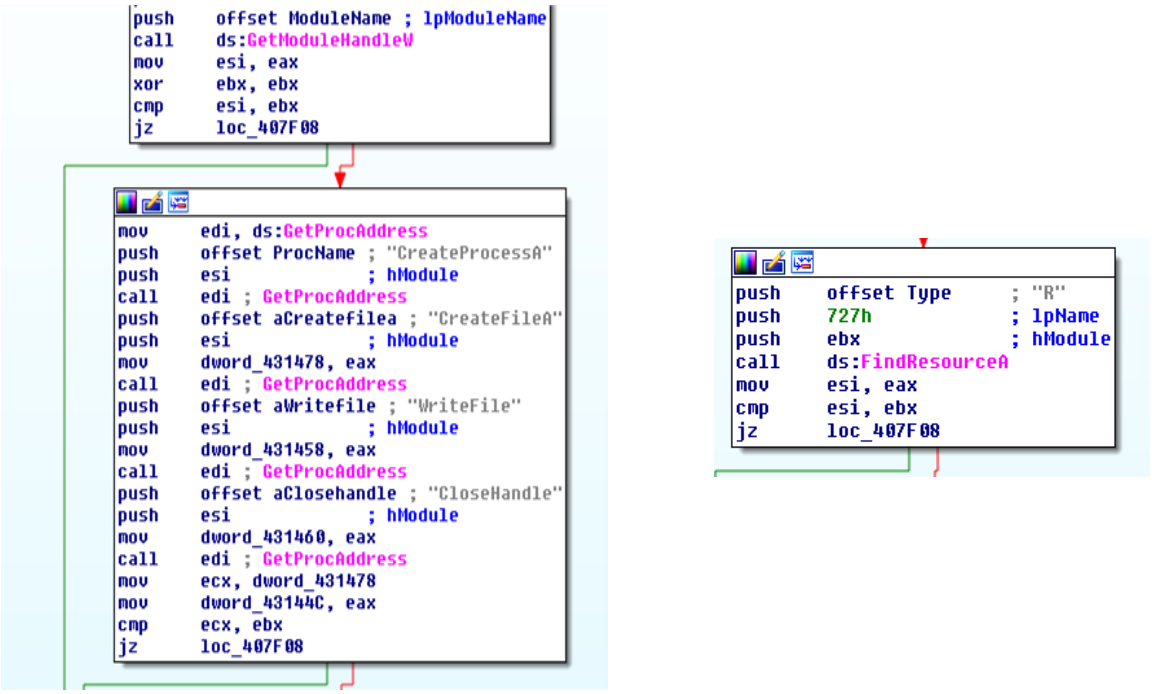


Ilustración 22: Campos necesarios para la resolución API (a) y extracción de recurso R (b)

A continuación, el software crea dos variables con los nombres *Dest y NewfileName*, donde posteriormente, se introducen paths del sistema tal y como se observa en la imagen 23. Esta asignación, queda representada en la tabla 3. Para terminar con esta subrutina, se realiza una copia de C:\WINDOWS\tasksche.exe en C:\WINDOWS\qeriuwjhrf y se copia el recurso R en C:\WINDOWS\tasksche.exe. Mas adelante, en los puntos 4.5 y 4.6 se analiza de forma exhaustiva el archivo tasksche.exe, debido a que en este instante del análisis, todavía no se sabe que tipo de archivo se trata.

Desp	C:\WINDOWS\tasksche.exe
NewFileName	C:\WINDOWS\qeriuwjhrf

Tabla 3: Paths creados en el sistema

```
xor     eax, eax
lea     edi, [esp+270h+var_207]
mov     [esp+270h+Dest], bl
rep stosd
stosw
stosb
mov     ecx, 40h
xor     eax, eax
lea     edi, [esp+270h+var_103]
mov     [esp+270h+NewFileName], bl
rep stosd
mov     esi, ds:sprintf
push    offset aTasksche_exe ; "tasksche.exe"
stosw
stosb
push    offset aWindows ; "WINDOWS"
lea     eax, [esp+278h+Dest]
push    offset aCSS     ; "C:\\%s\\%s"
push    eax             ; Dest
call    esi ; sprintf
add     esp, 10h
lea     ecx, [esp+270h+NewFileName]
push    offset aWindows ; "WINDOWS"
push    offset aCSQeriuwjhrf ; "C:\\%s\\qeriuwjhrf"
push    ecx             ; Dest
call    esi ; sprintf
add     esp, 0Ch
lea     edx, [esp+270h+NewFileName]
lea     eax, [esp+270h+Dest]
```

Ilustración 23: Creación de path tasksche.exe *y queriuwjhrf en el sistema*

Terminada la subrutina, se vuelve a sub_408090 donde continuando con el código, se llama a sub_407FA0. Mediante la función ChangeServiceConfig2A, esta subrutina cambia los parámetros de configuración opcionales del servicio, representado en la ilustración 24, para seguidamente ejecutar sub_40800 registrándose el servicio en el sistema y haciéndolo persistente [48].

```
var_1C= dword ptr -1Ch
var_18= dword ptr -18h
Info= dword ptr -14h
var_10= dword ptr -10h
var_C= dword ptr -0Ch
var_8= dword ptr -8
var_4= dword ptr -4
hService= dword ptr  4
arg_4= dword ptr  8

sub     esp, 1Ch
mov     eax, [esp+1Ch+arg_4]
xor     edx, edx
mov     [esp+1Ch+var_1C], 1
mov     [esp+1Ch+Info], 0
lea     ecx, [eax+eax*4]
lea     ecx, [ecx+ecx*4]
lea     ecx, [ecx+ecx*4]
shl     ecx, 3
cmp     eax, 0FFFFFFFFh
mov     [esp+1Ch+var_18], ecx
setnz   dl
mov     eax, offset unk_70F87C
lea     ecx, [esp+1Ch+Info]
mov     [esp+1Ch+var_8], edx
mov     edx, [esp+1Ch+hService]
mov     [esp+1Ch+var_C], eax
mov     [esp+1Ch+var_10], eax
push    ecx             ; lpInfo
lea     eax, [esp+20h+var_1C]
push    2               ; dwInfoLevel
push    edx             ; hService
mov     [esp+28h+var_4], eax
call    ds:ChangeServiceConfig2A
add     esp, 1Ch
retn
sub_407FA0 endp
```

Ilustración 24: Cambio de los parámetros de configuración de servicio

Se ejecuta sub_407BD0. En este punto, se lleva a cabo el proceso de distribución del software, que se producirá por dos hilos. Un hilo para propagación local y otro para propagación por internet, como se representa en la ilustración 25.

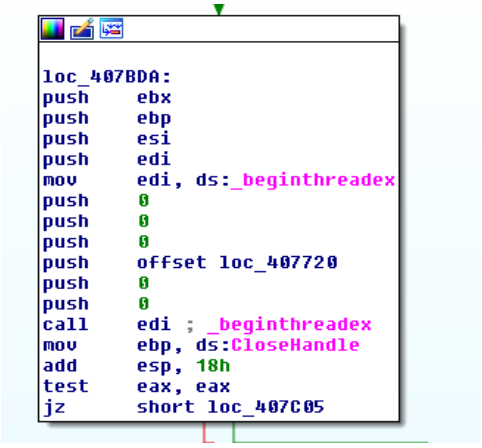


Ilustración 25: Creación de 2 hilos para propagarse

En la Ilustración 26 (a), se observa como la propagación por la red local obtiene información del adaptador de red y genera direcciones IP de su rango, mientras que en la figura 26 (b) se puede observar como la propagación por internet genera IP aleatorias.

(a) (b)

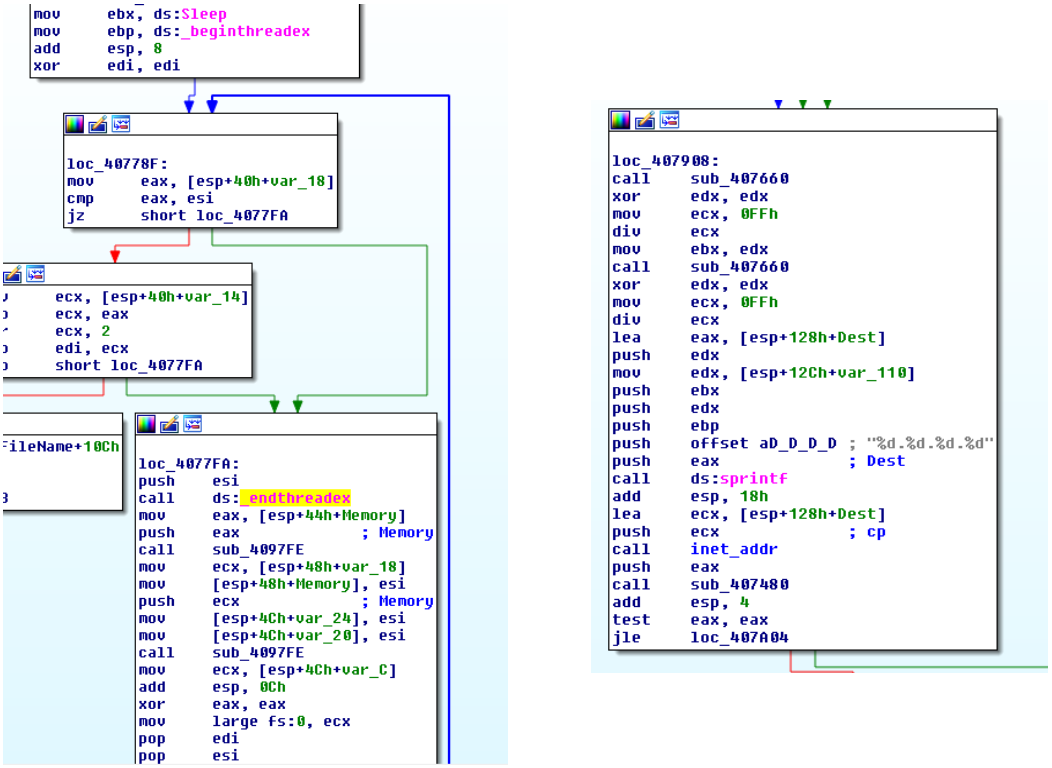


Ilustración 26: Propagación de WannaCry por la red local (a) y Propagación de WannaCry por internet (b)

Por último, ambos hilos terminan llamando a sub_407480 para intentar conectar con el puerto 445 según se muestra en la Imagen 27. Si la conexión es exitosa, se llama a Sub_407440, que es el *exploit* comúnmente conocido como EthernalBlue.

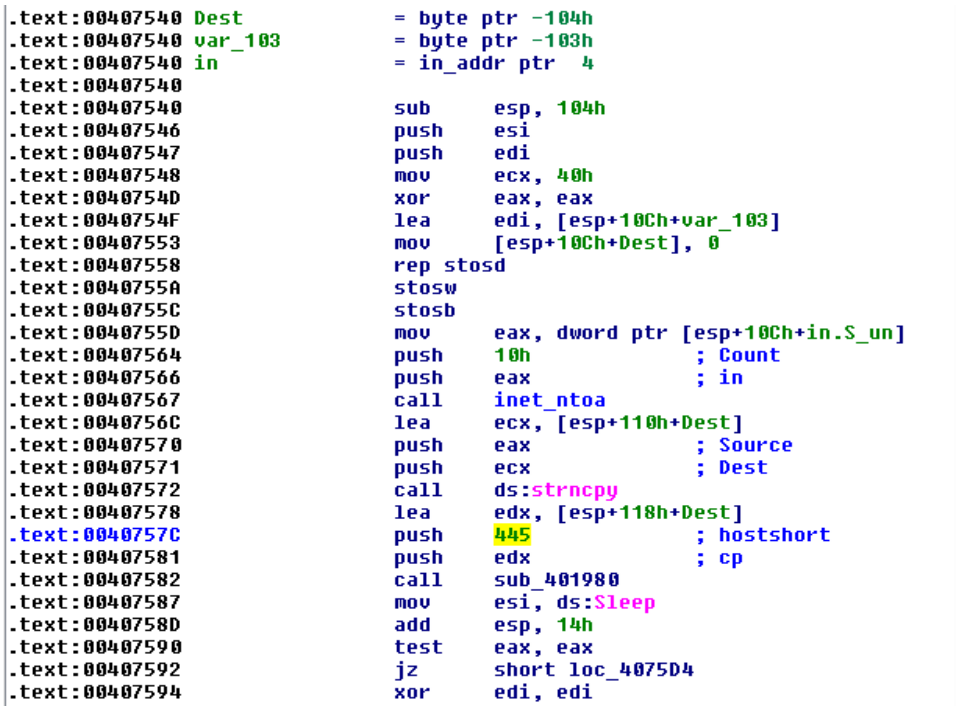


Ilustración 27: Conexión del programa al puerto 445

4.4. Análisis dinámico de muestra1.bin.exe parte 1

En este apartado, se presenta el análisis dinámico del malware. Este consiste en depurarlo paso a paso, terminando de conocer los detalles sobre las acciones que realiza el programa, poniendo puntos de ruptura a lo largo del código, mientras en los programas Procexp [40], que es una herramienta para monitorizar los recursos y Procmon [41], que se trata de una herramienta que monitorea y muestra en tiempo real, toda la actividad del sistema de archivos en un sistema operativo Microsoft Windows donde, observamos los eventos, registros y servicios que se van añadiendo o modificando. Se hace uso también de Wireshark [49], que es un analizador de protocolos de red, para capturar posibles conexiones a internet por parte del software. A continuación, se enumeran los procesos llevados a cabo para el análisis.

1) Se abre Wireshark y se inicia la captura de paquetes.
2) Se ejecuta el programa Ida pro con el malware y, se sitúa un punto de ruptura en la función InternetOpenA y otro, en la llamada a la función sub_408090 que, como se observó anteriormente en el análisis estático, llamaba en caso de no encontrar conexión.

3) Se inicia la depuración del programa y, pasando el punto de ruptura InternetOpenA, se percibe la finalización del programa.
4) Se procede a la parada de la captura de paquetes de Wireshark y se filtra por DNS, viendo en la ilustración 28, cómo se resuelve la conexión DNS del dominio www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com.

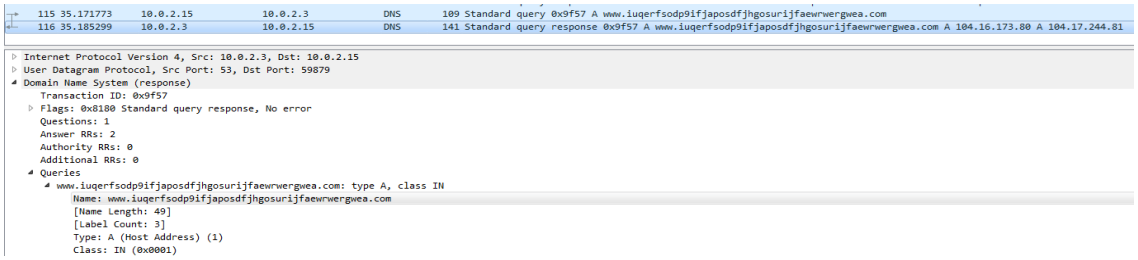


Ilustración 28: Conexión DNS del programa muestra1.bin.exe

5) Se filtra en Wireshark por HTTP. En la imagen 29, se puede ver como se obtiene una conexión favorable con el dominio anteriormente mencionado. La resolución de una conexión favorable es debido, a que el dominio se registró para frenar la propagación de WannaCry, por lo que se debe cambiar el dominio por otro que no pueda resolver una conexión DNS con la finalidad de entrar en la subrutina sub_ 408090.

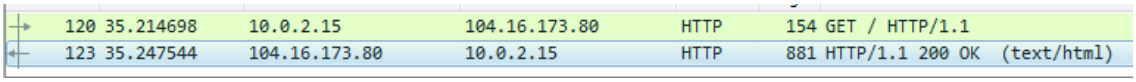


Ilustración 29: Conexión HTTP del programa muestra1.bin.exe

6) En IDA pro se busca y selecciona la URL www.iuqerfsodp9ifjaposdfjhgosurijfaewrwergwea.com mencionada en el paso 5) . Se visualiza la dirección de memoria en la que es asignada esta URL a la variable ahttpwww_iuqeru. Seguidamente se va al programa HxD, realizando una búsqueda de la dirección de memoria, donde se obtiene el resultado representado en la ilustración 30.

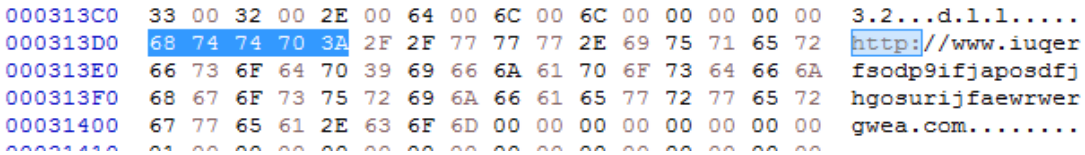


Ilustración 30: Búsqueda del dominio kill switch en HxD

7) Se cambian los Bytes en HEX con el fin de modificar el dominio, como puede observarse en la ilustración 31.

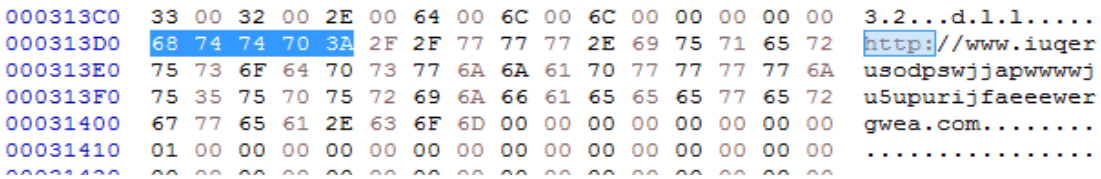


Ilustración 31: Modificación del dominio Kill switch en HxD

8) Se repite el proceso para observar en la imagen 32, como la resolución DNS del dominio es desfavorable.

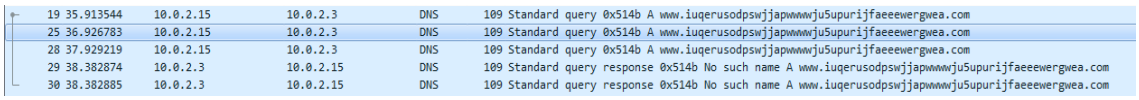


Ilustración 32: Resolución DNS con dominio modificado fallida

9) Se fijan el resto de breakpoints en el programa con el fin de depurarlo paso a paso y ver el funcionamiento total. En la ilustración 33 y 34, se pueden ver los puntos de ruptura más característicos, siendo estos la creación del servicio mssecsvs2.0 y la creación de las rutas C:\WINDOWS\tasksche.exe y C:\WINDOWS\qeriuwjhrf.

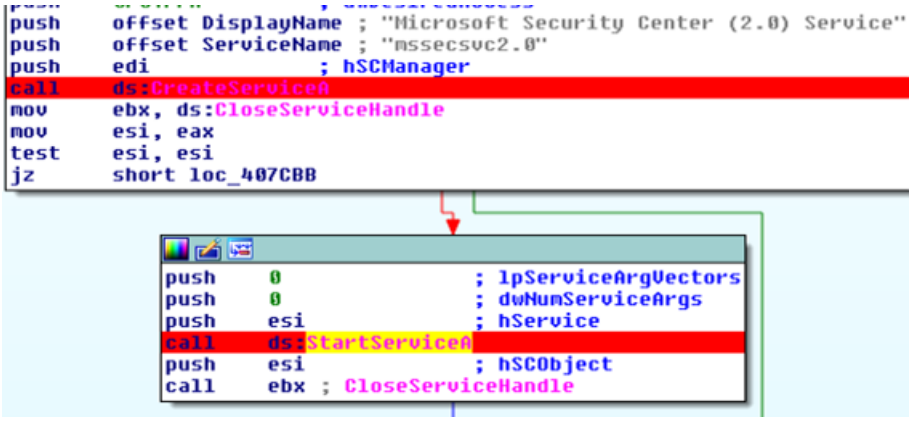
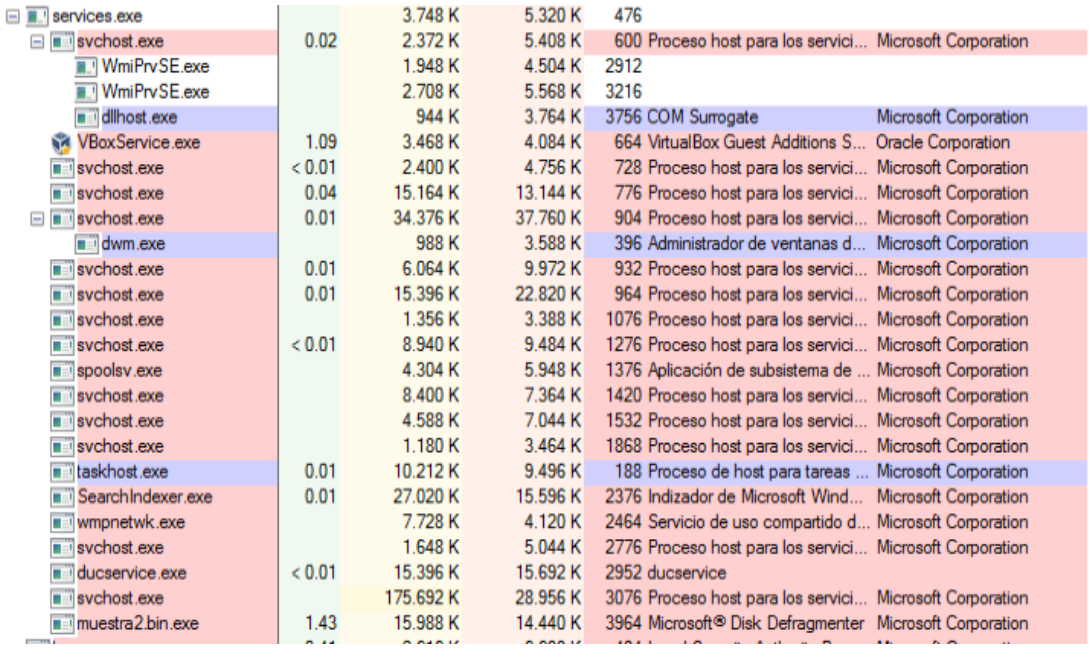


Ilustración 33: Breakpoint en la creación de servicio mssecsvc2.0

Ilustración 34: Breakpoint para la creación de paths en el sistema

10) Se reanuda la depuración y llegando al primer breakpoint, se depura el código línea por línea, para entender su funcionamiento. Se observa en la aplicación Procexp, representado en la ilustración 35, como se registra el servicio dentro de services.exe, cuando antes se encontraba colgando de idaq.exe.



services.exe		3.748 K	5.320 K	476		
svchost.exe	0.02	2.372 K	5.408 K	600	Proceso host para los servici...	Microsoft Corporation
WmiPrvSE.exe		1.948 K	4.504 K	2912		
WmiPrvSE.exe		2.708 K	5.568 K	3216		
dllhost.exe		944 K	3.764 K	3756	COM Surrogate	Microsoft Corporation
VBoxService.exe	1.09	3.468 K	4.084 K	664	VirtualBox Guest Additions S...	Oracle Corporation
svchost.exe	< 0.01	2.400 K	4.756 K	728	Proceso host para los servici...	Microsoft Corporation
svchost.exe	0.04	15.164 K	13.144 K	776	Proceso host para los servici...	Microsoft Corporation
svchost.exe	0.01	34.376 K	37.760 K	904	Proceso host para los servici...	Microsoft Corporation
dwm.exe		988 K	3.588 K	396	Administrador de ventanas d...	Microsoft Corporation
svchost.exe	0.01	6.064 K	9.972 K	932	Proceso host para los servici...	Microsoft Corporation
svchost.exe	0.01	15.396 K	22.820 K	964	Proceso host para los servici...	Microsoft Corporation
svchost.exe		1.356 K	3.388 K	1076	Proceso host para los servici...	Microsoft Corporation
svchost.exe	< 0.01	8.940 K	9.484 K	1276	Proceso host para los servici...	Microsoft Corporation
spoolsv.exe		4.304 K	5.948 K	1376	Aplicación de subsistema de ...	Microsoft Corporation
svchost.exe		8.400 K	7.364 K	1420	Proceso host para los servici...	Microsoft Corporation
svchost.exe		4.588 K	7.044 K	1532	Proceso host para los servici...	Microsoft Corporation
svchost.exe		1.180 K	3.464 K	1868	Proceso host para los servici...	Microsoft Corporation
taskhost.exe	0.01	10.212 K	9.496 K	188	Proceso de host para tareas ...	Microsoft Corporation
SearchIndexer.exe	0.01	27.020 K	15.596 K	2376	Indizador de Microsoft Wind...	Microsoft Corporation
wmpnetwk.exe		7.728 K	4.120 K	2464	Servicio de uso compartido d...	Microsoft Corporation
svchost.exe		1.648 K	5.044 K	2776	Proceso host para los servici...	Microsoft Corporation
ducservice.exe	< 0.01	15.396 K	15.692 K	2952	ducservice	
svchost.exe		175.692 K	28.956 K	3076	Proceso host para los servici...	Microsoft Corporation
muestra2.bin.exe	1.43	15.988 K	14.440 K	3964	Microsoft® Disk Defragmenter	Microsoft Corporation

Ilustración 35: Registro de mssecsvc2.0 en el sistema

11) Se abre el programa de Windows services.msc para buscar el servicio. En la figura 36, se muestra como viene registrado el mismo por el nombre de Microsoft Security Center(2.0) Service y, sin embargo, abriendo el servicio aparece con el nombre de mssecsvc2.0.

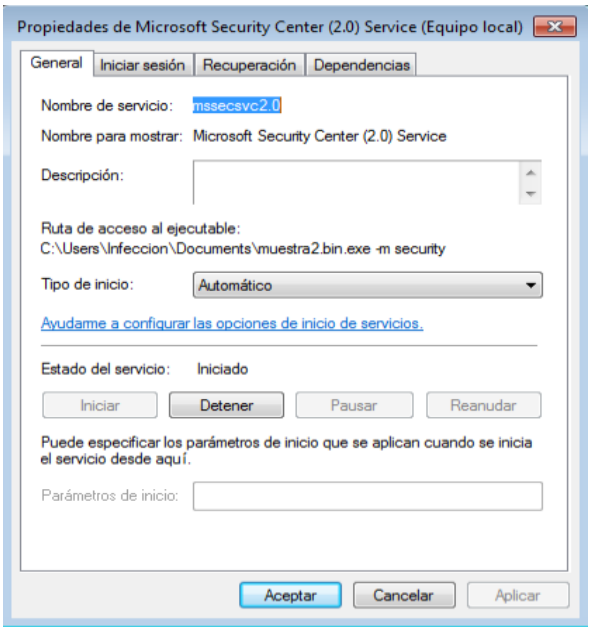


Ilustración 36: Muestra del servicio mssecsvc2.0 en el sistema

12) Se avanza hasta el segundo breakpoint, mostrando en el programa Procmon la comprobación de la existencia del fichero C:\windows\tasksche.exe, y al no existir, lo crea tal y como se representa en la ilustración 37. En caso de existir el path anteriormente mencionado, lo mueve a la segunda ruta C\windows\qeriuwjhfr y seguidamente crea el path de nuevo.

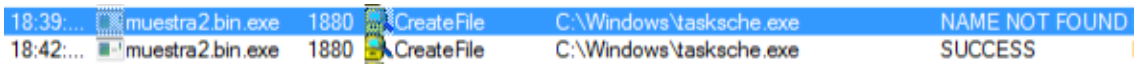


*Ilustración 37: Creación del path C:\windows*tasksche.exe *en el sistema*

Una vez creado el path, copia el recurso R en C:\windows\tasksche.exe y lo ejecuta mediante una query empezando el proceso de cifrado del equipo, así como se muestra en la ilustración 38.

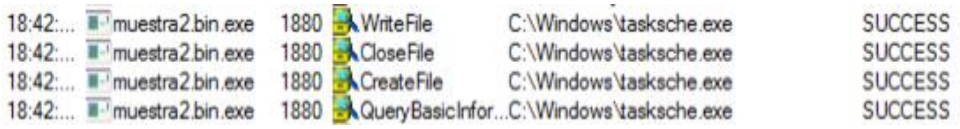


Ilustración 38: Ejecución de tasksche.exe *en el sistema*

Mediante el análisis dinámico del software malicioso, se comprende que el programa principal ejecuta un segundo programa llamado tasksche.exe que, es el encargado de cifrar el equipo, es decir, no solo hay un programa implicado en el objetivo de este *ransomware*, si no que muestra1.bin.exe que, se inicia como servicio en el sistema con el nombre mssecsvc2.0.exe, ejecuta otro programa llamado tasksche.exe. Se toma muestra de este nuevo programa y, se procede a analizarlo primero de forma estática y a posterior dinámicamente.

4.5. Análisis estático básico de tasksche.exe

En este apartado se representa el análisis estático básico del ejecutable tasksche.exe. Para este cometido, se repite el proceso llevado a cabo cuando se realizó el análisis de muestra1.bin.exe. A continuación, se enumeran los pasos llevados a cabo para el análisis.

1) Obtención de los hashes. En la imagen 40, se muestra la obtención de los hashes de programa.

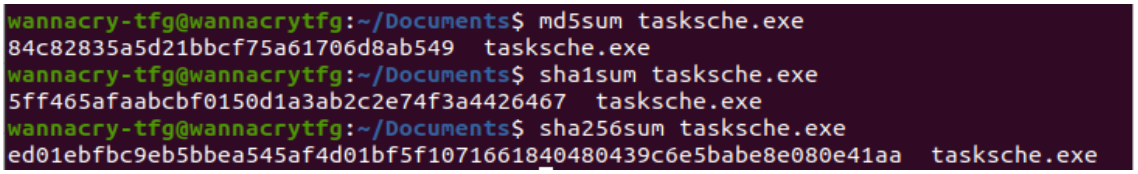


Ilustración 39: Obtención de hashes de programa tasksche.exe

2) En este paso, se localiza el tipo de archivo que es, a través el comando file en Terminal. En la ilustración 41, se muestra como el resultado es el mismo que en el análisis de la muestra1.bin.exe.

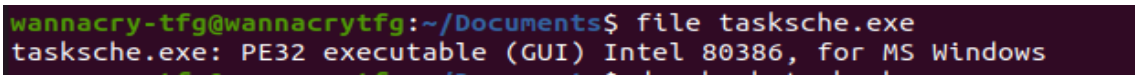


Ilustración 40: Tipo de archivo tasksche.exe

3) Se obtiene mediante el comando du de Terminal el tamaño de la muestra, siendo este 3,6M.
4) No sería necesario sacar los strings, debido a que como muestra1.bin.exe crea tasksche.exe, los strings que se obtendrían, serían una parte de los obtenidos en el anterior ejecutable.
5) Se carga la muestra en el programa HxD, con el fin de visualizar el resto de atributos. Se observa como se muestra en la ilustración 42, que se trata de un PE de 32 bits, exactamente igual que muestra1.bin.exe y tiene las mismas secciones de programa.

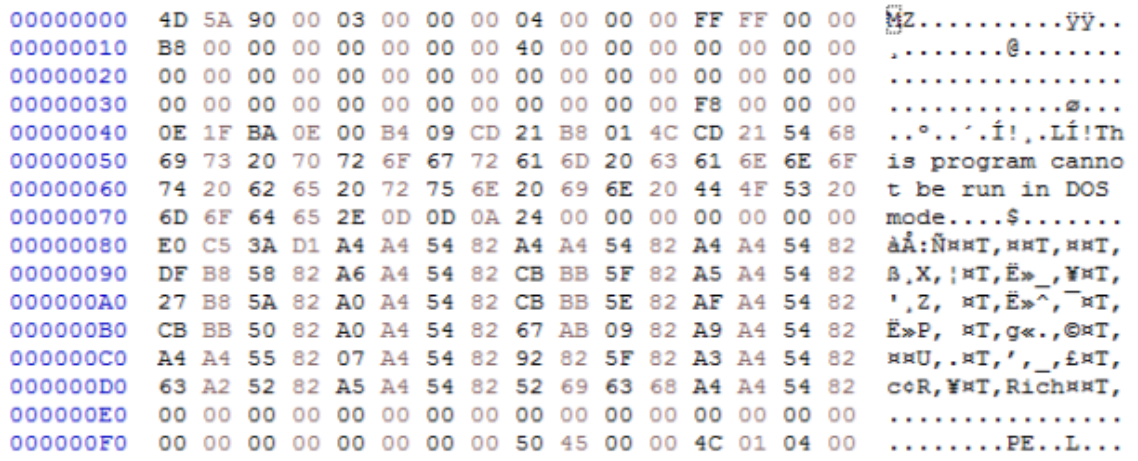


Ilustración 41: Lectura del binario de tasksche.exe *desde HxD*

4.6. Análisis estático avanzado de tasksche.exe

En este apartado, la tarea a realizar es el análisis estático avanzado del ejecutable tasksche.exe. Para llevar a cabo este análisis, se van a seguir los mismos pasos que se realizaron para analizar muestra1.bin.exe. A continuación, se enumeran los pasos que se han seguido para la realización de este análisis.

1) Se procede a la ejecución de IDA pro y cargar el programa tasksche.exe. Lo primero que realiza el programa al ejecutarse, es acceder al nombre de la carpeta en la cual se encuentra, llamando justo después a la subrutina sub_401225.
2) Tras pulsar el botón F5 para que IDA pro cree un seudocódigo y así entender mejor la ejecución del programa en este fragmento de código, se visualiza el seudocódigo y se estudian las funciones GetComputerNameW, Wcslen y Srand que usa el software. En la ilustración 43, se representa como el código genera un valor pseudoaleatorio numérico mediante las funciones wcslen y srand.
    - GetComputerNameW: recupera el nombre NETBIOS del equipo local, es decir, se intenta obtener el nombre del equipo [51].
    - Wcslen: obtiene la longitud de una cadena [52].
    - Srand: generación de números pseudoaleatorios [53].

```
int __cdecl sub_401225(int a1)
{
  unsigned int v1; // ebx@1
  WCHAR *v2; // edi@2
  size_t v3; // eax@3
  int v4; // edi@4
  int v5; // esi@4
  int v6; // esi@6
  int result; // eax@9
  WCHAR Buffer; // [sp+Ch] [bp-198h]@1
  char v9; // [sp+Eh] [bp-196h]@1
  __int16 v10; // [sp+19Ah] [bp-Ah]@1
  DWORD nSize; // [sp+19Ch] [bp-8h]@1
  unsigned int v12; // [sp+1A0h] [bp-4h]@1

  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;
```

Ilustración 42: Creación directorio pseudoaleatorio en el sistema

El programa verifica si hay un parámetro /i representado en la ilustración 44. Si existe este parámetro, se autocopia en C:/programData o C:\intel o en el directorio temporal del sistema más el directorio con valor pseudoaleatorio calculado anteriormente. Estas últimas tres rutas se extraen de la subrutina sub_401B5F.

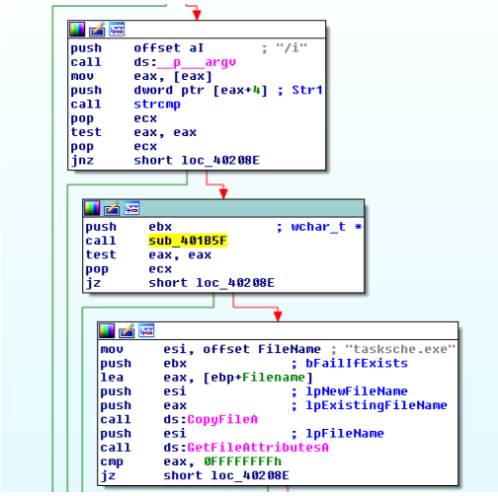


Ilustración 43: Verificación de existencia \i y copia en ruta pseudoaleatoria

Cuando el fichero se ha copiado en alguno de los tres paths, llama a la subrutina sub_401F5D. En esta subrutina, se guarda el path completo en el que se encuentra y llama a sub_401CE8, donde como puede verse en la imagen 45, se registra como servicio y se inicia creando persistencia.

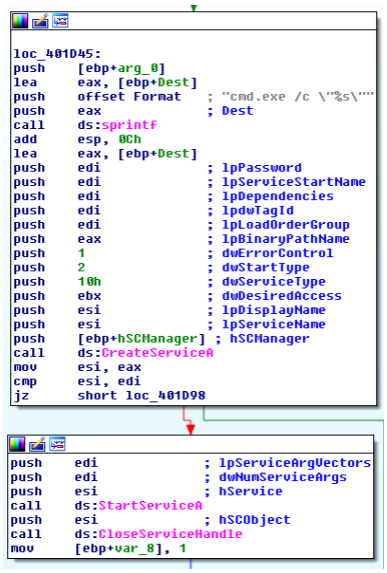


Ilustración 44: Registro de tasksche.exe *como servicio en el sistema*

Cuando el programa ya es persistente en el sistema, realiza un Mutex global determinando si hasta el momento ha sido exitosa la infección del equipo, si no es así, acaba el programa [54]. En caso de que la infección sea exitosa,

como se puede ver en la ilustración 46, se libera un recurso llamado XIA al que se le transfiere un parámetro como clave: “*WNcry@2ol7”*.

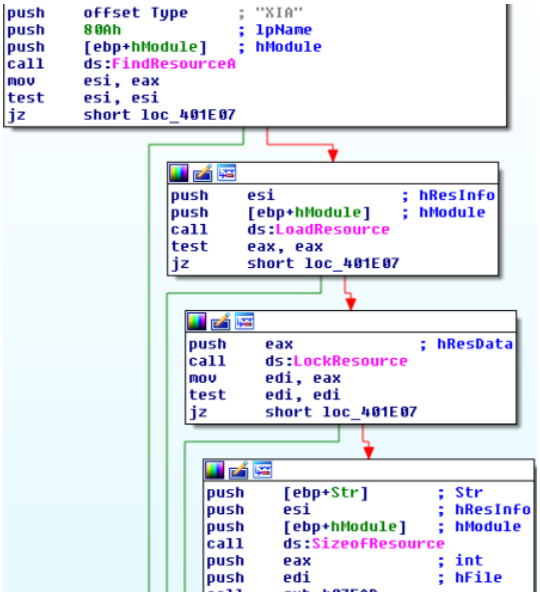


Ilustración 45: Liberación de recurso XIA en el sistema

Posteriormente, tasksche.exe lee un archivo llamado c.wnry del directorio actual y, a continuación, como se muestra en la ilustración 47, ejecuta dos comandos. El primero es attrib, es un comando para establecer o quitar atributos a archivos o directorios [55]. El segundo comando es icacls que, muestra o modifica las DACL en los archivos especificados [56].

```
push    offset CommandLine ; "attrib +h ."
call    sub_401064
push    ebx             ; lpExitCode
push    ebx             ; dwMilliseconds
push    offset aIcacls_GrantEv ; "icacls . /grant Everyone:F /T /C /Q"
call    sub_401064
```

Ilustración 46: Ejecución de comandos atrib e icacls

A continuación, en la figura 47, se puede observar cómo carga dos librerías, Kernel32.dll en la imagen (a) , es una librería que tiene como objetivo hacer que las funciones del sistema estén servibles para los programas con el

fin de poder iniciar o detener procesos, gestionar memoria o entrada y salida [57]. Por otro lado, la segunda librería es Avapi32.dll en la imagen (b), se trata de una librería diseñada para admitir varias APIs, incluidas las llamadas de seguridad y de registro. Para realizar la obtención de los campos de cada una de las APIs, se usa la función GetProcAdress [58].

(a) (b)

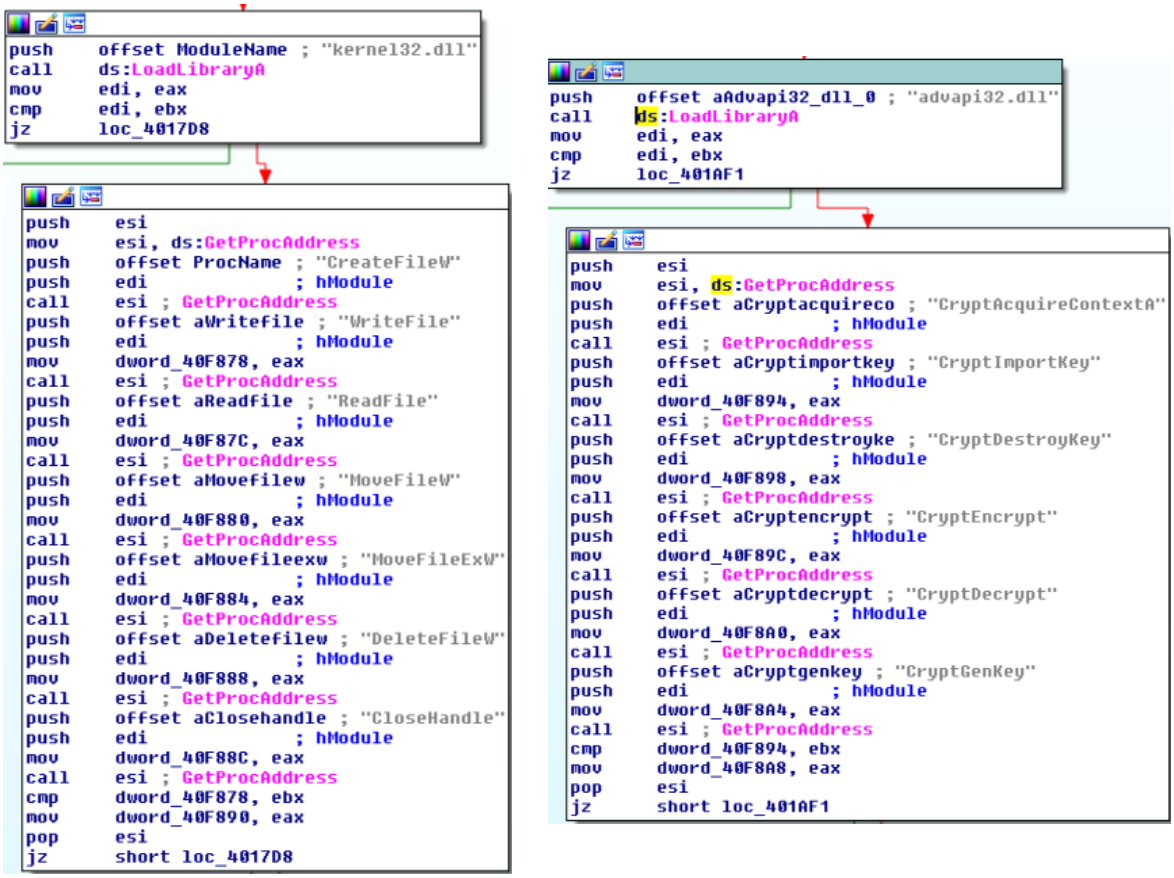


Ilustración 47: Librería Kernel32.ddl y librería Advapi32.ddl (b)

Por último, como se representa en la imagen 52, el programa lee un archivo t.wncry, lanza un taskstart y comienza a cifrar todos los datos. Para entender con mas profundidad el funcionamiento de este ejecutable, a continuacion, se procede a realizar el analisis dinamico de la muestra.

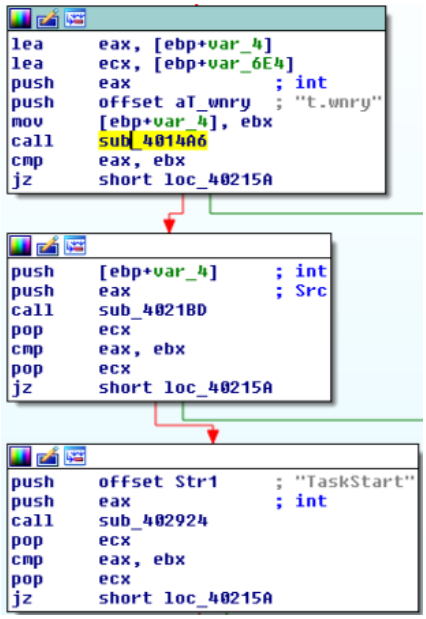


Ilustración 48: Lectura de recurso t.wnry y comienzo de cifrado

4.7. Análisis dinámico de tasksche.exe

En este apartado, se va a proceder a realizar el análisis dinámico de tasksche.exe. Para la realización de este análisis, se repite el mismo proceso que cuando se realizó el análisis de muestra1.bin.exe., enumerando los pasos realizados para llevar a cabo el estudio del malware.

1) Se fija un punto de interrupción en sub_401CE8, subrutina donde se hace persistente y se registra como servicio en el sistema.
2) Se depura el software y, como se observa en la imagen 50, que procede del programa Procexp, como se crea como servicio mediante una query.

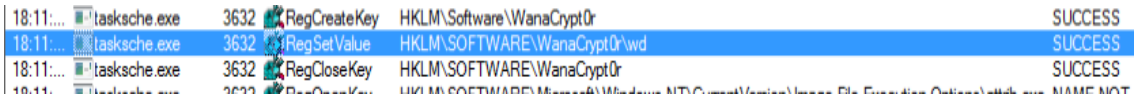


Ilustración 49: Creación como servicio de tasksche.exe

En la subrutina sub_ 401CE8 se divisa, cómo crea una entrada en el registro, apuntando a la carpeta donde está el malware. Este directorio, posee un nombre aleatorio que corresponde con *"fdskxvzjxaauxcaw817"* que es el

calculado anteriormente en sub_401225. De esta manera, es capaz de garantizar una persistencia tras el reinicio del equipo [59].

3) Al pasar al siguiente punto de ruptura del programa, se produce la liberación del recurso “Xia”. Este recurso, es un .zip que se desempaqueta con la contraseña WNcry@2ol7, que se encuentra en texto claro en el código y extrae del mismo los siguientes archivos.

    - **b.wnry**: imagen de fondo de pantalla.
    - **c.wnry:** archivo de configuración.
    - **r.wnry**: contiene una nota de rescate de texto, que luego sustituye el nombre por *@Please_Read_me@.txt* que se encontrará en cada directorio junto a *@WanaDecryptor@.exe.*
    - **s.wnry:** zip con las librerías de TOR.
    - **u.wnry:** código para crear *@WanaDecryptor@.exe.*
    - **@WanaDecryptor@.exe**: programa encargado de descifrar el equipo y dejarlo como estaba antes de la infección.
    - **00000000.pky :** en caso de no existir, crea un par de claves RSA.
    - **00000000.eky :** la clave privada generada se cifra con la pública y se guarda aquí.
    - **00000000.res:** archivo con la información de contadores de tiempo.

Tras haber realizado el análisis de todo el código, se puede entender la cadena de llamadas entre ejecutables que sigue el programa. El primer ejecutable llamado muestra1.bin.exe que, se inicia como servicio en el sistema con el nombre de mssecsvc2.0.exe, se encarga de propagar el ransomware por la red. El segundo ejecutable llamado tasksche.exe, es llamado por muestra1.bin.exe, que tiene como cometido cifrar todos los datos del equipo, y este a su vez, ejecuta *@WanaDecryptor@.exe* que es el encargado de descifrar todos los archivos y dejar la maquina tal y como se encontraba antes de la infección [60].

5. Prevención y medidas de seguridad

La infección de un equipo por este ransomware, puede ser realmente devastador , debido al riesgo que existe de perder información . Del mismo modo,

el riesgo aumenta potencialmente para las empresas, puesto que, estos datos sustraídos y cifrados podrían ser de suma importancia o de extrema privacidad para las funciones de determinadas empresas. Por esta razón, es muy importante seguir las pautas de prevención y medidas de seguridad, que se enumeran a continuación y que sirven para prevenir la infección de los equipos por este malware.

1) Actualizar o crear copias de seguridad periódicas (backups). Es necesario la creación y mantenimiento de copias de seguridad de todos los archivos potencialmente importantes. Es recomendable tener estas copias de seguridad aisladas y sin conectividad con otros sistemas o internet.
2) Mantener el sistema operativo actualizado con los últimos parches de seguridad. Es de los pasos más importantes puesto que, estos parches de seguridad son para reparar vulnerabilidades que se podrían explotar para llevar a cabo la infección o propagación. Para evitar la propagación de WannaCry, es necesario tener instalado el parche MS17-010.
3) Deshabilitar el puerto 445 , ya que utiliza esta vía para propagarse, y activar el cortafuegos de Windows, para que frene este tipo de softwares maliciosos.
4) Disponer de un antivirus instalado en el equipo. Este antivirus debe estar actualizado para que contenga las ultimas firmas en su base de datos.
5) Impedir la ejecución de ficheros desde directorios utilizados por el *ransomware* como App Data o Windows.
6) Utilizar una cuenta que no tenga permisos de administrador, de este modo, el *ransomware* no podrá ejecutarse por completo, reduciendo así el impacto.
7) Eliminar los correos sospechosos o analizar los enlaces de los Emails recibidos y no abrir adjuntos de usuarios desconocidos, debido a que pueden ser un enlace de descarga o un archivo “trampa” que busca ser descargado o ejecutado para infectar el equipo.
8) Si el equipo está siendo infectado por el *ransomware*, se debe matar el proceso de ejecución del malware con el fin de frenarlo. En caso de no poder matarlo o no tener los conocimientos suficientes para ejecutar esta tarea, se debe apagar el equipo de inmediato para parar su infección.

6. Conclusiones y trabajo futuro

En este trabajo de fin de grado, se ha estudiado en profundidad el *ransomware* WannaCry aplicando técnicas de reversing con el fin de entender los procesos llevados a cabo para propagarse, infectar y cifrar uno o varios equipos. Para realizar el proceso de ingeniería inversa del malware, se ha estudiado el código desensamblado de una muestra de WannaCry que, corresponde con el hash MD5 *DB349B97C37D22F5EA1D1841E3C89EB4* y, se ha ejecutado en un ambiente controlado para comprobar y analizar la afectación que tiene sobre el equipo.

Se ha aprendido sumamente de este análisis mediante técnicas de reversing y, no menos importante, sobre el propio *ransomware* WannaCry, también sobre la importancia que merece tener los equipos actualizados ante vulnerabilidades conocidas, dando lugar a escenarios más favorables en los cuales este software malicioso no habría podido propagarse.

En el mercado se encuentran infinidad de programas para realizar este tipo de análisis. Dependiendo de la muestra que se necesite analizar, existen programas más adecuados de los que se han utilizado para este propósito. Sin embargo, en esta ocasión, los programas elegidos han sido los más idóneos, debido a las capacidades de los mismos. Un ejemplo de esto, es la capacidad que tiene IDA pro para crear mapas de ejecución, mostrándolo en un formato gráfico parecido a ilustraciones.

Después de la realización de este trabajo, puedo resumir, que lo más destacable y, que me ha llamado la atención son, por un lado , que este proceso completo no se lleva a cabo por un solo ejecutable, sino que se van llamando de uno a otro para completar el proceso de cifrado. Por otro lado, la forma en que interactúan los ejecutables con los servicios y permisos de Windows. Igualmente, quiero destacar como escanean las redes con el propósito de propagarse usando el *exploit* EthernalBlue. Pero sin lugar a duda, lo que más me ha llamado la atención, con relación al análisis, es el “Kill switch” y, el modo en que se consigue parar la infección de esta “anomalía” independientemente del equipo, gracias a una consulta DNS.

Por la relativa complejidad y disposición del software para este tipo de análisis, se abre una gran variedad de futuras líneas de análisis, que podrían mejorar sustancialmente el trabajo realizado, utilizando otros programas que aporten más información al respecto. Aun así, el procedimiento llevado a cabo en este TFG para el análisis sería totalmente válido para analizar otras muestras de malware, ya sean o no de tipo *ransomware*. Igualmente, sería de gran ayuda, analizar distintas versiones del mismo *ransomware* para hacernos una idea de la evolución seguida y como podría evolucionar en un futuro, documentado las vulnerabilidades que aprovecha y los métodos utilizados para propagarse o infectar, y así encontrar formas más eficaces de detenerlo o prevenirlo.

7. Referencias

[1] N. Latto, «Avast,» Avast, 19 Mayo 2021. [En línea]. Available: https://www.avast.com/es-es/c-wannacry.

[2] Anonimo, «Ethernalblue,» wikipedia, 6 Julio 2020. [En línea]. Available: https://es.wikipedia.org/wiki/EternalBlue.

[3] J. Jiménez, «redeszone,» 25 Junio 2020. [En línea]. Available: https://www.redeszone.net/tutoriales/seguridad/reversing-malware-ciberseguridad/.

[4] © 2021 AO Kaspersky Lab. Todos los derechos reservados., «kaspersky,» 2021. [En línea]. Available: https://www.kaspersky.es/resource-center/threats/what-is-cybercrime.

[5] I. Belcic, «Avast,» 2019 Septiembre 2019. [En línea]. Available: https://www.avast.com/es-es/c-malware.

[6] © 2021 AO Kaspersky Lab. Todos los derechos reservados, «Kaspersky,» 2021. [En línea]. Available: https://www.kaspersky.es/resource-center/threats/trojans.

[7] M. S. a. A. Honing, Practical Malware Analysis,, No Starch Press: The Hands-On Guide to Dissecting Malicious Software, 2012.

[8] G. O. a. G. McDonald, Ransomware: A Growing Menace, ymantec Corporation, 2012.

[9] INCIBE, «Incibe,» 30 Mayo 2019. [En línea]. Available: https://www.incibe.es/protege-tu-empresa/blog/hace-antivirus-detectar-el-malware.

[10] N. F. Ignacio Esmite, «edu.uy,» 2017 Mayo 2017. [En línea]. Available: https://www.fing.edu.uy/inco/grupos/gsi/thesis/analisis-malware/.

[11] m. h. sat, «zonavirus,» 26 Junio 2020. [En línea]. Available: https://zonavirus.com/noticias/2020/que-es-el-reversing-de-malware-en-la-ciberseguridad_70030.

[12] Anonimo, «wikipedia,» 14 Marzo 2021. [En línea]. Available: https://es.wikipedia.org/wiki/WannaCry.

[13] « Campus Internacional de Ciberseguridad,» 10 ABR 2020. [En línea]. Available: https://www.campusciberseguridad.com/blog/item/140-que-es-el-reversing#:~:text=%E2%80%9CEl%20Reversing%20de%20malware%20es,%2C%20desarrollar%20medidas%20de%20protecci%C3%B3n.%E2%80%9D.

[14] H. M. &. U. L. C. García, «Hidden Tear: Análisis del primer Ransomware Open Source.,» Academia de Sistemas Computacionales, Instituto Tecnológico Superior Progreso, 2015. [En línea]. Available: https://d1wqtxts1xzle7.cloudfront.net/52336759/Hidden_Tear.pdf?1490668841=&response-content-disposition=inline%3B+filename%3DHidden_Tear_Analisis_del_primer_Ransomwa.pdf&Expires=1620671170&Signature=fzk9dQ9f3tZlVHQFQBK~Y7pMvDyblrvqL2~Iq7bXV9do4zR7uRdsEEI~.

[15] D. Jaimovich, «Infobae América,» Infobae América, 12 Mayo 2018. [En línea]. Available: https://www.infobae.com/america/tecno/2018/05/12/como-surgio-y-se-propago-wannacry-uno-de-los-ciberataques-mas-grandes-de-la-historia/#:~:text=WannaCry%20comenz%C3%B3%20un%2012%20de,en%20150%20pa%C3%ADses%20fueron%20afectadas.&text=WannaCry%20no%20se%20si.

[16] Anonimo, «Ciberataque masivo: ¿quiénes fueron los países e instituciones más afectados por el virus WannaCry?,» BBc Mundo, 15 Mayo 2017. [En línea]. Available: https://www.bbc.com/mundo/noticias-39929920.

[17] E. consultores, «¿QUÉ ES ETERNALBLUE? ¿CÓMO LO UTILIZAN LOS CIBERDELINCUENTES PARA HACKEAR MILLONES DE COMPUTADORAS CON WINDOWS?,» BLOG EHC GROUP, 13 Septiembre 2019. [En línea]. Available: https://blog.ehcgroup.io/2019/09/13/14/55/37/6420/que-es-eternalblue-como-lo-utilizan-los-ciberdelincuentes-para-hackear-millones-de-computadoras-con-windows/delitos-informaticos/ehacking/.

[18] © Microsoft 2021, « © Microsoft 2021,» 2017 Marzo 14. [En línea]. Available: https://support.microsoft.com/es-es/topic/ms17-010-actualizaci%C3%B3n-de-seguridad-para-windows-server-de-smb-14-de-marzo-de-2017-435c22fb-5f9b-f0b3-3c4b-b605f4e6a655.

[19] R. Velasco, «Redes Zone,» 27 Junio 2018. [En línea]. Available: https://www.redeszone.net/2018/06/27/doublepulsar-exploit-nsa-iot/.

[20] N. Latto, «¿Qué es WannaCry?,» Avast, 27 Febrero 2020. [En línea]. Available: https://www.avast.com/es-es/c-wannacry#:~:text=WannaCry%20se%20propagaba%20por%20medio,al%20uso%20del%20c%C3%B3digo%20EternalBlue..

[21] R. BARCELONA, «Éste es Marcus Hutchins, el joven que paró el ciberataque WannaCry,» La vanguardia, 15 Mayo 2017. [En línea]. Available: https://www.lavanguardia.com/tecnologia/20170515/422599023159/marcus-hutchins-ciberataque-virus-wannacry.html.

[22] J. F. Reyes, «Quadernos de criminología: revista de criminología y ciencias forenses,» Firma invitada: El virus" Wannacry"., 2017. [En línea]. Available: dialnet.unirioja.es.

[23] «¿QUÉ ES EL REVERSING DE MALWARE?,» Campus Internacional de Ciberseguridad, 10 Abril 2020. [En línea]. Available: https://www.campusciberseguridad.com/blog/item/140-que-es-el-reversing#:~:text=%E2%80%9CEl%20Reversing%20de%20malware%20e

s,%2C%20desarrollar%20medidas%20de%20protecci%C3%B3n.%E2%80%9D.

[24] R. Marín, «Recomendaciones para realizar el proceso de Ingeniería Inversa.,» Revista digital inesem, 28 Enero 2019. [En línea]. Available: https://revistadigital.inesem.es/informatica-y-tics/ingenieria-inversa/.

[25] © 2021 Heaventools Software. , «Heaventools,» 2021. [En línea]. Available: http://www.pe-explorer.com/.

[26] © 2021 Full Stack Technology FZCO, «FileHorse,» 18 Febrero 2021. [En línea]. Available: https://www.filehorse.com/es/descargar-w32dasm/.

[27] 2021 Slashdot Media, «SourceForge,» 16 04 2015. [En línea]. Available: https://sourceforge.net/projects/distorm/.

[28] Hex Rays, «Hex-rays,» 2021. [En línea]. Available: https://hex-rays.com/ida-pro/.

[29] 2001-2021 Softpedia, «Softpedia,» 9 Julio 2014. [En línea]. Available: https://www.softpedia.com/get/Programming/Debuggers-Decompilers-Dissasemblers/Borg.shtml.

[30] 2021 Slashdot Media, «SourceForge,» 1 Noviembre 2015. [En línea]. Available: https://sourceforge.net/projects/hte/.

[31] Program-Transformation.Org: Wiki de transformación de programas, «Program-Transformation.Org: Wiki de transformación de programas,» 02 Marzo 2015. [En línea]. Available: http://program-transformation.org/Transform/DccDecompiler.

[32] .sourceforge., «boomerang.sourceforge.net,» 28 Octubre 2012. [En línea]. Available: http://boomerang.sourceforge.net/.

[33] 1997 - 2015 Backer Street Software , «Backer Street,» 16 Noviembre 2015. [En línea]. Available: http://www.backerstreet.com/rec/rec.htm.

[34] 2000-2014 Oleh Yuschuk, «ollydbg,» 05 Mayo 2014. [En línea]. Available: https://www.ollydbg.de/.

[35] microsoft, «microsoft,» 05 Enero 2020. [En línea]. Available: https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/debugger-download-tools.

[36] Mediaprogramas 2021, «programas,» 2021. [En línea]. Available: https://www.programas.com/descargar_/visual-duxdebugger.

[37] Wikipedia®, «wikipedia,» 24 Febrero 2020. [En línea]. Available: https://es.wikipedia.org/wiki/SoftICE.

[38] 2003-2020 Maël Hörz., «Nexus,» 11 Febrero 2021. [En línea]. Available: https://mh-nexus.de/en/hxd/.

[39] x64dbg, «x64dbg,» 2021. [En línea]. Available: https://x64dbg.com/#start.

[40] © Microsoft 2021, «Microsoft,» 06 Enero 2021. [En línea]. Available: https://docs.microsoft.com/en-us/sysinternals/downloads/process-explorer.

[41] © Microsoft 2021, «Microsoft,» 06 Enero 2021. [En línea]. Available: https://docs.microsoft.com/en-us/sysinternals/downloads/procmon.

[42] A. (. KZKG^Gaara), «GNU/Linux, Tutoriales/Manuales/Tips,» Desde Linux, [En línea]. Available: https://blog.desdelinux.net/con-el-terminal-comandos-de-tamano-y-espacio/.

[43] L. Paus, «WebLiveSecurity,» 15 octubre 2015. [En línea]. Available: https://www.welivesecurity.com/la-es/2015/10/01/extension-de-un-archivo-cabeceras/.

[44] Snifer@L4b's, «Snifer@L4b's,» 17 Enero 2017. [En línea]. Available: https://sniferl4bs.com/2017/01/identificando-si-un-ejecutable-es-de-32-o-64-bits-con-un-editor-hexadecimal/.

[45] Virus Total, «Virus Total,» Junio 2021. [En línea]. Available: https://www.virustotal.com/gui/file/24d004a104d4d54034dbcffc2a4b19a11f39008a575aa614ea04703480b1022c/detection.

[46] Any.run, «Any.run,» 2021. [En línea]. Available: https://any.run/why-us/.

[47] I. C. o. A. C. Technology, «http://icact.org/,» 14 Noviembre 2018. [En línea]. Available: http://icact.org/upload/2018/0411/20180411_finalpaper.pdf.

[48] Kartone, «Kartone.ninja,» 23 Mayo 2019. [En línea]. Available: https://blog.kartone.ninja/2019/05/23/malware-analysis-a-wannacry-sample-found-in-the-wild/.

[49] wireshark, «wireshark,» 21 Abril 2021. [En línea]. Available: https://www.wireshark.org/.

[50] Anonimo, «Wikipedia,» 13 Mayo 2021. [En línea]. Available: https://en.wikipedia.org/wiki/Process_Monitor.

[51] Microsoft, «Microsoft,» 5 Diciembre 2018. [En línea]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-getcomputernamew.

[52] Microsoft, «Microsoft,» 2 Abril 2020. [En línea]. Available: https://docs.microsoft.com/es-es/cpp/c-runtime-library/reference/strlen-wcslen-mbslen-mbslen-l-mbstrlen-mbstrlen-l?view=msvc-160.

[53] cplusplus, «cplusplus.com,» [En línea]. Available: https://www.cplusplus.com/reference/cstdlib/srand/.

[54] P. security, «Panda security,» 22 Mayo 2017. [En línea]. Available: https://www.pandasecurity.com/es/mediacenter/src/uploads/2017/05/1705-Informe_WannaCry-v160-es.pdf.

[55] M. docs, «Microsoft docs,» 16 Octubre 2017. [En línea]. Available: https://docs.microsoft.com/es-es/windows-server/administration/windows-commands/attrib.

[56] M. docs, «Microsoft docs,» 21 Agosto 2018. [En línea]. Available: https://docs.microsoft.com/es-es/windows-server/administration/windows-commands/icacls.

[57] A. Kindig, «techlandia,» 2020. [En línea]. Available: https://techlandia.com/kernel32dll-sobre_70060/.

[58] file.net, «file,» [En línea]. Available: https://www.file.net/process/advapi32.dll.html#:~:text=and%20security%20calls.-,Advapi32.,including%20security%20and%20registry%20calls..

[59] G. F. Navarrete, «Linkedit,» 14 Enero 2019. [En línea]. Available: https://www.linkedin.com/pulse/reversing-wannacry-gerardo-fern%C3%A1ndez-navarrete/?originalSubdomain=es.

[60] CNI, «CNI-CERT,» Abril 2017. [En línea]. Available: https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/2169-ccn-cert-id-17-17-codigo-danino-wannacry-1/file.html.

[61] Anonimo, «Wikipedia,» 2021 Febrero 10. [En línea]. Available: https://es.wikipedia.org/wiki/Malware..

[62] P. &. S. T. Rogaway, «Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In International workshop on fast software encryption,» Febrero 2014. [En línea]. Available: https://link.springer.com/chapter/10.1007/978-3-540-25937-4_24.