Sociedad Andaluza para el Desarrollo de las Telecomunicaciones
CONSEJERÍA DE EMPLEO, EMPRESA Y COMERCIO

seguridad+ Y CONFIANZA DIGITAL | AndalucíaCERT CENTRO DE SEGURIDAD TIC

Informe de divulgación

Laboratorio para el análsis de malware (III): Análisis manual - Análisis dinámico

Tipo de documento:	*Informe*
Autor del documento:	*AndalucíaCERT*
Código del Documento:	*CERT-IF-8176-150723*
Edición:	*0*
Categoría	*Público*
Fecha de elaboración:	*23/07/2015*
N° de Páginas	*1 de 15*

1 TABLA DE CONTENIDOS

2 OBJETIVO

Éste es el tercero de una serie de documentos que describen la implementación de un laboratorio para el análisis de malware. Este laboratorio permitirá estudiar una muestra en un entorno controlado, y que cuente con las herramientas para obtener gran cantidad de información. Se hablará principalmente del proceso de análisis dinámico de malware, usando diversas herramientas de análisis, monitorización y depuración.

3 ALCANCE

Este documento va dirigido tanto al personal de la Junta de Andalucía, como al público en general. Pretende aportar las nociones necesarias para entender y tener conocimiento sobre las técnicas que se suelen usar para combatir las amenazas de malware.

En el enfoque de este análisis se describen algunas de las herramientas que se pueden usar en plataformas Microsoft Windows y GNU/Linux para realizar análisis dinámico, y con las herramientas de monitorización y depuración, obtener información de las características del fichero y el comportamiento del malware.

4 ANÁLISIS DINÁMICO

El análisis dinámico consiste en monitorizar la actividad de un malware después de ejecutarlo en una máquina. Se puede observar las funcionalidades de un software malicioso comprobando los ficheros o claves de registro que crea o modifica, la actividad de red que genera o estudiando la información que se almacena en memoria. Este tipo de análisis debería realizarse siempre después del análisis estático, cuando se tenga alguna idea de las características de la pieza, y en un entorno controlado, ya que puede poner en riesgo el sistema o la red.

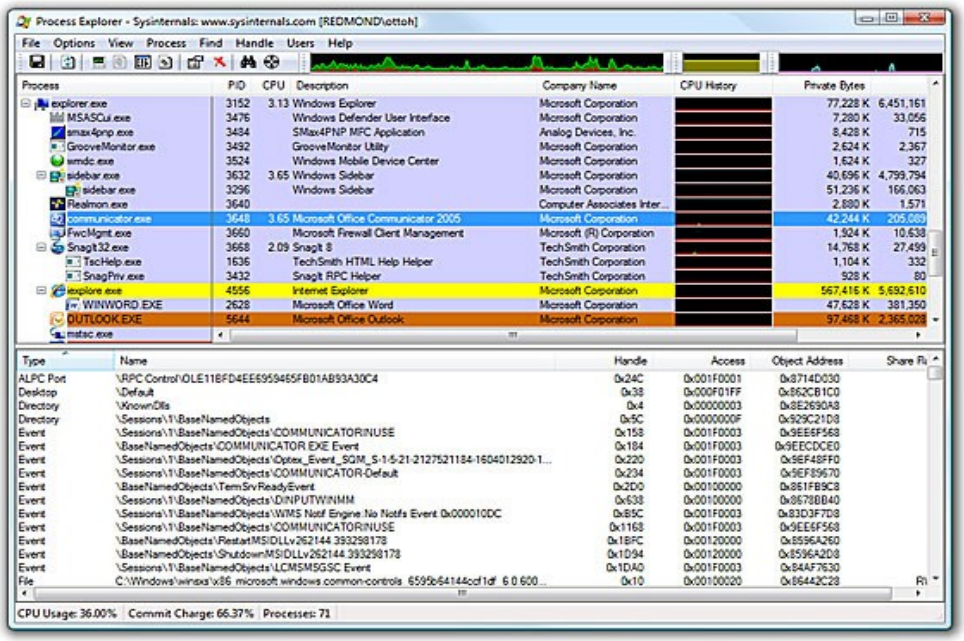
Este tipo de análisis consiste en la creación de una *sandbox* donde ejecutar la muestra de malware para ver su comportamiento. Esto es, un entorno controlado en el que el malware podrá mostrar su verdadera cara, sin que afecte realmente a algún sistema real.

4.1 PROCESS EXPLORER

Process Explorer es una herramienta gratuita de Microsoft para sistemas Windows que proporciona información sobre los procesos que se están ejecutando en el sistema. Se puede usar Process Explorer para listar procesos activos, DLLs cargadas por un proceso, ver las propiedades de un proceso, etc. También se puede usar para finalizar la ejecución de los procesos, sacar de la sesión a usuarios logados en el sistema y validar los procesos que se están ejecutando.

Se puede descargar de la siguiente URL:

- http://technet.microsoft.com/es-es/sysinternals/bb896653.aspx

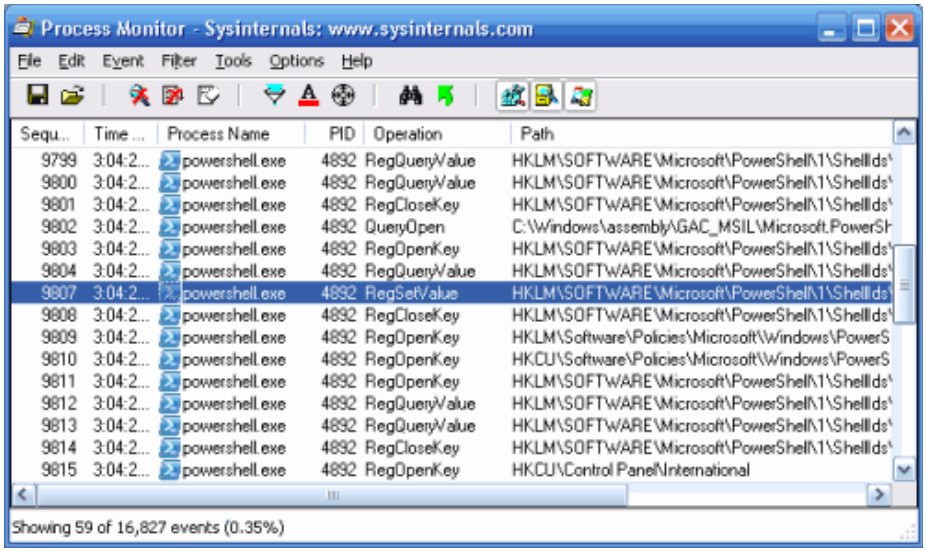


4.2 PROCESS MONITOR

Process Monitor es una herramienta avanzada para Windows que permite monitorizar en tiempo real el sistema de ficheros, el registro y la actividad de procesos e hilos en ejecución. Combina las características de dos antiguas herramientas, Filemon y Regmon, y añade algunas características más.

La herramienta está disponible en la siguiente URL:

- http://technet.microsoft.com/es-es/sysinternals/bb896645.aspx



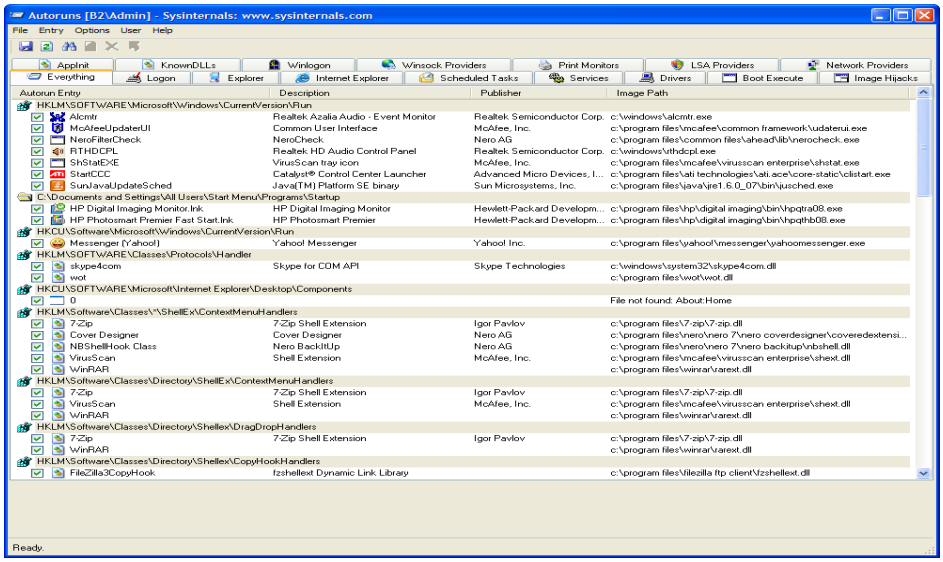
4.3 AUTORUNS

Autoruns es un programa perteneciente a la suite Sysinternals que Microsoft ofrece de forma gratuita para sistemas Windows. Este software permite obtener un listado completo y detallado de todos los procesos que se inician de forma automática en el sistema: aquéllos que se ejecutan de forma autónoma al iniciar el sistema, o al iniciar sesión; los programas de la carpeta Inicio, ...

También muestra el orden en que el Windows ejecuta las entradas de sistema, así como las claves de tipo Run, RunOnce y otras claves del registro similares.

Se puede descargar desde la siguiente URL:

- https://download.sysinternals.com/files/Autoruns.zip



4.4 REGSHOT

Regshot es una herramienta Open Source que permite tomar una copia del registro de un sistema operativo Windows y compararla con otra copia realizada previamente. Se suele usar en el análisis de malware para comprobar las modificaciones realizadas en el registro por una muestra de malware al ejecutarla en el sistema.

Se puede descargar de la página del proyecto:

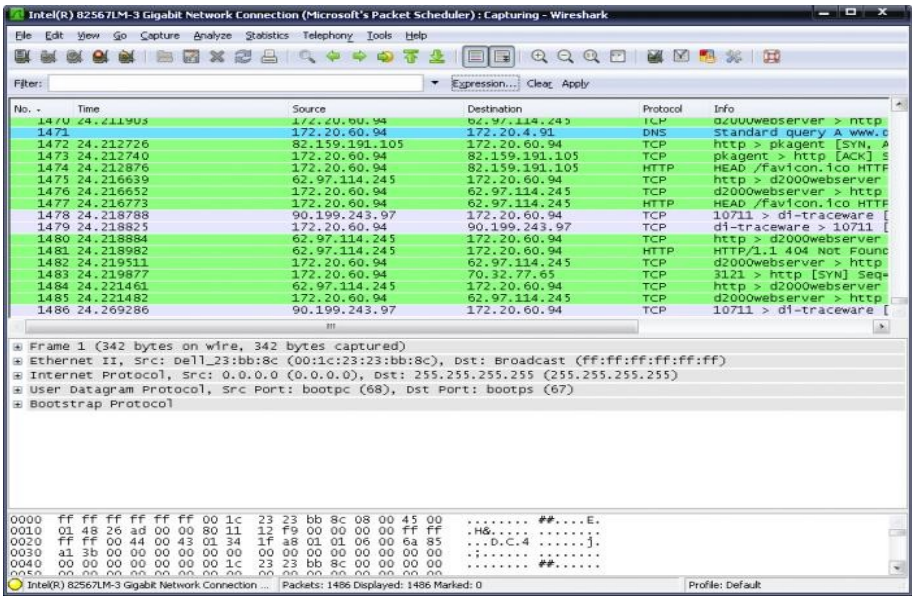
- https://code.google.com/p/regshot/

Para comparar dos copias del registro simplemente habría que:

1. Tomar una copia del registro pulsando en el botón "1st shot".
2. Ejecutar la muestra de malware en el sistema.
3. Lanzar una segunda copia pulsando "2nd shot".
4. Por último realizar una comparación de las dos copias pulsando en "Compare".

4.5 WIRESHARK

Wireshark es un analizador de protocolos de red Open Source. Captura paquetes de red y permite visualizar los flujos de paquetes e inspeccionar el contenido de éstos de forma individual. Wireshark puede ayudar a entender las comunicaciones que está realizando un malware.

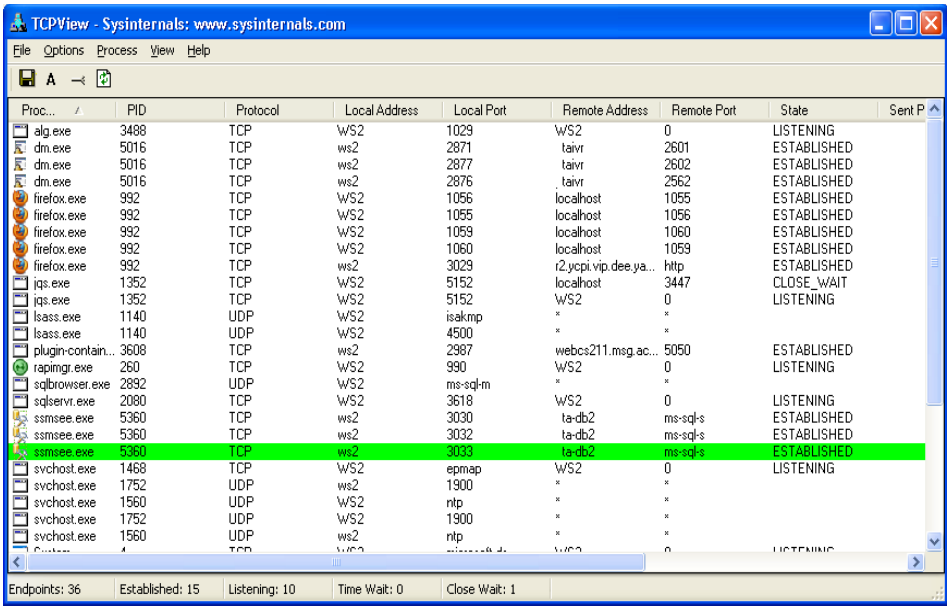


Se puede descargar y encontrar más información en la web del proyecto:

- http://www.wireshark.org/

4.6 TCPVIEW

TCPView es una utilidad de monitorización de red que permite representar de forma gráfica todas las conexiones TCP y UDP presentes en un sistema Windows.



Este software también pertenece a la suite Sysinternals de Microsoft y se puede descargar desde la siguiente URL:

- https://download.sysinternals.com/files/TCPView.zip

4.7 VOLATILITY FRAMEWORK

Volatility Framework es un conjunto de herramientas implementadas en python para la extracción de información de volcados de memoria RAM. La ventaja de este método de análisis es que es independiente de programas tipo rootkit alojados en el sistema, y que puedan estar falseando la información que devuelven las llamadas al sistema operativo. Con esta herramienta podemos acceder a la información directamente de la memoria, sin consultar al sistema operativo. Con Volatility se puede extraer la siguiente información de memoria:

- Identificación del sistema.
- Listado de procesos en ejecución.
- Puertos abiertos.
- Puertos conectados.
- DLLs cargadas por proceso.
- Claves de registro utilizadas por proceso.
- Módulos del kernel.

- Direccionamiento de memoria por proceso.
- Extracción de ejecutables.
- Historial de comandos.
- Historial de navegación.
- Y mucha más información.

Se puede descargar y encontrar más información de la página del proyecto:

- https://code.google.com/p/volatility/

Por ejemplo, podemos comprobar las conexiones de red que han sido terminadas:

```
$ vol.py -f sality.vmem connscan
Volatility Foundation Volatility Framework 2.3.1
Offset(P)  Local Address             Remote Address            Pid
---------- ------------------------ ------------------------- ---
0x02214988 172.16.176.143:1034      131.107.115.254:80        1260
0x06015ab0 172.16.176.143:1037      131.107.115.254:443       1260
```

O el listado de procesos en ejecución en el momento de realizar el volcado de la RAM:

```
$ vol.py -f sality.vmem pslist
Volatility Foundation Volatility Framework 2.3.1
Offset(V)  Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start
Exit
---------- -------------------- ------ ------ ------ -------- ------ ------ ------------------------------
------------------------------
0x810b1660 System                    4      0     58      190 ------      0
0xff2ab020 smss.exe                544      4      3       21 ------      0 2010-08-11 06:06:21 UTC+0000
0xff1ecda0 csrss.exe               608    544     11      434      0      0 2010-08-11 06:06:23 UTC+0000
0xff1ec978 winlogon.exe            632    544     19      513      0      0 2010-08-11 06:06:23 UTC+0000
0xff247020 services.exe            676    632     16      269      0      0 2010-08-11 06:06:24 UTC+0000
0xff255020 lsass.exe               688    632     21      349      0      0 2010-08-11 06:06:24 UTC+0000
0xff218230 vmacthlp.exe            844    676      1       24      0      0 2010-08-11 06:06:24 UTC+0000
0x80ff88d8 svchost.exe             856    676     18      203      0      0 2010-08-11 06:06:24 UTC+0000
0xff217560 svchost.exe             936    676     11      268      0      0 2010-08-11 06:06:24 UTC+0000
0x80fbf910 svchost.exe            1028    676     88     1426      0      0 2010-08-11 06:06:24 UTC+0000
0xff22d558 svchost.exe            1088    676      6       80      0      0 2010-08-11 06:06:25 UTC+0000
0xff203b80 svchost.exe            1148    676     15      212      0      0 2010-08-11 06:06:26 UTC+0000
0xff1d7da0 spoolsv.exe            1432    676     13      135      0      0 2010-08-11 06:06:26 UTC+0000
0xff1b8b28 vmtoolsd.exe           1668    676      5      221      0      0 2010-08-11 06:06:35 UTC+0000
0xff1fdc88 VMUpgradeHelper        1788    676      5      102      0      0 2010-08-11 06:06:38 UTC+0000
0xff143b28 TPAutoConnSvc.e        1968    676      5      100      0      0 2010-08-11 06:06:39 UTC+0000
0xff25a7e0 alg.exe                 216    676      7      108      0      0 2010-08-11 06:06:39 UTC+0000
0xff364310 wscntfy.exe             888   1028      4       32      0      0 2010-08-11 06:06:49 UTC+0000
0xff38b5f8 TPAutoConnect.e        1084   1968      4       66      0      0 2010-08-11 06:06:52 UTC+0000
0x80f60da0 wuauclt.exe            1732   1028      7      178      0      0 2010-08-11 06:07:44 UTC+0000
0xff3865d0 explorer.exe           1724   1708     18      414      0      0 2010-08-11 06:09:29 UTC+0000
0xff3667e8 VMwareTray.exe          432   1724      4       53      0      0 2010-08-11 06:09:31 UTC+0000
0xff374980 VMwareUser.exe          452   1724     11      208      0      0 2010-08-11 06:09:32 UTC+0000
0x80f94588 wuauclt.exe             468   1028      7      139      0      0 2010-08-11 06:09:37 UTC+0000
0xff22f3d0 aelas.exe              1984   1724     19      139      0      0 2010-08-15 17:43:26 UTC+0000
0x80f167b8 cmd.exe                1368   1668      0 --------      0      0 2010-08-15 17:43:45 UTC+0000
2010-08-15 17:43:45 UTC+0000
```

En el siguiente enlace hay una referencia completa de los comando que se pueden usar:

- https://code.google.com/p/volatility/wiki/CommandReference21

4.8 DEPURADORES

Un depurador es una herramienta que se usa para testear o examinar la ejecución de otro programa. Son muy utilizados para encontrar errores durante el proceso de desarrollo de software. Permite a los desarrolladores controlar el estado interno y ejecución de un programa.

Los depuradores proporcionan información que difícilmente se puede obtener con un desensamblador. Los desensambladores ofrecen una foto de un programa inmediatamente antes de ejecutarse. Los depuradores proporcionan una vista dinámica de un programa en ejecución. Pueden mostrar las variables almacenadas en memoria y cómo cambian a lo largo de la ejecución del programa.

Hay dos maneras de depurar un programa. La primera es arrancar el programa con el depurador. Cuando se arranque el programa se cargará en memoria, se parará inmediatamente antes de la ejecución de su punto de entrada. En este punto se tendrá un control completo del programa.

También se puede enlazar un depurador a un programa que ya se esté ejecutando. Todos los threads serán parados y se podrá depurar. Es un buen enfoque cuando se quiere depurar un programa que ya se ha arrancado o para depurar un proceso que ha sido afectado por un malware.

Algunos de los depuradores más utilizados son los siguientes:

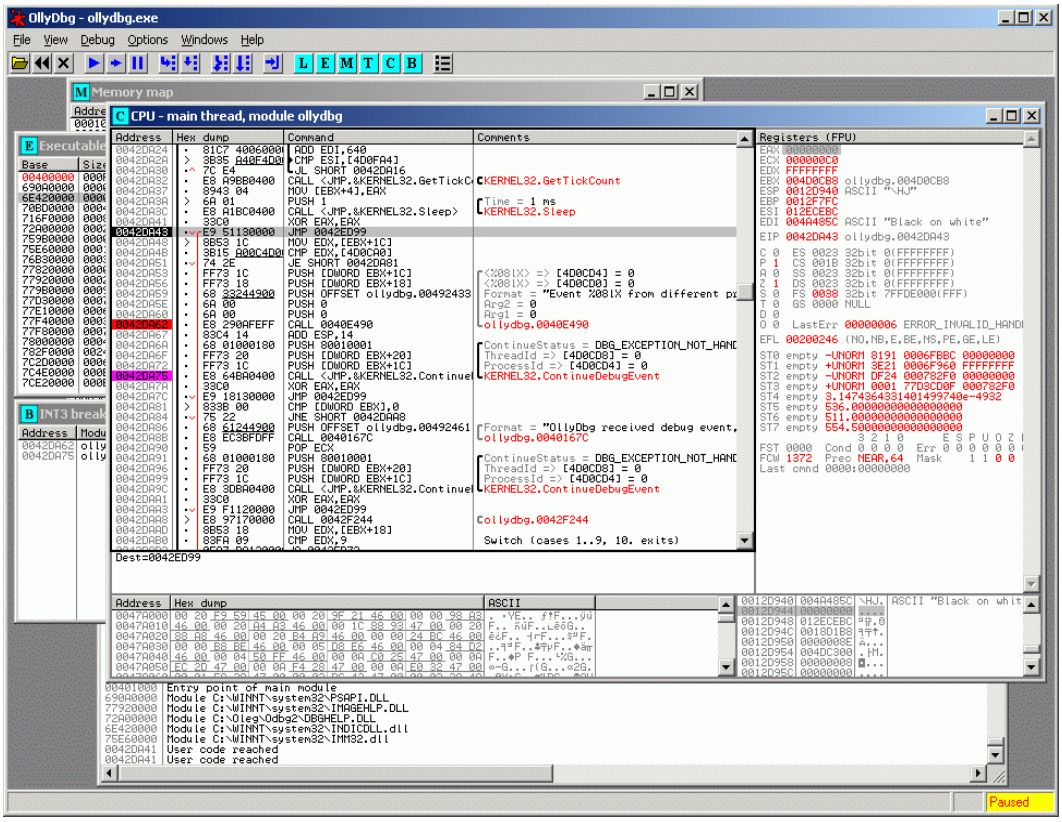
- OllyDbg
- Immunity Debugger
- WinDbg
- gdb
- Intel Debugger
- LLDBG

4.8.1 OllyDbg

OllyDbg es un depurador a nivel de aplicación. La interfaz OllyDbg muestra el código ensamblador, volcado hexadecimal, la pila y registros de la CPU. También soporta rastreo, puntos de interrupción condicionales, visión de cabecera PE, edición hexadecimal, y plug-in de soporte.

En la primera puesta en marcha, OllyDbg pide configurar el directorio de datos del usuario (UDD) y el directorio de plug-ins. UDD se utiliza para guardar información específica de la aplicación como puntos de interrupción. Ofrece amplias opciones de depuración como la configuración de *breakpoints* en la carga de nuevos módulos, la creación de *threads*, la forma de procesar las excepciones, etc.

También soporta el establecimiento de puntos de interrupción de hardware, puntos de interrupción de software, puntos de interrupción de memoria e incluso puntos de interrupción condicionales.

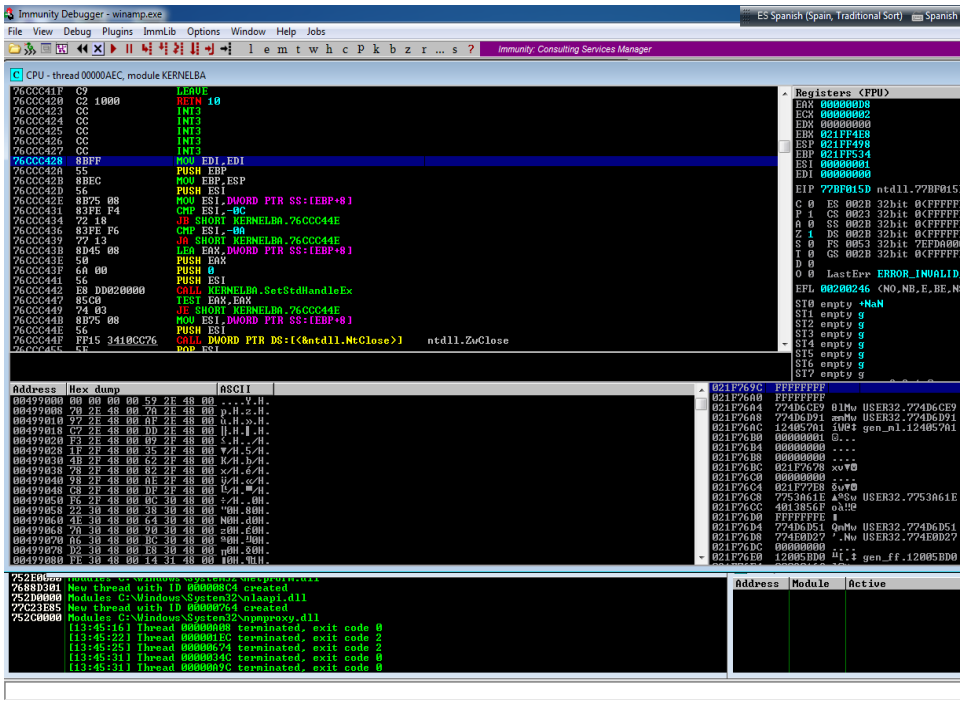


4.8.2 Immunity Debugger

Se trata de un depurador basado en el clásico OllyDBG. Esta compuesto por varias herramientas que realizan técnicas de ingeniería inversa con archivos binarios, pero sobre todo hay que destacar la herramienta llamada mona.py. Tiene una interfaz gráfica con herramientas de análisis en heap apoyado por una API Python. Mediante el uso de scritp's automatiza la depuración de manera más inteligente, rápida y fácil para la explotación de herramientas de desarrollo.

Es una aplicación distribuida en la web de Microsoft. Se puede utilizar para depurar en modo de usuario las aplicaciones, los controladores y el sistema operativo en modo kernel.

WinDbg se puede utilizar para la depuración de volcados de memoria y también tiene la capacidad de cargar de forma automática lo que se llama 'mapa de símbolos' traduciendo fechas, horas, CRCs haciendo coincidir diversos criterios. Otra de las ventajas es que pueden relacionarse con el código fuente del binario. La aplicación dispone de dos formas de depurar un proceso; en local y en remoto.



Los módulos son partes del programa (en depuración) que se cargan con la ejecución del programa original. Muchas veces es mejor utilizar o depurar un módulo para saber cómo funciona el programa.

4.8.3 Gdb

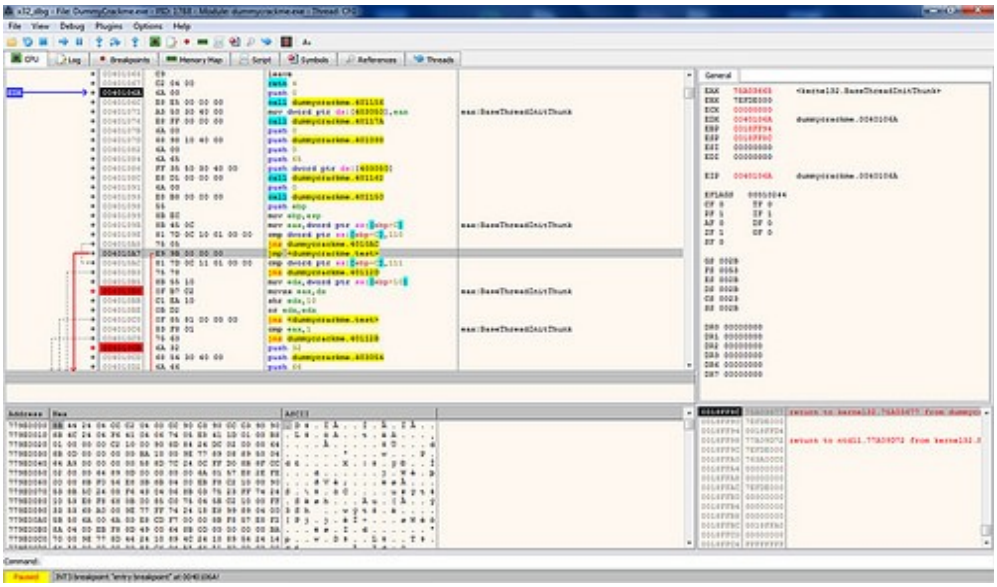
Es un poderoso depurador que permite "ver" qué esta sucediendo dentro de programas escritos en diversos lenguajes de programación, como C, C++, Fortran, ...

Entre las capacidades más notorias que este debugger posee están:

- Debugging de programas complejos con múltiples archivos.
- Capacidad para detener el programa o ejecutar un comando en un punto especifico (breakpoints), según una condición (watchpoints) o al llegar una señal (catchpoints).
- Capacidad para mostrar valores de expresiones cuando el programa se detiene automáticamente (displays).
- Es posible examinar la memoria y/o variables de diversas formas y tipos, incluyendo estructuras, arreglos y objetos.
- Es posible igualmente cambiar los valores de las variables para estudiar el comportamiento del programa sin necesidad de recompilar.
- Posibilidad de realizar debugging a programas en ejecución (procesos).
- Posibilidad de realizar debugging a programas que han finalizado.

4.8.4 Intel Debugger

Es un depurador para arquitecturas Intel de 32 y 64 bits para sistemas Windows. La interfaz es muy parecida a la de OllyDbg.



Las características que x64_dbg nos ofrece en este momento son:

- Código abierto.
- Interfaz de usuario intuitiva y familiar.
- Analizador de expresiones C-Like.
- Con todas las funciones de depuración de los archivos DLL y EXE (TitanEngine)
- Mapas de memoria
- Visor de símbolos.
- Visores de secuencias.
- Visores de Contenido de registros
- Reconocimiento dinámico de módulos.
- Reconstructor de importación integrado (Scylla)
- Desensamblador rápido (BeaEngine)
- Base de datos de usuario (JSON) para comentarios , etiquetas, marcadores etc.
- Soporte a plungin's API
- Volcado de memoria de múltiples tipos de datos
- Símbolos de depuración básico de apoyo (AP)
- Ensamblador (XEDParse)
- Visor de parches y copias de seguridad en disco.
- Editor Hexadecimal Built-in
- Búsquedas de patrones en la memoria.

5 CONCLUSIONES

En el presente documento se continúa con la serie iniciada en comunicados anteriores en la que se pretende dar a conocer las técnicas y herramientas que se usan durante el análisis de malware y que permitirán detectar infecciones en equipos y en la red, determinar el impacto que pueda ocasionar una infección y las acciones que se podrán realizar para contenerla y erradicarla.

Tras el análisis estático de una muestra, la siguiente fase para obtener más información del fichero bajo estudio es el análisis dinámico. A lo largo del documento se describen varias herramientas para obtener información del comportamiento que realiza la muestra una vez ejecutada.

La información obtenida de este análisis nos permitirá:

- Crear firmas de detección, tanto locales como de red, que evidencien la presencia de un código malicioso en la organización.
- Desarrollar un procedimiento de desinfección.
- Conocer las acciones realizadas por el malware en cuestión.
- Determinar el impacto que puede ocasionar en la organización.
- Obtener indicadores de compromiso.
- Conocer las técnicas usadas para informar a usuarios en una campaña de concienciación y sensibilización.
- Conocer los vectores de acceso usados.
- Prevenir la ocurrencia de nuevos incidentes de seguridad.

Las técnicas de análisis descritas hasta ahora entran dentro de los análisis manuales. En futuros informes se seguridad se mostrarán técnicas que permitirán analizar grandes cantidades de muestras de malware de forma automática.