

Introducción a la informática forense en entornos Windows (I)

Este documento no pretende ni tiene la finalidad de convertirse en manual de referencia. Este documento relata los aspectos básicos relacionados en el campo de la informática forense, explicando de una forma simple y concisa las herramientas utilizadas para la captura de evidencias. También explicará los aspectos técnicos relativos a la arquitectura de un sistema Windows. Probablemente esto ayudará al lector a comprender mejor cómo un sistema Windows recopila y almacena la información, ayudando también a entender la arquitectura del sistema.

Prohibida la reproducción total o parcial de este texto sin poner la fuente
(<http://www.elhacker.net>)

Prohibida la modificación o eliminación de enlaces e imágenes en este documento.

Redactado por [Silverhack](#) desde el 08 de Agosto de 2006 al 07 de Marzo de 2007

Versión 0.1

General

Definición de análisis forense

Evidencia Digital

RFC3227 (Recolección y manejo de evidencias)

Buenas prácticas a la hora de analizar datos

Entorno Microsoft

Cuentas de usuario y perfil de usuario

Tipos de Logon en un sistema basado en Windows

La Papelera de Reciclaje. Estructura y funcionamiento

Archivos de Registro. Estructura

Index.dat e Internet Explorer. Estructura y funcionamiento

Service Pack, HotFix. Qué es y para qué sirve?

Introducción y definición de análisis forense

En este tutorial, vamos a explicar de la forma más sencilla posible el uso de algunas herramientas que nos pueden facilitar la tarea a la hora de realizar un análisis forense en entornos Windows.

Existen muchísimas herramientas destinadas a éste propósito, comerciales y gratuitas. En este tutorial vamos a ver como enfocaríamos un análisis partiendo de herramientas gratuitas.

Cuando un usuario **no autorizado** toma el control de un sistema, éste puede instalar múltiples backdoors (puertas traseras) que le permitan entrar al sistema en un futuro,

aunque **parcheemos** la vulnerabilidad original.

Se denomina **análisis forense** al proceso de analizar una copia completa de un sistema que ha sufrido una intrusión o ataque.

El **análisis forense** permite obtener la mayor cantidad posible de información sobre:

- **El método utilizado por el atacante para introducirse en el sistema**
- **Las actividades ilícitas realizadas por el intruso en el sistema**
- **El alcance y las implicaciones de dichas actividades**
- **Las “puertas traseras” instaladas por el intruso**

Realizando un **análisis forense** nos permitirá, entre otras cosas, recuperarnos del incidente de una manera más segura y evitaremos en la medida de lo posible que se repita la misma situación en cualquiera de nuestras máquinas.

Un buen análisis forense debe dar respuestas a varias cuestiones, entre las que se encuentran las siguientes:

- **¿En que fecha exacta se ha realizado la intrusión o cambio?**
- **¿Quién realizó la intrusión?**
- **¿Cómo entró en el sistema?**
- **¿Qué daños ha producido en el sistema?**

Si una vez realizado el análisis forense no conocemos con exactitud las respuestas a estas preguntas, no tendremos un análisis funcional. Esto puede derivar en futuros ataques, bien por la misma persona, o bien por diferentes medios de intrusión que desconozcamos.

Evidencia digital

Uno de los pasos a tener en cuenta en toda investigación, sea la que sea, consiste en la captura de la/s evidencia/s. **Por evidencia entendemos toda información que podamos procesar en un análisis.** Por supuesto que el único fin del análisis de la/s evidencia/s es saber con la mayor exactitud qué fue lo que ocurrió.

Bueno, y ¿qué entendemos por evidencia digital? Podemos entender evidencia como:

- El último acceso a un fichero o aplicación (unidad de tiempo)
- Un Log en un fichero
- Una cookie en un disco duro
- El uptime de un sistema (Time to live o tiempo encendido)
- Un fichero en disco
- Un proceso en ejecución
- Archivos temporales
- Restos de instalación
- Un disco duro, pen-drive, etc...

RFC3227. Recolección y manejo de evidencias

El propósito de este documento (RFC3227) no es otro que proveer a los administradores de sistemas unas pautas a seguir en el aspecto de recolección de evidencias, si se diese

el caso de un incidente de seguridad.
En este rfc se trata los siguientes aspectos:

- Principios para la recolección de evidencias
- Orden de volatilidad
- Cosas a evitar
- Consideraciones relativas a la privacidad de los datos
- Consideraciones legales
- Procedimiento de recolección
- Transparencia
- Pasos de la recolección
- Cadena de custodia
- Como archivar una evidencia
- Herramientas necesarias y medios de almacenamiento de éstas

Algunos de los principios que rige el documento para la recolección de evidencias son:

- Comprometer al personal de aplicación de la ley y manejo de incidentes apropiados
- Capturar la imagen tan exacta del sistema como sea posible
- Anotar todo lo que se vaya investigando
- Recolectar las evidencias en función de la volatilidad de la misma. Primero se recogerán las de mayor volatilidad

El orden de volatilidad que recoge el rfc es el siguiente:

- Registros, Cache
- Tabla de ruta. ARP Cache, Tabla de Proceso, Núcleo de estadísticas, memoria
- Sistema de Archivo temporales
- Disco
- Datos de monitoreo y Log's remotos relativos al caso
- Configuración física, topología de red
- Medio de Archivos

Si se quiere leer más sobre el RFC3227 sobre la recolección y manejo de evidencias, puede hacerlo en el siguiente enlace:

<http://rfc.net/rfc3227.html>

Buenas prácticas a la hora de la recogida y análisis de los datos

Estudio Preliminar

Es el primer paso de cualquier análisis forense. Nos deben o debemos explicar con la mayor exactitud posible qué ha ocurrido, qué se llevaron o intentaron llevar y cuándo ocurrió.

También tendremos que recoger información sobre la organización, ya sea organización, casa, etc...

Recogeremos información sobre la tipología de red y de gente directa o indirectamente implicada.

También podríamos recoger información sobre el tipo de escenario y el/los sistema/s afectado/s.

¿Apagamos el equipo?

Podemos presentarnos con dos casos. El primero es el de no apagar el equipo. Si no apagamos el equipo, podremos ver todos los procesos en ejecución, los consumos de memoria, las conexiones de red, los puertos abiertos, los servicios que corren en el sistema, etc.

También se nos presenta el problema de que si apagamos el equipo, se perderá información volátil que puede ser esencial para el curso de la investigación.

La parte mala de esta situación es que el sistema, al poder estar contaminado, éste puede ocultar la información. También se nos presenta el problema de que si no apagamos el sistema, éste puede comprometer a toda la red.

Si no apagamos el sistema tendremos que controlar este aspecto de la seguridad, y aislarlo completamente de la red, lo cual llega a ser prácticamente imposible en determinados escenarios.

Tipo de Herramientas

Una de las cosas más importantes a la hora de realizar un análisis forense es la de no alterar el escenario a analizar. Esta es una tarea prácticamente imposible, porque como mínimo, alteraremos la memoria del sistema al utilizar cualquier herramienta.

Las herramientas que utilicemos deben de ser lo menos intrusivas en el sistema, de ahí que se huya de las herramientas gráficas, las que requieren instalación, las que escriben en el registro, etc...

Lo normal y lógico sería utilizar herramientas ajenas al sistema comprometido, ya sean herramientas guardadas en cualquier soporte (CD-ROM, USB, etc...). Esto lo hacemos para no tener que utilizar las herramientas del sistema, ya que pueden estar manipuladas y arrojar falsos positivos, lecturas erróneas, etc...

Tipo de Copia del Sistema

En el caso de que se pueda realizar, lo ideal sería hacer más de una copia de seguridad. Una de ellas se podría guardar herméticamente junto con algún sistema de fechado digital como el proporcionado por RedIris <http://rediris.es/app/sellado>. Otra copia la podría guardar algún responsable de la compañía afectada, y una copia se destinaría a trabajar sobre ella.

En el caso que sea posible, la imagen obtenida podremos montarla sobre un hardware similar al de la máquina afectada.

Destacaremos los siguientes aspectos:

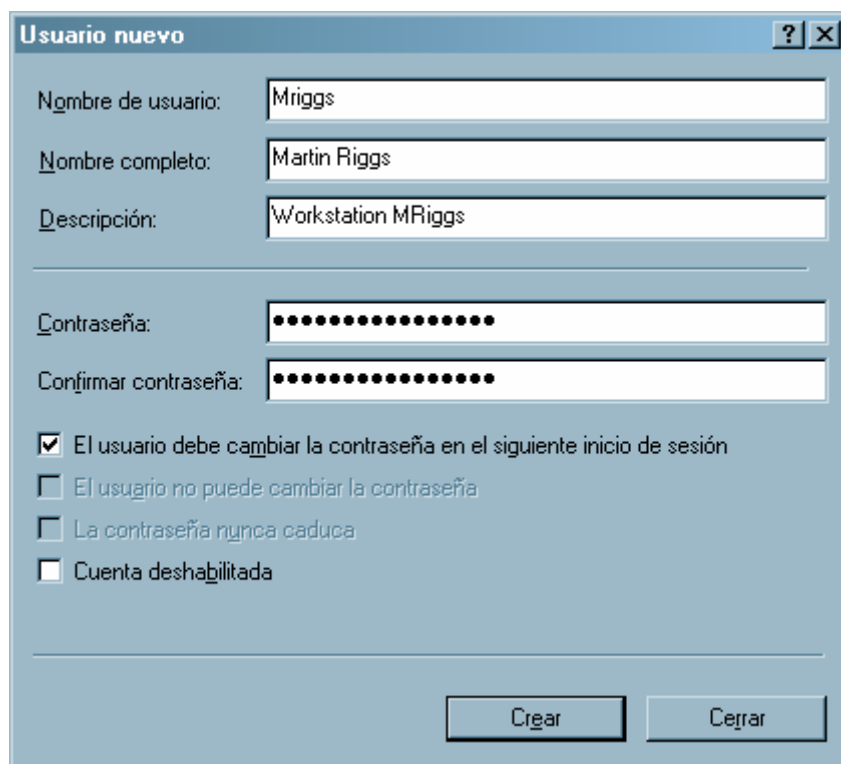
- La copia que realicemos debería ser lo más exacta posible
- Si es posible, haremos varias copias de seguridad
- Una de ellas se guardará herméticamente, para aislarla de todo tipo de agente exterior
- A ser posible se fecharán digitalmente y sobre el papel
- En el caso que sea posible, la imagen obtenida la montaremos sobre hardware similar

Cuentas de usuario y perfiles de usuario

Dejando a un lado si se accede legítima o ilegítimamente, un usuario no es más que cualquier persona que pueda acceder al sistema.

En una cuenta de usuario almacenaremos información acerca del usuario. Algunos datos que se guardan son:

- Nombre de usuario: Nombre con el que nos identificaremos en el sistema
- Nombre completo: Nombre completo del usuario (Siempre que se rellene)
- Contraseña: Palabra cifrada para autenticarnos en el sistema
- SID: Código de identificación de seguridad *
- Directorio: Es el lugar donde en un principio se guardará toda información relevante al usuario.



Usuario nuevo

Nombre de usuario: Mriggs

Nombre completo: Martin Riggs

Descripción: Workstation MRiggs

Contraseña:

Confirmar contraseña:

El usuario debe cambiar la contraseña en el siguiente inicio de sesión

El usuario no puede cambiar la contraseña

La contraseña nunca caduca

Cuenta deshabilitada

Crear Cerrar

***Nota.- A diferencia de los demás, este es el único dato que no podemos especificar manualmente**

El perfil de usuario contiene las preferencias y las opciones de configuración de cada usuario. En la tabla siguiente se puede ver un ejemplo de la configuración que contienen los perfiles de usuario.

Fuente	Parámetros guardados
Explorador de Windows	Todos los valores definibles por el usuario en el Explorador de Windows.
Mis documentos	Documentos almacenados por el usuario.
Mis imágenes	Imágenes almacenadas por el usuario.
Favoritos	Accesos directos a las ubicaciones favoritas de Internet.
Unidad de red asignada	Asignaciones de unidades de red creadas por el usuario.
Mis sitios de red	Vínculos a otros equipos de la red.
Contenido del escritorio	Elementos almacenados en el Escritorio y en los accesos directos.
Colores y fuentes de pantalla	Toda la configuración de colores y textos presentables en pantalla y definibles por el usuario.
Datos de aplicación y sección del Registro	Datos de aplicación y configuraciones definidas por el usuario.
Configuración de impresoras	Conexiones de impresoras de red.
Panel de control	Todas las configuraciones definidas por el usuario en el Panel de control.
Accesorios	Todas las configuraciones de aplicación definidas por el usuario que afectan al entorno de usuario de Windows, incluidos Calculadora, Reloj, Bloc de notas y Paint.
Programas de instalación de la familia Windows Server 2003	Cualquier programa escrito específicamente para la familia Windows Server 2003 se puede diseñar para que haga un seguimiento de las configuraciones propias de cada usuario. Si dicha información existe, se guarda en el perfil de usuario.
Marcadores de formación en pantalla para el usuario	Los marcadores del sistema de Ayuda de la familia Windows Server 2003.

En Windows 2003 Server, los perfiles de cada usuario se almacenan en el directorio Documents and Settings de la raíz. Si nuestro equipo estuviese montado en la unidad C:\, el directorio de los perfiles se encontrará en el directorio siguiente:

C:\Documents and Settings\usuario

Tipos de Logon en un sistema basado en Windows

Los sucesos de inicio de sesión en un sistema Windows se generan en los controladores de dominio para la actividad de cuentas de dominio y en los equipos locales para la actividad de cuentas locales. Si están habilitadas ambas categorías de directiva, los inicios de sesión que utilizan una cuenta de dominio generan un suceso de inicio o cierre de sesión en la estación de trabajo o servidor, y generan un suceso de inicio de

sesión de cuenta en el controlador de dominio.

La categoría de inicio de sesión en Windows registrará la entrada con un evento ID 528 que contendrá una serie de datos importantes, como son el tipo de entrada y el ID de inicio de sesión.

Dependiendo del inicio de sesión que hagamos en la máquina, ya sea a través de recursos compartidos, de forma remota o de forma física, Windows registrará ese inicio de sesión con una numeración u otra.

Algunos tipos de inicio de sesión son:

Tipo 2. Interactivo. Entrada a un sistema desde la consola (teclado)

Tipo 3. Red. Entrada al sistema a través de la red. Por ejemplo con el comando net use, recursos compartidos, impresoras, etc...

Tipo 4. Batch. Entrada a la red desde un proceso por lotes o script programado.

Tipo 5. Servicio. Cuando un servicio arranca con su cuenta de usuario.

Tipo 7. Unlock. Entrada al sistema a través de un bloqueo de sesión.

Tipo 10. Remote Interactive. Cuando accedemos a través de Terminal Services, Escritorio Remoto o Asistencia Remota.

La Papelera de Reciclaje. Estructura y funcionamiento

Al contrario de lo que se piensa mucha gente, cuando un archivo se borra de una computadora, realmente no se borra. Los archivos se modifican por decirlo de alguna manera, para que el sistema operativo no los vea. Windows utiliza un almacén para los archivos eliminados llamado Papelera de Reciclaje. La existencia de este almacén permite que un usuario pueda recuperar la información, si ésta ha sido borrada accidentalmente por ejemplo. Cuando Windows da orden de eliminar cierto archivo o directorio, la información se guarda en expedientes, por si el usuario se arrepiente y quiere recuperar sus datos. El archivo que contiene esta información se llama INFO2 y reside en el directorio de la Papelera de Reciclaje, es decir, está dentro de la Papelera. Es necesario explicar cómo funciona la Papelera de Reciclaje antes de que discutamos las estructuras del archivo INFO2. Cuando un usuario suprime un archivo a través del explorador de Windows, una copia del archivo se mueve al almacén de la Papelera de Reciclaje. La localización de este directorio es distinta, dependiendo de la versión de Windows que tengamos. En versiones NT/XP/2003, el archivo INFO2 se encuentra en el siguiente directorio:

C:\Recycler\<<USER SID>\INFO2

Cuando eliminamos un fichero, Windows lo renombra siguiendo este parámetro:

D <Unidad raíz del sistema> <número> .Extensión del archivo

Es decir, que si nosotros quisiésemos eliminar el archivo **Contabilidad.doc** y lo mandásemos a la Papelera de Reciclaje, Windows lo renombraría de la siguiente manera:

DC1.Doc

Si borrásemos otro archivo, a éste nuevo archivo se le pondría el número 2, y así sucesivamente.

```

C:\WINDOWS\system32\cmd.exe
C:\RECYCLER\S-1-5-21-1482476501-1532298954-1801674531-1005>dir /a
Volume in drive C has no label.
Volume Serial Number is 3452-DE76

Directory of C:\RECYCLER\S-1-5-21-1482476501-1532298954-1801674531-1005

04/11/2003  11:35 AM    <DIR>          -
04/11/2003  11:35 AM    <DIR>          -
04/08/2003  06:39 PM             1,926  De1.lnk
04/08/2003  06:39 PM             1,952  De2.lnk
04/08/2003  05:13 PM              779  De3.lnk
04/11/2003  11:17 AM          1,897,672  De4.exe
04/11/2003  11:32 AM          125,173,760  De5
04/11/2003  11:07 AM    <DIR>          De6
04/11/2003  11:33 AM          600,000,000  De7.sa
04/08/2003  06:40 PM              65  desktop.ini
04/11/2003  11:35 AM              5,620  INFO2
      8 File(s)          727,081,774 bytes
      3 Dir(s)          11,881,926,656 bytes free

C:\RECYCLER\S-1-5-21-1482476501-1532298954-1801674531-1005>_

```

Si al menos un archivo se ha movido a la Papelera de Reciclaje, el archivo INFO2 existirá. Cuando se vacía la Papelera de Reciclaje, el contenido del archivo INFO2 se limpiará, y el número se establecerá de nuevo a 1. Es decir, el archivo INFO2 se suprime y se crea un nuevo y vacío INFO2.

Archivos de Registro de Windows. Estructura

Windows define al registro como una base de datos jerárquica central utilizada en todas las versiones de Windows, con el fin de almacenar información necesaria para configurar el sistema para uno o varios usuarios, aplicaciones y dispositivos hardware. El registro contiene información que Windows utiliza como referencia constantemente, como por ejemplo los perfiles de usuario, las aplicaciones instaladas, los parches o HotFixes instalados, etc... Los archivos del registro de Windows se almacenan en archivos binarios, es decir, que si abrimos estos ficheros con un editor de texto, como puede ser notepad, no podremos leerlo.

El registro se puede manipular desde muchos medios, tanto en línea de comandos como por la propia interfaz gráfica de Windows. Evidentemente la forma más fácil de manipular el registro es de forma gráfica. Sólo tendríamos que ejecutar la herramienta regedit.

El Registro está organizado en una estructura jerárquica compuesta por subárboles con sus respectivas claves, subclaves y entradas.

Las claves pueden contener subclaves y éstas, a su vez, pueden contener otras subclaves. Generalmente, la mayor parte de la información del Registro se almacena en disco y se considera permanente, aunque en determinadas circunstancias hay datos que se almacenan en claves llamadas volátiles, las cuales se sobrescriben cada vez que se inicia el sistema operativo.

Toda información relativa al sistema operativo y al PC se encuentra recogida en los archivos del sistema del registro de Windows, los cuales se localizan en %systemroot%\system32\config, y atienden a los nombres siguientes:

- SECURITY
- SOFTWARE
- SYSTEM
- SAM

- DEFAULT

Cada sección del Registro está asociada a un conjunto de archivos estándar. En la tabla siguiente se muestran las secciones y archivos asociados a estas secciones:

Sección del Registro	Nombres de archivo
HKEY_LOCAL_MACHINE\SAM	Sam y Sam.log
HKEY_LOCAL_MACHINE\SECURITY	Security y Security.log
HKEY_LOCAL_MACHINE\SOFTWARE	Software y Software.log
HKEY_LOCAL_MACHINE\SYSTEM	System y System.log
HKEY_CURRENT_CONFIG	System y System.log
HKEY_CURRENT_USER	Ntuser.dat y Ntuser.dat.log
HKEY_USERS\DEFAULT	Default y Default.log

Index.dat e Internet Explorer. Estructura y funcionamiento

Internet Explorer es el navegador por excelencia de Microsoft. A partir de su versión XP, este navegador viene integrado en el sistema operativo, es decir, que no se puede desinstalar. Internet Explorer guarda una copia de las páginas visitadas en el disco duro. Si vas a una página ya visitada, Internet Explorer busca primero en la caché, y la compara con la página del servidor, mostrándote la página desde tu disco duro, si no ha habido actualizaciones. Con esto conseguimos una carga mucho más rápida de las páginas Web, o como dirían los expertos, **Una mejor experiencia para el usuario final**. Podemos borrar el caché de disco desde el propio Internet Explorer (herramientas, opciones de Internet, eliminar archivos). El problema es que esta opción borra todo el contenido del historial de Internet (los archivos html, los gráficos, etc.) pero no borra el índice de referencia que Internet Explorer usa para buscar dentro de su historial: el archivo index.dat. Estos archivos (hay varios index.dat) están definidos como ocultos y de sistema; por eso no podemos acceder a su contenido desde el propio Windows, a no ser que quitemos el atributo de ocultos a esos directorios. En ellos se guarda una lista de todos los sitios Web que hemos ido visitando (aunque hayamos borrado el historial, esta lista no está sincronizada, luego no borra esas Urls). Esto supone un problema de privacidad, ya que cualquiera que sepa localizar y leer estos archivos index.dat tendrá un listado completo de los sitios que hayamos visitado (aunque hayamos borrado el historial del navegador). Además este archivo está creciendo constantemente, y puede llegar a ocupar varios megas de la forma más innecesaria. Aparte, si por cualquier razón su contenido se corrompe, puede ocasionar que Internet Explorer no pueda visualizar correctamente algunas páginas o no pueda descargar ficheros. La ruta en donde se encuentran estos archivos (index.dat) es la siguiente:

```
Windows 2K/XP  \Documents and Settings\<<username>\Local Settings\Temporary
                Internet Files\Content.IE5\
                \Documents and Settings\<<username>\Cookies\
                \Document and Settings\<<username>\Local
                Settings\History\History.IE5\
```

Service Pack, HotFix, ¿Qué es y para qué sirve?

Un Service Pack mantiene la versión de Windows y/o aplicaciones actualizados, corrigen problemas conocidos así como ampliar funcionalidad al equipo. En un Service Pack se incluyen drivers o controladores, herramientas y actualizaciones, así como algunas mejoras realizadas después de la puesta al público del producto. Y todo esto incluido en un paquete.

Cada nuevo Service Pack contiene todas las soluciones incluidas en los anteriores, es decir, cada Service Pack es acumulativo. Para mantener actualizado nuestro sistema sólo necesitaremos instalar el último Service Pack para cada producto o versión de Windows, ya que los Service Packs son específicos para cada producto. No se utiliza el mismo Service Pack para actualizar un Windows XP, que para actualizar un Windows 2000, por ejemplo.

Un HotFix básicamente es una revisión de un producto. Estas revisiones se realizan con el fin de subsanar errores específicos para los que no existe una solución viable.

Un HotFix no se somete a pruebas rigurosas, por lo que se recomienda aplicar estas revisiones, si se experimenta el problema exacto.

Cada cierto tiempo, al incorporarse nuevas funcionalidades y actualizaciones en los Service Packs, estos HotFix se someten a comprobaciones más exhaustivas, y se ponen a disposición del público en general.

La rama encargada de programar HotFix en Microsoft se denomina Ingeniería de corrección rápida o QFE (**Quick Fix Engineering**).

Introducción a la informática forense en entornos Windows (II)

General

[Recogida de archivos Log del sistema](#)

[Rebuscando en la Papelera de Reciclaje](#)

[Sacando información del archivo Index.dat](#)

[NtUser.dat y archivos de registro de Windows](#)

[Cookies. Estructura, funcionamiento y metodología de acceso a la información](#)

[El archivo de paginación de Windows. El archivo pagefile.sys](#)

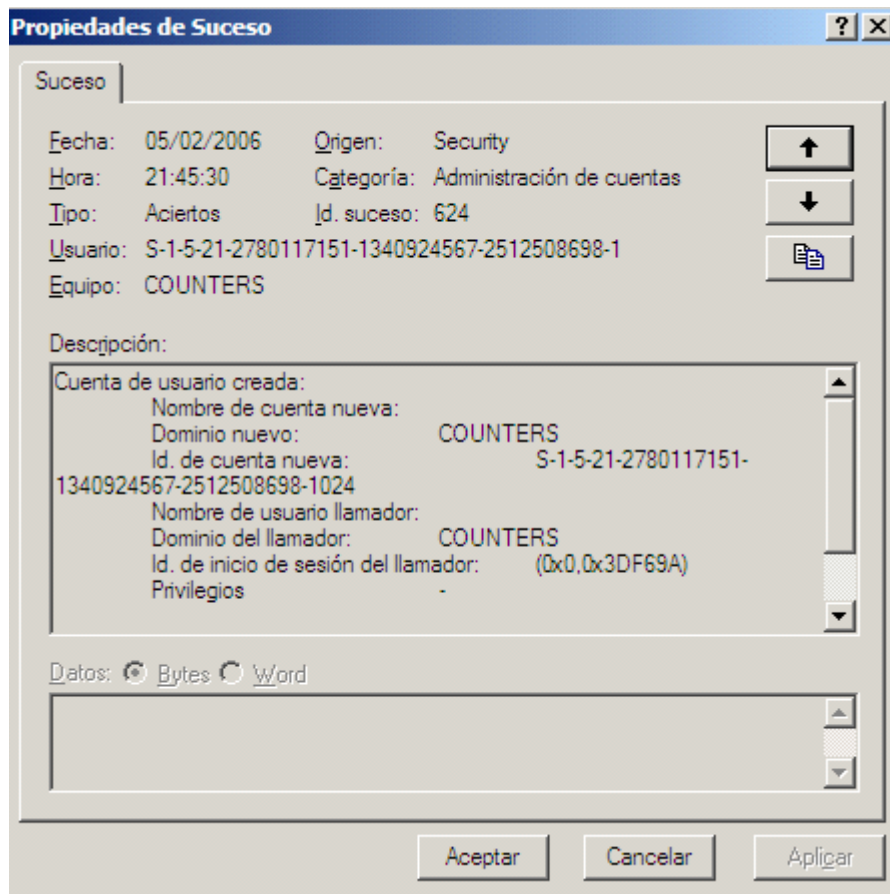
Recogida de archivos Log del sistema

Los ficheros Log de una máquina, sea la que sea, son una fuente de información importantísima en un análisis forense. Empezaremos con estos ficheros. Los sistemas Windows basados en NT tienen su principal fuente de Log en los archivos de sistema siguientes:

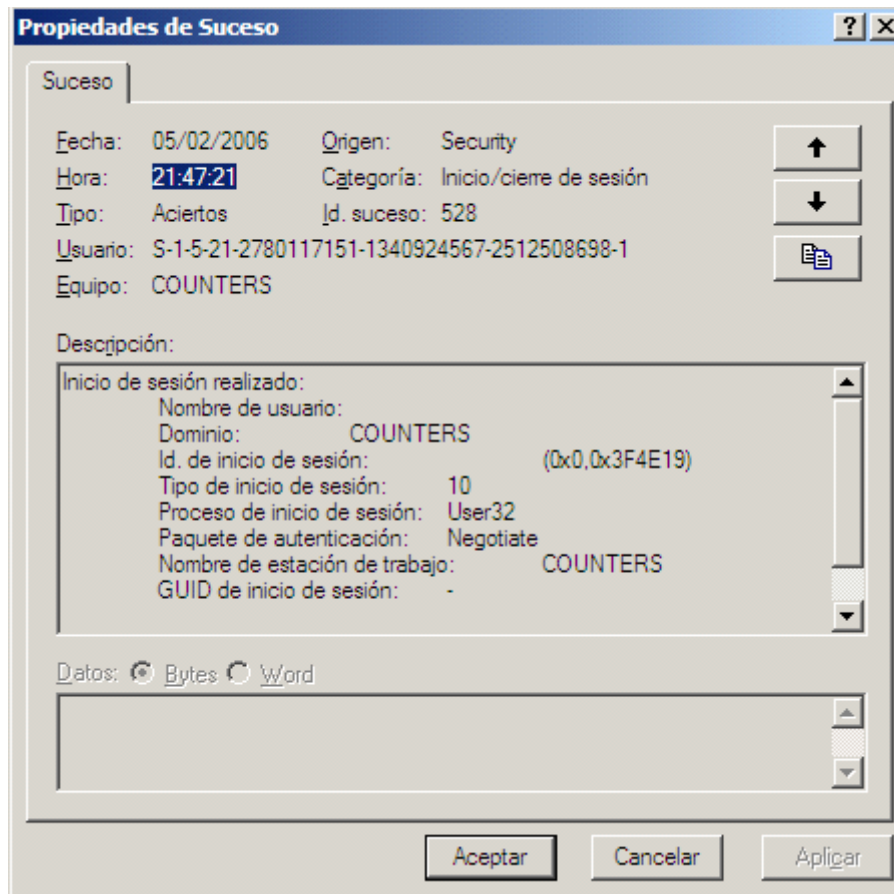
- SysEvent.Evt. Registra los sucesos relativos al sistema
- SecEvent.Evt. Registra los sucesos relativos a la seguridad
- AppEvent.Evt. Registra los sucesos relativos a aplicaciones

Estos ficheros se encuentran en el directorio `%systemroot%\system32\config`. Si están auditadas las opciones de inicio de sesión, cambio de directivas y permisos, nos centraremos con especial atención en el archivo Log **SecEvent.Evt**. Para visualizar este fichero podremos utilizar la herramienta de Windows **eventvwr.msc**, comúnmente llamada Visor de Sucesos. Abriremos con esta herramienta el archivo SecEvent.Evt, que es el encargado de almacenar los sucesos relativos a la seguridad, tales como ingresos en la máquina, cambio de directivas, etc... Por ejemplo, podríamos buscar todo acceso físico a la máquina, cambio de directivas y creación de cuentas de usuario. Eso nos podría dar una idea de quién toca el sistema.

Por ejemplo, el **evento 624** es el referido por Windows para un suceso de creación de cuenta de usuario. En la siguiente imagen podréis ver como lo registra Windows:



Otro ejemplo de suceso, sería el relativo al inicio de sesión. Windows almacena este suceso con el **identificador 528**.



Como podréis comprobar, Windows almacena información sobre el usuario que ha iniciado sesión, ID de sesión, tipo de inicio de sesión, nombre de estación de trabajo, etc...

Windows tiene distintos archivos Log para auditar los posibles sucesos y/o errores que puedan surgir en la vida útil del sistema operativo. Algunos de ellos son:

- WindowsUpdate.log (Log de Windows Update)
- memory.dump (Archivos de volcado de memoria)
- Archivos de registro de Windows (Software, System, etc...)

En internet disponemos de muchos sitios en donde buscar información y/o solución a un evento determinado. Algunos de ellos son:

[EventID](#)
[Knowledge base Microsoft](#)

Rebuscando en la Papelera de Reciclaje

Sabiendo más o menos como funciona la estructura de la papelera de reciclaje y de cómo Windows indexa sus archivos, entenderemos mejor la información que podamos sacar de la papelera.

Para recomponer un archivo INFO2 utilizaremos la herramienta llamada **Rifiuti**, palabra italiana que se utiliza para llamar a la basura. La estructura de comandos de Rifiuti es tan simple como este comando:

Rifiuti.exe INFO2 >INFO2

En una ventana de MS-DOS teclearemos lo siguiente:

```
Rifiuti c:\recycler\S-1-5-21-27800051451-1340124367-2516573698-1128\INFO2 >
C:\INFO2.TXT**
```

Podremos redirigir la salida del comando a un archivo .txt para poder visualizarlo con el notepad o con cualquier editor de texto.

****** El **UserID** es diferente para cada máquina y para cada usuario. Este número es creado por Windows y diferencia de forma unívoca a cada usuario.

Sacando información del archivo Index.dat

Sabiendo más o menos como Internet Explorer indexa sus páginas, y sabiendo la ruta en donde se encuentran, vamos a ver el contenido de lo que hay en los archivos Index.dat. Hay bastantes soluciones para hacerlo. Aquí explicaremos dos formas:

- Desde línea de comandos (MS-DOS) y sin aplicación de terceros
- Desde línea de comandos con aplicación de terceros

Son dos técnicas muy sencillas y que cualquier persona sin amplios conocimientos sobre informática puede realizar.

Desde línea de comandos

Si quisiésemos visualizar que tenemos en nuestro Index.dat desde línea de comandos y sin aplicación de terceros podríamos utilizar este comando:

```
C:\Documents and Settings\tu_user_name\Configuración
local\Historial\History.IE5> find /i "http://" index.dat | sort > C:\history.txt
```

Explicación: Nos situaremos en el directorio en donde se encuentra el archivo Index.dat de nuestra carpeta **Historial**, y una vez dentro utilizaremos el comando **find** para buscar cadenas dentro del archivo que empiecen por **http://**. La opción **/i** indica que no distinga entre mayúsculas y minúsculas y el comando **sort** lo utilizaremos para ordenar la salida.

Desde línea de comandos con Herramientas de terceros

La segunda opción que podemos utilizar para visualizar este archivo sería con aplicaciones de terceros. Por ejemplo **Pasco**. Esta aplicación reporta la salida en un fichero con texto delimitado. Nuestra vista lo agradecerá, ya que podremos visualizar este archivo en cualquier hoja de cálculo como Excel, y nos facilitará mucho la tarea a la hora de **husmear**. Otra cosa interesante que nos brinda este programa es la opción **undelete**. El modo undelete hace caso omiso a la información que hay en la tabla HASH y reconstruye cualquier dato válido de actividad. Gracias a esto podremos recuperar información que con otra aplicación no podríamos.

La línea de comandos que utiliza Pasco es muy sencilla:

```
[kjones: pasco] kjones% ./pasco
```

```
Usage: pasco [options] <filename>  
-d Undelete Activity Records  
-t Field Delimiter (TAB by default)
```

Tan solo tendríamos que poner este sencillo comando:

Pasco -d -t index.dat >index.csv

Una vez creado index.csv, podremos importar este texto a cualquier hoja de cálculo, como Microsoft Excel por ejemplo.

Un ejemplo de salida con este comando podemos verlo en el siguiente párrafo:

**URLindex.dat Visited: Silverhack@http://170.110.240.150:8080/AccountUser.wmf
index.dat Sun Feb 9 22:51:10 2006**

Ntuser.dat y archivos de Registro de Windows

Dado que Windows utiliza como referencia toda la información que se encuentra en el registro, un analista forense puede utilizar como referencia esta gran base de datos para recabar información sobre la máquina. En la base de datos de registro que se encuentra en el sistema Windows, podremos averiguar:

- Versión del Sistema Operativo
- Ruta de instalación
- Clave de Producto
- Tipo de Procesador de la máquina
- Aplicaciones Instaladas
- HotFix y parches instalados
- Servicios corriendo en la máquina
- Configuración y enumeración de los adaptadores de red

Podemos utilizar varias herramientas para analizar el registro. Algunas de ellas son:

- WRR (Windows Registry Recovery de MiTec)
- Comando nativo XP FC (Compara ficheros)
- Access Data Registry Viewer
- Windows Registry File Viewer
- Windiff (Herramienta para comparar ficheros)

Una vez abierto estos archivos de sistema que referencian al registro de Windows (SECURITY, SYSTEM, SOFTWARE, SAM, DEFAULT) con alguna de estas aplicaciones, podremos movernos por sus distintas ramas, y poder así analizar los datos que contienen estos ficheros.

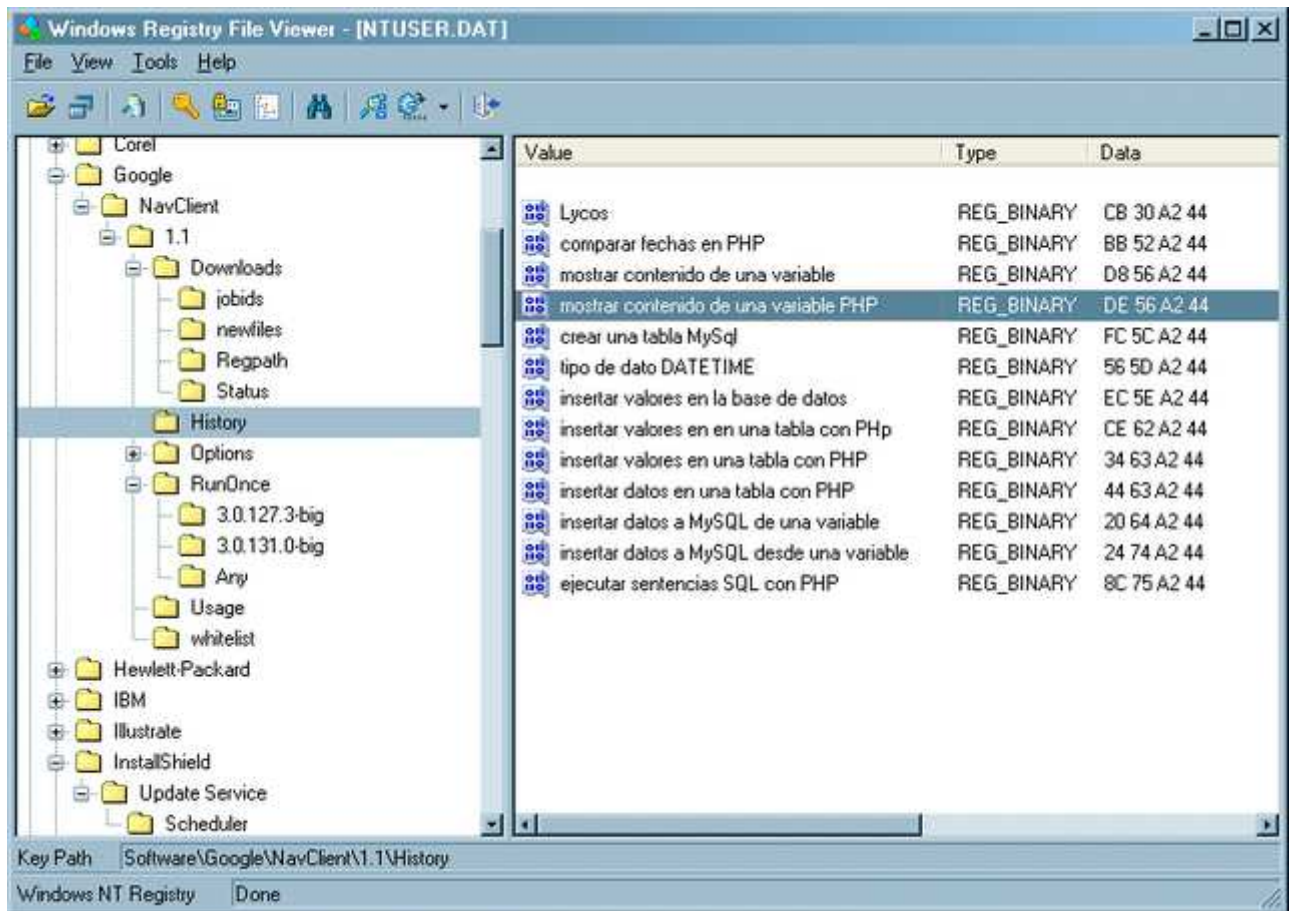
Ejemplo práctico: Por ejemplo podríamos averiguar los criterios de búsqueda de un usuario en particular, buscando en el registro del perfil de usuario (ntuser.dat) la clave

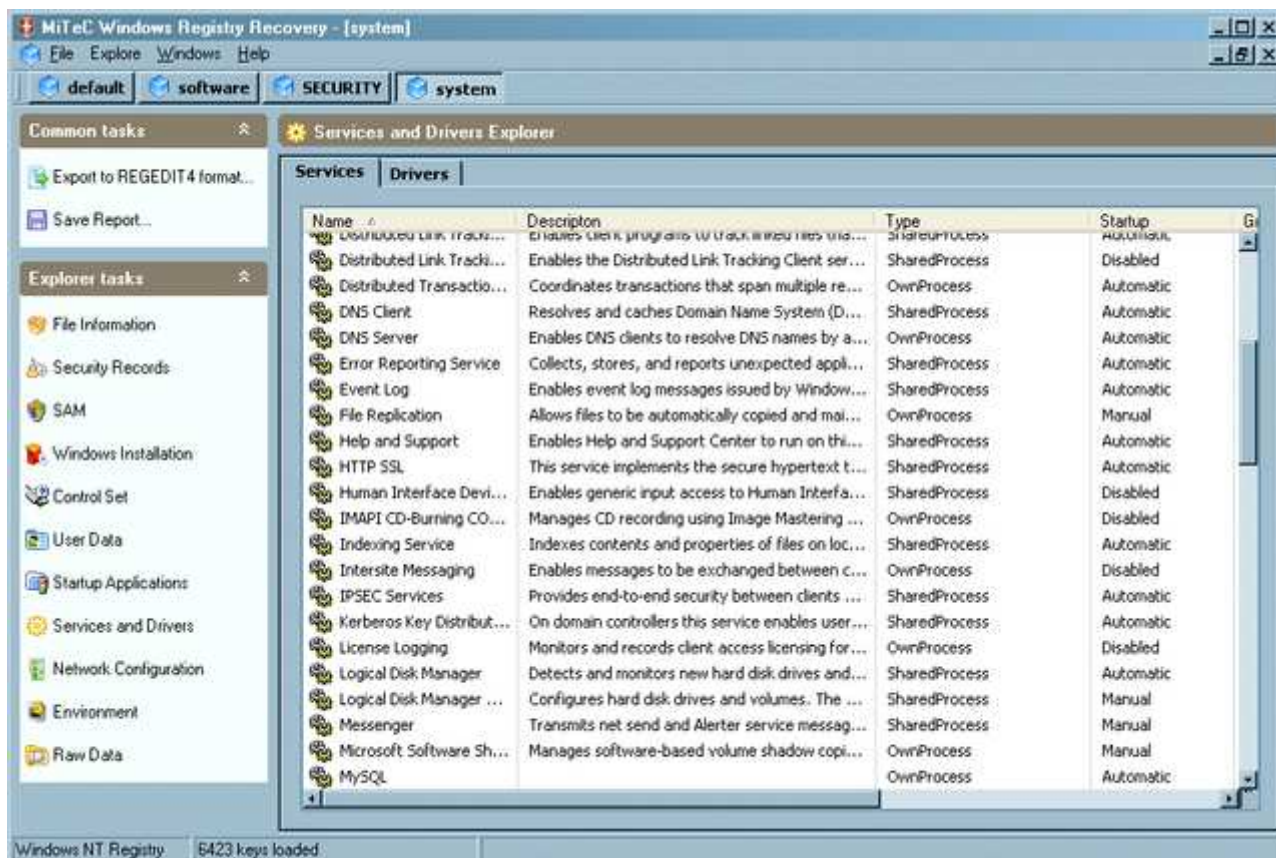
del historial de navegación de la barra de búsqueda de Google (si la tuviese instalada)

HKCU\Software\Google\NavClient\1.1\History

Ejemplo práctico2: Podríamos también investigar si un usuario determinado utilizó alguna cuenta de mensajero (Messenger) sin autorización en un sistema comprometido. La clave que nos mostraría esta información (cuenta de correo utilizada) se encontraría en la parte del registro perteneciente al usuario (ntuser.dat), y en la clave siguiente:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Unread Mail





Cookies. Estructura, funcionamiento y metodología de acceso a la información

Una cookie no es más que un fichero de texto. El funcionamiento es bastante sencillo. Algunos servidores piden a nuestro navegador que escriba en nuestro disco duro, y nuestro navegador escribe en un fichero de texto información acerca de lo que hemos estado haciendo por sus páginas.

Una de las mayores ventajas de las cookies es que se almacenan en el equipo del usuario. Con esto conseguimos liberar al servidor de una sobrecarga en su espacio de disco. Cuando el servidor solicite esa información, nuestro navegador se la enviará en forma de cookie. Las cookies poseen una fecha de caducidad, que puede oscilar desde el tiempo que dure la sesión hasta una fecha especificada, a partir de la cual dejan de ser operativas.

En un sistema Windows 2K/XP/2K3, las cookies se encuentran en la ruta:

C:\Documents and Settings\Nombre_Usuario\Cookies

En Internet Explorer existe una cookie por cada dirección de Internet que visitamos, con la nomenclatura siguiente:

Identificador_Usuario@dominio.txt

Para ver el contenido de una cookie, sólo tenemos que editarla con un editor de texto. Las cookies se almacenan en memoria hasta que salimos del navegador, momento en el que se escriben en el disco. Para ver las cookies que nos pide un determinado sitio Web cuando estamos conectados, podremos escribir en la barra de direcciones este simple comando:

JavaScript:alert(document.cookie);

Acto seguido nos saldrá un cuadro de alerta, con el contenido de la cookie que nos pide el servidor.

Limitaciones de las Cookies

- Trescientas cookies en total en el archivo de cookies. Si llega la número 301, se borra la más antigua
- Veinte cookies por servidor o dominio
- 4 Kb por cookie, para la suma del nombre y valor de la cookie
- Ninguna máquina fuera del dominio de la cookie puede leerla

Riesgos reales de una cookie

- Visualizar los datos contenidos en las cookies
- Utilizar los servicios a los que permiten acceder los username y passwords almacenados en una cookie
- Conocer las preferencias del usuario (posibilidad de envío de propaganda personalizada)

Galleta (Aplicación para visualizar Cookies)

Galleta es una herramienta diseñada por FoundStone que analiza todas las cookies y genera un reporte con el detalle de cada una en formato *.csv (Se puede visualizar en Excel).

El funcionamiento de esta herramienta es bastante sencillo.

Usage: galleta [options] <filename>
-t Field Delimiter (TAB by default)

Ejemplo:

galleta [silverhack@dominio.txt](#) >CookieSilverhack.txt

El archivo de paginación de Windows. Pagefile.sys

El archivo de paginación (Pagefile.sys) es un archivo que Windows utiliza como si fuera memoria RAM (Memoria de acceso aleatorio). Este archivo está situado en la raíz del disco duro y por defecto lo marca como **oculto**. El archivo de paginación y la memoria física conforman la memoria virtual. De manera predeterminada, Windows almacena el archivo de paginación en la partición de inicio, que es la partición que contiene el sistema operativo y sus archivos auxiliares. El tamaño predeterminado o recomendado del archivo de paginación es igual a 1,5 veces la cantidad total de memoria RAM.

Para visualizar el contenido del archivo de paginación, podemos utilizar una herramienta de **SysInternals**, llamada **Strings.exe**.

Esta herramienta busca cadenas de texto (ASCII o Unicode) en archivos. Podemos conjugar esta herramienta con la herramienta nativa de Windows [findstr](#). Esta herramienta busca patrones de texto en archivos utilizando expresiones regulares.

Por ejemplo, podríamos buscar información relativa a conversaciones de Messenger, buscando cadenas de texto que contengan las palabras MSNMSGR o MSNSLP.

El comando resultante:

```
strings C:\pagefile.sys | findstr /i MSNMSGR
```

El resultado que podríamos sacar sería algo similar a esto:

```
MSNMSGR:silverhack@elhacker.net MSNSLP/1.0  
To: <msnmsgr:silverhack@elhacker.net>  
From: <msnmsgr:silverhacker@hotmail.com>  
Via: MSNSLP/1.0/TLP; branch= {5B3DFCC7-7BB5-4C2E-9351-5B8033C6A0CF}  
CSeq: 0  
Call-ID: {43C326E3-6109-4B3D-9E4D-F9AB0C5FF0C}  
Max-Forwards: 0  
Content-Type: application/x-msnmsgr-sessionreqbody  
Content-Length: 219
```

Introducción a la informática forense en entornos Windows (III)

General

[Recogida de información \(puertos, servicios, procesos\)](#)

[Analizando una pantalla azul \(BSOD\)](#)

[Alternate Data Streams \(ADS\)](#)

Material relacionado a este documento

[Comportamiento de Virus en plataformas Windows](#)

Recogida de información (puertos, servicios, procesos)

La evidencia digital, por su naturaleza, es bastante frágil. Ésta puede ser alterada, dañada y/o destruida, principalmente por un mal manejo y/o recogida de estos datos. Por estas razones (intentamos hacer) tanto hincapié en la recogida y manejo de estos datos. La falta de este principio, puede ocasionar que estas evidencias sean inutilizables o no válidas en un proceso judicial, por no decir que podemos llegar a una conclusión inexacta. Debemos recopilar las evidencias que nos encontremos de tal forma que las protejamos y mantengamos su fiabilidad.

Una guía para este propósito, podemos encontrarla en esta URL del departamento de Justicia de EE.UU.

<http://www.ncjrs.gov/txtfiles1/nij/187736.txt> (Inglés)

En este capítulo nos centraremos más en la recogida práctica de información que en la teoría, para no hacer tan extenso el paper. En internet hay muchísima información sobre aspectos legales en este tipo de materia, y abundante información también en lo referente al correcto manejo y recogida de evidencias.

El primer paso que vamos a realizar será una captura de los datos físicos de la máquina, es decir, recoger diversos datos como:

- Dueño de la máquina (Organización)
- Tipo de BIOS
- Uptime del sistema (Time to live)
- Directorios del sistema
- Número de tarjetas de red
- Número de servipacks o updates del sistema
- Ubicación del archivo de paginación
- Tipo de procesador
- Fabricante y modelo del sistema
- Número de procesadores
- Tamaño de la memoria RAM
- Versión del sistema operativo
- Etc...

Esto lo podemos hacer con varias herramientas, de las cuales vamos a describir dos.

[SystemInfo \(Nativa de Windows\)](#)

Aplicación nativa de Windows que nos muestra información acerca de la configuración del sistema y el tipo y versión del sistema operativo. También nos muestra información relevante a seguridad, propiedades del Hardware, memoria RAM, espacio total del disco e información sobre las tarjetas de red.

Su sintaxis es la siguiente:

```
Systeminfo [/s equipo [/u dominio\nombreUsuario [/p contraseña]]] [/fo {TABLE | LIST | CSV}] [/nh]
```

Podríamos redireccionar la salida del comando a un fichero de texto y fechado, para saber con exactitud cuándo se tomó la evidencia. El comando que podríamos poner sería el siguiente:

```
Systeminfo /FO list >SystemInfo.txt &date /t >>SystemInfo.txt &time /t >>SystemInfo.txt
```

[PsInfo \(SysInternals\)](#)

Esta herramienta es similar a la nativa de Windows, y podremos conseguir prácticamente los mismos resultados. Ambas herramientas permiten su uso tanto en local como en remoto. Su sintaxis es la siguiente:

```
psinfo [[\computer [, computer [...]] | @file [-u user [-p psswd]]] [-h] [-s] [-d] [-c [-t delimiter]] [filter]
```

No necesita más explicación.

El segundo paso que vamos a realizar será recopilar información acerca de los servicios que hay corriendo en la máquina con sus estadísticas. En este punto voy a utilizar el comando nativo de Windows net y el comando SC.

Con el comando [net statistics](#) vamos a recabar información acerca de los bytes recibidos por el sistema, el número de inicios de sesión fallidos, las cuentas de uso fallidas, etc... Toda esta información la almacenaremos en un archivo de texto con fecha y hora incluida para su posterior análisis. El comando resultante podría ser el siguiente:

```
Net statistics Workstation >Estadisticas.txt &date /t >>Estadisticas.txt &time /t >>Estadisticas.txt
```

El comando [SC](#) me va a permitir conseguir una lista de los servicios que actualmente están corriendo en la máquina. Aunque SC posee muchos comandos para poder regular su salida, un comando resultante válido podría ser el siguiente:

```
SC query >ServiceOpen.txt &date /t >>ServiceOpen.txt &time /t >>ServiceOpen.txt
```

El tercer paso que vamos a realizar será la recopilación de supuestos procesos maliciosos, puertos de escucha, identificación de aplicaciones no autorizadas y la finalización de procesos legítimos.

Para saber cuantas conexiones tengo abiertas puedo utilizar el comando nativo de Windows netstat.

Utilizaré la opción -a para conocer todas las conexiones y puertos de escucha, y la juntaré con la opción

-b para conocer el ejecutable que crea la conexión necesaria para llegar al TCP/IP. El comando resultante fechado para su posterior análisis quedaría:

```
netstat -ab >Conexiones.txt &date /t >>Conexiones.txt &time /t Conexiones.txt
```

Para saber los procesos que tenemos actualmente corriendo en nuestro sistema utilizaremos la aplicación nativa de Windows tasklist, o en su defecto pslist (sysinternals). También utilizaremos la herramienta [Fport](#). Ambas herramientas (tasklist, pslist) permiten realizar esta operación en local como en remoto. El comando resultante quedaría:

```
Tasklist >Procesos.txt &date /t >>Procesos.txt &time /t >>Procesos.txt
```

También vamos a recopilar información sobre los servicios que dependen de los procesos que están en funcionamiento. Tasklist también contempla esta situación. El comando sería:

```
Tasklist /SVC >ProcesosYServicios.txt &date /t >>ProcesosYServicios.txt &time /t >>ProcesosYServicios.txt
```

Muchas veces cuando en el sistema hay determinados rootkits, virus o troyanos, éste no nos muestra una salida “coherente”, de ahí a que siempre que podamos utilicemos aplicaciones que sean lo menos intrusivas en el sistema. Si somos un poco “paranoicos” en ese tema, para ver los puertos abiertos en un sistema podemos utilizar la herramienta fport. Por regla general, no nos debemos fiar de un sistema en el que haya corriendo este tipo de virus.

Básicamente fport nos muestra la misma salida que si ejecutásemos el comando nativo netstat con el filtro -a y -n. También puede identificar puertos desconocidos que estén abiertos, con sus correspondientes procesos y PID. La salida a este comando sería la siguiente:

```
C:\>fport
```

```
FPort v2.0 - TCP/IP Process to Port Mapper
```

```
Copyright 2000 by Foundstone, Inc.
```

```
http://www.foundstone.com
```

```
Pid Process Port Proto Path
```

```
392 svchost -> 135 TCP C:\WINNT\system32\svchost.exe
```

```
8 System -> 139 TCP
```

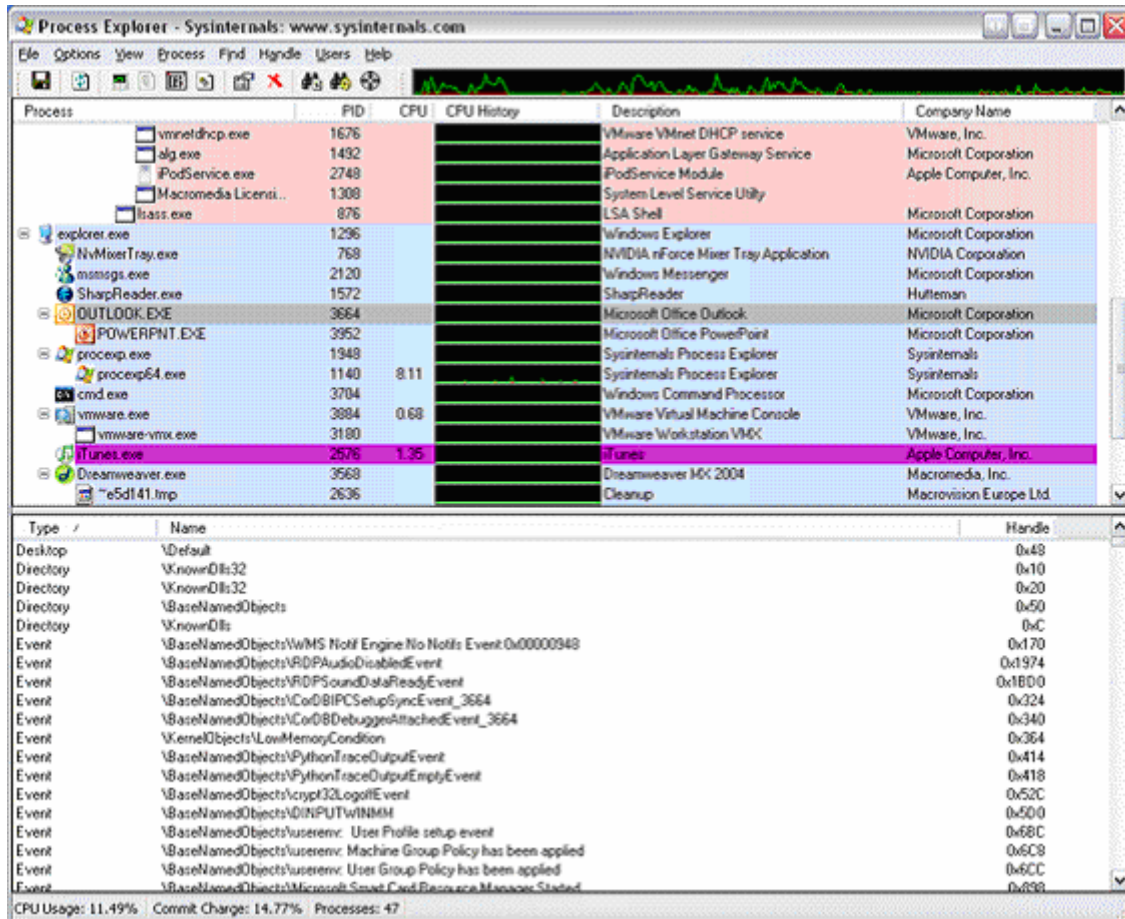
```
8 System -> 445 TCP
```

```
508 MSTask -> 1025 TCP C:\WINNT\system32\MSTask.exe
```

```
392 svchost -> 135 UDP C:\WINNT\system32\svchost.exe
```


8 System -> 137 UDP
 8 System -> 138 UDP
 8 System -> 445 UDP
 224 lsass -> 500 UDP C:\WINNT\system32\lsass.exe
 212 services -> 1026 UDP C:\WINNT\system32\services.exe

También podremos ver los procesos que corren en la máquina de forma gráfica con la herramienta de sysinternals [Process Explorer.exe](#), la cual nos mucha más información pero de forma gráfica. Una captura de pantalla de la aplicación:



También podríamos recabar información sobre los módulos que cargan estos procesos. Podemos averiguar por ejemplo qué DLL están asociadas a un determinado proceso. Así tendremos un control más exhaustivo sobre los procesos. Para recabar esta información podemos utilizar la herramienta de sysinternals [ListDLLs.exe](#). Por ejemplo, si quisiésemos averiguar qué DLL dependen del proceso con PID 1548 utilizaríamos la siguiente sintaxis:

```
ListDLLs.exe 1548 >DLL1548.txt &date /t >>DLL1548.txt &time /t>>DLL1548.txt
```

El cuarto paso que vamos a realizar será una recopilación de los últimos accesos a ficheros, clasificados por fechas. Esta lista nos servirá de referencia a la hora de realizar el análisis, y podremos comprobar qué ficheros se modificaron en el día o los días en los que el sistema estuvo comprometido.

Podremos utilizar varias herramientas destinadas a tal fin, pero vamos utilizar sólo dos. En una primera instancia podremos utilizar el comando nativo de Windows DIR, con algunas reglas para que nos muestre los ficheros modificados conforme a la fecha. Podríamos utilizar el siguiente comando:

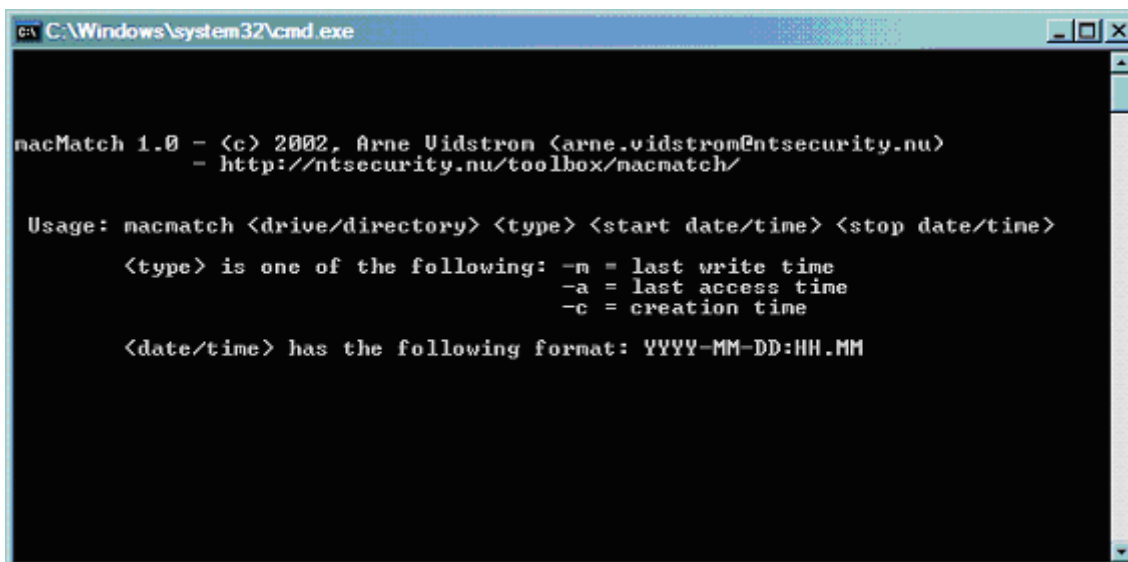
```
DIR /t: a /a /s /o: d c:\ >Directory.txt &date /t >>Directory.txt &time /t >>Directory.txt
```

- /t:a Nos muestra el campo del último acceso (Fecha)
- /a Muestra todos los ficheros
- /s Muestra todos los archivos del directorio especificado, incluidos los subdirectorios
- /o Lista los archivos indicados
- d Muestra los más antiguos primero (Por fecha y hora)

Como siempre poniéndole la fecha al final para saber cuándo tomamos esa prueba.

En varias ocasiones esta lista puede ser larguísima y el fichero puede ocuparnos unos cuantos megas. La herramienta que vamos a describir a continuación puede ayudarnos a buscar archivos en fechas concretas. La herramienta en sí se llama [MacMatch.exe](#). Ésta herramienta básicamente buscará ficheros modificados en un intervalo de tiempo, que lógicamente se lo daremos nosotros.

Una captura de imagen con la sintaxis:



```
C:\Windows\system32\cmd.exe

macMatch 1.0 - (c) 2002, Arne Vidstrom (arne.vidstrom@ntsecurity.nu)
- http://ntsecurity.nu/toolbox/macmatch/

Usage: macmatch <drive/directory> <type> <start date/time> <stop date/time>

<type> is one of the following: -m = last write time
                                -a = last access time
                                -c = creation time

<date/time> has the following format: YYYY-MM-DD:HH.MM
```

Por ejemplo, si quisiésemos saber qué ficheros se “tocaron” o modificaron entre el 10 y el 12 de Noviembre, sobre las 15 horas, el comando a poner sería el siguiente:

```
Macmatch.exe c:\ -a 2006-11-10:15.00 2006-11-12:15.59
```

Recopilemos información sobre diversos aspectos de los usuarios. Trataremos de recabar la siguiente información:

- Relación de usuarios creados en el sistema
- Relación de las últimas sesiones (LogOn) fallidas en el sistema
- Relación de las últimas sesiones establecidas remotamente en el sistema
- Relación de las actividades de los usuarios remotos
- Tiempo de logeo en el sistema
- Histórico de los usuarios logeados localmente en el sistema

Para utilizar estas herramientas es necesario verificar que estén activados los controles de auditoría.

Para realizar estas operaciones vamos a utilizar tres aplicaciones. Una aplicación llamada [netusers](#), otra llamada [nlast](#) y una aplicación llamada [psloggedon](#).

Con `netusers` podremos comprobar los usuarios que están conectados actualmente a una máquina remota o local. Su sintaxis es la siguiente:

```

F:\WINDOWS\system32\cmd.exe
Network Users [Version 1.21]
Displays a list of users logged on to a specified Windows system.

The syntax of this command is:
netusers [\\computername \\...] [/history] [/local] [/verbose]

\\computername Allows you to query remote systems. Multiple computers
                can be specified, each separated by a space.
/history:/h     Displays the user accounts that have logged on in the past. If
                history is omitted, currently logged on users are displayed.
/local:/l      Includes local user accounts in the output, by default they
                are skipped. In a domain these are usually service accounts.
/verbose:/v    Reports non-fatal problems as errors (such as deleted accounts)

An argument of /? or -? displays this syntax and returns 1.
A successful completion will return 0.

Copyright 2000-2002 Marty List, Marty@OptimumX.com

```

Para ver los usuarios que actualmente están conectados a la máquina local tan solo tendríamos que poner el comando:

netusers >usuarios.txt &date /t >>usuarios.txt &time /t>>usuarios.txt

Si queremos que nos muestre un histórico de usuarios que se han logeado anteriormente, pondríamos:

netusers /history >UsersHistory.txt &date /t >>UsersHistory.txt &time /t >>UsersHistory.txt.

Otra aplicación que nos sirve para el mismo propósito es `PsLoggedOn`. Esta herramienta la podemos utilizar tanto en local como en remoto, y la salida que nos muestra por defecto es la siguiente:

- Usuarios locales autenticados en el sistema
- Usuarios logeados a través de recursos compartidos
- Hora de inicio de sesión

En artículos anteriores, hemos podido comprobar cómo funciona el archivo de registro de Windows, y cómo podemos sacar información de ellos.

Para auditar los sucesos relativos a los inicios de sesión, el archivo de seguridad del visor de sucesos nos mostrará todos los sucesos auditados, como inicios de sesión fallidos, inicios de sesión correctos, alguna operación con privilegios, etc...

Si tuviésemos que mirar uno a uno todos esos sucesos en un entorno de producción, prácticamente nos sería casi imposible de terminar, debido a la longitud del fichero. Aquí es donde actúa la herramienta NtLast.

Por medio de esta herramienta podremos averiguar de forma sencilla todos y cada uno de los sucesos del sistema.

En una primera instancia vamos a sacar un fichero con los últimos 100 inicios de sesión exitosos, incluidas las sesiones nulas en caso de que el sistema no tuviese parcheada esta opción. El comando resultante sería:

```
NtLast -null -v -n 100 >>SuccessfulLogons.txt &date /t >>SuccessfulLogons.txt  
&time /t >>SuccessfulLogons.txt
```

Ahora vamos a sacar un listado con los últimos 100 inicios de sesión fallidos. El comando resultante sería:

```
NtLast -v -n 100 >>LogonsFailed.txt &date /t >>LogonsFailed.txt &time /t  
>>LogonsFailed.txt
```

Puede que también queramos sacar un listado con los últimos 100 inicios de sesión remotos. El comando resultante sería:

```
NtLast -v -n 100 -r >>RemoteLogin.txt &date /t >>RemoteLogin.txt &time /t  
>>RemoteLogin.txt
```

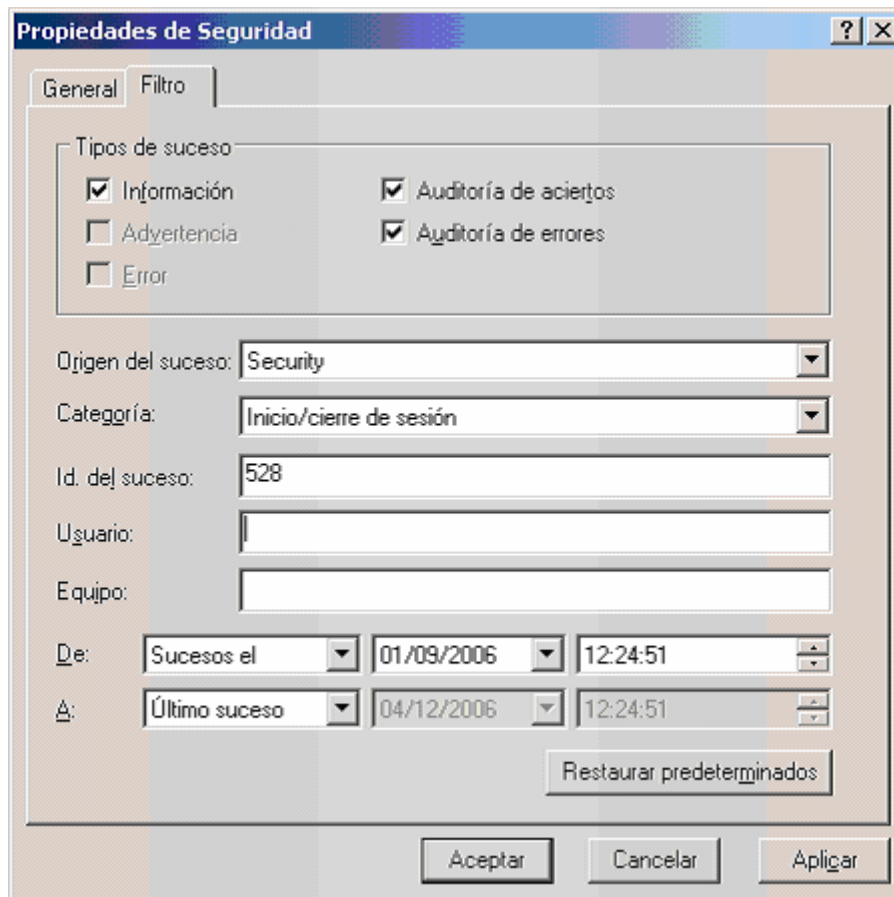
NtLast es una aplicación que puede dar mucho juego a la hora de sacar listados. Además de todo esto, que ya es mucho, podremos sacar todo esto pero referente a un solo usuario, por si tuviésemos alguna pista y tuviésemos que estrechar el cerco.

Por poner algún ejemplo, y si en la empresa X tuviésemos sospechas de que el usuario Silverhack está cometiendo actividades impropias o no encomendadas, podríamos hacer lo siguiente:

```
NtLast -v -n 100 -r -u Silverhack >>RemoteLoginSilverhack.txt &date /t  
>>RemoteLoginSilverhack.txt &time /t >> RemoteLoginSilverhack.txt
```

Con este comando lo que obtenemos es una lista con los últimos 100 inicios de sesión remotos del usuario Silverhack.

Desde el propio visor de sucesos ([eventvwr.exe](#)) también podemos hacer esto mismo, aplicando un filtro al archivo. Por ejemplo, si quisiésemos mirar quién inició sesión de forma exitosa en los últimos 100 días, podríamos poner:



Analizando una pantalla azul (BSOD)

Cuando Windows encuentra una sentencia que pueda llegar a comprometer el sistema, éste se para. Esta sentencia se llama KeBugCheckEx. Esta llamada al sistema la podríamos llamar fallo de sistema, error de kernel, STOP, etc.. , y toma 5 argumentos:

Código de STOP

Cuatro parámetros que indican el código de STOP

Si hemos configurado nuestro sistema para que nos vuelque el contenido de la memoria, éste nos generará un archivo para su posterior análisis. Estos archivos se dividen en:

Small Memory Dump.- El más pequeño de todos y el más limitado (en cuanto a investigación). Solo ocupa 64 Kb y en este archivo irá incluida la siguiente información:

- El mensaje de detención, parámetros y otros datos
El contexto del procesador (PRCB) para el procesador que se colgó.
- La información de proceso y contexto del kernel (EPROCESS) del proceso que se colgó.
- La información de proceso y contexto del kernel (ETHREAD) del thread que se colgó.
- La pila de llamadas del modo kernel para el subproceso que se colgó. Si éste tiene un tamaño mayor que 16 Kb, sólo los primeros 16 Kb serán almacenados.
- Una lista de controladores cargados

- Una lista de módulos cargados y no cargados
- Bloque de datos de depuración. Contiene información básica de depuración acerca del sistema.
- Páginas adicionales de memoria. Windows también almacena estos datos para que así podamos obtener una mayor “versión” de lo que cascó. Esto nos ayudará a identificar mucho mejor el error ya que contienen las últimas páginas de datos a las que señalaban los registros cuando el sistema se colgó.

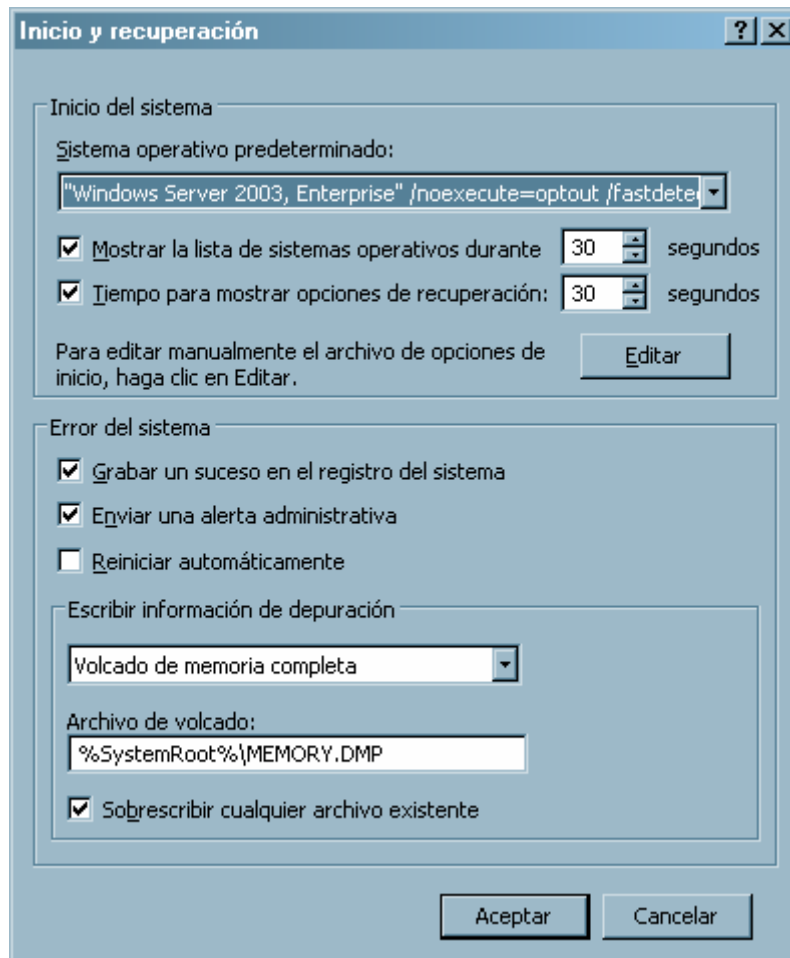
Kernel Memory Dump.- Escribe el contenido de la memoria excepto los procesos.

Complete Memory Dump.- Escribe todo el contenido de la memoria. [/li][list]

En muchos casos, la información que viene contenida en un MiniDump, no es suficiente para analizar en profundidad un error. Hace años el espacio en disco podría ser un problema para almacenar este tipo de datos, pero hoy en día esto ya no es un problema. Si queremos analizar mejor el error, configurad vuestro sistema para generar o un volcado de memoria completo (Complete Memory Dump) o un volcado del Kernel (Kernel Memory Dump).

Para configurar el volcado de memoria iremos a Inicio à Ejecutar à sysdm.cpl y pulsaremos Enter.

Una vez dentro nos dirigiremos a la pestaña **Configuración**, y una vez dentro, en la parte de **Inicio y Recuperación** pulsaremos **Configuración**



Como podréis ver en la imagen, hemos configurado nuestro Windows de prueba, para que NO reinicie cuando muestre una pantalla de STOP, así podremos ver la pantalla azul en todo su esplendor, y aparte le decimos que cuando muestre una pantalla de error, nos vuelque el contenido completo de la memoria, para que podamos debugearlo, y así practicar!

Y para poder practicar, qué mejor que una maquina virtual no? Podemos instalar una máquina virtual desde 0, o descargarnos una de las muchas imágenes para Virtual PC que podemos encontrar en Microsoft. En cuestión yo utilizaré esta:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=21EABB90-958F-4B64-B5F1-73D0A413C8EF&displaylang=en>

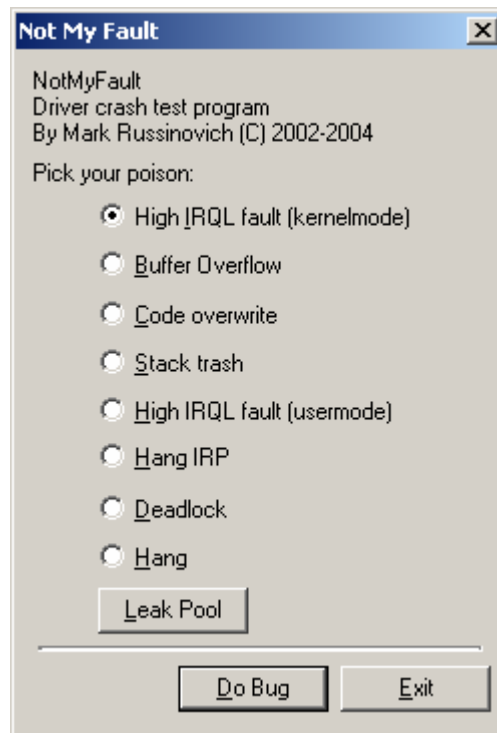
En este blog también hay puestas algunas imágenes en descarga directa con algunas plataformas servidoras, por si queréis descargarlas también. Y gratis!!

<http://windowstips.wordpress.com/2007/01/18/practica-practica/>

Bien. Ya tenemos nuestra máquina virtual funcionando. Hemos configurado nuestro sistema para que nos haga un volcado de memoria completo. Qué hacemos ahora? Esperar? Instalar drivers y aplicaciones hasta que pete por algún lado?

Como no queremos esperar a tener un BSOD para practicar, Mr. Mark Russinovich,

autor de grandes herramientas, principal fundador de la Web Sysinternals y fichado por Microsoft, se ha codeado una herramienta muy chula para que podamos practicar. La herramienta en cuestión se llama NotMyFault.



Esta herramienta nos ayudará a crear los típicos escenarios que nos podemos encontrar en nuestro trabajo. Esta herramienta carga un driver llamado MyFault.sys, y este es el que va a implementar las diferentes BSOD que nos podemos encontrar.

La herramienta en cuestión la podéis descargar de la siguiente dirección:

<http://download.sysinternals.com/Files/Notmyfault.zip>

También necesitaremos un debuggeador, para poder analizar los volcados de memoria. Utilizaremos el debuggeador de Microsoft WinDbg, el cual lo podemos descargar de:

<http://www.microsoft.com/whdc/devtools/debugging/installx86.msp>

José Manuel Tella Llop, MVP de Windows, escribió un artículo en su día sobre cómo debíamos comportarnos ante una pantalla azul. El artículo original lo podéis encontrar aquí y nos servirá de guía en todo momento:

<http://multingles.net/docs/jmt/bsod.htm>

En el artículo de Tella viene cómo instalar tanto las herramientas de debugeo como los símbolos necesarios para poder analizar un volcado de memoria, así como un casque real, a modo de ejemplo, en el que nos muestra la sencillez con la que se saca un error. A partir de aquí asumiremos que tienes instalado WinDbg y tienes configurado correctamente el path de símbolos.

Manos a la obra! Abrimos nuestra máquina virtual, iniciamos nuestro Windows, ejecutamos NotMyFault.exe y pulsaremos la primera opción que viene en el Interface. **High IRQL fault (kernel mode)**. Acto seguido pulsamos en **Do Bug**.

```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

DRIVER_IRQL_NOT_LESS_OR_EQUAL

If this is the first time you've seen this stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacturer
for any windows updates you might need.

If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use Safe Mode to remove or disable components, restart
your computer, press F8 to select Advanced Startup options, and then
select Safe Mode.

Technical information:

*** STOP: 0x000000D1 (0xE1071800,0x0000001C,0x00000000,0xF7CDA403)

*** myfault.sys - Address F7CDA403 base at F7CDA000, DateStamp 43774e1d

Beginning dump of physical memory
Dumping physical memory to disk: 7
```

Upppssss. Pantallaco azul al canto. Formateamos el equipo? Reiniciamos? Reinstalamos el sistema operativo? Por qué Windows es tan malo con nosotros?? J

Detengámonos un momento a analizar el BSOD.

Primera pista: DRIVER_IRQL_NOT_LESS_OR_EQUAL

Explicación: Un driver en modo kernel ha intentado acceder a memoria paginable desde un proceso o aplicación

Causa: Un driver que ha intentado acceder a memoria paginable mientras había una interrupción, o incluso intentó acceder a memoria inválida o no presente.

Efecto: Casque real del sistema. Estamos jodidos...

Como podemos ver en el pantallazo azul, éste nos muestra varias cosas que tenemos que tener en cuenta en un primer momento:

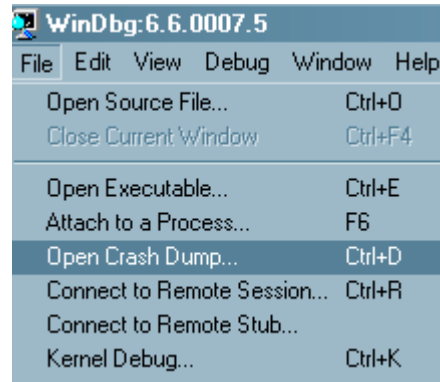
- Nos muestra el BugCheck o mensaje de STOP.

- Nos muestra el posible causante del fallo (MyFault.sys)

Pero como no estamos seguros, vamos a analizar el error debugeando un poquito.

Una vez que el volcado de memoria haya finalizado, reiniciaremos la máquina y en nuestro directorio Windows, tendremos un archivo llamado MEMORY.DUMP, el cual contiene toda la memoria, ejecutables, threads, etc..

En WinDbg, pulsaremos en **File-->Open Crash Dump (Acordaos del path de los símbolos explicados perfectamente en el artículo de JM Tella)**



Esto es lo que nos sale en primera instancia:

```
Use !analyze -v to get detailed debugging information. BugCheck D1,
{e1071800, 1c, 0, f7cda403} *** ERROR: Module load completed but
symbols could not be loaded for myfault.sys*** WARNING: Unable to
verify checksum for NotMyfault.exe*** ERROR: Module load completed but
symbols could not be loaded for NotMyfault.exe Probably caused by :
memory_corruption Followup: memory_corruption
```

El debugeador nos dice que casi con toda seguridad es una corrupción de memoria. El código del **BugCheck** es **D1 (DRIVER_IRQL_.....)**

Microsoft tiene registrado más de 150 posibles códigos de error. Todos ellos podemos encontrarlo en la ayuda de la misma aplicación. Más concretamente en: **Debuggin tools for Windows-->Debuggin Techniques-->Blue Screen-->Bug Check Code Reference**

También nos pone que si queremos un análisis más exhaustivo podemos escribir el comando **!analyze -v**. Así que escribimos en **kd>** el mencionado comando, con lo que obtendremos una respuesta parecida a esta:


```
Dump C:\Documents and Settings\Silverhack\Escritorio\MEMORYHighIRQL.DMP - WinDbg:6.6.0007.5
File Edit View Debug Window Help
Command
DRIVER_IRQL_NOT_LESS_OR_EQUAL (d1)
An attempt was made to access a pageable (or completely invalid) address at an
interrupt request level (IRQL) that is too high. This is usually
caused by drivers using improper addresses.
If kernel debugger is available get stack backtrace.
Arguments:
Arg1: e1071800, memory referenced
Arg2: 0000001c, IRQL
Arg3: 00000000, value 0 = read operation, 1 = write operation
Arg4: f7cda403, address which referenced memory
Debugging Details:
-----
READ_ADDRESS: e1071800 Paged pool
CURRENT_IRQL: 1c
FAULTING_IP:
myfault+403
f7cda403 8b06          mov     eax,dword ptr [esi]
DEFAULT_BUCKET_ID: CODE_CORRUPTION
BUGCHECK_STR: 0xD1
PROCESS_NAME: NotMyfault.exe
TRAP_FRAME: f3eb4b80 -- (.trap ffffffff3eb4b80)
ErrCode = 00000000
eax=e10705b0 ebx=8428ccb0 ecx=0000001c edx=00000000 esi=e1071800 edi=00000000
kd> !analyze -v
```

Vaya. Nos da el código exacto de la pantalla azul, el bugcheck reference (D1), los 4 argumentos restantes de la pantalla azul y nos dice que el proceso que cascó es el proceso NotMyfault.exe. Ya tenemos a nuestro culpable, pero no nos dice nada del driver (MyFault.sys) que provoca el casque real.

Cuando el debugeador no nos dice realmente quién es el culpable, y queremos averiguar más sobre dicho casque, podemos hacer varias cosas:

Ejecutar el comando **!Thread**

Con el comando **!Thread** conseguimos que nos muestre qué llamadas hizo el subproceso en el sistema.

Llamando al comando **!Thread**, el debugeador nos muestra las siguientes líneas:

```
kd> !threadTHREAD 842f6600 Cid 049c.04ac Teb: 7ffdf000 Win32Thread:
e1a99168 RUNNING on processor 0 IRP List: 8428cc98: (0006,0094) Flags:
00000000 Mdl: 00000000 WARNING: Stack unwind information not
available. Following frames may be wrong.f3eb4c58 80579a8a 841d8f18
8428cc98 842b6ad0 myfault+0x403
```

El proceso NotMyFault.exe hace una serie de llamadas, hasta que una pone en peligro la estabilidad del sistema. En nuestro caso es:

f3eb4c58 80579a8a 841d8f18 8428cc98 842b6ad0 myfault+0x403

Una vez que realiza la llamada, el sistema nos avisa de que la pila posiblemente esté corrompida y que los frames pueden estar corruptos.

WARNING: Stack unwind information not available. Following frames may be wrong. También nos muestra la IRP (The I/O request packet) que solicitó. IRP List: **8428cc98: (0006,0094) Flags: 00000000 Mdl: 00000000**

Con lo que tendríamos que llamar a esa IRP para que nos muestre el contenido de esa IRP.

```
kd> !irp 8428cc98 Irp is active with 1 stacks 1 is current (=
0x8428cd08) No Mdl: No System Buffer: Thread 842f6600: Irp stack
trace. cmd flg cl Device File Completion-Context>[ e, 0] 5 0 841d8f18
842b6ad0 00000000-00000000 \Driver\MYFAULT Args: 00000000 00000000
83360018 00000000
```

Ya tenemos a nuestro culpable. El driver MYFAULT.SYS.

También podremos ver los procesos que actualmente corrían en esa máquina antes del casque, con el comando **!process 0 0**. La salida es parecida a esta:

```
kd> !process 0 0**** NT ACTIVE PROCESS DUMP ****PROCESS 843cab98
SessionId: none Cid: 0004 Peb: 00000000 ParentCid: 0000 DirBase:
00039000 ObjectTable: e1000d68 HandleCount: 232. Image: System PROCESS
841d8550 SessionId: none Cid: 0134 Peb: 7ffdf000 ParentCid: 0004
DirBase: 091af000 ObjectTable: e1007790 HandleCount: 21. Image:
smss.exe
```

Pero qué pasa cuando el pantallazo no nos muestra información relevante? Ni muestra driver que cause el error, ni aplicación, ni tipo de error... Nada. Qué hacemos en este caso? Pues lo mismo. J Con una serie de comandos en el depugeador podremos sacar toda la información que necesitemos para un análisis post-mortem.

En este caso vamos a utilizar de nuevo la aplicación NotMyFault. En este caso vamos a marcar el apartado **Stack Trash** y pulsamos en **Do Bug**, con lo que nos debería de salir el siguiente BSOD.

```
A problem has been detected and windows has been shut down to prevent damage to your computer.
```

```
If this is the first time you've seen this Stop error screen, restart your computer. If this screen appears again, follow these steps:
```

```
Check to be sure you have adequate disk space. If a driver is identified in the stop message, disable the driver or check with the manufacturer for driver updates. Try changing video adapters.
```

```
Check with your hardware vendor for any BIOS updates. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select Safe Mode.
```

```
Technical information:
```

```
*** STOP: 0x0000008E (0xc0000005,0x00000000,0xf3c61b8c,0x00000000)
```

```
Beginning dump of physical memory  
Dumping physical memory to disk: 27
```

Sabemos por la misma ayuda que nos brinda el debuggeador, que el código de STOP (0X0000008E) se corresponde al error `KERNEL_MODE_EXCEPTION_NOT_HANDLED`, y que la primera dirección que nos muestra el error (0XC0000005), se corresponde a `STATUS_ACCESS_VIOLATION`. La misma ayuda nos dice que ha habido una violación en el acceso a memoria. Y todo esto con sólo mirar la ayuda!

Cargando el CrashDump en el debuggeador, vemos que este no nos muestra nada a simple vista:

```
BugCheck 8E, {c0000005, 0, f3c61b8c, 0} *** WARNING: Unable to verify checksum for NotMyfault.exe*** ERROR: Module load completed but symbols could not be loaded for NotMyfault.exeProbably caused by : memory_corruption Followup: memory_corruption
```

Nos dice que es un error de tipo 8E y que la causa probable es una corrupción de memoria. Toca analizar con el comando **!analyze -v**

Analizando con este comando, vemos que la salida ya nos va mostrando otras cosas interesantes:

```
EXCEPTION_CODE: (NTSTATUS) 0xc0000005 - La instrucció n en "0x%08lx"
hace referencia a la memoria en "0x%08lx". La memoria no se puede
"%s". FAULTING_IP: +0 00000000 ?? ??? TRAP_FRAME: f3c61b8c - (.trap
fffffffff3c61b8c) ErrCode = 00000000 eax=00000120 ebx=841c53c8
ecx=83360010 edx=83360010 esi=841c53c8 edi=00000000 eip=00000000
esp=f3c61c00 ebp=f3c61c58 iopl=0 nv up ei ng nz na pe nc cs=0008
ss=0010 ds=0023 es=0023 fs=0030 gs=0000 efl=00010286 0008:00000000 ??
???Resetting default scope DEFAULT_BUCKET_ID: CODE_CORRUPTION
BUGCHECK_STR: 0x8E PROCESS_NAME: NotMyfault.exe
```

Nos dice que la causa del cuelgue se podría dar por haber utilizado la aplicación NotMyFault.exe. Pero nuevamente tampoco nos dice nada del driver. Podemos utilizar el comando **!thread**, para que nos muestre el subproceso que logró colgar a nuestro sistema. Llamando a este comando conseguimos más información, como por ejemplo las llamadas que realizó la aplicación al sistema, así como también nos muestra la IRP:

```
THREAD 842f3610 Cid 0118.0328 Teb: 7ffde000 Win32Thread: e1b11968
RUNNING on processor 0IRP List: 841c53b0: (0006,0094) Flags: 00000000
Mdl: 00000000Not impersonatingDeviceMap e183ac28Owning Process
842f2608 Image: NotMyfault.exeWait Start TickCount 12895 Ticks:
0Context Switch Count 557 LargeStack UserTime 00:00:00.0050 KernelTime
00:00:00.0140 Win32 Start Address NotMyfault (0x00401ccb)
```

Llamando al comando **!irp <IRP>**, este nos muestra el driver que nos faltaba por descubrir:

```
kd> !irp 841c53b0Irp is active with 1 stacks 1 is current (=
0x841c5420) No Mdl: No System Buffer: Thread 842f3610: Irp stack
trace. cmd flg cl Device File Completion-Context>[ e, 0] 5 0 842f76a8
84153028 00000000-00000000 *** ERROR: Module load completed but
symbols could not be loaded for myfault.sys \Driver\MYFAULT Args:
00000000 00000000 83360010 00000000kd> !irp 841c53b0Irp is active with
1 stacks 1 is current (= 0x841c5420) No Mdl: No System Buffer: Thread
842f3610: Irp stack trace. cmd flg cl Device File Completion-Context
>[ e, 0] 5 0 842f76a8 84153028 00000000-00000000 \Driver\MYFAULT Args:
00000000 00000000 83360010 00000000
```

Nuevamente tenemos a nuestro culpable!!

Otros comandos que podemos utilizar:

!cpuinfo.- Muestra información acerca del procesador instalado.

```
kd> !cpuinfo CP F/M/S Manufacturer MHz PRCB Signature MSR 8B Signature
Features TargetInfo::ReadMsr is not available in the current debug
session 0 15,2,4 GenuineIntel 1195 0000000d00000000 80073fff Cached
Update Signature 0000002000000000 Initial Update Signature
0000000d00000000
```

!peb.- Válido para comprobar por ejemplo el nombre de equipo, path de instalación, número de procesadores, ruta de instalación, etc...

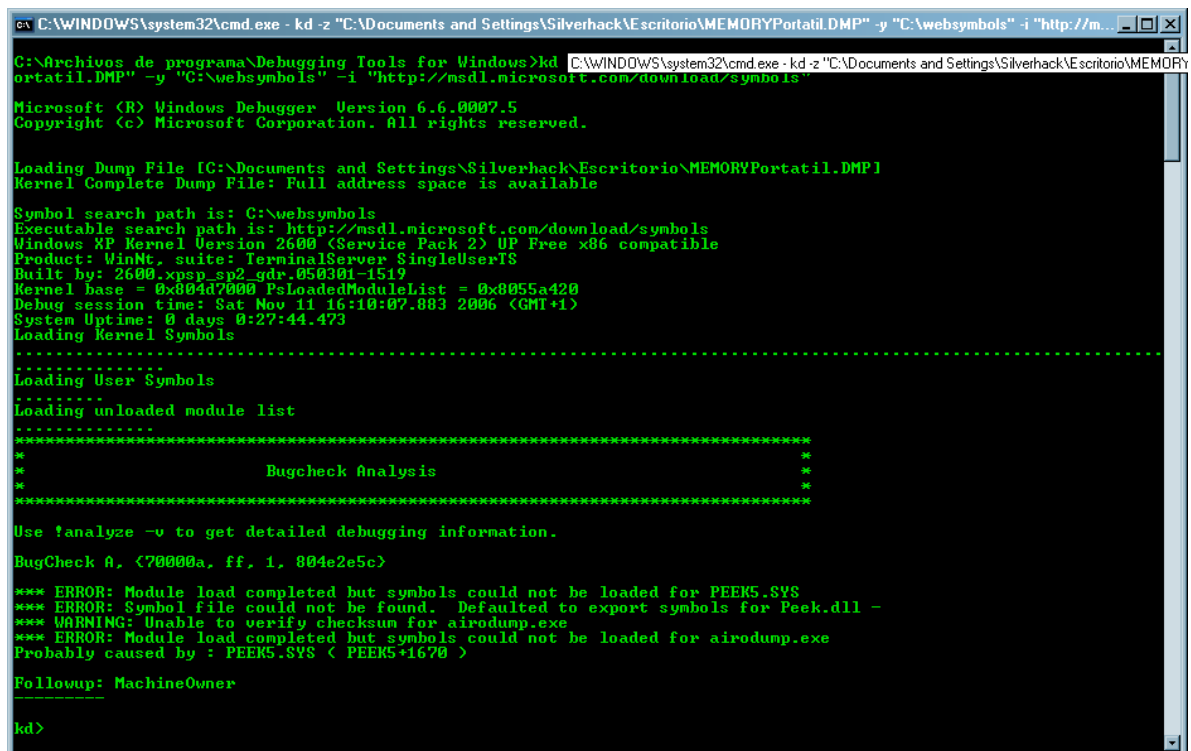
!token.- Muestra información sobre los objetos de seguridad

.cls.- Igual que la shell de Windows. Limpia la pantalla

Se puede analizar desde el entorno de comandos (cmd.exe)??

El debuggeador de Windows viene con una herramienta llamada kd.exe, la cual la podemos invocar desde la shell de Windows. Los comandos son prácticamente los mismos, al igual que la información que nos es mostrada por la shell. La forma de invocar un crashdump desde la shell de Windows es la siguiente:

kd.exe -z “Ruta donde está el volcado de memoria en formato DMP” -y “Ruta donde se encuentran los ficheros de símbolos” -i “Ruta de búsqueda de símbolos”



```
ex C:\WINDOWS\system32\cmd.exe - kd -z "C:\Documents and Settings\Silverhack\Escritorio\MEMORYPortatil.DMP" -y "C:\websymbols" -i "http://m...
C:\Archivos de programa\Debugging Tools for Windows>kd C:\WINDOWS\system32\cmd.exe - kd -z "C:\Documents and Settings\Silverhack\Escritorio\MEMORY
ortatil.DMP" -y "C:\websymbols" -i "http://msdl.microsoft.com/download/symbols"
Microsoft (R) Windows Debugger Version 6.6.0007.5
Copyright (c) Microsoft Corporation. All rights reserved.

Loading Dump File [C:\Documents and Settings\Silverhack\Escritorio\MEMORYPortatil.DMP]
Kernel Complete Dump File: Full address space is available

Symbol search path is: C:\websymbols
Executable search path is: http://msdl.microsoft.com/download/symbols
Windows XP Kernel Version 2600 (Service Pack 2) UP Free x86 compatible
Product: WinNt, suite: TerminalServer SingleUserTS
Built by: 2600.xpsp_sp2_gdr.050301-1519
Kernel base = 0x804d7000 PsLoadedModuleList = 0x8055a420
Debug session time: Sat Nov 11 16:10:07.883 2006 (GMT+1)
System Uptime: 0 days 0:27:44.473
Loading Kernel Symbols

.....
Loading User Symbols
Loading unloaded module list
*****
*
*               Bugcheck Analysis
*
*****
Use !analyze -v to get detailed debugging information.

BugCheck A, (70000a, ff, 1, 804e2e5c)

*** ERROR: Module load completed but symbols could not be loaded for PEEK5.SYS
*** ERROR: Symbol file could not be found.  Defaulted to export symbols for Peek.dll -
*** WARNING: Unable to verify checksum for airodump.exe
*** ERROR: Module load completed but symbols could not be loaded for airodump.exe
Probably caused by : PEEK5.SYS ( PEEK5+1670 )

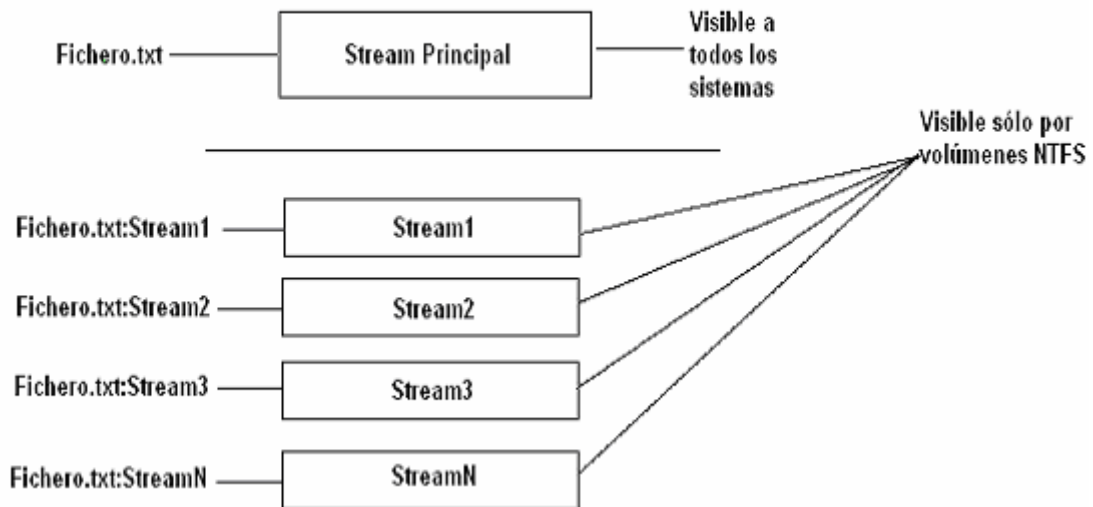
Followup: MachineOwner

kd>
```

Tienes las herramientas para poder hacerlo, y ya sabes cómo enfocar un problema de estas características. Lo único que te queda es practicar!

Alternate Data Streams (ADS). Qué es y como funciona

Hablando desde el punto de vista NTFS, un fichero no es más que un conjunto de data streams. Un stream por ejemplo, puede contener información acerca del nivel de acceso a ese fichero (directivas ACL, permisos, etc..), otro stream contendrá los datos del fichero en sí almacenados en el contenedor. Si el fichero es un acceso directo o link, un stream puede contener la dirección en donde se ubica el fichero original, etc, etc.. Si por ejemplo leyésemos un fichero en un sistema de archivos que no fuese NTFS, ej: FAT16 o FAT32, el sistema leería tan sólo el stream que contiene los datos del fichero. La estructura de un fichero con múltiples streams es parecida a esta:



Aunque muchos piensen que es una falla o Bug del sistema, lo cierto es que ADS no es más que un feature (característica) del sistema, y generalmente se usa para mantener información asociada a un fichero.

Microsoft tiene amplia documentación sobre ADS, en el que se incluye tutoriales, scripts para creación de ADS, etc..

Limitaciones de un ADS

Por defecto cualquier usuario del sistema puede usar esta característica. Tan sólo se limita a aquellos ficheros en los que tengamos permiso de escritura. Es decir, un Administrador, podrá añadir un ADS en prácticamente todos los ficheros del sistema, mientras que un usuario se limitará sólo a los ficheros y directorios en donde tenga acceso de escritura (Por defecto su perfil).

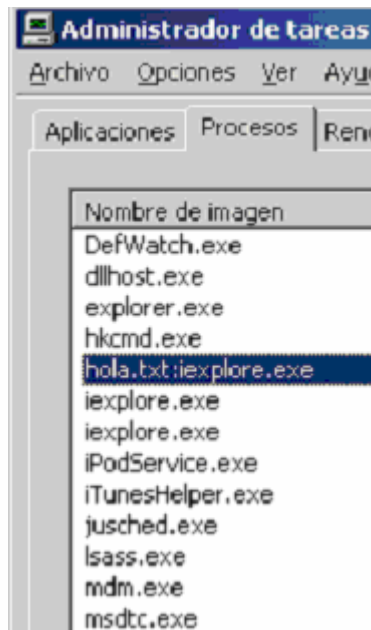
Por defecto ADS sólo está limitado a volúmenes NTFS como hemos visto. A través de red local (LAN) podremos mandar ficheros con ADS, siempre y cuando los volúmenes intermedios tengan el sistema de archivos NTFS.

La limitación teórica es mandar un ADS a través de Internet. Teóricamente no podemos mandar un fichero con ADS (sea malicioso o no), ya que nuestro cliente de correo, el medio (Internet) y el destinatario, sólo mandarían el stream con los datos, sin procesar los demás. A lo largo de este documento podremos ver que en la práctica y bajo una serie de factores, se podría mandar ADS ocultos en un fichero.

No vamos a hablar sobre creación, modificación y eliminación de ADS, porque es una característica ampliamente documentada en la red. Al final de este documento se proporcionarán enlaces a investigaciones pasadas sobre éste tema en concreto.

Microsoft no tiene de forma directa ninguna aplicación para el escaneo ADS. De forma indirecta tenemos dos opciones:

A través del administrador de tareas (taskmgr.exe)



Como se puede comprobar en la imagen adjunta, el Administrador de tareas de XP muestra el proceso ejecutado.

Nota: En versiones anteriores a XP (NT,2K), este proceso de visualización no es del todo transparente. NT y 2K tratan de forma diferente el manejo de los ADS.

A través de la librería StrmExt.dll

Esta librería, añade un nuevo Tab a las propiedades del Explorer, que nos permitirá ver los posibles Streams que pueda tener un archivo, directorio o unidad.

Para añadir un Tab a las propiedades de Explorer descargamos de Microsoft el siguiente código fuente:

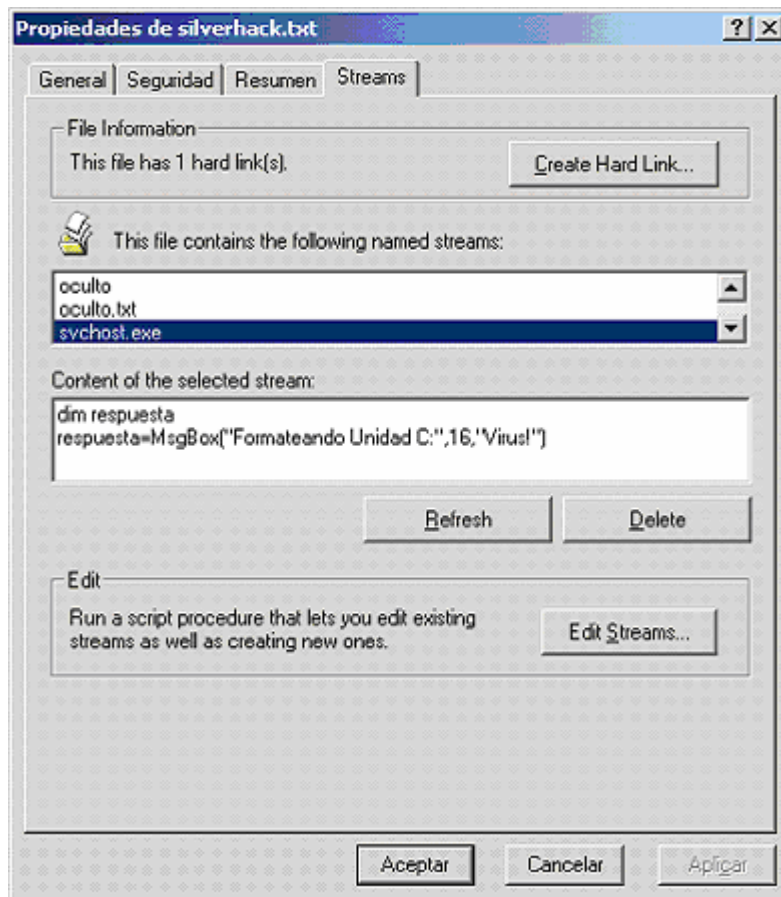
[NtfSext.exe](#)

Dentro nos encontramos con algunos ejemplos de creación de Streams. El fichero que nos interesará es la librería compilada **StrmExt.dll** (**Se adjunta código fuente de la librería**).

La extraemos y la depositaremos en el directorio System32 de nuestro sistema. Acto seguido la registramos en el registro con el siguiente comando:

Regsvr32.exe StrmExt.dll

Una vez ejecutado el comando Windows registrará la librería, y podremos disponer en las propiedades del Explorer de un nuevo Tab para visualizar si un fichero tiene alojado algún Stream.



Registrando esta librería, conseguimos visualizar el Tab Streams sólo para ficheros. Si queremos que nos muestre el Tab Streams para analizar los streams de alguna unidad (ej. C:) o por ejemplo de algún directorio, tendremos que incluir un par de entradas en el registro.

-----No copies esta línea-----

Windows Registry Editor Version 5.00

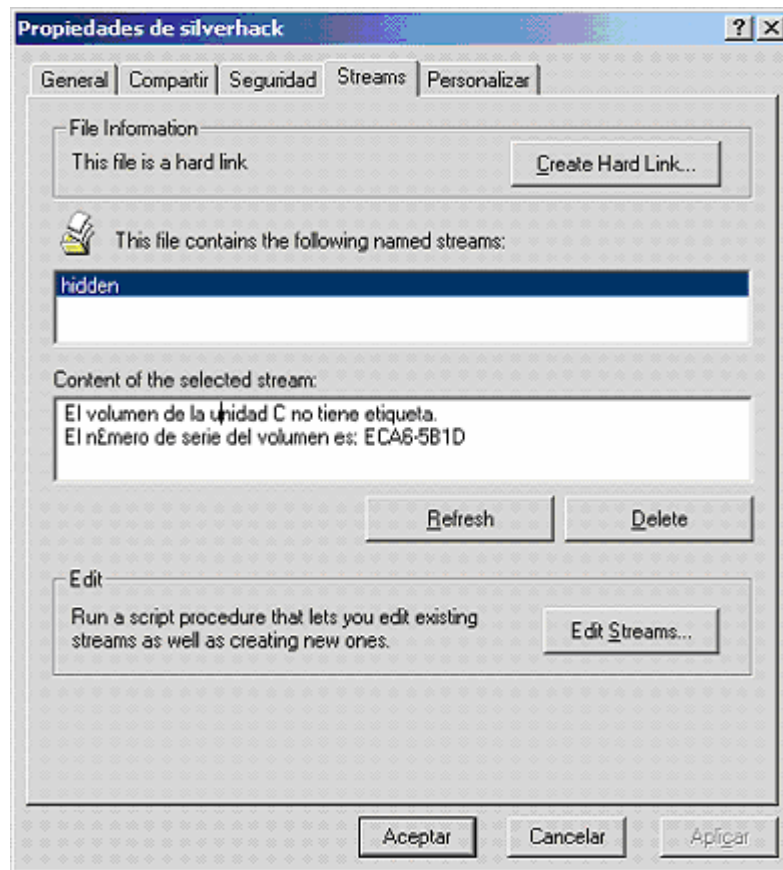
```
[HKEY_CLASSES_ROOT\Directory\shellex\PropertySheetHandlers\{C3ED1679-814B-4DA9-AB00-1CAC71F5E337}]
```

```
[HKEY_CLASSES_ROOT\Drive\shellex\PropertySheetHandlers\{C3ED1679-814B-4DA9-AB00-1CAC71F5E337}]
```

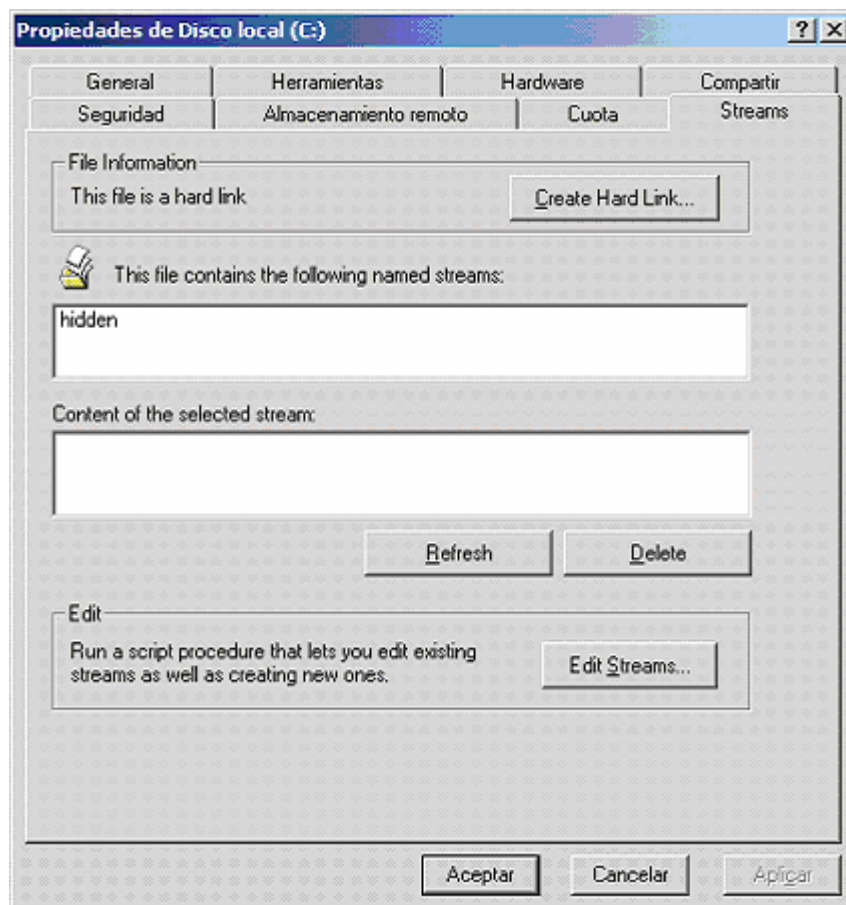
-----No copies esta línea-----

Guarda este fichero como **AddTabStream.reg**. Clickea dos veces en el archivo que has creado y añadirá las entradas al registro necesarias para añadir el Tab en el explorer.

Para un directorio:



Para una unidad:



Escaneo de ADS

Frank Heyne ha desarrollado una herramienta que escanea documentos en busca de ADS. La herramienta en cuestión es LADS.

Su funcionamiento es muy sencillo:

IMAGEN

Si quisiésemos escanear nuestra unidad C: (incluyendo subdirectorios) en busca y captura de ADS, ejecutaríamos un comando tal que:

Lads C:\ /S

Cómo un usuario malicioso podría mandarnos un fichero con ADS? Ejemplo práctico

Hemos explicado antes, que si creamos un archivo con más de un stream de datos (sea malicioso o no), éste sólo será soportado por un sistema que utilice el sistema de archivos NTFS, lo que lo limita sólo a ese campo. No podríamos guardarlo en un pendrive, mandarlo por mail, colgarlo en una Web para su descarga, etc...

El ejemplo que os pongo a continuación, de libre descarga, es una prueba de concepto que demuestra que efectivamente se pueden mandar ficheros que contengan ADS, aprovechando un feature (característica) de una herramienta ampliamente utilizada por usuarios y empresas corporativas. La utilidad de copia de seguridad de Windows

([ntbackup](#)).

Esta herramienta **sí incluye tanto el fichero como sus streams**, lo que la convierte en una gran herramienta para usuarios con perfiles muy distintos:

- Una gran herramienta de copia de seguridad
- Una potencial herramienta para usuarios maliciosos

En el ejemplo (empaquetado en zip) se incluyen estos archivos:

- Un fichero de texto que contiene un stream con código VB inofensivo
- Un archivo Bat para ejecutarlo
- Todo ello va empaquetado en un archivo .bkf (Copia de seguridad)

Sólo tenéis que descargar la copia de seguridad, desempaquetarla en algún directorio y ejecutar el archivo setup.bat.

Es un archivo inofensivo, a modo de broma, que prueba la capacidad de esta técnica, utilizada hoy en día por muchos virus y troyanos.

Un ejemplo de este tipo de virus, y de aparición reciente, lo podemos visualizar aquí:

[Rustock.NAH Descarga otros archivos, utiliza rootkit](#)

Recordad que este riesgo **se minimiza en más de un 90% si navegamos y utilizamos nuestro equipo bajo una cuenta con permisos mínimos**. Este y otros ejemplos parten sobre la base de un usuario medio navegando e iniciando sesión bajo una cuenta administrativa.

[Descarga Ejemplo ADS](#) (Guardar destino como...)

Si queréis saber en mayor profundidad qué es ADS, podéis visitar estas Webs:

[Características desconocidas del NTFS. José Manuel Tella](#)

[A Programmer's Perspective on NTFS 2000 Part 1 by Dino Esposito](#)

[Hidden Threat: Alternate Data Streams by Ray Zadjmool](#)