Análisis forense de la información contenida en los volcados de memoria

Adith Bismarck Pérez O.

Una guía metodológica que permita identificar la evidencia digital presente en los volcados de memoria de diferentes sistemas operativos.

En el desarrollo de la investigación se pretende profundizar en los conceptos de computación forense y cómo afecta esta definición el desarrollo específico de la guía metodológica. Posteriormente, se propone una guía metodológica para realizar correctamente un análisis de los volcados de memoria. Se concluye estableciendo una comparación entre la información rescatada de los volcados de diferentes sistemas Operativos.

La computación forense es un área de investigación orientada a la generación de reglas y estándares, que describan el comportamiento de las herramientas de software y al desarrollo tecnológico asociado a ellas.

El objetivo principal de la computación forense es identificar evidencia digital con el fin de usarla en una investigación que emplea tanto evidencia física como digital, examinada con el método científico, para deducir o concluir un comportamiento en dicha información.

Este artículo examina la existencia de esta evidencia dentro de los volcados de memoria de los sistemas operativos más comunes, Windows y Linux [9]. La investigación provee una guía que contiene los requerimientos y los pasos a seguir para lograr extraer la mayor cantidad de información posible de los diferentes archivos objetos de estudio.

Análisis Forense Digital

El objetivo principal de la computación forense es identificar, analizar y

presentar evidencia digital con el fin de ser usada en una investigación a partir de un estudio basado en el método científico.

El método científico es usualmente definido como el "... uso repetido y objetivo de procesos rigurosos y sistemáticos por medio de los cuales lo que creemos de la realidad es probado contra lo observado, y lo que observamos es examinado a la luz de lo que sabemos..." [3].

La computación forense también incluye la preservación, colección, validación, identificación, análisis, interpretación, documentación y presentación de evidencia recuperada de fuentes digitales (i.e. discos duros, cintas magnéticas, etc.) con el fin de asistir en la reconstrucción posterior de eventos o ayudar a anticipar acciones no autorizadas [1]

Al nivel básico la computación forense está compuesta de 4 etapas [4]: Adquisición, Autenticación, Análisis y Presentación.

La fase de Adquisición se encarga de guardar el estado de un sistema digital de tal manera que la información copiada pueda ser analizada posteriormente. Esta fase es similar a tomar fotos o buscar huellas dactilares en la escena del crimen. El objetivo principal de esta fase es salvar la mayor cantidad posible de información, debido a que en ese instante es imposible conocer la utilidad de la información para el propósito de la investigación. [1] En la presente investigación se pretende recuperar la mayor cantidad de información a partir de los archivos creados por el sistema operativo, esta fase quedara satisfecha en la medida que se pueda recuperar estos archivos [7] (Memory.dmp en Windows, o Core en GNU/Linux).

La fase de Autenticación verifica que la información salvada es precisa y no difiere de la información original, garantizando la consistencia de los datos obtenidos. Esta fase es llevada a cabo con el fin de probar que la información no ha sido modificada durante el proceso de copiado, lo cual la haría inadmisible en una corte. [4]

La fase de Análisis examina la información recolectada para identificar la información más relevante para la investigación. Esta información, denominada evidencia, es clasificada de la siguiente forma [1]:

Evidencia Inculpatoria: es la evidencia que soporta una teoría dada.

Evidencia Exculpatoria: es la evidencia que contradice una teoría dada.

Evidencia de Manipulación: es la evidencia que no puede relacionarse

a ninguna teoría, pero muestra que el sistema fue modificado para evitar la identificación de la persona o sistemas que accedieron a la información.

El método científico es usado aquí para enunciar conclusiones basadas en la evidencia encontrada.

La fase de Presentación se basa enteramente en las políticas o leyes de la entidad a la cual le será presentada la evidencia. Esta fase presenta las conclusiones y la evidencia recolectada de la investigación.

La guía propuesta ofrece elementos que muestran cómo la metodología aplicada para la recolección y explotación de la evidencia son claras, y que el resultado de la investigación es confiable.

Admisibilidad de la evidencia digital

Para que la evidencia sea aceptada en una alta corte debe pasar el "Test de Daubert", el cual consta de cinco categorías básicas enunciadas por Carrier y Ryan así [1], [2], [4]:

Prueba: Verifica que las técnicas y teorías empleadas para efectuar las fases de la investigación forense han sido probadas anteriormente.

Tasa de Error: Verifica y explica si las técnicas y teorías aplicadas para efectuar la investigación forense tienen una rata de error conocida.

Publicación: Verifica que las técnicas y teorías han sido sujetas a revisión científica y subsiguiente publicación.

Aceptación: Verifica que las técnicas y teorías usadas son de uso común en la comunidad científica.

Estandarización: Verifica que las técnicas y teorías aplicadas estén gobernadas por estándares claramente definidos [4].

Pruebas: La directiva de pruebas identifica si la herramienta forense captura la información correctamente (i.e. directorios, nombres de archivos, etc.) y adicionalmente que no introduzca información adicional. Algunas veces la herramienta debe ser probada contra otras herramientas cuyo comportamiento es bien conocido. La información de prueba debe ser creada para cada tipo de herramienta así como la prueba correspondiente que cumpla con los requerimientos de cada herramienta. Esta tarea puede ser difícil pues en general las herramientas forenses primarias son dependientes del sistema operativo en el que son ejecutadas, haciendo la prueba de estas herramientas una tarea bastante compleja [1] [2].

Tasa de Error: Esta directiva identifica si hay alguna tasa de error cono-

cida una vez el proceso se ha llevado a cabo. Carrier [2] ha mostrado dos tasas de error básicas que pueden presentarse en una herramienta forense:

Error de implementación de la Herramienta: El error que viene de fallas o bugs de la especificación de la herramienta o por usar la especificación o versión errónea de la herramienta.

Error de abstracción: El error que viene de la simplificación usada para generar la capa correcta de datos. Una capa de abstracción provee una forma fácil y efectiva de analizar la información. Por ejemplo el formato ASCII es una capa de abstracción, pero decir que los caracteres ASCII juntos forman un documento HTML provee una capa más alta de abstracción de la misma información.

Publicación: La directiva de publicación trata de identificar si el método o técnica ha sido publicado en un medio público. Por ejemplo, el Federal Boreu of Investrigation (FBI), requiere que cada herramienta forense haya sido objeto de publicación en un medio científico y que además sea entregada con anterioridad a los peritos expertos, comisionados por abogados, con el fin de probar la efectividad del proceso forense [1].

Aceptación: La directiva de aceptación está altamente relacionada con la directiva de publicación y consiste en la aceptación general de la herramienta por la comunidad científica y judicial. Para que este procedimiento sea válido debe asumirse que algún medio de comunicación científico ha publicado y probado el procedimiento o la herramienta. Es importante notar que es diferente aprobar o aceptar la herramienta a aprobar o aceptar los procedimientos que esta lleva a cabo. [2]

Estandarización: La directiva de estandarización se relaciona no solo con las directivas de publicación y aceptación sino también con la aprobación de la herramienta o método por entidades federales, científicas o gubernamentales en general (i.e. National Institute of Standards and Technology (NIST), American National Standards Institute (ANSI), etc.).

En estos principios la investigación encuentra otros inconvenientes a superar, debido a la naturaleza no determinista de los objetos de estudio, como son los volcados de memoria. La distribución de los datos dentro de los archivos de prueba no depende directamente de las entradas al sistema, es decir, a pesar de que sea el mismo equipo, ejecutando los mismos procesos, los datos del volcado nunca se repetirán, siempre serán diferentes.

Para realizar los análisis presentados en este documento se ha utilizado una

herramienta conocida y aceptada por la comunidad relacionada como es: WinDBG de Microsft. [10]

Requerimientos de las herramientas de análisis

Se han presentado los requerimientos legales necesarios para el uso y presentación de información forense. Adicionalmente algunos requerimientos técnicos que debe incluir una herramienta forense se han recolectado y presentados por Carrier [1], [2] así:

Usable: La herramienta debe proveer al usuario la información en el nivel correcto de abstracción para que sea útil para el investigador.

Exhaustivo: La herramienta debe proveer un nivel adecuado de abstracción tanto para la evidencia inculpatoria como para la exculpatoria.

Precisión: La herramienta debe asegurar un bajo porcentaje de error para asegurar la precisión de la información que está siendo procesada por ella.

Determinista: Para asegurar la precisión, la herramienta debe siempre entregar los mismos resultados para la misma información de entrada y cada capa de abstracción dadas.

Verificable: Los resultados dados por la herramienta deben poder ser verificados para probar su precisión en cada capa de abstracción que provea la herramienta.

Solo-Lectura: La herramienta debe leer la fuente de la información forense sin modificarla y sin adicionarle información cuando esta es copiada.

Origen de los volcados de memoria

El diseño de los sistemas operativos debe contemplar la posibilidad de permitir que todos los eventos y operaciones que se realicen queden debidamente registrados [9] y pueda hacerse seguimiento de estos sucesos. Para resolver esta problemática, los sistemas operativos poseen un servicio de almacenamiento de eventos en un archivo de bitácora. Sin embargo, en caso de una falla general del sistema, este servicio no podrá cumplir su cometido, [8] de esta manera se originan los volcados de memoria; estos últimos son una copia de diferentes sectores de la memoria a un archivo en el disco duro del equipo correspondiente.

En los sistemas operativos Microsoft Windows, existen tres tipos de volcados: Mini, Kernel, y Full.

El primer tipo de volcado es mini, o de memoria pequeña, ocupa solo

64kb y corresponde a la información de las estructuras del sistema operativo tal como se estaba ejecutando, sin embargo, no almacena la información del sistema operativo como tal, no guarda el estado completo de los registros, ni el contenido de las páginas de la memoria principal (RAM), ni la memoria virtual.

El segundo tipo de volcado, corresponde a la información del volcado pequeño o mini, más la información correspondiente al núcleo del sistema operativo, el estado de los registros, el ciclo de vida de los procesos, y normalmente posee una longitud de 20 Megas.

Finalmente, el tercer tipo de volcado corresponde a el vaciado completo de la memoria RAM y virtual del equipo al archivo plano; es importante mencionar, que cuando se selecciona esta opción el sistema puede tomarse unos minutos para reiniciar debido a que debe copiar toda la información a un archivo, antes de permitir el inicio de las tareas. [7]

En los sistemas operativos Unix y derivados (Solaris, GNU/Linux) existen dos tipos de volcados.

El primero es conocido como “core” este corresponde a la información de una aplicación cuando ocurre un error en su ejecución, estos datos se almacenan en un archivo plano, de nombre core, en el directorio de trabajo correspondiente a la aplicación, este tipo de volcado tiene el inconveniente que si otra aplicación falla se sobrescribe el anterior, por esta razón no se realiza un volcado de todo el sistema, debido a que por diseño de estos sistemas operativos la falla de una aplicación no implica la falla general del sistema.

El segundo tipo de volcado, corresponde a la información completa del sistema operativo, pero este módulo no se instala por defecto en el sistema operativo, para que este tipo de seguimiento funcione es necesario que se modifiquen algunas líneas del kernel del sistema operativo, y posteriormente se recompile y se ejecute, actualmente este comportamiento es soportado hasta la versión 2.5. (no la 2.6 y posteriores, que es la que se instala con las actuales distribuciones). Después de recompilar el kernel

es necesario instalar un paquete de aplicaciones para realizar el análisis deseado. [11] [12]

Como resultado preliminar de esta etapa, la investigación se enfoca en los volcados completos (full) de los sistemas operativos Microsoft Windows. Porque solo en estos tipos de archivos se encuentra la cantidad necesaria y suficiente para realizar un correcto análisis forense.

Requerimientos para analizar un volcado de memoria

En este apartado, se estudian los requerimientos para poder realizar análisis forense en un volcado de memoria.

En su naturaleza más intrínseca los volcados de memoria corresponden a una representación en binario de la información contenida en la memoria RAM y virtual del sistema operativo, por tal motivo el estudio de patrones dentro de estos archivos corresponde a una labor compleja.

Para poder entender que contiene un volcado de memoria, en los sistemas operativos Microsoft Windows es necesario instalar los Symbols de las aplicaciones que se ejecutan en el equipo.

Los Symbols son archivos adicionales que son creados cuando se compila una librería, un controlador o una aplicación.

Los Symbols son archivos que contienen la información relativa a la distribución de datos en memoria de una aplicación; esta variedad de datos no son imprescindibles para ejecutar la aplicación, pero es información útil en el proceso de depuración.

Típicamente un archivo de Symbol puede contener: variables globales, variables locales, nombres de las funciones, direcciones de los puntos de entrada de las funciones, el número de líneas de código fuente.

Los archivos binarios pueden resultar más pequeños y rápidos si el resultado de estos symbols se mantiene por separado. Sin embargo, esto significa que para poder realizar el seguimiento de la aplicación es necesario conseguir la representación de todos y cada una de las aplicaciones que se pretendan vigilar.

Para poder descifrar el contenido de los volcados de memoria es imperativo obtener los símbolos correspondientes. Estos archivos se pueden conseguir en la página de Microsoft:

http://www.microsoft.com/whdc/devtools/debugging/debugstart.mspx

Es muy importante mencionar que cada sistema operativo posee su propia representación de las aplica-

ciones, es decir que posee su propio conjunto de Symbols. Por tal razón, se debe poseer el conjunto de symbols correspondientes al sistema operativo y service pack instalado, de la maquina que genero el volcado de memoria para poder realizarlo.

Después de instalar el conjunto de symbols necesario, se debe disponer de una herramienta para interpretar los volcados de memoria, en este caso se ha utilizado el depurador windbg, es cual esta disponible en el sitio web de Microsoft.

El Depurador Windbg

En este punto se analiza el depurador windbg como herramienta para interpretar el contenido del archivo de volcado de memoria, y se resalta la información que se puede extraer. [10]

Windbg es una herramienta poderosa para la depuración de los sistemas

operativos Microsoft Windows, después de configurarla apropiadamente, se puede cargar el archivo del volcado de memoria que se desea y se le aplicaran los siguientes comandos:

!vm: muestra las aplicaciones que estaban utilizando memoria virtual y su correspondiente distribución. Gracias a este comando, se tienen los datos de: memoria física total, nombre del archivo de paginación, tamaño actual del archivo de paginación, espacio libre disponible, tamaño mínimo y máximo del archivo de paginación, paginas disponibles para nueva asignación, y un listado con el nombre del proceso y la cantidad de páginas que estaba utilizando.

!memusage: resume la distribución por páginas del uso y asignación de memoria. A partir de este comando se puede determinar ¿qué porcentaje de la memoria RAM se estaba utilizando? ¿Cuántas páginas se estaban utilizando? ¿Cuántas páginas estaban libres? ¿Cuántas se encontraban en espera de ser guardas en disco? ¿Cuántas paginas se habían modificado? ¿Cuántas páginas se modificaron y no se escribieron en disco? ¿Cuántas páginas se encontraban en transición al disco? y ¿Cuántas se encontraban en un estado desconocido?.

!dlls: permite ver las dlls que se encontraban cargadas en memoria. Se tiene el nombre de la dll, su dirección

base en memoria, la dirección de la función de entrada para atender la interrupción correspondiente, y el tamaño de la dll.

!imgreloc: reporta la dirección base de cada aplicación en la memoria RAM.

!process 0 7: elabora un listado de los procesos que se encontraban en ejecución al momento de la falla y los datos asociados a estos, este reporte permitiría generar el árbol de procesos existentes. Entre otros datos se puede obtener: el nombre del proceso, su dirección de ubicación en memoria, su dirección base, el tamaño que ocupa, y el identificador del padre.

!errlog: muestra el contenido del reporte de errores del sistema operativo antes de la falla que ocasiono la caída del sistema.

!analize: realiza una búsqueda de incongruencias en el archivo y sugiere cuales pueden ser las causas de la falla. Sin embargo, el análisis realizado no contempla incongruencias en la asignación de memoria, se limita a identificar las aplicaciones que tienen Symbol asignado.

Propuesta del análisis

Con los resultados obtenidos por el windbg, se pretende inferir el comportamiento del sistema operativo antes de la falla y poder determinar si existiese alguna aplicación maligna o virus en ejecución.

Con base en los resultados obtenidos de realizar el análisis con windbg, se construye una relación triangular entre los procesos, la memoria asignada, y los recursos asignados a cada proceso.

Se determina entonces, si la sumatoria de la memoria asignada a los procesos corresponde exactamente, al reporte del uso de memoria. Sí existe una incongruencia, se puede afirmar que en ese sector de memoria es posible que exista algún código malicioso. [5] Posteriormente, se revisa esta sección de memoria y se identifica que aplicación o dll a causado la falla.

También, debe revisarse las direcciones correspondientes a las dlls que se encuentren cargadas en memoria. Se debe observar si las direcciones de las librerías winsock, win32 o kernel32 corresponden a las características de ellas. Si no es así, es posible, que se este ejecutando un shellcode en nuestro ordenador. [6][14]

Análisis de resultados

Durante el desarrollo de la investigación se realizaron pruebas con 9 volcados de memoria de un equipo

Windows 2000 profesional, con service pack 2 instalado.

Se siguieron los pasos metodológicos planteados, se obtuvieron resultados satisfactorios en todos los casos. Efectivamente, Se pudo determinar que aplicación había ocasionado el fallo general del sistema.

Sin embargo, los resultados aun son susceptibles de mejorar, debido a que en la investigación no se cubre la infección del equipo con virus. En caso de infectar el equipo, es necesario determinar el patrón de código malicioso para poder detectarlo en la sección de memoria identificada.

Con este artículo, se busca promover la investigación específica en esta área del conocimiento, porque se demuestra que aun existen múltiples posibilidades de estudio; permitiendo profundizar en diferentes aspectos como en informática forense, herramientas de depuración y análisis, automatización del proceso de análisis, etc.

Conclusiones

Se han presentado algunas definiciones y directivas para evaluar la validez de herramientas de análisis forense dentro de un marco legal. También se ha presentado algunas características técnicas importantes que el software de investigación forense digital debe poseer para ser útil en un proceso investigativo, y como se relacionan con los volcados de memoria de los sistemas operativos.

Se han determinado los requerimientos para realizar análisis forense sobre los volcados de memoria en los sistemas Microsoft Windows y se ha entregado una guía general para el

proceso de instalación y configuración de las herramientas necesarias.

Se ha establecido el contenido de los volcados de memoria, y estipulado un plan a seguir para encontrar patrones, incoherencias, tendencias y relaciones existentes entre los rastros dejados en los volcados de memoria por los sistemas operativos, como son: procesos, memoria, recursos de hardware, y dispositivos de entrada y salida.

Se estableció que en la actualidad los sistemas operativos GNU/Linux no brindan la posibilidad de generar volcados de memoria acordes con las necesidades de una investigación forense. Lo cual constituye un grave riesgo a los indicadores de seguridad informática y a la trazabilidad de los sucesos.

Referencias

[1] B. Carrier, Defining Digital Forensics Examination and Analysis Tools. 2002. Disponible en: http://www.dfrws.org/2002/papers/Papers/Brian_carrier.pdf (Última consulta: Agosto de 2005)

[2] B. Carrier. Open Source Digital Forensic Tools. Disponible en: www.digital-evidence.org/papers/opensrc_legal.pdf (Última consulta: Agosto 2005)

Computer Forensics Tool Testing, Disponible en: http://www.cftt.nist.gov/ (Última consulta: Agosto 2005)

[3] D.J. Ryan, Legal Aspects of Digital Forensics, Disponible en: http://www.danjryan.com/papers.htm (Última consulta Agosto 2005)

[4]Burger Doug. Memory Systems. ACM Computing Survey

[5]Wolak Marcin, Hackin 9 N°4 2004. Exploit Remoto para el Sistema Windows 2000

[6]Stuhl Hill, Demystifying Blue Screens and Crash Dump Analisys. revista Exploring Windows NT

[7] Knuth Donald, El Arte de Programar Ordenadores. Addison-Wesley, 1997

[8]Stallings William. Sistemas operativos. Person, Prentice Hall. 4 edición 2001.

[9]http://www.microsoft.com/whdc/devtools/debugging/debugstart.mspx, (Última consulta octubre 2005)Debugging Tools for Windows. Microsoft Windows.

[10] http://lkcd.sourceforge.net/ (Última consulta octubre 2005)

[11] http://lkcd.sourceforge.net/doc/lkcd_tutorial.pdf (Última consulta octubre 2005)

[12]http://www.smidgeonsoft.prohosting.com/pebrowse-crash-dump-analyzer.html (Última consulta noviembre 2005)

[13] Wolak Marcin,Hackin9. N1° 2 2004 Escribimos el shellcode para los sistemas MS Windows.

***Adith Bismarck Pérez O**. Ingeniero de Sistemas, Pontificia Universidad Javeriana; actualmente se desempeña como profesor de tiempo completo del departamento de Ingeniería de Sistemas de la misma universidad; ha estado encargado de las cátedras de Sistemas Operativos, Sistemas Basados en el Conocimiento, Estructuras de Datos, Programación Orientada a Objetos además de coordinar las Practicas Profesionales y Proyectos de Grado. Realiza su Proyecto de grado de Maestría en el área de Inteligencia Artificial.*