El Estado Actual de las Técnicas de File Carving y la Necesidad de Nuevas Tecnologías que Implementen Carving Inteligente

Bruno Constanzo[1], Julian Waimann[2]

[1] Técnico en Informática. Auxiliar de Investigación Alumno, Facultad de Ingeniería de la Universidad FASTA. bru.constanzo@gmail.com

[2] Analista en Informática. Auxiliar de Investigación Alumno, Facultad de Ingeniería de la Universidad FASTA. Desarrollador de Software, Making Sense. julianw@ufasta.edu.ar

Abstract. El *file carving* es una técnica que permite lograr la recuperación de información en la ausencia de metadatos de un sistema de archivos. Pese a su comprobada eficacia, aún presenta problemáticas y aspectos que pueden optimizarse. El proyecto Carving Inteligente Aplicado a Recuperación de Archivos (CIRA) busca investigar distintos algoritmos de file carving, reunir y formalizar ese conocimiento, para en una etapa posterior desarrollar una herramienta de carving. En este documento se presenta CIRA y los avances logrados hasta el momento.

Keywords: Informática forense – Recuperación de la información – File carving.

1 Introducción

En la actualidad, y de forma cada vez más acentuada, la información juega un rol fundamental en la vida diaria. En actividades variadas intervienen sistemas informáticos manejando, administrando y procesando datos. De esta dependencia, surge la necesidad de contar con medios para que dichos datos puedan ser recuperados en caso de pérdida.

A bajo nivel, los datos se almacenan utilizando alguna cualidad eléctrica, magnética u óptica de un material para representar un estado, 0 o 1. A esa unidad elemental de información la llamamos bit, y los datos se representan como secuencias de bits que toman significado de acuerdo a como las interpretemos. Sin embargo, no resulta práctico que los programas, programadores y usuarios de computadoras tengan que trabajar a ese nivel, y por eso los sistemas operativos brindan abstracciones que permiten un trabajo independiente del *hardware* y de la forma en que físicamente se almacena la información. Los sistemas de archivos son una serie de reglas y estructuras que permiten a un sistema operativo ordenar y organizar cómo se

almacena la información en un dispositivo de almacenamiento, y funcionan en conjunto con los *drivers* de dispositivo para brindarnos una forma accesible de almacenar y acceder a nuestros datos: los archivos. Los sistemas de archivo almacenan información sobre los archivos que contienen, a saber: qué bloques del dispositivo ocupan, fecha de creación y acceso, entre otros datos. Toda esta información administrativa se conoce con el nombre de *metadatos*.

En este contexto, se requiere de una técnica que permita recuperar archivos de un dispositivo en el que los metadatos del sistema de archivos no se encuentren disponibles, situación que puede darse cuando el dispositivo resulta dañado, pero permanece funcional, cuando se dañan las particiones a un nivel lógico sin daño físico, o cuando se eliminan archivos accidental o intencionalmente.

2 File Carving: conceptos y algoritmos

El *Digital Forensics Research Workshop* [3] ha propuesto una definición formal de *File Carving*, que se presenta a continuación:

> *Es el proceso de extraer una colección de datos de un conjunto de datos más grande. Las técnicas de carving frecuentemente se utilizan en una investigación forense cuando se analiza el espacio no asignado de un sistema de archivos para extraer archivos. Las estructuras del sistema de archivos no se utilizan durante el proceso.*

Se puede ampliar esta definición con aspectos discutidos en [4] donde se indica que el proceso se basa en características específicas de los formatos de archivo. Dado que es usual trabajar con imágenes de los dispositivos de almacenamiento a analizar, la definición debería poder aplicarse tanto a un dispositivo físico como a una imagen. Entonces se dirá sobre el *file carving*:

> *Es el proceso de extraer archivos de un dispositivo de almacenamiento, analizando el contenido de sus bloques, teniendo en cuenta características específicas de los formatos de archivo, e ignorando las estructuras del sistema de archivos.*

Esta definición da un buen punto de partida, aunque más adelante se verá que, debido a la escala del problema de *carving*, es posible no respetarla estrictamente para lograr mejores resultados o mejor performance.

2.1 Origen y evolución

Para entender el origen del *file carving,* puede considerarse el proceso de eliminación de archivos en un sistema de archivos FAT[1]. Al eliminar un archivo se sobrescriben los punteros a los *clusters* (siendo un clúster la unidad más pequeña de almacenamiento de un sistema de archivos) del archivo con el valor hexadecimal "00", para indicar que están disponibles. La entrada de directorio del archivo todavía apunta al primer *clúster* del archivo, aún luego de reinicializar los enlaces. Finalmente, modifica el primer caracter del nombre de archivo para indicar que corresponde a un archivo eliminado.
Otros sistemas de archivos realizan acciones distintas al eliminar un archivo, sin embargo hay una característica común a varios de ellos, y es que en ningún momento se sobrescribe el contenido de los *clusters*, ya que esto tendría un impacto negativo en la performance por el tiempo requerido para realizar una escritura en el dispositivo. Los *clusters* correspondientes a un archivo eliminado conservan su contenido, pero se marcan como disponibles en la tabla de asignación, o la estructura pertinente del sistema de archivos. Los contenidos solamente se perderán cuando los *clusters* se asignen y se los ocupe con el contenido de un nuevo archivo.
Un caso particular es el de los discos de estado sólido con *garbage collection*, una tecnología que permite mantener el rendimiento del dispositivo reinicializando el contenido de los bloques de archivos eliminados. Esta característica sólo funciona en sistemas operativos que soportan el comando TRIM[2].
Haciendo un análisis histórico, la posibilidad de recuperar en un dispositivo de almacenamiento aquellos archivos que estuvieran desvinculados de los metadatos que los representaban en el sistema de archivos fue aprovechada en el año 1999 por el Laboratorio de Informática Forense del Departamento de Defensa de Estados Unidos (*Defense Computer Forensics Lab*), con el programa CarvThis. Una serie de trabajos sucesivos, fomentados por éste trabajo inicial, resultaron en el desarrollo de Foremost por la Oficina de Investigaciones Especiales de la Fuerza Aérea del mismo país, (*US Air Force Office of Special Investigations*), una herramienta *open source* para realizar *file carving*. En un principio, Foremost implementaba únicamente *header/footer carving*, pero en el año 2005 se extendió su funcionalidad para trabajar con la estructura interna de los archivos.
También en el año 2005, Richard y Roussev reimplementaron Foremost con una nueva base de código, creando Scalpel, un *carver* enfocado en la velocidad de procesamiento y el bajo consumo de recursos. Scalpel logró ubicarse como una herramienta de referencia en el ámbito forense. En el año 2011, con el *release* de Scalpel 2.0, se agregó la capacidad de trabajar en multiprocesadores y procesadores gráficos de propósito general (GPGPUs), incrementando su rendimiento.
Paralelamente, existen dos ramas de investigación importantes para el file carving. Por un lado, en el año 2007 Garfinkel realizó una incursión en la temática. Su trabajo consistió en desarrollar un nuevo algoritmo, basado en las técnicas de *header/footer carving*, pero más complejo y capaz de recuperar archivos fragmentados, bajo condiciones específicas pero estadísticamente relevantes.

Por otro lado, comenzando en el año 2003, Memon, Shanmugasundaram y Pal comenzaron a presentar trabajos describiendo algoritmos para recuperar archivos fragmentados considerando al *carving* como un problema de grafos. Su trabajo resultó en algoritmos que fueron implementados en el llamado *Smart Carving*™. Las características del *Smart Carving* hacen que sea una técnica altamente eficaz, y capaz de recuperar archivos fragmentados en diversas condiciones. Además, se separa el proceso de *carving* en etapas, lo que brinda flexibilidad y permite aplicar diversas optimizaciones para la performance.
Pese a todos estos avances, en la actualidad hay pocas herramientas que implementen los algoritmos de *carving* avanzado. Muchas de ellas implementan variantes del *header/footer carving*, la técnica más básica. Por otro lado, las herramientas que implementan algoritmos avanzados en pocos casos dejaron la etapa teórica para convertirse en programas maduros y usables en un entorno forense. Además, aún está presente el problema de los falsos positivos que presentan algunos algoritmos.
Debido a estos problemas y a las falencias presentes en las herramientas, es oportuno realizar el análisis y la investigación de los algoritmos y técnicas de *carving* para hacer un aporte que tenga tanto un valor académico como de aplicación real.

2.2 Algoritmos de carving

Puede realizarse una clasificación inicial de los algoritmos de *carving* en:

- *Carving* básico: se clasificará así los algoritmos que solamente pueden recuperar archivos contiguos en el disco o imagen, y que tampoco pueden trabajar con archivos comprimidos. La mayoría de los algoritmos clasificados en ésta categoría son variantes del *header/footer carving*, la técnica más básica.
- *Carving* avanzado: se clasificará así a los algoritmos que pueden recuperar archivos fragmentados, posiblemente con fragmentos desordenados o faltantes. Algunos ejemplos de carving avanzado son el *Bifragment Gap Carving, Graph Theoretic Carving o Smart Carving*™.

A continuación se presenta un listado de algoritmos de *carving*, según [6] y [7]:

- *Header/Footer Carving*: se busca entre los bloques del dispositivo un *header* que indica el comienzo de un archivo de determinado tipo. Luego se añaden en secuencia los bytes contiguos hasta encontrar el *footer* que corresponde a ese tipo de archivo.
    - *Header/Maximum Size Carving*: en ocasiones no se encuentra presente el *footer*, o el tipo de archivo no tiene *footer*, pero se puede recuperar un archivo válido o parcialmente recuperado haciendo un corte cuando se alcanza un tamaño determinado en el archivo.

- *Header/Embedded Length Carving:* algunos tipos de archivos contienen en su cabecera un campo que indica la longitud del archivo. Para estos archivos puede utilizarse un *carver* que analice ese campo y decida en función de él cuántos bytes añadir a partir del *header* detectado.

- *File Structure Based Carving*: una evolución del anterior, se analizan las estructuras internas del tipo de archivo además de las que se encuentran en la cabecera. Especialmente útil en formatos que guardan sus datos en bloques, como AVI, MP3, PNG, entre otros.

- *Semantic Carving*: identifica el idioma utilizado en un bloque, para poder relacionarlo con bloques en el mismo idioma. Adicionalmente, un *semantic carver* debería relacionar los bloques en forma coherente para reconstruir un texto. Especialmente útil para formatos que almacenan texto. S. Garfinkel ha presentado, en DFRWS 2006, un *carver* semántico llamado S2, escrito en C++, que permite incorporar distintos tipos de *carvers* y validadores para tipos de archivo específicos.

- *Fragment Recovery Carving*: se encuentran dos (o más) fragmentos de un archivo, y se descartan los bloques intermedios a los fragmentos que no pertenecen al mismo.
    - *Bifragment Gap Carving*: variante enfocada en los archivos divididos en dos fragmentos que tienen bloques de otros archivos entre ellos. Trabaja con validadores rápidos como parte del proceso y es altamente eficaz.

- *Statistical Carving*: se calcula una función matemática de los bloques buscando similitudes entre ellos para reconstruir los archivos. Algunas variantes usan funciones estadísticas, hashes, y/o *signatures*, entre otras.

- *Smart Carving*™: es una técnica de *carving* basada en grafos que separa el proceso de *carving* en etapas definidas. Es una de las técnicas más eficaces, capaz de recuperar archivos fragmentados. Intentar reproducirla es uno de los elementos motivadores del proyecto. La primer etapa pre-procesa los bloques, los descomprime y quita los bloques pertenecientes a archivos conocidos. La segunda etapa clasifica los bloques restantes de acuerdo a su tipo de archivo, ubicándolos en uno o más grafos de reconstrucción. Finalmente la tercera etapa recorre los grafos buscando cómo rearmar los archivos.

Además de los algoritmos enfocados en la reconstrucción de los archivos, hay técnicas y algoritmos que se utilizan para acelerar y mejorar los resultados del *carving*:

- Identificación de archivos conocidos: se aplica un hash a los bloques del dispositivo analizado y se comparan con los hashes de archivos conocidos (archivos de configuración de los sistemas operativos, por ejemplo). Cuando se identifica un bloque conocido, se lo quita del conjunto de bloques a analizar.
- Identificación del espacio no asignado: si el dispositivo analizado tiene un sistema de archivos conocido, pueden analizarse sus metadatos para dejar en el conjunto de bloques a analizar sólo aquellos que corresponden a espacio no asignado.
- Carving con validación: se utilizan visores o validadores que comprueban la estructura de los archivos y verifican que respete el formato al que pertenece. Puede utilizarse tanto durante la etapa de reconstrucción de archivos como en una etapa posterior para filtrar falsos positivos [7].
- *Repackaging Carving*: se utiliza cuando hay archivos parcialmente recuperados. Esta técnica agrega bloques al archivo para obtener un archivo válido. Aunque el archivo resultante no es idéntico al archivo original, permite recuperar partes del mismo [5].
- *In-place File Carving*: en lugar de extraer los archivos de la imagen de disco analizada, se crean metadatos nuevos que referencian a los bloques presentes en la imagen. Utilizando un sistema de archivos virtual se accede a los archivos como si se hubieran extraído las copias de la imagen. Esta técnica mejora la performance de la etapa de extracción, ya que no se realiza o se realiza diferida, y reduce el costo en espacio de almacenamiento que acarrea el uso de un *file carver* [8].

2.3 Métricas para el File Carving

Hay una variedad de métricas que se pueden utilizar para evaluar el rendimiento de un file *carver*, su velocidad de procesamiento y la calidad de sus resultados. Algunas de ellas son:

- Cantidad de archivos recuperados.
- Cantidad de archivos válidos recuperados.
- Cantidad de archivos parcialmente recuperados.
- Cantidad de archivos no recuperados.[1]
- Cantidad de falsos positivos.
- Precisión de *carving*: del total de los archivos recuperados, cuántos son válidos o relevantes.

---

1 Estas métricas sólo pueden considerarse cuando se trabaja en imágenes pre-armadas, o que se han analizado profundamente y se conoce todo su contenido.

- Carving *recall*: del total de archivos relevantes presentes en el dispositivo, cuántos se han recuperado.[1]
- *Overcarving*: es la relación entre el tamaño del dispositivo analizado y el tamaño de los archivos recuperados. Idealmente debería ser un número <= 1, aunque usualmente es mayor.

De estas métricas se busca obtener un *recall* igual a 1, una precisión cercana a 1 y un *overcarving* lo más pequeño posible. La métrica más importante es el *recall*, ya que es la que asegura recuperar la totalidad de los archivos presentes en el dispositivo. Además, los inconvenientes asociados con tener precisión baja u *overcarving* alto pueden ser contrarrestados con la aplicación de técnicas complementarias al *carving*, como *In-place File Carving*.

2.4 .Beneficios y limitaciones de las herramientas actuales

Entre los beneficios que puede brindar una herramienta de *file carving*, de acuerdo con [9], podemos mencionar:

- Identificar y recuperar archivos de interés, que hayan sido borrados, o se encuentren en un sistema de archivos dañado, en memoria, o información que se encuentre en el archivo de paginación o tráfico de red.

- Asistir en un proceso forense de recuperación de archivos y datos, así como en un proceso normal de recuperación de datos.

Las herramientas actuales se encuentran sujetas a una serie de limitaciones, a saber:

La mayoría de ellas solo permiten recuperar archivos que no se encuentran fragmentados, producen un gran número de falsos positivos, solo permiten recuperar el contenido de los archivos pero no sus metadatos ni la estructura de directorios, pueden ser fácilmente engañadas con el uso de técnicas anti-forenses, y el proceso de recuperación es muy lento y requiere mucho espacio. A todo ello se suma la inexistencia de técnicas estandarizadas.

3 El Proyecto

El proyecto CIRA, Carving Inteligente para la Recuperación de Archivos, surge como una respuesta a las deficiencias que se pueden encontrar aún hoy en las herramientas de *file carving*, buscando realizar un aporte al área. Se habla de "*carving* inteligente" porque el foco está puesto en las técnicas más avanzadas, y de "recuperación de archivos" porque se buscará minimizar la ocurrencia de falsos positivos cuando llegue el momento de implementar un prototipo de herramienta.

El proyecto se divide en dos etapas, una dedicada a la investigación y otra dedicada al desarrollo de un prototipo.
El objetivo de la etapa de investigación es reunir, aprender y formalizar el conocimiento relacionado al *carving* disponible en el dominio a través de publicaciones y libros.
Comienza con la investigación de técnicas básicas de carving, para luego ir progresando hacia los algoritmos más complejos. También se complementa con una investigación sobre la forma de operar de filesystems de uso extendido. Esta etapa culmina con dos documentos: un documento de formalización del conocimiento adquirido y un estudio de factibilidad para el desarrollo de una herramienta prototipo.
El objetivo de la etapa de desarrollo es programar y documentar el prototipo de una herramienta moderna, flexible y extensible, que implemente uno o más de los algoritmos investigados.
Debido a que depende directamente de la etapa de investigación, aún no se ha definido completamente qué forma va a tomar la herramienta, aunque se cuenta con dos propuestas tentativas. La primera de ellas consiste en desarrollar un programa que implemente una técnica avanzada, probablemente *semantic carving* o *carving* basado en grafos, similar al *Smart Carving*™. La otra propuesta es desarrollar un *framework* que organice el proceso de *carving* en etapas definidas y que sea capaz de guiar e implementar una variedad de algoritmos, y que brinde la flexibilidad de extender el proceso para poder adaptarlo a las necesidades de casos particulares.

4 Conclusiones y Trabajo Futuro

Se presentó el conocimiento adquirido hasta el momento en el desarrollo del proyecto. Al momento de escribir este trabajo, se está comenzando con la segunda instancia de la etapa de investigación, donde se profundizará el análisis sobre los algoritmos y técnicas de *carving* presentadas. Además del trabajo de investigación, comienza el desarrollo de un prototipo de herramienta capaz de realizar *header/footer carving*.
En lo que resta de la etapa actual se continúa con la investigación de los algoritmos, filesystems y tipos de archivo, para poder incorporar funcionalidades a la herramienta que se debe desarrollar en la siguiente etapa del proyecto. Al finalizar la etapa de investigación, se definirá qué tipo de herramienta se desarrollará en base al estudio de factibilidad.
El proyecto está orientado a brindar una mejora para el *file carving* en general, y los avances logrados hasta el momento sobre la bibliografía que se utiliza para la investigación dan una buena perspectiva para la etapa de desarrollo.

Agradecimientos

Nuestro agradecimiento a los ingenieros Ana Di Iorio, Fernando Greco y Sebastián Sznur por sus aportes y opiniones que ayudaron a mejorar este trabajo.

Referencias

1. Pal, A., Memon, N.: The Evolution of File Carving. IEEE Signal Processing Magazine, (2009) 59 – 71
2. Bell, G., Boddington, R.: Solid State Drives: The Beginning of the End for Current Practice in Digital Forensic Recovery? The Journal of Digital Forensics, Security and Law, Vol. 5(3), (2010).
3. Merola A.: Data Carving Concepts, SANS Institute (2008) 4
4. Kloet, B.: Advanced file carving How much evidence are you ignoring?,Hoffmann Investigations (2010)
5. Zirnstein, R.: Advances in file carving. American Society of Digital Forensics & eDiscovery -ASDFED (2011)
6. Hulin, K.: Digital Forensics III - File Carving, Department of Computer Science, University of Texas at Dallas (2011)
7. Garfinkel, S.: Carving contiguous and fragmented files with fast object validation. Digital Forensics Research Workshop - DFRWS (2007)
8. Richard, G., Roussev, V., Marziale, L.: In-place File Carving. International Federation for Information Processing - IFIP (2007)
9. Carreño, J., Torres, D.: Recuperación de Datos: Data Carving y Archivos Fragmentados. VII Jornada Nacional de Seguridad Informática en Bogotá, Colombia (2007)