UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS

"DESARROLLO DE UN SOFTWARE DE ANÁLISIS FORENSE PARA LA IDENTIFICACIÓN, RECOLECCIÓN Y CLASIFICACIÓN DE EVIDENCIA DIGITAL"

TESIS

Que para obtener el título de:

Licenciado en Ciencias Computacionales

Presenta:

CARLOS RAMÓN VEGA GONZÁLEZ

Ensenada, Baja California, México. **Febrero del 2009.**

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
FACULTAD DE CIENCIAS

DESARROLLO DE UN SOFTWARE DE ANÁLISIS FORENSE PARA LA IDENTIFICACIÓN, RECOLECCIÓN Y CLASIFICACIÓN DE EVIDENCIA DIGITAL

TESIS PROFESIONAL

QUE PRESENTA

CARLOS RAMÓN VEGA GONZÁLEZ

APROBADO POR:

M.C. CARLOS GONZÁLEZ SÁNCHEZ
Presidente

M.C. EVELIO MARTÍNEZ MARTÍNEZ
Sinodal

M.C. ADRIÁN VÁZQUEZ OSORIO
Sinodal

M.C. ADÁN HIRALES CARBAJAL
Sinodal

M.I. ADRIÁN ENCISO ALMANZA
Secretario

Resumen de la Tesis de Carlos Ramón Vega González presentada como requisito parcial para la obtención de la Licenciatura en Ciencias Computacionales. Ensenada, Baja California, México. Febrero del 2009.

DESARROLLO DE UN SOFTWARE DE ANÁLISIS FORENSE PARA LA IDENTIFICACIÓN, RECOLECCIÓN Y CLASIFICACIÓN DE EVIDENCIA DIGITAL

Resumen aprobado por:

M.C. Carlos González Sánchez
Director de Tesis

M.C. Evelio Martínez Martínez
Co-Director de Tesis

En la actualidad las computadoras y el Internet están cambiando la forma de interactuar de las personas, creando una nueva visión y pensamiento acerca de muchas actividades cotidianas. Sin embargo todo este mundo de información digital trae como consecuencia algunos comportamientos inapropiados entre los que se encuentran la violación de la propiedad intelectual, invasión de la privacidad, phishing, fraudes electrónicos y robos de identidad, los cuales son considerados como delitos informáticos o cibercrímenes.

La investigación de este tipo de delitos relacionados con las computadoras y las redes nos lleva a hacer un análisis de los mecanismos de transmisión, almacenamiento e interacción de estos sistemas con el fin de obtener evidencia digital que nos permita conocer las causas del mismo, tal como lo hace un médico forense en una investigación post mortem.

Este trabajo presenta y describe las nuevas facetas de los crímenes tradicionales, pero que ahora son cometidos a través de computadoras, y define las técnicas que se emplean para llevarlos a cabo. En este documento también se describe el análisis, diseño y desarrollo de una herramienta computacional que funciona bajo la plataforma Microsoft Windows. Este software dirigido principalmente a administradores de red y a otros usuarios interesados en la seguridad en cómputo, permite reunir pistas acerca del origen de este tipo de delitos a partir de una identificación, recolección y clasificación de evidencia digital.

Palabras clave: informática forense, evidencia digital, cibercrimen.

Agradecimientos.

A mis padres por brindarme ese cariño, confianza, y apoyo en las decisiones que he tomado en mi vida.

A mis hermanas por darme ánimo y apoyo para terminar.

A mis directores de tesis por su paciencia y apoyo.

A todos los compañeros, profesores y amigos que me motivaron para realizarla.

A Dios por tenerla aquí.

Índice de Contenidos.

CAPÍTULO 1 Introducción

Las ciencias de la computación son un campo relativamente joven comparado con otras áreas de investigación. La sociedad de computación más antigua es la Association for Computing Machinery (ACM) fundada en 1947, poco después de haber acabado la segunda guerra mundial. No es de extrañarse que la guerra haya sido un detonante para el avance de la tecnología. El surgimiento de las Ciencias de la Computación es una muestra de ello.

En un principio los investigadores de esta área emergente estaban más interesados en saber que podría procesarse. Se crearon algoritmos, se diseñó nuevo hardware, se elaboraron diversas teorías que son fundamentales hasta en nuestros días. Después llegó otra nueva era donde los estudios se enfocaron en reducir costos y maximizar la velocidad de cómputo. En esta era surgió la ingeniería del software, los lenguajes de programación y se desarrollaron los primeros sistemas operativos. Luego en la década de los 80's hubo un enorme interés en hacer los sistemas robustos y seguros. Esto llevó a hacer implementaciones de la tolerancia frente a los fallos y de ahí que se incrementó el enfoque hacia la seguridad. Sin embargo, lo que vino hacer un concepto muy importante y revolucionario fue el de las redes, que aunque su origen tuvo que ver con fines militares, en la actualidad es la base de muchas tecnologías y servicios emergentes.

Hoy en día el uso ubicuo de computadoras y redes, está creando un crecimiento de la información digital almacenada, a su vez como una variedad de mecanismos y dispositivos electrónicos capaces de leer, escribir y transferir esos datos mediante diferentes tecnologías e infraestructuras. La proliferación masiva de la creación y movimiento de esta información en las organizaciones e individuos ha creado nuevos modos de trabajo y diversión, en la cual

continuamente se generan nuevas ideas y se ofrecen muchos beneficios. Al mismo tiempo se presentan oportunidades para el desgaste social y por consiguiente algunos riesgos que deben ser tomados en cuenta y mitigados de manera adecuada.

La revolución de la información que vivimos tal vez es comparable con la revolución industrial, quizás este lapso es tan significante como la invención de la escritura o de la imprenta, sin embargo el daño que puede ser hecho a través de las tecnologías de la información oscila en un gran espectro, debido al impacto que ha ocasionado en el mundo y la fuerte adaptación en la población.

1.1 Comportamiento Humano en una era de la información.

El número de personas conectadas a internet ha ido en un rápido aumento en los últimos años. Las estadísticas del uso de internet y población a nivel mundial de acuerdo a internet World Stats [IWS, 2007] la población mundial era de 6,574,666,417 y el número estimado de personas que usan el internet es de 1,244,449,601, es decir, ha habido una penetración en la población de un 18.9%. También reporta que del 2000 a la fecha hubo un incremento de 244.7%, siendo África la zona con más crecimiento en lo que refiere al uso, en cuanto a penetración, la región mundial con mayor índice es Norte América con un 69.1%, de ahí le sigue Oceanía, Europa, y en cuarto lugar Latinoamérica con un 18%. Cabe señalar que en nuestro país hubo un crecimiento del uso del internet en un 586.6% del 2000 a la fecha y una penetración del 17.7%. Pero más allá de las cifras todo esto nos lleva a una interrogante mayor, ¿Qué están haciendo estas personas con el uso de tal tecnología? La respuesta más corta y sencilla es: interactuar.

En los años de la proliferación de internet en los años 90's, se dieron muchos cambios constantes en las tecnologías, empezando por la forma de comunicación entre las redes. También se tuvo la necesidad de intercambio de la información, de ahí que surgieran los primeros navegadores de internet como Mosaic y los primeros lenguajes WEB como HTML. Se ingresaron gradualmente nuevas características a estos dos bastiones de las páginas web tales como: capacidad para organizar los datos en tablas, insertar imágenes, hipervínculos y demás. La popularidad en el uso de estas tecnologías dio origen a un sin número de servicios adicionales. A través del tiempo se fue asentando el servicio de internet, ya que fue accediendo a más zonas del mundo. Durante este periodo de expansión ocurrió un proceso de aceptación inicial en la población, que sin duda el desenlace fue a su favor. Nuevos mundos y fronteras se abrieron con el asentamiento de la WWW (World Wide Web), convirtiéndose en un sistema global social que ya no solo era para tener un servicio nuevo intrínseco de esa tecnología sino que se reflejaron los comportamientos e ideas de antes de la llegada de esta tecnología, tales como la compra-venta. En la actualidad se aprovechan todas las ventajas que la internet ofrece para hacer negocios no solamente a un área específica sino a cualquier parte del mundo y es un hecho que conforme avanza la tasa de conectividad más y mejores servicios se crean, combinándose distintos tipos de hardware e infraestructuras para innovar formas y procesos que facilitan las tareas cotidianas. Pero siempre la expansión del conocimiento y de la tecnología implica riesgos tales como: engaños, fraudes, ataques, robos, entre otros incidentes. La diferencia ahora es que todo esto pasa en un nuevo contexto y ambiente diferente al de otras épocas. La realidad de estos hechos requirió desarrollar una conducta de comportamiento, así como un sistema de monitoreo, y la creación de un conjuntos de reglas para evitar eventos indeseables.

Las redes corporativas tuvieron que invertir en temas respecto a la seguridad en cómputo, cuestiones como *firewalls (cortafuegos)*, control de acceso, canales virtuales, monitoreo de paquetes, filtro de direcciones, sistemas detectores de intrusos, antispam y demás procesos que traten de mitigar esos riesgos. También para el usuario doméstico es importante proteger sus datos, su información y su privacidad en general. Aprovechando la tecnología para compartir y redimensionar sus vidas pero también pagándose el costo de los eventos indeseables que pueden ocurrir y seguirán ocurriendo mientras haya un interés de por medio por parte de algún miembro de este gran sistema social.

1.2 La naturaleza del crimen informático.

Las computadoras han alentado a una oleada de nuevas formas de ataques y avivado conductas indeseables, tales como denegación de servicios y robo de datos personales. Este tipo de delitos se dan porque existe un mundo de oportunidades para que los delincuentes cumplan con algún objetivo trazado, ya sea para ganancias económicas o poder político. Sin embargo también hay otro tipo de estructura más compleja que tienen como motivación, por ejemplo los beneficios psicológicos que pudieran tener dentro de una sociedad de cibercrimen, como lo fuera en caso de que "si cumples ciertos ataques satisfactoriamente, ganarás más *rating* o puntos". Existen páginas de internet que ofrecen diversa información como tutoriales, manuales, ligas, herramientas a sus visitantes para llevar a cabo este tipo de actos. Además ofrecen un servicio de monitoreo de usuarios y su puntuación respectiva de acuerdo a sus aportes o ataques realizados satisfactoriamente. Éstos comúnmente presentados como forma de retos.

La tasa de quejas por delitos informáticos ha aumentado en todos los países. Esto no es casualidad ya que es lógico que si aumentan los ataques también aumenten el número de quejas por parte de empresas, independientes y usuarios domésticos. El FBI en su reporte anual de crimen en internet [FBI-IC3R, 2007] reporta que en el año 2007 hubo un total de 90,008 quejas referidas acerca de este tipo de delitos, un ligero aumento a comparación con 2006, y las pérdidas de dinero dadas por los quejosos se estimaron en 239.09 millones de dólares. Esto nos lleva a la interrogante de ¿Porqué aumentan las cifras? Si tomamos en cuenta que muchas empresas se están uniendo a esta era de la información y que cada vez hay más usuarios que usan una computadora con acceso a internet, entonces podríamos decir que está circulando una cantidad de dinero considerable representado por transacciones en línea. En México de acuerdo a datos recabados por la AMIPCI revelan que el importe de ventas total por comercio electrónico fueron de 955 millones de dólares en 2007 [AMIPCI, 2008], una cifra sustancial que permite ver la existencia de un mercado de posibles ganancias para los detractores. Por otra parte los negocios no se pueden dar el lujo de no tener presencia en el internet ya que esto implicaría riesgos en la competitividad de la misma. Un dato relevante es que de estas ganancias un 70% de los compradores en línea lo hacen mediante tarjeta de crédito, es decir, que existe un cierto nivel de confianza en las empresas que ofrecen este servicio de pago en línea. Por todo esto podríamos declarar que si la cantidad de usuarios en internet aumenta y también hay más servicios de pagos en línea, entonces por consiguiente aumentan las posibilidades para comprar en línea así como la posibilidad para realizar delitos informáticos.

Los delitos informáticos o cibercrímenes tienen las siguientes características en común:

- Se requiere de habilidades técnicas o tecnológicas.
- Tienen más alto nivel de globalización que los crímenes convencionales.

- Difíciles de perseguir.

Según Kshetri Nir (2007) las características de estos crímenes, las víctimas y las agencias de aplicación de la ley han creado un círculo vicioso del cibercrimen (ver figura 1.1). Las agencias como las fuerzas policíacas y unidades de investigación o inteligencia no tienen la experiencia requerida para manejar estas nuevas formas de crímenes. En realidad en la mayoría de los países no se cuenta con el suficiente equipo y personal capacitado para lidiar y combatir la naturaleza global de éstos. En Latinoamérica vemos un problema mayor, ya que las agencias no están bien capacitadas ni siquiera para los crímenes convencionales.

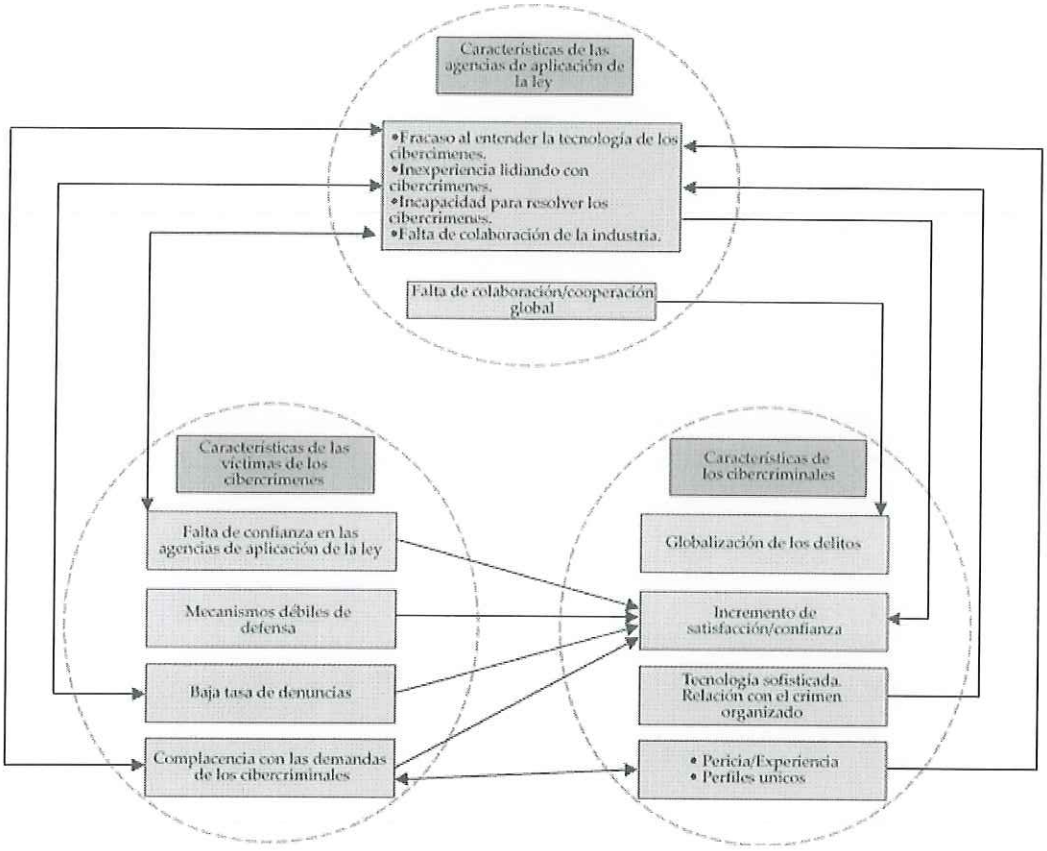

Figura 1.1 Círculo vicioso del cibercrimen, involucrados las agencias de aplicación de la ley, los criminales y las víctimas. [Kshetri Nir, 2007]

En el mundo convencional, la mayoría de crímenes ocurren cerca de la casa del delincuente, ellos viajan tan lejos solo si tienen suficientes incentivos para dejar el territorio conocido. Algunos crímenes como secuestro o robo a un banco, tienen un tinte lucrativo y de buena planeación. Los crímenes en el mundo digital si bien difieren en esta dimensión, no es por mucho la diferencia que se vislumbra en el pasar del tiempo.

Las tecnologías de comunicaciones e información han incrementado drásticamente la porosidad entre las fronteras nacionales, es decir han dificultado aún más la oportunidad de cooperación para llegar con el criminal. Además el anonimato en el internet impone una compleja interacción que permite a grupos de violencia, criminales, organizaciones terroristas transnacionales y compañías dedicadas al espionaje, expandir sus operaciones globalmente sin dejar su territorio. Una alta proporción de las investigaciones de los cibercrímenes, tienen asuntos jurisdiccionales significativos, que van más allá de los recursos humanos y tecnológicos que se tienen disponibles. En muchos casos estos crímenes que traspasan fronteras, reducen el tiempo de reacción de las autoridades, ya que requieren de una cooperación estrecha entre organismos de diferentes países.

1.3 Mitos detrás del cibercrimen.

Cuando nosotros quitamos los términos técnicos referidos hacia crímenes relacionados o cometidos en el ciberespacio o el internet, algo bueno sucede, empezamos a recordar que un crimen ha sido cometido. Tal vez la confusión que se da es por la palabra especial que reciben algunos delitos como: p*harming, phishing, DDOS* entre otros. Pero lo que realmente importa es lo abstracto y fundamental que hay detrás de esa palabra, es decir el *phishing* por ejemplo, es un

fraude, no tiene ninguna diferencia con los fraudes de antaño, simplemente que ahora ese fraude se comete de otra forma, mediante el uso de las tecnologías del internet y las redes. Por consiguiente no se puede seguir el mismo proceso de investigación para perseguir de una forma adecuada este tipo de delitos sino que debe asumirse un cambio.

El problema es el entendimiento de la semántica, ya que a veces hasta un investigador puede mitigar el efecto mediático u olvidarse que un crimen ha sido cometido [Anthony Reyes et al, 2007].

Otro ejemplo es el caso del *DNS spoofing*, si nos enfocamos en lo abstracto de este ataque vemos que es una suplantación de identidad. En este caso es la suplantación de una infraestructura legítima por una falsa. En realidad hay pocos "verdaderos cibercrímenes" que no podrían existir sin el uso de una computadora, como lo son: ataque de denegación de servicios, propagación de gusanos, spam, entre otros. Y aunque estos sean cometidos con la ayuda de una computadora los actos por sí mismo develan definiciones de los crímenes tradicionales.

Entonces si nosotros quitamos el caos que puede representar los nombres técnicos de un cibercrimen y eliminamos el aspecto computacional, estaríamos enfrente del crimen que ha ocurrido. La realidad es que cuando usamos terminología relacionada con la palabra *ciber*, internet o informática, el impacto o valor que tiene es reducido. La primera aparición de la palabra ciber fue en la palabra cibernética la cual fue introducida en el libro de Norbert Wiener en 1948 [Quinion, 2002]. Este prefijo implica la idea de conducción o de gobierno. Su origen viene del griego *kibernao* que significa pilotar una nave. De hecho ha habido un uso excesivo de este prefijo, como si fuera una moda y en ocasiones se percibe como si estuviera fuera de control o intercambiable con otras palabras. En la realidad, tiene más impacto escuchar "acosador en línea" que "cibernovio", el problema es que la palabra ciber tiende a sí misma hacia lo irreal,

falso o un lugar distante, después de todo el *ciberespacio* no es físicamente tangible, es virtual. Por todo lo anterior es importante para los investigadores o personas involucradas con los cibercrímenes, recalcar el fondo que parece ofuscado por una confusión lingüística.

1.4 Clasificación de los cibercrímenes.

Las computadoras estarán probablemente involucradas en crímenes que tal vez todavía desconozcamos. Nuevos tipos de crímenes emergen constantemente o evolucionan sus formas, sin embargo existe una clasificación aceptada general de los crímenes informáticos [Harlan Carvey, 2004]:

1. La computadora es el blanco del crimen, con la intención de dañar su integridad, confidencialidad y/o disponibilidad. Una manera obvia en la cual una computadora puede estar envuelta en una conducta fuera de la ley es cuando la confidencialidad, integridad o disponibilidad de la información de la computadora o servicios es atacada. Esta forma de objeto de crimen, generalmente el objetivo es que se quiere adquirir información almacenada en ese sistema, para controlarlo sin autorización, robo de recursos o la alteración a la integridad de los datos.
2. La computadora es un repositorio de información usada o generada en el acto del crimen. Una segunda manera en la cual las computadoras pueden ser usadas para ilegalidades involucra el uso de una computadora o dispositivo conectado a la misma, para que funcione como medio de almacenamiento. Por ejemplo los traficantes de drogas podrían usarla para guardar información respecto a sus ventas o clientes; otro ejemplo es el hacker que guarda contraseñas o credenciales robadas, ya sea números de

tarjetas de crédito, información corporativa propietaria, imágenes pornográficas o *warez* (software comercial pirateado).

3. La computadora es usada como herramienta para cometer el crimen. Otra forma que una computadora puede ser usada para un cibercrimen es como una herramienta de comunicación. Mucho de los crímenes que caen en esta categoría son simplemente los crímenes tradicionales realizados en línea.

1.5 Respuesta a los incidentes y malware.

Para poder responder a los incidentes relacionados con los cibercrímenes hay que empezar por definir incidente en el contexto de la seguridad informática. Los incidentes son eventos adversos que amenazan los sistemas de cómputo y las redes. Estos eventos incluyen cualquier cosa observable que pase en una computadora o red, como lo es conectarse con otro sistema, acceso a los archivos, desactivación de controles y demás. Eventos adversos pueden ser un bloqueo del sistema *(crashed), inundamiento de paquetes* dentro de una red, uso no autorizado de la cuenta de un usuario, falsificación de una o más páginas Web y ejecución de código malicioso; otros eventos adversos pueden ser inundaciones, fuegos, cortocircuitos, o un sobrecalentamiento del sistema que lleve al daño del equipo.

De los incidentes mencionados hay uno que sobresale por sus características: no requiere ir al lugar físico, relativamente fácil de hacer y muy globalizado, el concepto que describimos es el de la ejecución de código malicioso. Este incidente puede llevar a un atacante a tener el control total de un sistema de cómputo, lo cual es una seria amenaza para los principios básicos de la seguridad como: la autenticación, confidencialidad e integridad de nuestros datos.

Todos los objetos que están detrás de la ejecución de código malicioso se le conoce con el término de *malware*, el cual es definido como: *"un conjunto de instrucciones que se ejecutan sobre una computadora y hacen que tu sistema haga lo que el atacante quiere"* [Ed Skoudis, Lenny Zeltser, 2003]. El código puede actuar como un agente interno, por ejemplo con la instalación de un *troyano* o *rootkit,* y si el atacante puede instalar código malicioso en alguna computadora o engañar a algún usuario a cargar dicho programa, la computadora podría actuar como subordinada de las intenciones o deseos del atacante. Al mismo tiempo pudiera ser que el sistema ya no siga los comandos e instrucciones del usuario legítimo ni una vez más.

Entonces, ¿Quién necesita un colaborador humano interno cuando un atacante puede usar código malicioso para ejecutar instrucciones sobre el interior? Seres humanos se infiltran en organizaciones y pueden ser sorprendidos, arrestados e interrogados. El código malicioso, por otra parte, probablemente sea descubierto, analizado y borrado, todas ellas son mucho mejor para el atacante que tener un cómplice humano capturado en prisión empezando a revelar detalles que lleven a su captura. Ya sea una organización de negocio comercial, institución educativa, agencia de gobierno, división militar o tu propia computadora personal, el código malicioso puede hacer un daño real.

El malware puede ser implementado en casi cualquier lenguaje de programación, con la limitación de la imaginación del atacante, y ellos comúnmente tienden a ser muy creativos, muchos de ellos están a la vanguardia de la tecnología, tratando de encontrar nuevas vulnerabilidades para aprovechar. Toda esta información la pueden recabar de *blogs* de personalidades en la seguridad en cómputo, comunidades en internet, libros, artículos de diseño, lista de correos de páginas dedicadas a vulnerabilidades etc. Siempre hay una forma para seguir aprendiendo y hacer el proceso más fácil y certero en su intento. Los atacantes

vierten una enorme cantidad de tipos de ejecutables, lenguaje de script, lenguajes de macros, ensamblador y demás ocurrencias para llegar a sus objetivos.

El código malicioso sobre alguna computadora puede realizar los siguientes eventos:

1. Borrar archivos de configuración sensibles del disco duro o convertir tu computadora inoperable.
2. Infectar alguna computadora y usarla como intermediaria o fuente de infección para llegar a otras.
3. Monitorear lo que escribes con tu teclado y ver lo que estás viendo o haciendo clic con el ratón.
4. Reunir información acerca de algún usuario, ya sea de hábitos, como las páginas que visita, el tiempo que está conectado y demás.
5. Robar archivos.
6. Enmarcar o apuntar hacia alguna computadora acerca de un crimen, ya sea poniendo evidencia que involucre y apunte hacia el usuario.
7. Esconder procesos y archivos.

Las posibilidades son bastas, esta es una lista común y pequeña de lo que puede pasar, por eso es muy importante hablar de este tipo de incidentes y empezar a capacitarse. Teniendo conocimiento de cómo llevar una práctica competente de seguridad de la información, se podrían mitigar los riesgos que conlleva. Por lo tanto un profesional del área, debería tener una buena formación de respuesta a los incidentes, ya que ésta es una pieza clave para el ciclo de vida de la seguridad. Hay 3 partes importantes en el ciclo: detección, respuesta y contramedidas.

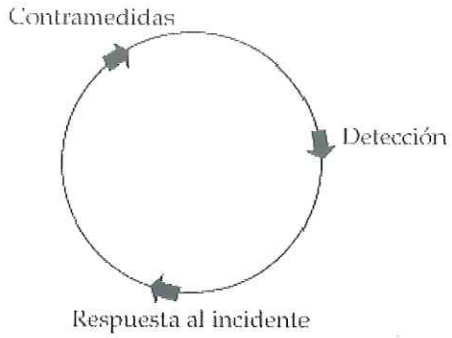

Figura 1.2 Ciclo de vida de la seguridad.

1. Detección. La detección de los incidentes de seguridad esencialmente quiere decir proveer una pauta para decir que la seguridad ha sido violada. La detección efectiva provee una retroalimentación respecto a la adecuación de las contramedidas que han sido desplegadas.
2. Respuesta a incidentes. Después de que un incidente ha sido percibido, la respuesta a incidentes es el siguiente paso lógico. Remarcando el tipo, severidad, ubicación, y frecuencia de ataques.
3. Contramedidas. Las contramedidas son defensas que manejan las amenazas tales como ataques de denegación de servicios, repudio, y demás. Usualmente son escogidas y desplegadas como resultado de un análisis de riesgo conducido, aunque otras alternativas han sido usadas ampliamente en lugar de las tradicionales. Ninguna contramedida es infalible, sin embargo si así fuera, los recursos para adoptar todas las contramedidas necesarias probablemente serían insuficientes.

Este trabajo de tesis estará enfocado en el área de detección y respuesta, en este caso haciendo el análisis forense podremos tener un mejor contexto de lo que ha pasado y determinar si se violó

la seguridad en un determinado sistema. Una vez detectado las causas podremos realizar una serie de actividades para tomar contramedidas al incidente en cuestión.

1.6 Justificación.

Como se acotó anteriormente, la informática forense viene a caer en nuestros días como una importante área de oportunidad para la investigación académica y comercial interesada en la seguridad en cómputo, es por ello que muchas agencias de aplicación de la ley, especialmente aquellas en grandes ciudades, están necesitadas de personal que ha sido entrenado con una especialización en informática forense. La industria, por otra parte, ha ido tomando esta área muy seriamente desde hace años, para proteger sus intereses y su información. Aunque la situación en nuestro país, no se parezca a la de otros países más desarrollados como Estados Unidos, Canadá o Inglaterra, donde aparte de tener legislación acerca del tema hay diversa oportunidad de crecimiento profesional, más temprano que tarde se asentarán como toda una realidad. Por ello que es conveniente ir encontrando áreas de oportunidad frescas y novedosas.

Otro punto de resaltar es que la falta de experiencia en el mundo real puede ser un impedimento para especializarse y tener una experiencia probada, aunque existen avances en cuanto a certificaciones, cursos académicos e investigación, es conveniente hacer más investigación en el campo y que mejor manera de empezar que desarrollando una herramienta de análisis forense para la identificación, recolección y clasificación de evidencia digital. De ahí la importancia de este trabajo que no solamente es desarrollar una herramienta como muchas sino que entre sus posibles aportes estará dando un granito de arena en la concientización de la comunidad universitaria para tomar estos temas con más interés y poder crear planes de

estudios más completos que incluyan estas áreas. Además se recalcará que los ataques o incidentes en el mundo de la seguridad en cómputo no involucran solamente crímenes nuevos y tecnologías difíciles de comprender sino que son los mismos delitos tradicionales realizados en un nuevo contexto. Por último con la realización de este sistema se aportarán algunas técnicas, algoritmos de cómo se realizan algunos procesos de informática forense, esto sirve ya que en la literatura existe poca documentación acerca de cómo realizar este tipo de software.

1.7 Objetivos.

El objetivo principal de este trabajo es desarrollar una herramienta de análisis forense para la identificación, recolección y clasificación de evidencia digital (Hefoclase). Hefoclase, significa Herramienta Forense de Clasificación de Evidencia. Además se establecieron otros objetivos específicos:

- Conocer y comprender como el uso de estas herramientas puede ayudar a mejorar las condiciones de seguridad en un sistema informático.
- Describir el proceso de desarrollo de este tipo de aplicaciones.
- Conocer la estructura y tipos de herramientas que existen en el mercado.
- Describir que tipo de incidentes son más fáciles de detectar.
- Describir el proceso de adquisición de evidencia digital.
- Publicar el trabajo en una revista o congreso.

1.8 Organización del trabajo.

Este documento consta de 8 capítulos que se pueden leer de una manera secuencial para un mejor entendimiento, sin embargo para aquellos lectores que tengan conocimiento previo acerca del marco teórico de los cibercrímenes e informática forense, pueden pasar directamente al capítulo de su interés a partir del capítulo 3, donde se describen las herramientas actuales gratis y comerciales existentes para llevar a cabo una investigación forense. También se acota alguna problemática que ataca a las mismas y define los paradigmas existentes. Como se mencionó anteriormente los primeros dos capítulos del trabajo hablan acerca del entorno que rodea a la informática forense, más específicamente, el capítulo 2 introduce los términos de informática forense y evidencia digital y describe sus orígenes. Posteriormente en el capítulo 4 se presenta la metodología que se llevó a cabo para el desarrollo computacional de la herramienta. En el capítulo 5 se documenta las fases de análisis y diseño del sistema, después en el capítulo 6 se muestran las pantallas y funcionamiento de Hefoclase. El capítulo 7 se enfoca a las pruebas de usabilidad, rendimiento y operación que se realizaron para evaluar el sistema y por último en el capítulo 8 se presenta una breve discusión acerca de las interrogantes planteadas y las conclusiones a las que se llegó con este trabajo.

CAPÍTULO 2 Informática Forense

En este capítulo hablaremos acerca del concepto que involucra este trabajo, la informática forense. Describiremos un poco como es que surgió esta rama en el contexto de las ciencias computacionales y acotaremos la importancia de la evidencia digital que es base para un investigador forense. Además se enlistará algunos lugares donde esa evidencia digital puede ser encontrada en un sistema.

2.1 ¿Qué es la Informática Forense?

La informática forense es el "proceso de identificar, preservar, analizar y presentar la evidencia digital en una manera que es legalmente aceptable" [McKemmish et al 2003]. La informática forense es por consiguiente un largo tema cubriendo varios aspectos referentes a la seguridad en cómputo.

Otra definición más completa es la siguiente: es la identificación, preservación y análisis de información almacenada, transmitida, o producida por un sistema de cómputo o red de computadora en orden para razonar acerca de la validez de una hipótesis, la cual intenta explicar las circunstancias o causas de una actividad bajo investigación, en una manera tal que cubra los requerimientos la evidencia presentada.

El término informática forense también lo pueden llamar análisis forense informático, forensia computacional o descubrimiento de evidencia digital, y ésta puede a menudo encontrar evidencia de información borrada ya sea intencional o no. La meta es recuperar los datos e

interpretar tanta información como sea posible, principalmente relacionados con incidentes de seguridad y/o de delitos informáticos.

Pero bien ¿Qué es la evidencia digital? En 1999 el Scientific Working Group on Digital Evidence dio una clara definición en su propuesta [SWGDE, 1999], *"Información de valor probado almacenada o transmitida en forma digital"*. Entonces así como otra ciencia forense, la informática forense involucra el uso de herramientas de tecnología sofisticadas y procedimientos que deben guiar a la garantía de la exactitud de la preservación de la evidencia y de los resultados concernientes al procesamiento de la evidencia. Y ¿qué evidencia es necesaria? la evidencia necesaria es la siguiente [Eugene Schultz et al, 2001]:

1. Toda la evidencia física (computadora, periféricos, documentación).
2. Salida visual del monitor (pantallas, *snaps,* videos)
3. Evidencia impresa.
4. Representaciones magnéticas (archivos en discos)

En resumen toda los datos, metadatos, y estructuras que nos ayuden a conectar los hechos *post mortem* del caso que se sigue.

2.2 Historia de la Informática Forense

La informática forense fue creada en demanda para servicio de la comunidad relacionada con la aplicación de la ley. A principios de 1984, el laboratorio del FBI y otras agencias empezaron a desarrollar programas para examinar la evidencia en sistemas de cómputo. Para manejar apropiadamente las crecientes demandas de investigadores y fiscales de una manera

estructurada y programática, el FBI estableció el Computer Analysis and Response Team (CART). Aunque CART fue único en su tiempo, sus funciones y organización general fueron duplicadas en otras agencias de aplicación de la ley en Estados Unidos y otros países [PC History, 2007].

Un problema que se vio al principio por estos organismos de la ley fue la de identificar recursos dentro de la organización que pudieran ser usados para examinar la evidencia en una computadora. Aunque la evidencia almacenada de una computadora fue usada en las cortes desde los años 70s [Kabay M. E., 2008] en aquella fase temprana la computadora era vista simplemente como un dispositivo para almacenar y reproducir registros de papel, el cual constituía la evidencia real. Oportunidades para el fraude informático estaban limitadas a la destrucción o robo de equipo y otros delitos locales. La evidencia en la computadora presentó un reto aún en esas condiciones limitadas, y debido a que en algunas jurisdicciones los trabajos del sistema que la producía tenían que ser explicadas en detalle en la corte, se derivó a la necesidad de crear reglas y organismos que estudiaran más a fondo este tipo de delitos. Por ejemplo, en Inglaterra se creó la *United Kingdom Police and Criminal* Act (PACE), sección 69 la cual era gobernada por admisibilidad y peso de la evidencia, introducir la evidencia computacional en una corte no fue tan directo. La computadora tenía que estar certificada de trabajar apropiadamente, en el mismo sentido como un dispositivo tal como una lámpara o un detector de velocidad por radar. La informática forense emergió a mediados de los años 80s, principalmente debido al incremento de casos comunes de robo y falsificación de hardware y software, una consecuencia del ascenso del mercado de las computadoras personales y además debido a que intrusos o extraños podían acceder a las *mainframes* remotamente y anónimamente. Los virus empezaron a proliferar y mutar vía las redes de área local [LAN] y las redes de área

amplia (WAN). Los hombres de negocio y el gobierno empezaron a mostrar un mayor interés en formalizar sus políticas de seguridad e implementando éstas con contramedidas apropiadas. Muchos de estos mecanismos de prevención o detección produjeron como efecto colateral, el material crudo y más importante para la informática forense: la evidencia basada en la computadora.

El término informática forense y la estandarización de los procesos de manejo de la evidencia empezaron a ganar aceptación a finales de los 80s. En la tabla 1, se puede ver que la informática forense como disciplina estandarizada se fue valorando poco a poco conforme con la evolución de los sistemas de cómputo.

Año	Tecnología	Crimen Informático	Informática Forense
1950	Transistores	Nada	
1960	Aplicaciones comerciales	Fraude local	
1970	Silicio	Crimen Interno	
	Líneas de 10 baudios	Crimen exterior	
	Bases de datos	Hacking	
	ARPANET	Robo de datos	
1980	Computadoras personales	Violación de estándares de seguridad	Unidades de crimen local
	Telnet	Acceso no autorizado	Organismos nacionales
	LAN, WAN	Virus	
1990	Internet se hace público	Fraude en línea	
	La Web	Hacking a páginas web	Organismos internacionales
	Ajax	Exploits	
	Comercio Electrónico	Robo de credenciales bancarias	
2000		Fraude corporativo	Entrenamiento y certificación
			Legislación de leyes
			Creación de estándares
2008	Web 2.0/ Virtualización	?	?

Tabla 1. Contexto Histórico de la Informática Forense.

En los últimos años apenas se han ido creando ambientes de trabajo globales para la prevención, detección y castigo del crimen informático. Los siguientes son una lista de áreas que existen y están apareciendo para la aplicación de la informática forense:

1. Seguridad Nacional. En países como Estados Unidos e Inglaterra.
2. Aduana e impuestos.
3. Abogados.
4. Cortes civiles
5. Policía
6. Negocios
7. Firmas de Seguros
8. Crimen corporativo e internacional

A nivel personal o de usuario final.

2.3 Naturaleza de la Evidencia Digital.

La evidencia es lo que distingue a una hipótesis de una aseveración infundada. La evidencia puede confirmar o refutar una hipótesis, así que la confiabilidad de la evidencia digital y su integridad son claves para la admisibilidad y peso en un tribunal. Hay ciertas características especiales de la evidencia digital, en la que los sistemas de cómputo y redes involucrados hacen que la interpretación de la evidencia sea todo un reto:

1. Muchos sospechosos. Cualquier persona que tenga una computadora y esté conectado a internet puede realizar un ataque y por lo tanto es un sospechoso, sin embargo lo que debemos ver es ¿quién ha sido la víctima?, ¿qué información tiene?, ¿qué motivos existen para perjudicarla?, entre otras preguntas esenciales.

2. Identificación del crimen. En el crimen informático o el relacionado con los sistemas de cómputo, la naturaleza de un evento es por lo general no tan obvia e inmediata. Por ejemplo, cuando un *hacker* roba información confidencial, las víctimas pueden tardarse en saber lo que se ha robado, hasta que alguien con mas información como lo puede ser un administrador de sistema, identifique que ha pasado. Sin embargo puede que ni siquiera él sepa lo que pasó.
3. Demasiada evidencia potencial. En la informática forense, por lo general cuando se lleva a cabo el proceso de adquisición y análisis, suele haber problemas para saber qué datos son importantes y cuáles no, lo cual puede ser un punto crítico en términos de eficiencia en la corte, por ejemplo un caso de que los tiempos sean sensibles y esté en riesgo una persona o una empresa.
4. La evidencia es fácil de contaminar. Tradicionalmente, la evidencia en la escena es enviada a laboratorios forenses independientes para probarse mientras que la investigación dirige sus preguntas hacia otro lado hasta que los resultados estén listos. Pero en la informática forense, todos los aspectos de la investigación tales como: nombrando el crimen, identificando al perpetrador, siguiendo el rastro de la evidencia y la construcción del *modus-operandi,* usan las mismas técnicas de análisis. De aquí que

el manejo de ésta es especialmente vulnerable a errores. Por ejemplo reiniciando un equipo, instalando un programa, etc.

5. Contaminando alguna evidencia puede arruinar todo. Cada objeto de la evidencia física es solo un simple componente del caso, a menudo independiente, y la fiscalía probablemente consiga algo satisfactorio sin ello. En contraste la evidencia digital está altamente interconectada. Probar una hipótesis, acerca de ¿Qué?, ¿Cómo? y ¿Quién? significa recrear el escenario paso a paso en una línea de tiempo, donde un paso en falso o inválido puede hacer que se pierda el caso entero.

2.4 ¿Dónde se puede esconder esa Evidencia Digital?

Una vez que un sistema ha sido comprometido, y el atacante ha ganado acceso, éste puede instalar malware y software adicional para tener mayor control sobre el sistema o poder llevar a cabo sus objetivos. Existen diversas formas en que el atacante puede esconder o borrar los rastros que pueda dejar. De manera similar, puede haber usuarios legítimos que escondan datos usando una variedad de métodos como el atacante externo. Comúnmente este tipo de mecanismos suelen ser específicos de cada sistema, es decir las características propias de cada computadora tales como: sistema operativo instalado, sistema de archivos, arquitectura de hardware, estructuras internas y demás. Todos estos son aspectos fundamentales que diferencian muchas veces la forma de llevar a cabo una investigación.

Para nuestro caso estamos contemplando que el sistema operativo es Microsoft Windows XP. En Windows como en otros sistemas operativos una parte fundamental de la gestión que realiza el mismo, es el sistema de archivos. Esta estructura virtual es la que permite organizar la

mayoría de datos que almacenamos en un disco duro. Los tipos de sistemas de archivos que tiene este sistema operativo son dos: NTFS y FAT en sus diversas versiones cada uno. Una generalidad que se observa de estos dos tipos de sistema de archivos son en que hay más maneras de ocultar datos sobre un sistema de archivos NTFS que uno FAT, esto debido a su diseño interno de uno que es más sencillo que del otro.

Los aspectos clave del ocultamiento de datos tienen que ver acerca de que estás tratando de ocultar y de quién. Existen algunos trucos sencillos para esconder datos y ejecutables para el usuario común, a su vez que también los hay para ocultar archivos de un administrador que quizás esté buscando algo específicamente sospechoso. También hay maneras para esconder datos de un analista forense. Sin embargo, si nosotros conocemos como esos datos pueden ser ocultados podremos desarrollar mecanismos para protegernos o detectarlos.

Estas son algunas de las formas o lugares en donde se pueden ocultar datos en un sistema Windows:

1. Atributos en el sistema de archivos. Sin duda la manera más fácil de ocultar datos sobre un sistema es simplemente cambiando el nombre o extensión del archivo en cuestión. Cambiando el nombre de un programa de "malware.exe" a algo inocuo como "win.exe" podría ser una técnica buena para esconderse del ojo del observador casual. Un administrador buscando algo específicamente sospechoso podría pasarle desapercibido, si particularmente ese archivo esta donde debe ir y con el nombre correspondiente. Hay varios métodos para detectar los cambios en el sistema de

archivos de esta naturaleza, por ejemplo, cada archivo dentro del sistema tiene un conjunto de meta datos (tales como fecha de creación, fecha de último acceso y de última modificación) asociados. Cuando un archivo es creado por primera vez sobre un sistema, el tiempo de creación del archivo deberá reflejar la fecha y hora apropiada. Los archivos pueden ser buscados usando estos parámetros. Aunque cabe destacar que la fecha de creación probablemente no refleje exactamente la fecha y hora de creación del archivo. Por ejemplo en el caso de que haya sido incluido vía una rutina de instalación, los archivos puestos sobre el sistema de esta manera, generalmente reflejan la fecha de creación correspondiente a la fecha en que fueron anexados o empaquetados al instalador. También existen programas específicos que modifican estos metadatos a conciencia por parte de algún usuario o perpetrador del sistema.

2. Firma digital. Otro atributo o propiedad de un archivo donde puede encontrarse evidencia es en la firma digital del mismo. Sin embargo, esta propiedad generalmente no es usada para esconder datos, más bien y a menudo es como un método para descubrir datos ocultos. La firma del archivo es una secuencia de caracteres localizada en los primeros 20 bytes de un archivo. En un sistema Windows, los archivos tienen firmas específicas dependiendo del tipo de archivo. Los archivos ejecutables, aquellos con extensión .exe, .dll y .sys por ejemplo, tienen de firma la cadena de caracteres "MZ". Las firmas son útiles cuando se buscan datos escondidos. Si un usuario cambia el nombre y extensión de un archivo pero no modifica la firma, una herramienta puede abrir el archivo, leer sus primeros 20 bytes y comparar con su lista o base de datos basada en la extensión del archivo para verificar que concuerda. Aún así aunque

parezca que todo está en orden con el archivo y sus firmas, no significa que no se haya violado su integridad.

3. Enlace de archivos. Es un método de ligar o enlazar dos ejecutables en uno, sin afectar el funcionamiento del programa. Existen herramientas para hacer esto de tal forma que ambos archivos son ejecutados cuando se corre el archivo primario. Tales herramientas han tenido popularidad con los lanzamientos de troyanos como *Back Orifice*.
4. Flujos de datos alternos NTFS (New Technology File System). El sistema de archivos NTFS es recomendado para su uso en los servidores Windows. El sistema de archivos tiene un número de ventajas sobre el sistema de archivos FAT (File Allocation Table), ya que soporta particiones más grandes, mejoras en cuanto a tolerancia a fallos y la habilidad para poner permisos y auditar archivos y directorios. Desde un inicio NTFS tenía incluido soportar el Apple's Hierarchical File System (HFS). Los archivos sobre HFS se dividen en dos, recursos y datos. Esto significa que en NTFS, una entrada de archivo dentro del Master File Table (MFT) puede tener atributos adicionales, especialmente flujos adicionales asociadas al flujo primario.
5. Escondiendo datos en el registro. El registro es otra parte donde se pueden ocultar datos dentro de un sistema Windows. Los datos almacenados en el registro consisten de varios formatos, incluyendo cadenas y datos binarios. Muchos tipos de datos pueden ser escondidos dentro del registro, como información de texto, contraseñas, direcciones URL (Uniform Resource Locator) e información binaria. La información binaria puede incluir segmentos de programas o incluso programas enteros. Programas pequeños puede ser ocultados como datos binarios en una llave del registro, o un programa más grande puede ser segmentado en varias llaves.

6. Archivos temporales de internet. En un sistema Windows, cuando se navega por internet hay un repositorio fijo donde se guardan los datos que se descargan de la red, ya sea imágenes, archivos, páginas, etc., en ese repositorio puede haber información que interese a un investigador, por ejemplo los sitios que haya visitado un usuario específico.
7. Cookies del Sistema. Las cookies sirven para realizar una comunicación entre algunos sitios y el cliente (la máquina del usuario) y generalmente son usadas para mantener persistencia con ciertas operaciones de los servicios de algún servidor, ya sea mantener la sesión abierta o cerrarla automáticamente después de un tiempo, entre otras. Sin embargo en la actualidad ha habido un uso malicioso de ellas, como por ejemplo esconder *ad-aware* o *spyware* dentro de ellas.
8. Esteganografía. Es el arte y ciencia de escribir mensajes ocultos de tal manera que nadie excepto el receptor sabe de la existencia del mensaje [Wikipedia, 2008]. Aquí podemos encontrar una variedad de métodos esteganográficos como lo son esconder texto en texto, imagen en audio, audio en imagen, texto en imagen y demás. Lo cual puede ser peligroso si es que se oculta código malicioso dentro de algo que parece legítimo.

Hay que tener en mente que estos son solo algunos métodos, y conforme avanza la tecnología habrá otros más. Tal vez en algún punto un investigador ande en busca de evidencia que le indique ciertas actividades específicas, pongamos el caso de pornografía infantil. El investigador puede que se enfoque en encontrar conversaciones por correo o fotos, sin embargo si éste ha escrito mal ciertas palabras clave o el atacante haya usado un lenguaje extranjero o ajeno al

investigador probablemente la búsqueda fallará, en estos casos depende mucho la experiencia que tenga el forense.

Por último, hay que resaltar que no solamente se puede ocultar información en el sistema de archivos. Viéndola desde una perspectiva general un sistema de cómputo consiste de uno o más unidades de procesamiento central para ejecutar código. Módulos de memoria para almacenamiento dinámico, un área de respaldo-apoyo para datos, dispositivos de entrada-salida, etc. Durante la ejecución, el estado de la información que describe qué es lo que es y que se está haciendo está distribuido a través de varios componentes:

- La unidad de procesamiento central. Es la que ejecuta la operación actual sobre los datos en el almacenamiento dinámico.
- Almacenamiento Dinámico. Contiene fragmentos del sistema operativo, aplicaciones ejecutando, páginas de datos temporal, etc.
- El almacenamiento de apoyo o secundario (como discos removibles o discos compactos) contienen datos en persistencia, archivos retenidos de una sesión a otra o datos cargados en memoria en el espacio de trabajo actual como la memoria volátil (RAM).
- Componentes de red. Contiene los paquetes enviados, conexiones, tablas de la ruta de acceso, puertos y bitácoras de las peticiones.

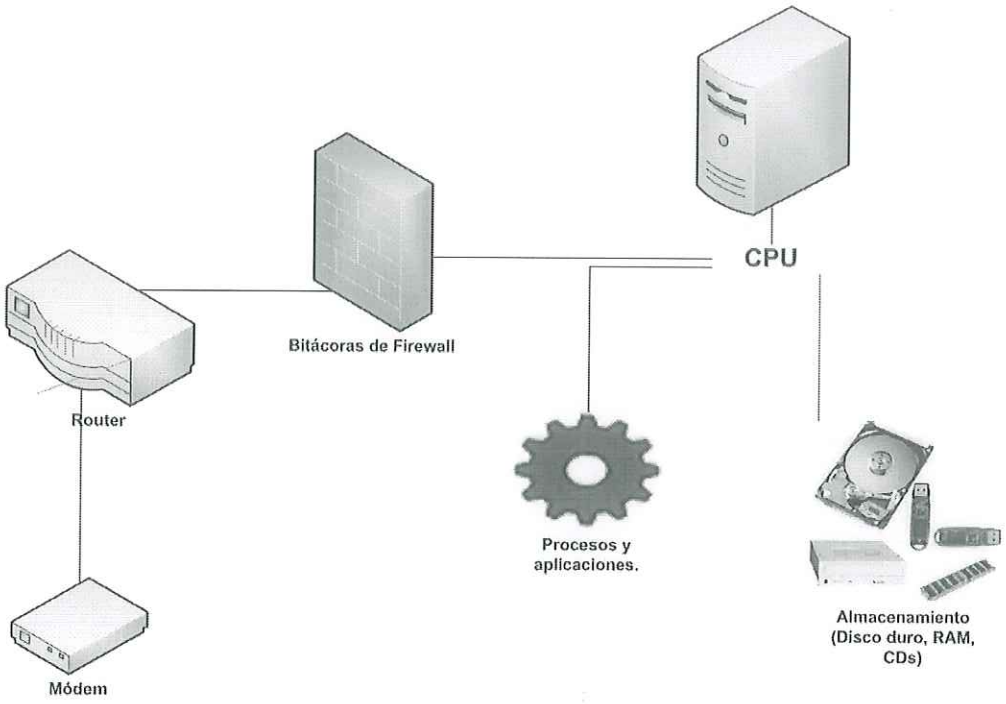

Figura 2.1 Fuentes de evidencia en una computadora básica de escritorio.

CAPÍTULO 3 Herramientas de Análisis Forense

En este capítulo describiremos el medio principal con el cual se realiza una investigación de informática forense, las herramientas de software. Describiremos algunas categorías en la que pueden ser clasificadas las herramientas de informática forense. También hablaremos del paradigma tradicional y la discusión que se genera cuando se quiere tomar la decisión acerca de cómo llevar a cabo una investigación, ya sea de forma tradicional o en vivo. Después enlistaremos una serie de herramientas que son utilizadas en el mundo real y veremos algunas de sus características y defectos que tienen.

Podemos en general identificar 3 categorías de funcionalidad en las herramientas de análisis forense computacional: obtención de imágenes (respaldo), análisis y visualización. Estas categorías pueden clasificarse en:

1. **Obtención de imágenes:**
    - Obtener imágenes de la memoria volátil (incluyendo PDAs y teléfonos móviles);
    - Imágenes de archivos y el disco.
    - Bloqueadores de escritura.
    - Generadores y verificadores de integridad.
2. **Análisis**
    - Recuperación de datos de ambiente y la búsqueda cruda en el disco, para textos de archivos o por sector.
    - Recuperación de archivos.

- Herramientas para verificar integridad en el sistema de archivos y el disco.
- Conversión de archivos (por ejemplo conversión de archivos propietarios en texto y viceversa o entre diferentes formatos).
- Filtro de datos por fecha de última modificación y otras propiedades de archivo, tal como tipo de archivo o aplicación como correo, gráfica, procesamiento de palabras, hojas de cálculo, archivos de presentación.
- Herramientas de búsquedas, motores de búsqueda sofisticados con lógica difusa.
- Herramientas de minería de datos.

3. **Visualización**
   - Línea de tiempo.
   - Herramientas de análisis de puntos de referencia o ligas.
   - Métodos gráficos de visualización.

3.1 En vivo o Tradicional.

En la informática forense, una de las discusiones más populares es acerca de dos paradigmas, dos formas de llevar a cabo la recolección de evidencia digital, el famoso debate acerca de "Live vs. Postmortem". Tomar datos de un sistema corriendo o en vivo (live) o apagar el sistema y sacar una imagen total del disco duro. La primera opción puede proveer evidencia que no está disponible en una imagen de un disco, este paradigma ve como si se tomara una foto del sistema dinámico que no puede ser reproducida después. Además que ya han sido aceptados casos en la corte con este tipo de técnica, aunque los disidentes argumenten que en su método este aspecto

se podría solucionar con una imagen de la memoria RAM, para recabar algunos datos acerca del contexto en el que se ejecutaba el sistema. Comúnmente estas técnicas solo extraen cadenas de texto en formatos ASCII o UNICODE y la interpretación de estas imágenes es inconsistente y limitada [Adelstein P., 2006].

Ahora imaginen un caso en el que un servidor de una compañía X que ha sido comprometido y donde mantiene información sensible de millones de cuentas de usuarios, datos personales, de contacto, tarjetas de crédito etc. Y que su disco duro es de 630 Terabytes, ¿cuáles son los problemas a los que nos enfrentaríamos? Primero, ¿Dónde se almacenarán tantos datos? Se puede guardar una sección del disco duro, pero la realidad es que no se sabe donde puede estar lo importante, ¿Es necesario tener otro disco duro de 630 TB? Segundo, ¿cuánto tiempo me tardaría en hacer la imagen de ese disco duro? Tercero, la máquina no puede ser apagada solo para hacer la imagen ya que la compañía tendría pérdidas millonarias. En este ejemplo hacer la imagen del disco no es para nada práctico, las razones son de que 630 TB = 6,926,923,254,988,880 bytes. Suponiendo que se cuenta con un dispositivo que duplica discos duros a una tasa de 6GB por minuto, después de un poco de aritmética, el tiempo que tardaríamos son: 2 años para hacer la imagen completa, aún si ya hubieras logrado ese pequeño paso el análisis de toda esa información sería exhaustivo y de nuevo impráctico.

Por lo anterior, es mejor hacer un análisis en vivo de ese sistema, para recabar información volátil que puede consistir de procesos corriendo, bitácoras de eventos, información de la red, librerías cargadas, servicios registrados, entre otros. En la práctica uno de los mecanismos que parecen indispensables en una investigación en vivo, son los procesos corriendo con el número de puertos que están utilizando.

A pesar de tener ciertas ventajas la adquisición en vivo, también tiene sus desventajas como es el hecho de que tenga que instalar el programa en la máquina y esto lleve a que renombre áreas del disco que puedan ser vitales para la investigación, es decir, podemos perder evidencia digital llevando a cabo este proceso. Una solución a esto son los llamados discos en vivo, que en lugar de instalarlo, corren en un medio de sólo lectura ya sea un CD o DVD, otro tipo de soluciones son las que se conectan a través de la red, tienen el mismo objetivo que las basadas en un disco de inicio solo que se cargan por medio de la interface de red, tal es el caso de herramientas como Auditor Remote Exploit.

No cabe duda que por el aumento de la cantidad de información digital almacenada y por consiguiente de posible evidencia digital, habrá más casos donde realmente sea imposible hacer el análisis forense de la manera tradicional, por eso que el paradigma de adquisición en vivo ya es una realidad. Sin embargo para elegir cuál es el mejor método para cada investigador, dependerá de las precondiciones y contexto dado, por ejemplo en el caso de una investigación de pornografía infantil por lo general se buscan bitácoras de chats o imágenes, entonces este podría ser un caso candidato para hacerlo de la forma tradicional. Por eso es importante recalcar que para cada caso puede ser diferente, pero la decisión final la tiene el forense.

3.2 EnCASE

EnCase es una herramienta comercial integral de análisis forense dirigido para plataformas Windows principalmente, distribuida por Guidance Software, y es ampliamente usada por profesionales de la seguridad en cómputo. El ambiente integrado de EnCase significa que el software adquiere la evidencia mediante una imagen orientada a flujos de bits propietaria

llamada Evidence File (EF). EnCase monta la imagen EF como una unidad virtual de sólo lectura y reconstruye la estructura del sistema de archivos utilizando los datos lógicos en la imagen. El EF adquirido está disponible como una imagen compresa ligera e incluye verificación de redundancia cíclica y valores hash con MD5 para asegurar la integridad de los datos.

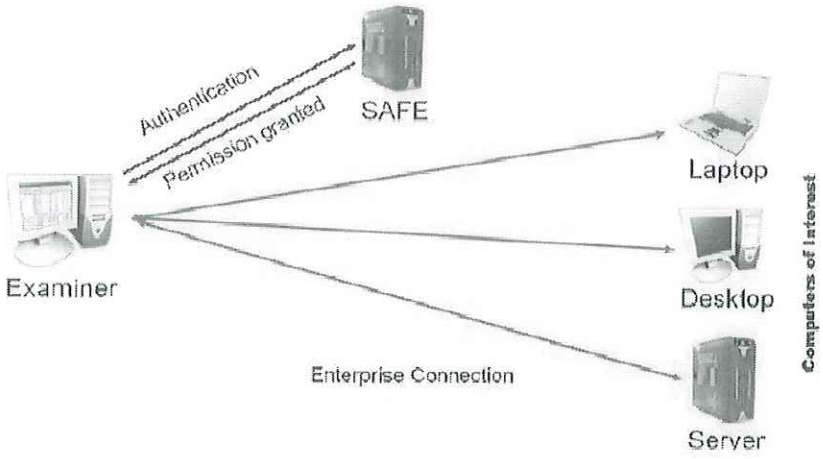

Figura 3.1 Componentes de EnCASE.

Algunas Características.

1. Permite hacer análisis de respuesta a incidentes.
2. Captura automática de los datos volátiles
3. Enumeración de los procesos incluyendo el controlador y servicio.
4. Muestra los archivos actualmente en uso.
5. Análisis del Registro de Windows en vivo.
6. Análisis de impacto de incidente.
7. IDS (Intrusion Detection System) integrado para la respuesta a incidentes automático.

8. Identificación de procesos no autorizados.
9. Detección de procesos ocultos y rootkits
10. Soporte a sistema de archivos y sistema operativo.
11. Varios tipos de adquisición de la evidencia.
12. Condiciones de filtros, consultas y MAC times.
13. Visualización de archivos eliminados y en espacios no asignados.
14. Extracción de archivos, análisis de hardware.
15. Análisis de bitácoras y eventos.
16. Análisis de firma de archivos, claves hash.

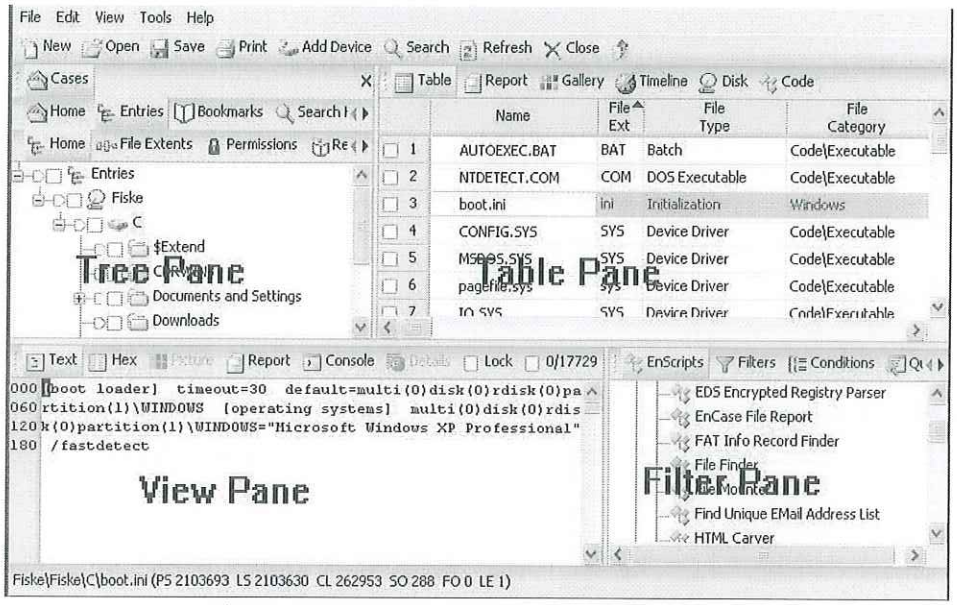

Figura 3.2 Interfaz principal de EnCASE.

La última versión que salió al mercado es la serie 6.x. Sin embargo desde la versión 4.x se han reportado *bugs* y vulnerabilidades, desde corrupción de memoria hasta denegación de servicios. Otras vulnerabilidades reportadas son: problema con las particiones en la que no detecta más de

25 [US-CERT, 2007,1] y el uso de autenticación débil para conectarse con el equipo a examinar [US-CERT, 2007, 2]. Fuera de este tipo de fallos, en general la herramienta es muy buena.

3.3 ILook Investigator.

Una herramienta desarrollada por Elliot Spencer y el Criminal Investigation Division of the U.S Internal Revenue Service U.S Treasury Department. ILook es una herramienta comercial y está diseñada para permitir a un investigador acceder a la partición del sistema de archivos durante el proceso de reunión de evidencia y análisis forense extendido. ILook puede ser usado para crear una imagen de cualquier dispositivo conectado, sin embargo se basa en un mecanismo de escritura de bloques. Los investigadores pueden investigar el mapa de la imagen simplemente recorriéndola para examinar las estructuras de la partición y poder probar la imagen para meta-estructuras específicas o registros raíz. Soporta los siguientes sistemas de archivos: FAT, VFAT, NTFS, Macintosh's HFS, HFS+, Linux's Ext2FS y Ext3FS, Novell's NWFS, CDF, Reiser FS e ISO9660.

Algunas características:

- Posee una ventana de evidencia que te permite navegar la estructura de las particiones y el sistema de archivo del disco en cuestión.
- Despliega un conjunto adicional de directorios virtuales que contienen apuntadores hacia los flujos de archivo que han sido borrados, o están en sectores no asignados.

- Tiene una ventana de archivos que enlista todos los archivos con sus respectivas propiedades, almacenados en el directorio seleccionado en la ventana de evidencia.
- Posee ventana de información que le da al investigador acceso a grupos de información relacionados con los objetos seleccionados en la ventana de evidencia y de archivos.
- ILook también permite al investigador buscar archivos basados en atributos específicos. Búsquedas basadas en fecha-hora usando un calendario básico como las bases para la selección de fecha/sellos de tiempo y visualización.
- Puede realizar búsquedas de los valores hash del contenido del archivo. Reconstrucción de archivos, esto es, la interpretación de un conjunto limitado de formatos de archivos compuestos, y la extracción de datos.
- Categorización y etiquetado de datos, análisis de los flujos de datos, soporta indexación para la búsqueda de archivos, visor de objetos en miniatura.

Para disponer de esta herramienta hay que enviar una carta para ver si alguien es candidato, incluyendo la agencia donde trabaja, nombre especial de la unidad e información de contacto. Al parecer solo está disponible en Estados Unidos.

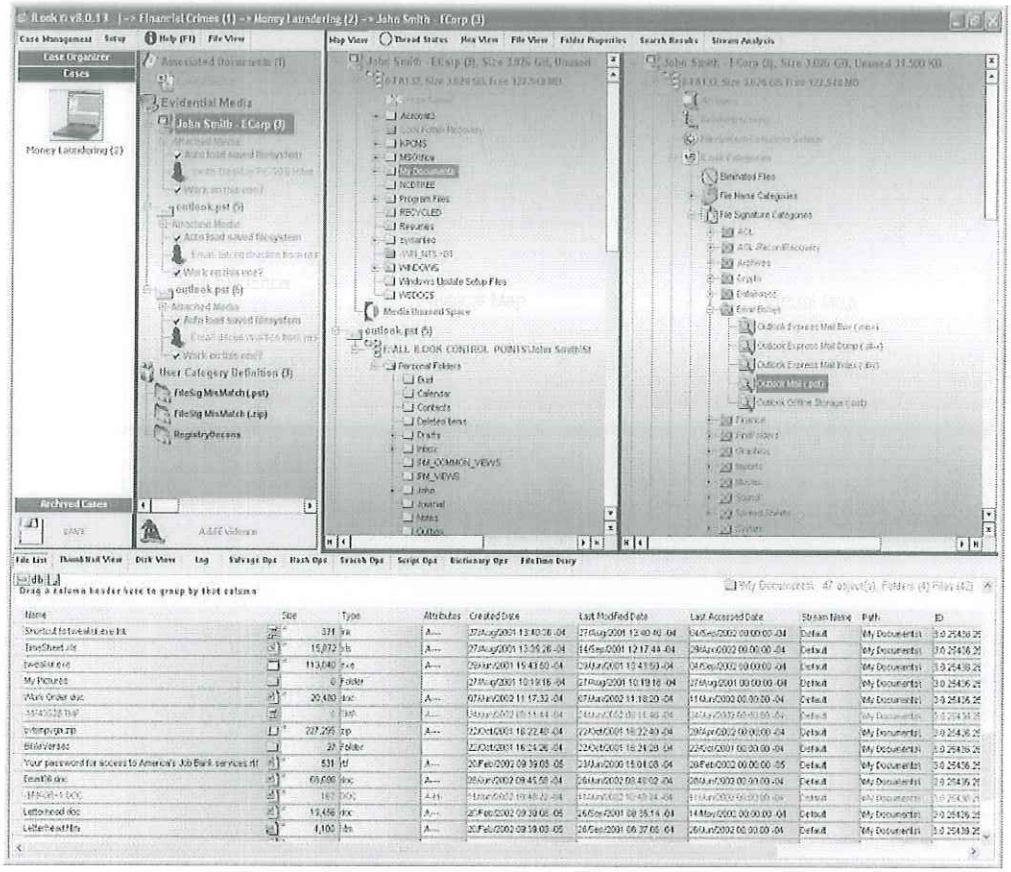

Figura 3.3 Ambiente gráfico principal de ILook Investigador.

3.4 CFIT

Es una herramienta de análisis forense integral desarrollado por DSTO, Departamento de Defensa, Australia. CFIT provee eficiencia y flexibles métodos forenses automatizados para analizar el contenido de los flujos de datos tales como unidades de disco, datos de la red y discos. Las principales ventajas de CFIT son:

1. La habilidad para integrar múltiples herramientas forenses interactivas en una en común.

2. La facilidad para agregar nuevas herramientas forenses al ambiente de trabajo.
3. La habilidad para capturar la historia de una investigación de manera visual.

El ambiente investigativo básico en CFIT es el caso, en el cual los investigadores pueden trabajar individualmente o en equipo, para resolver uno o más casos criminales. La plataforma CFIT incluye administración de casos, manipulación y acceso a los flujos de datos, visualización de datos y procesamiento forense. También incorpora un lenguaje de ambiente visual en 2 dimensiones llamado Picasso, para expresar gráficamente un caso sobre un ambiente de trabajo visual. Herramientas forenses que incluye son analizador de disco duro, analizador del sistema de archivos, extractor de bitácoras, motor de búsquedas ontológicas, extractor del espacio no asignado, organizador de eventos en el tiempo y herramienta para análisis en línea de tiempo.

Un ejemplo de herramienta forense es Ferret Discovery Engine, una herramienta para generación ontológica de conceptos textuales, navegación y búsqueda. Puede ser usado para buscar archivos o documentos para conceptos particulares e identificar aquellos documentos que probablemente tendrían significado.

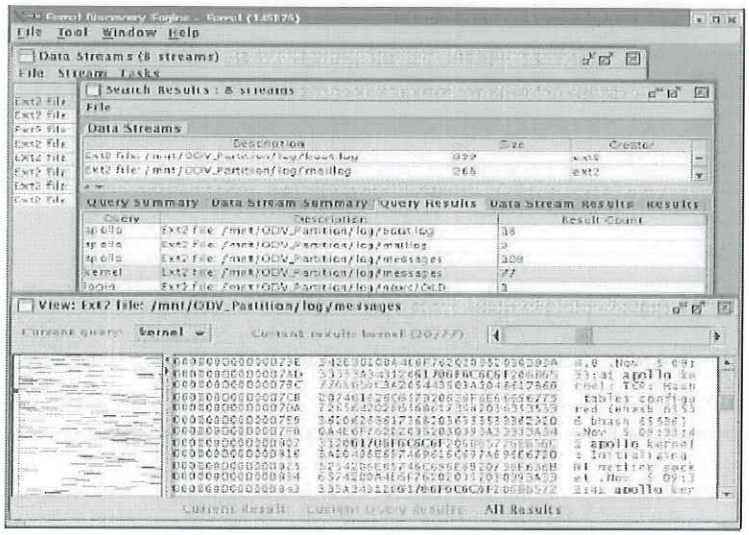

Figura 3.4 Interfaz del motor de descubrimiento Ferret de CFIT.

3.5 Helix.

Helix es una distribución gratuita especial del disco en vivo Knoppix de Linux. Helix son muchas aplicaciones dedicadas para la respuesta a incidentes y análisis forense. Con Helix puedes obtener una imagen de la memoria física ya sea el disco duro, memoria RAM o cualquier dispositivo conectado a la computadora. También tiene capacidad para gestionar la respuesta a incidentes de seguridad en cómputo. Una característica común es que contiene una serie de herramientas por diversos programadores, que ofrecen cierta funcionalidad como: realizar un escaneo del registro de Windows por patrones, una herramienta para revelar los procesos ocultos y librerías cargadas en el sistema, un método para la detección de rootkits, visualización de la lista de usuarios conectados al sistema, buscador de patrones de archivos, escuchador de modificación en un determinado directorio. También te permite analizar las imágenes mediante patrones, maneja una sección para agregar notas a la investigación o el caso en cuestión y

permite visualizar el historial de internet e información de software y hardware relevante del sistema.

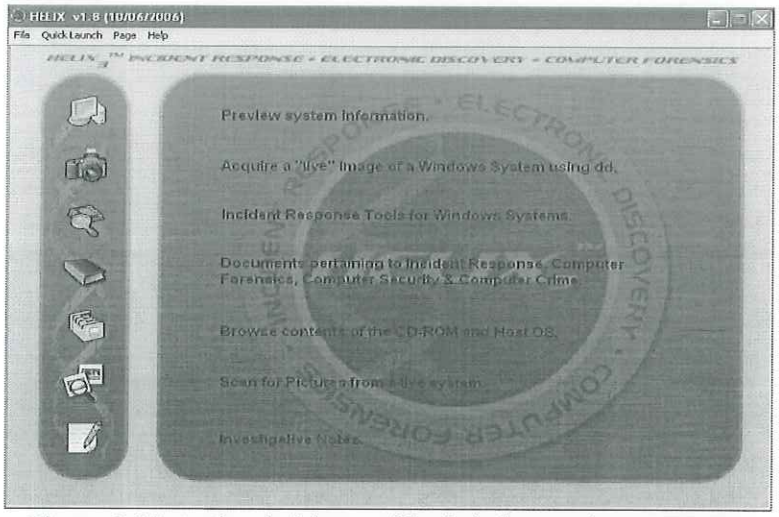

Figura 3.5 Interfaz de bienvenida de la herramienta HELIX.

3. 6 F.I.R.E.

FIRE es un CD de inicio portable, que es gratis. Está basado en una distribución Linux con la meta de proveer un ambiente inmediato para llevar a cabo un análisis forense, respuesta a incidentes, recuperación de datos, escaneo de virus y aseguramiento de vulnerabilidades. También provee herramientas necesarias para análisis forense en vivo en Windows XP, SPARC Solaris y distribuciones Linux x86.

F.I.R.E es un conjunto de herramientas, como buscador de archivos por atributos MAC, recuperación de contraseñas del CMOS y del sistema operativo, visualización de los procesos ocultos, historial de internet entre otras características. Similar a Helix.

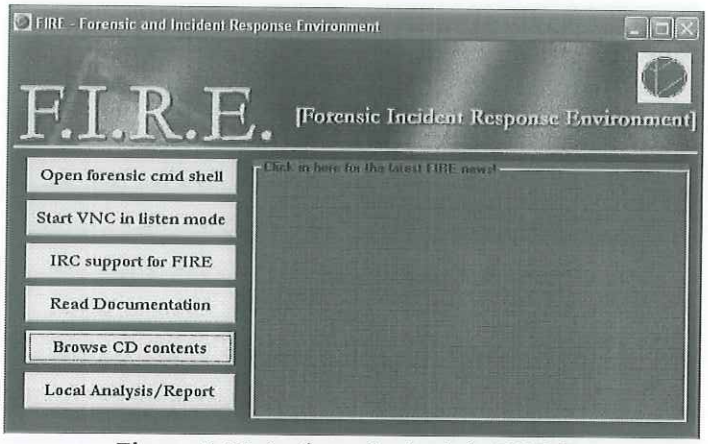

Figura 3.6 Interfaz principal de F.I.R.E.

3.7 ProDiscover.

ProDiscover es una herramienta de análisis forense en vivo, que nos permite realizar imagen del disco duro o de la memoria RAM, su organización está dirigida por proyectos. Entre la funcionalidad que ofrece este programa se encuentra: recuperación de archivos que han sido eliminados, visualización de su contenido, consultas en el registro, buscador de actividad de internet y la caché de los navegadores. ProDiscover tiene una vista general de resultados por módulo de búsqueda, agregando una cadena de custodia muy general.

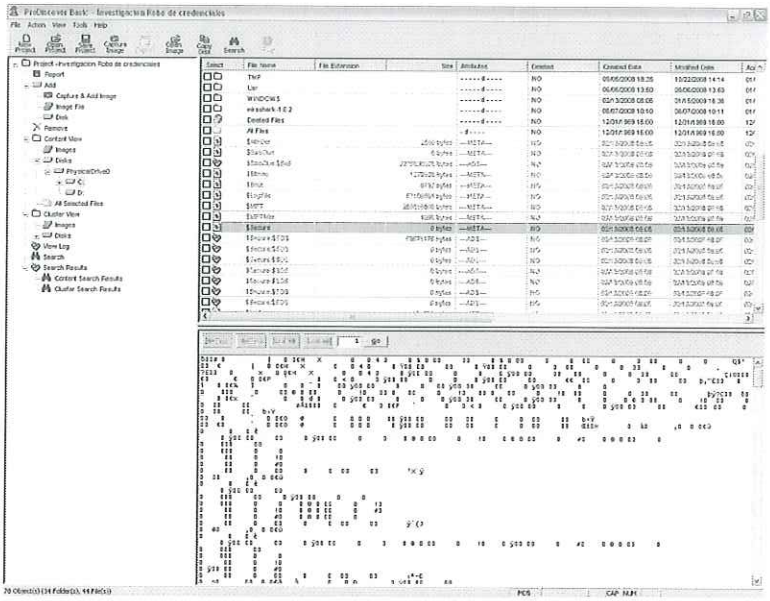

Figura 3.7 Interfaz de trabajo principal de ProDiscover de TechPathWays.

Lo malo de esta herramienta es que no tiene una buena organización para agregar metadatos a algún tipo de evidencia. Solo nos ofrece agregar una descripción y un número de seguimiento.

Otras herramientas: Auditor, Operator, P.H.L.A.K, BackTrack, Disk Investigator.

Como vemos hay diversas herramientas con diferentes características, nuestra herramienta tendrá las características que se pueden ver marcadas por una X en la tabla 2.

3.8 Características propuestas vs Herramientas presentadas.

Característica/Herramienta	EnCASE	ILook	CFIT	Helix	ProDiscover	Hefoclase
Visualizar procesos corriendo	—	—	—	X	—	X
Visualizar procesos ocultos	X	—	—	X	—	X
Búsqueda en el sistema de archivos	X	X	—	X	X	X
Búsqueda mediante MAC Times	X	X	X	—	X	X
Criterio de búsqueda hasta por hora	—	—	—	—	—	X
Verificación de firma del archivo	X	X	X	—	X	X
Visualización del archivo modo texto	X	X	X	X	X	X
Visualización del Historial de IE	X	/	/	X	/	X
Recuperación archivos en la papelera	X	/	X	X	X	X
Visualización de Cookies de IE	X	/	/	X	/	X
Explorador de registro de sistema	X	—	/	X	/	X
Búsqueda de claves/llaves en registro	X	—	/	X	/	X
Explorador de registro de sistema	X	—	/	X	/	X
Búsqueda de claves ocultas	—	—	—	—	—	X
Múltiple administración de casos	X	X	X	—	X	—
Captura automática de la RAM	X	X	X	X	X	—
Visualización archivos eliminados	X	X	X	X	X	X
Visualización espacio no asignado HD	X	X	—	—	X	—
Adquisición imagen del disco duro	X	X	X	X	X	—
Búsqueda de patrones en disco duro	X	X	X	—	X	—
Encriptación de evidencia (HASH)	X	X	X	—	X	—
Soporte HFS, Linux ext2, ext3	X	X	—	X	X	—
Visualizador Hexadecimal	X	X	X	X	X	—
Soporte montar unidad remota	X	—	—	X	—	—
Extracción de EXIF de las imágenes	X	—	—	—	—	—

Tabla 2. Características de la herramienta propuesta vs Herramientas presentadas.

- X – Cumple, / - Lo hace de otra forma, — - No cumple.

CAPÍTULO 4 Metodología

En este capítulo se describe cual es la metodología, procesos y modelos que se siguieron para el desarrollo de la herramienta Hefoclase.

Para desarrollar el sistema propuesto, es necesario tener una idea de cómo llevarlo a cabo, tener definido que proceso de software se va a seguir. Partiendo de estos conceptos, describiremos a continuación, el proceso que se siguió para desarrollar la herramienta Hefoclase, partiendo desde la investigación preliminar avanzando por el análisis, diseño y evolución de la herramienta hasta su estado actual.

4.1 Modelo de desarrollo.

Una definición clara de modelo de proceso de software puede ser la siguiente: "es una representación simplificada de un proceso del software, presentada desde una perspectiva específica" [Pressman R., 2001]. La mayoría de los modelos de proceso de software se enfocan en uno de los 3 modelos generales o paradigmas:

1. En Cascada. Considera las actividades anteriores y las representa como fases de procesos separados, tales como la especificación de requerimientos, el diseño del software, la implementación, las pruebas, etc. Después de que cada etapa queda definida se "notifica" y el desarrollo continúa con la siguiente fase.
2. Iterativo. Se concibe con pequeñas entregas del software que van acercándose cada vez más al sistema en su totalidad. El sistema inicial se desarrolla rápidamente a partir de

especificaciones muy abstractas (entregas). Luego se refina a partir de las peticiones del cliente para producir un sistema que satisfaga las necesidades del mismo.

3. Métodos Formales. Son enfoques matemáticos para resolver los problemas del software en los requerimientos, especificación y niveles. Comúnmente utilizan teoría de autómatas.

El modelo que utilizamos para el desarrollo del sistema Hefoclase, es un modelo de desarrollo en cascada pero con algunas variaciones que a continuación describiremos.

4.2 Modificación del modelo en cascada (sashimi).

En respuesta a los problemas percibidos con el modelo en cascada normal, varias modificaciones han sido introducidas. Una de las más importantes es el llamado modelo sashimi (como el pescado japonés sashimi) o cascada con retroalimentación originalmente propuesto por DeGrace [Wikipedia, 2007]. La principal característica de este modelo es el traslape entre las fases, es decir que de una fase a otra se puede regresar. Por ejemplo, problemas de la fase de implementación se pueden descubrir en la fase de diseño, entonces se podría regresar para rediseñar y continuar con el desarrollo. Con esta característica este modelo alivia los principales problemas del modelo en cascada normal.

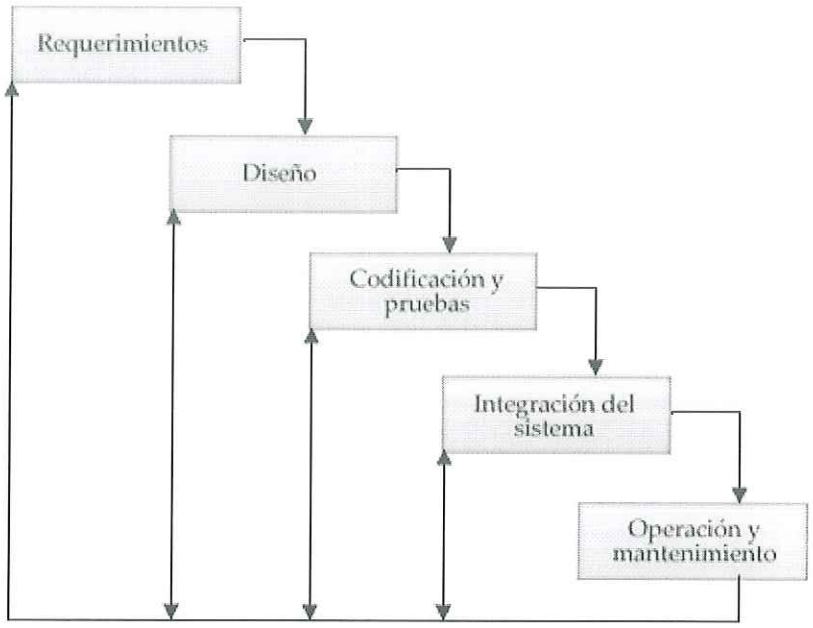

Figura 4.2. Modelo de cascada con retroalimentación (sashimi).

El modelo en cascada con retroalimentación fue el que se utilizó para el desarrollo del sistema Hefoclase.

4.3 Herramientas CASE (Ingeniería del software asistida por computadora).

El software que se utiliza para ayudar a las actividades del proceso del software como el análisis de requerimientos, diseño, implementación y pruebas son llamado herramientas CASE, haciendo alusión a que se está asistiendo al ejecutor del proceso de desarrollo para completar más fácilmente el producto final de software. Estas herramientas incluyen editores de diseño, diccionario de datos, compiladores, depuradores, ambientes integrados, generadores de diagramas, generador de documentación, etcétera.

La tecnología CASE proporciona ayuda al proceso del software automatizando algunas de sus actividades, así como proporcionando información acerca del software en desarrollo.

Esta tecnología está disponible para la mayoría de actividades rutinarias del proceso de desarrollo del software. Hay que decir que estas herramientas ayudan en la calidad o productividad de los desarrolladores en un cierto grado, que es difícil medirlo en porcentajes ya que depende de cada equipo de trabajo o de cada persona y del tipo de requerimientos de su sistema, ya que por lo general el proceso de desarrollo depende muchas veces de la creatividad o también de aspectos sociales que se salen de los orígenes de las herramientas CASE.

4.4 Preparación del ambiente de trabajo.

Para llevar a cabo el desarrollo del sistema, se necesitó configurar e instalar un equipo de cómputo con el sistema operativo Windows XP, se instalaron las herramientas CASE que se utilizaron a lo largo del desarrollo, como lo son: Star UML y Visio 2007. También se requirió de la instalación del ambiente de desarrollo de Microsoft Visual Studio 2005.

Además se tuvo la necesidad de conexión a internet para consultas de literatura e investigación.

CAPÍTULO 5 Análisis y Diseño

En este capítulo se describen las estructuras estáticas del sistema mediante modelos gráficos que permitan un mejor entendimiento. De acuerdo al modelo de desarrollo que utilizamos (cascada con retroalimentación), la primera fase es la de Análisis de requerimientos que consiste en definir las tareas que debe hacer nuestro sistema. El propósito del análisis es definir todas las clases que son relevantes al problema que se va a resolver, las operaciones y atributos asociados, así como las relaciones y comportamientos con ellas.

5.1 Definición de Requerimientos.

Los requerimientos son una descripción de las necesidades o deseos del producto. El principal objetivo de la fase de requerimientos es de identificar y documentar lo que realmente se necesita en una forma que claramente entienda el cliente y el grupo de desarrollo. El reto aquí es definir los requerimientos sin ambigüedad, identificando el riesgo que se tiene y no que haya sorpresas cuando el producto esté en etapa de desarrollo. A continuación se definen los requerimientos y se dividen en 2 tipos, los requerimientos funcionales, que es lo que realmente se quiere hacer y los requerimientos no funcionales que son restricciones operativas del sistema. También cabe señalar que cada requerimiento tiene una nomenclatura o notación que nos permite identificar a cada requerimiento como único.

Las características de la herramienta propuesta se definen a continuación en los requerimientos funcionales.

Requerimientos Funcionales:

RFT-1 El sistema deberá tener un mecanismo para recopilar información acerca del hardware y software del sistema en el que se ejecuta.

RFT-2. El sistema deberá tener un mecanismo para hacer búsquedas en el registro del sistema operativo.

RFT-3. El sistema deberá tener un módulo para realizar búsquedas por atributos en el sistema de archivos.

RFT-4. El sistema deberá tener un módulo para mostrar una lista de los procesos y rutas de los binarios en ejecución.

RFT-5. El sistema deberá tener un proceso para verificar las firmas de los archivos por su extensión.

RFT-6. El sistema deberá tener un proceso para mostrar una lista de los archivos eliminados y ocultos.

RFT-7. El sistema deberá tener un mecanismo para mostrar el contenido de los archivos.

RFT-8. El sistema deberá tener un mecanismo para seleccionar, leer y desplegar el contenido de las cookies del sistema.

RFT-9 El sistema deberá tener un mecanismo para visualizar el historial de páginas visitadas.

RFT-10. El sistema deberá poder desplegar los resultados de cada módulo.

RFT-11. El sistema deberá contar con un mecanismo para que el usuario pueda clasificar los resultados.

RFT-12. El sistema deberá desplegar un sumario total de los resultados.

RFT-13. El sistema deberá tener un mecanismo para guardar los resultados.

Requerimientos No Funcionales:

RNF-1. El sistema deberá poder ejecutarse en un medio de almacenamiento de sólo lectura.

5.2 Descripción General.

La herramienta que se diseñará tendrá el mecanismo de adquisición en vivo, se ejecutará en un medio no volátil o volátil, ya sea un CD o un USB. Se podrá ejecutar en un disco de arranque o simplemente con la ejecución de un archivo binario en el dispositivo de almacenamiento volátil. También será posible copiarlo al sistema de archivos, lo ideal es no copiarlo para no sobrescribir áreas en el disco duro y perder información importante, aunque en realidad el binario final no debe ocupar mucho espacio en disco. Al empezar a ejecutarse el sistema, presentará su interfaz con un menú de los módulos correspondiente de análisis forense, de ahí el usuario podrá seleccionar que tipo de búsqueda o análisis que quiere realizar, ya sea búsqueda en el registro, consulta de los tiempos de creación, modificación o acceso de los archivos, recuperar archivos eliminados o los demás criterios de adquisición propuestos; una vez terminada la búsqueda, se podrá clasificar su tipo. Esta clasificación son como notas relacionadas a la evidencia digital, después se podrá crear un sumario de los resultados de los análisis de cada módulo con su respectiva clasificación, con el propósito de que el investigador tenga una vista global y contextual de la evidencia reunida, y que pueda definir cuál fue la causa del probable incidente, finalmente se podrá guardar esa evidencia e información contextual a un medio volátil como una memoria USB o al sistema de archivos, en caso de que no se contara con la primera opción.

5.3 Diagrama de Casos de uso.

Ningún sistema existe en aislamiento, todos los sistemas interactúan con los humanos o actores automatizados que usan el sistema para algún propósito, y estos actores esperan que el sistema se comporte de maneras predecibles. Un caso de uso especifica el comportamiento de un sujeto o parte del sistema, descrita en secuencia de acciones, incluyendo variantes que el sujeto puede llevar a cabo para cosechar un resultado observable de valor hacia o de un actor.

A continuación se muestran los casos de uso del sistema Hefoclase.

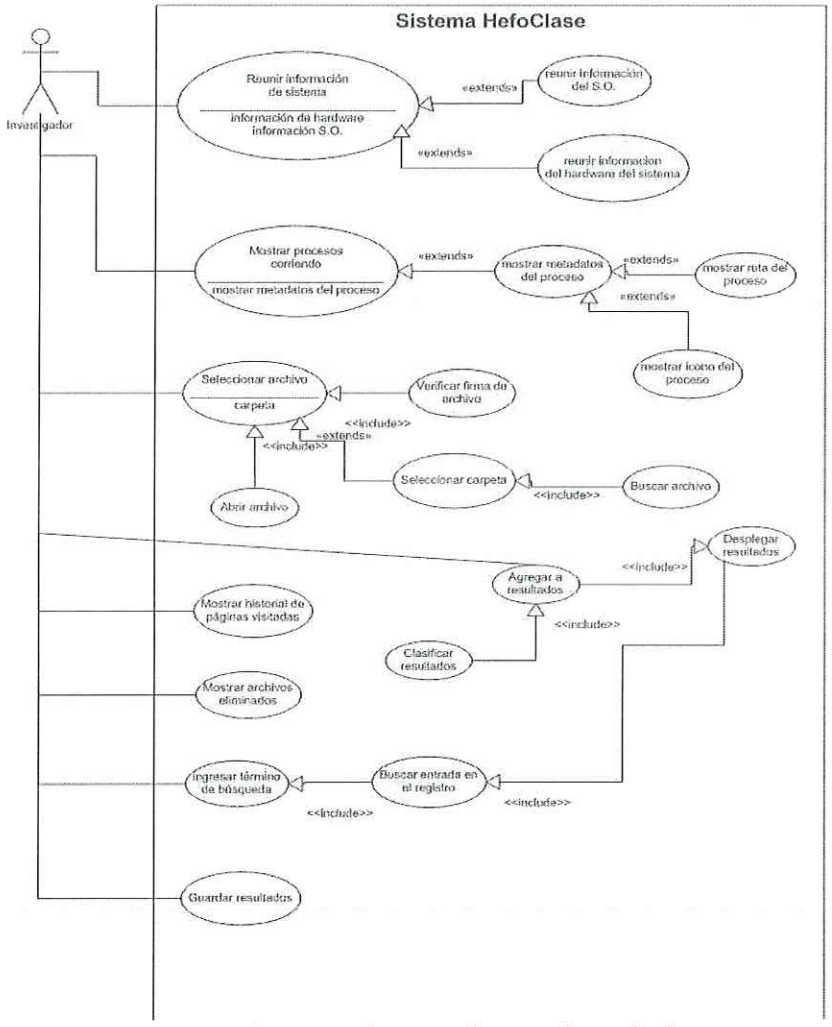

Figura 5.1 Diagrama de casos de uso de Hefoclase.

Como vemos en la figura 5.1, estamos describiendo los requerimientos del sistema Hefoclase, identificamos al actor principal y único, que es el investigador.

5.4 Diagrama de Clases.

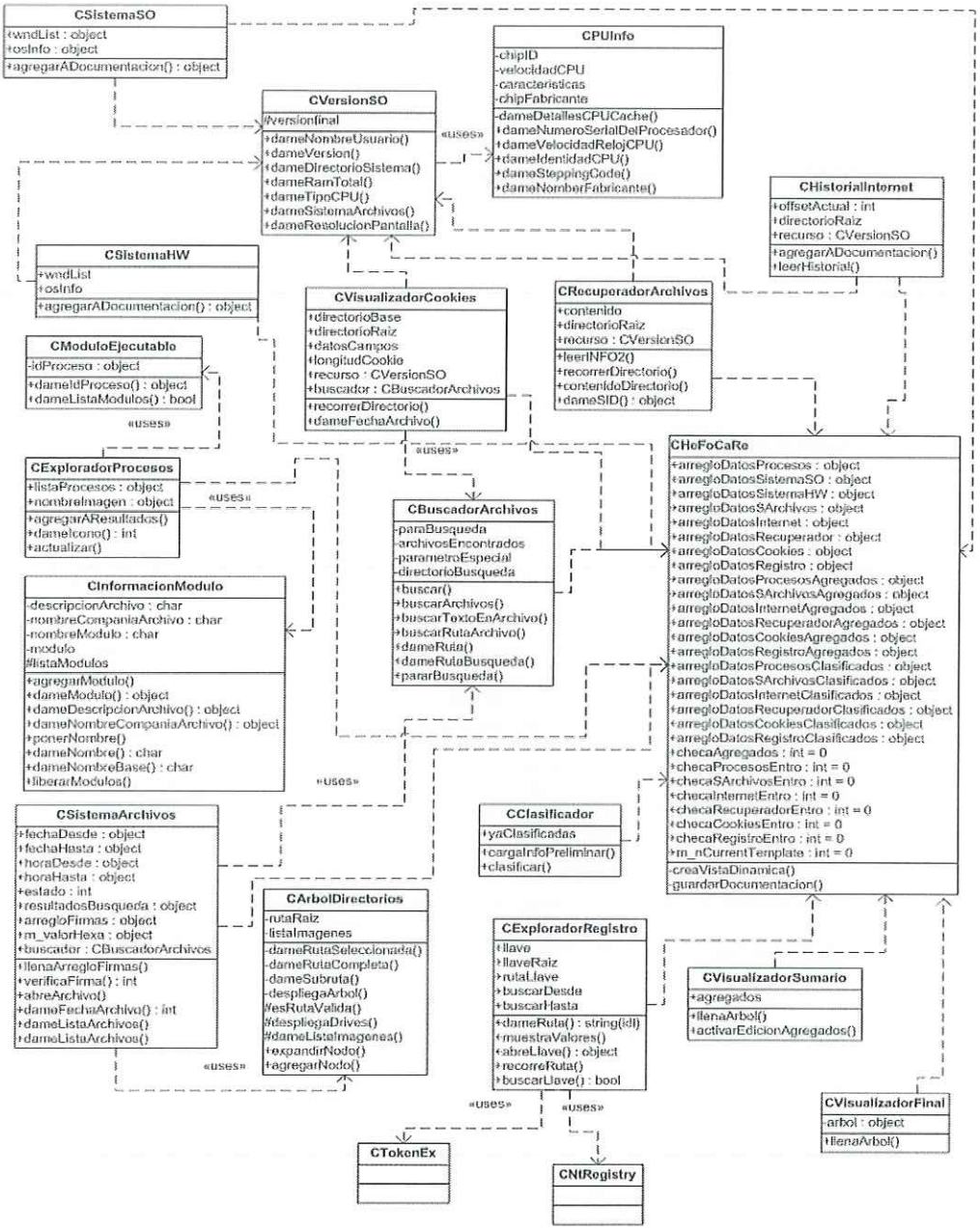

Figura 5.2. Diagrama de clases del sistema Hefoclase.

5.5 Diagramas de Secuencia.

Un diagrama de secuencia es un diagrama de interacción que enfatiza el orden en el tiempo de los mensajes. Describe las partes dinámicas del sistema.

Diagrama de Secuencia Mostrar información del S.O.

En la figura 5.3 se muestra el diagrama de secuencia para obtener información del sistema operativo. Éste comienza cuando el usuario selecciona la opción de S.O, entonces el sistema escucha la acción y manda el mensaje de crear vista dinámica al manejador principal Hefoclase. Hefoclase crea el objeto de la vista, pero falta agregarle los elementos necesarios. Para ello el objeto creado se registra con el manejador de vistas, si todo está bien entonces Hefoclase manda que la vista se actualice para cargar los datos de la misma. Luego la vista crea un objeto osInfo que es el que llamará algunos métodos para cargar los datos requeridos, si cada llamada al método es correcta entonces se actualiza la vista, una vez ejecutados todos los métodos se notifica a Hefoclase y este repinta la vista y la muestra al usuario.

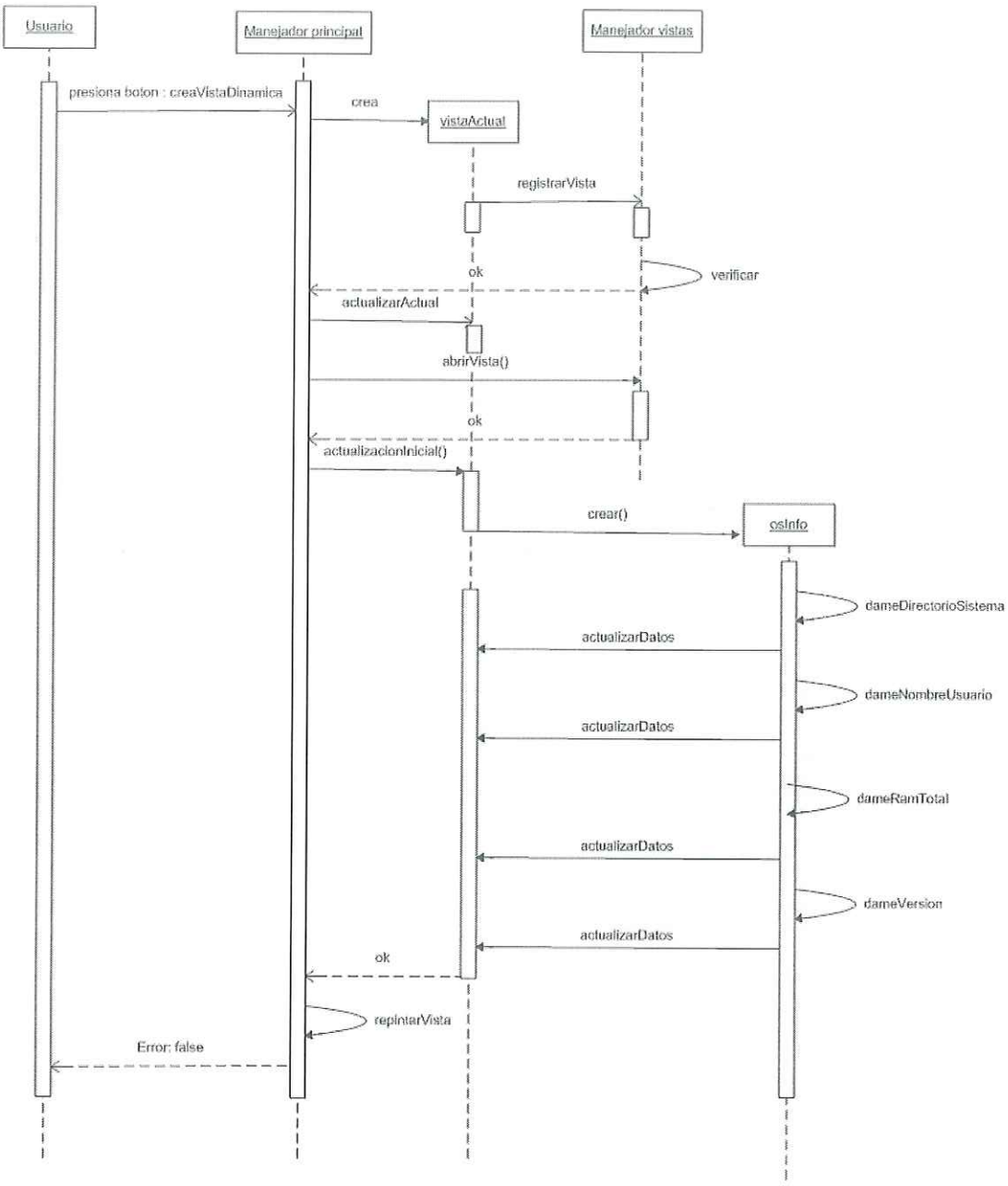

Figura 5.3 Diagrama de secuencia muestra información del S.O.

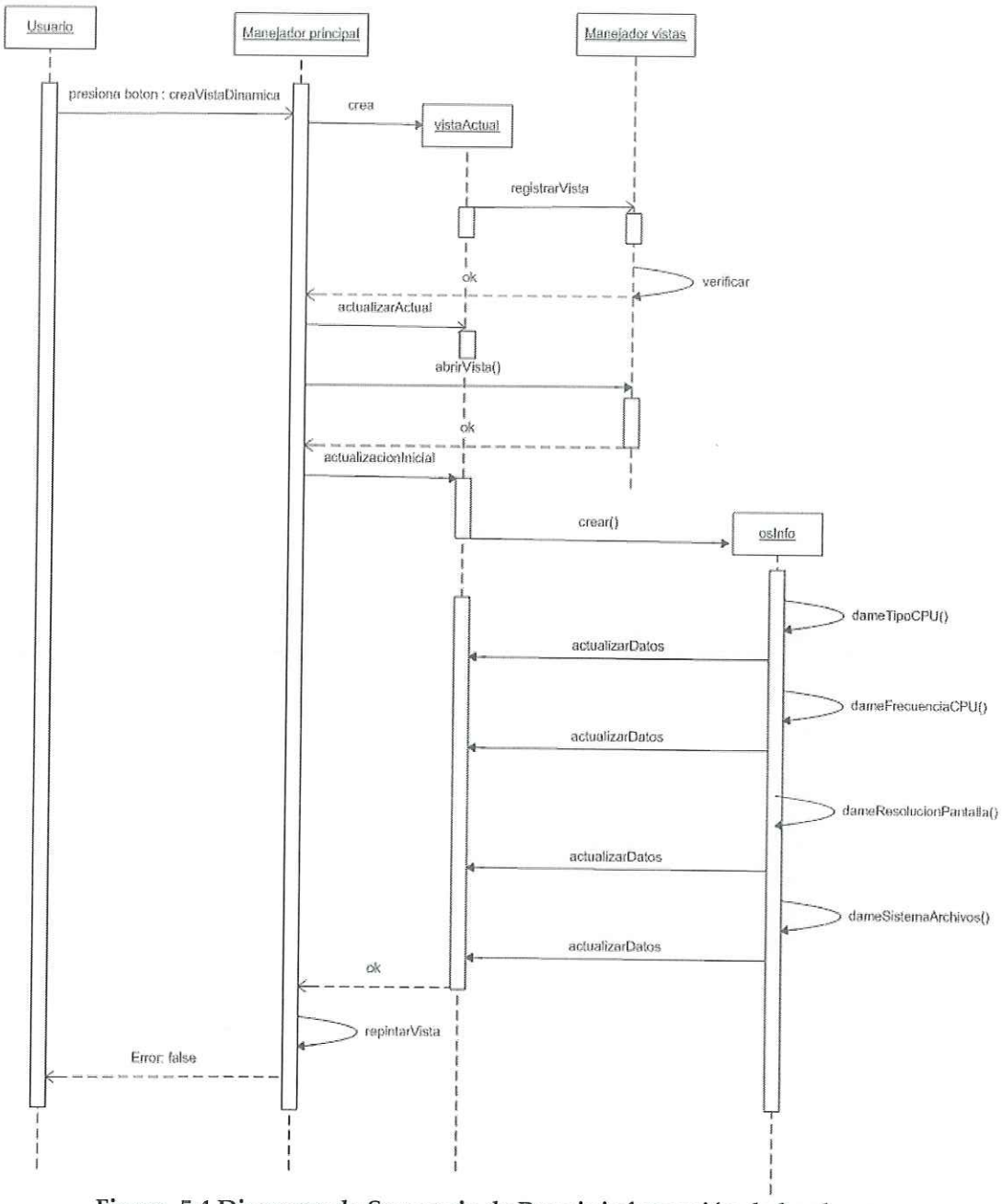

Figura 5.4 Diagrama de Secuencia de Reunir información de hardware.

En el diagrama anterior se muestra el diagrama de secuencia Reunir información de hardware. Este describe un comportamiento similar al anterior caso de uso, la única diferencia

son los métodos llamados por el objeto osInfo. Hay que decir que estos métodos siempre regresan una cadena ya que está inicializada en vacío y que si no se actualiza quedará como tal y por lo tanto en la vista no se verá ningún resultado ya que el objeto cargado será vacío, simplemente se verá un fondo blanco que falta desplegar los datos.

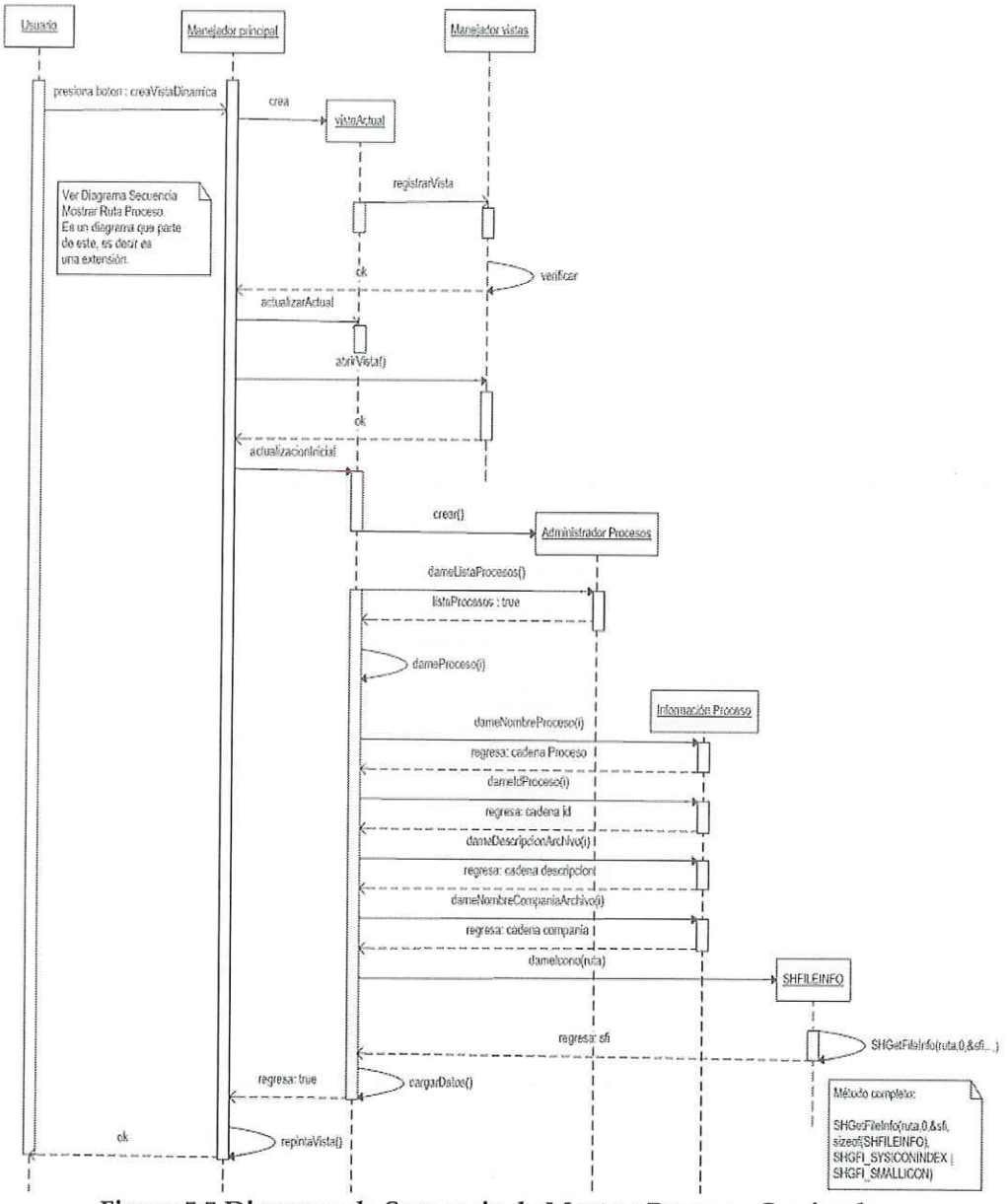

Figura 5.5 Diagrama de Secuencia de Mostrar Procesos Corriendo.

Este módulo mostrará los procesos que se están ejecutando al momento de cargar la aplicación, aunque también se detecta en la lista de procesos. Similar que los anteriores diagramas, este empieza dando un clic en el botón de explorador de procesos, para ello envía un mensaje de crear la vista dinámica al manejador principal, de nuevo este crea el objeto donde estará almacenada la vista y éste último se registra ante el manejador de vistas y manda al principal un mensaje de que ya se pudo registrar. Luego el manejador principal obliga a inicializar los datos de la vista y se actualiza. Se abre la vista para poderse cargar los datos y entonces se crea un objeto administrador procesos, la vista le manda un mensaje, le pide al administrador de procesos que le mande la lista, el administrador se la manda. La vista entonces empieza un ciclo en el cual para cada proceso se pide información para sacar sus metadatos, para lo cual se lo pide al objeto *información proceso*. Después de una serie de mensajes y contestaciones para adquirir los metadatos del proceso, el último dato pedido es el de la imagen del icono de cada proceso, la vista se lo pide al objeto SHFileInfo, éste a su vez manda a llamar el método para cargar el icono en memoria y devolverlo, si no encuentra un icono en ese proceso entonces manda uno predeterminado. Por último la vista carga estos datos y le avisa al manejador principal que esta lista para que lo repinte y los datos se desplieguen en la vista del usuario.

Diagrama de Secuencia Mostrar Ruta del proceso.

El objetivo del mismo es que muestra la ruta completa y actual del proceso que se escoja de la lista. Este empieza cuando el usuario da clic sobre algún proceso de la lista o bien lo seleccione con el teclado, después la lista de procesos lee el objeto que ha sido seleccionado y entonces

manda un mensaje al área de texto para que se actualice, si el proceso de actualización ha sido correcto entonces regresa verdadero sino regresa falso.

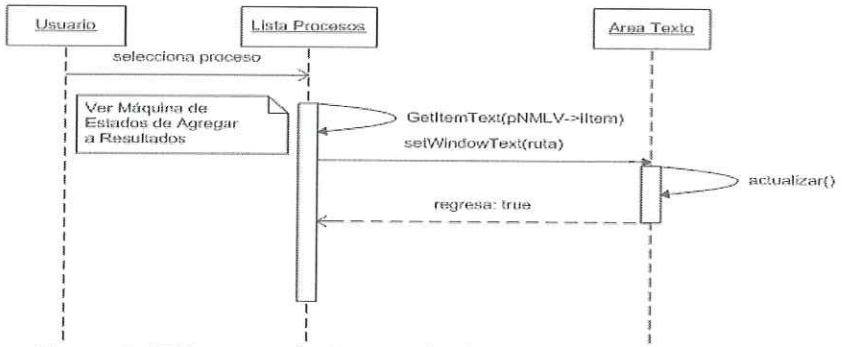

Figura 5.6 Diagrama de Secuencia de Mostrar Ruta del proceso.

Diagrama de Secuencia Verificar Firma.

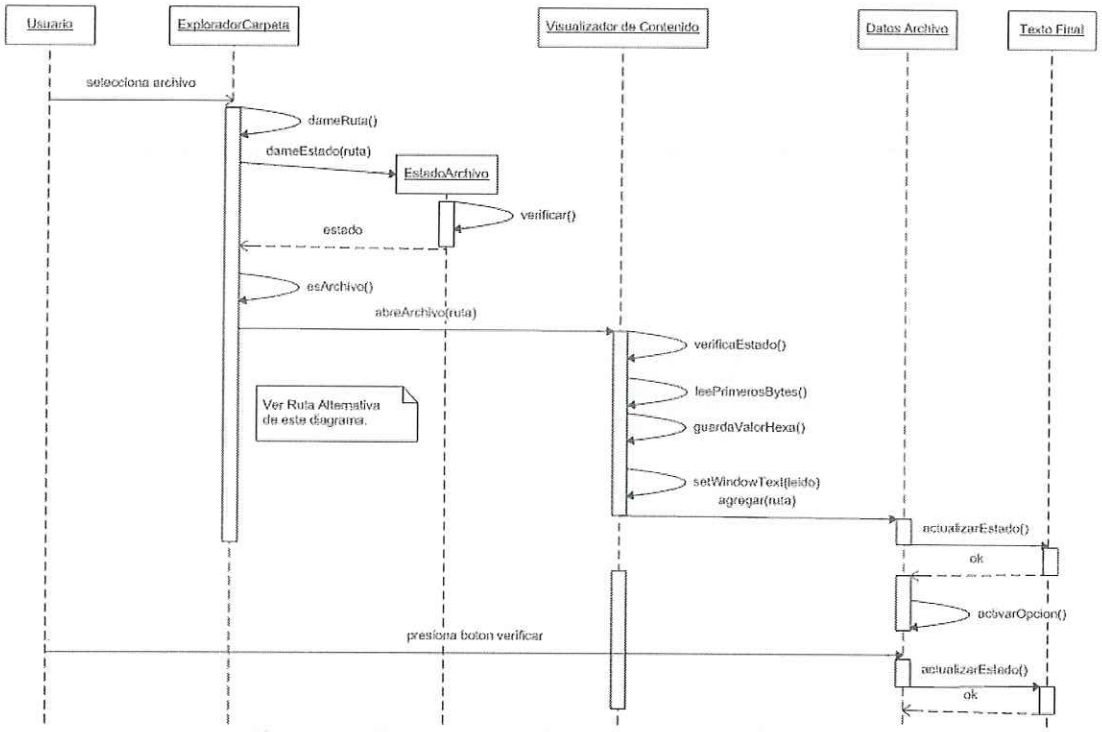

Figura 5.7 Diagrama de Secuencia de Verificar firma.

En el diagrama 5.7 mostramos una manera de llevar a cabo la verificación de una firma de un archivo, empieza cuando se selecciona algún elemento del explorador de carpeta, después se

verifica que sea un archivo lo seleccionado con la ayuda de un objeto estado, una vez hecho esto, el sistema abre el archivo para ver si se puede leer y entonces comienza a leer los primeros bytes del mismo para sacar su firma y la guarda para luego compararla con la base de datos interna del sistema. También lo leído se agrega en forma de texto al visualizador de contenido del archivo. Después se agrega la ruta al visor de datos de archivo y se actualiza el estado final, entonces se activa la opción verificar firma para que el usuario pueda seleccionarla y así terminar de cargar el resultado de este proceso en la vista del estado final.

Diagrama de Secuencia Alternativo de Verificar Firma.

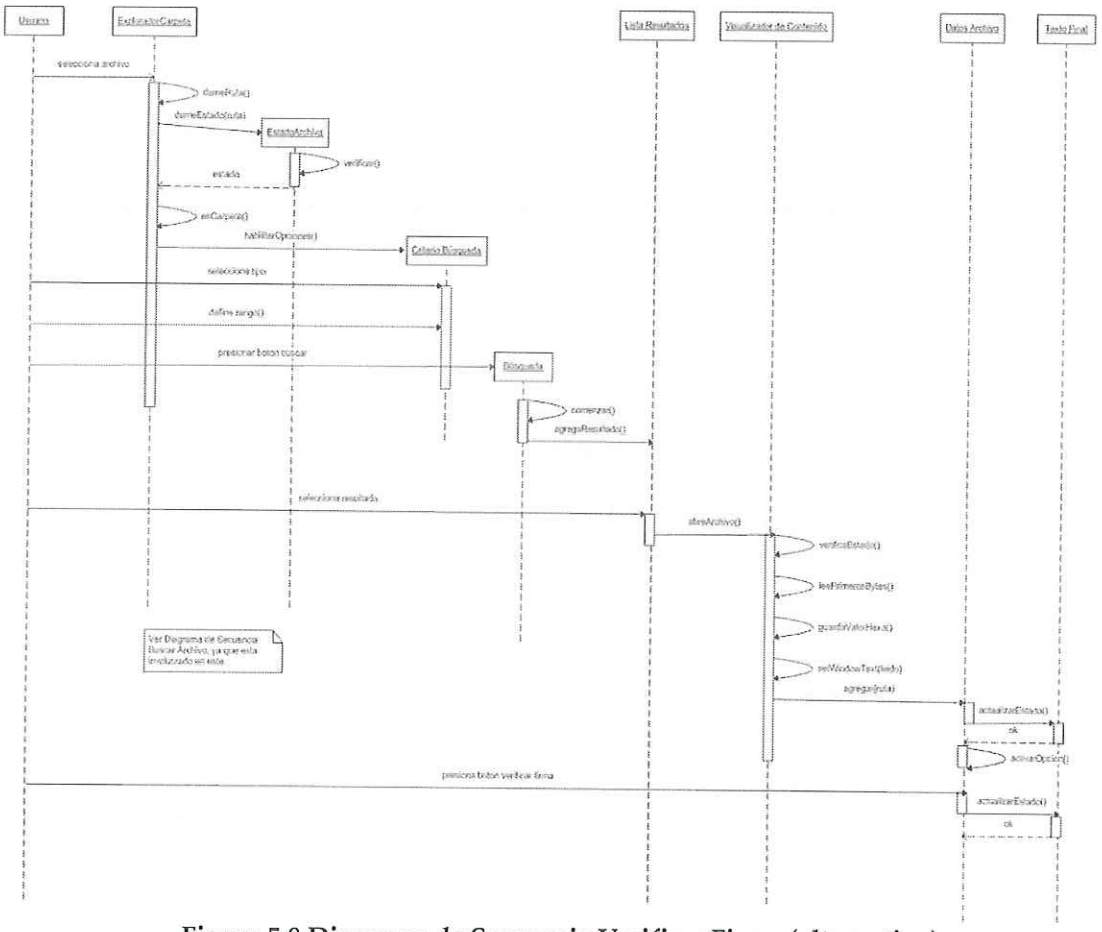

Figura 5.8 Diagrama de Secuencia Verificar Firma (alternativo)

En este diagrama, se sigue otro camino para verificar una firma, aunque el resultado es el mismo. Es importante describir la otra posibilidad de hacer este proceso, el cual comienza como el anterior seleccionando algo en el explorador de carpeta. En el caso anterior verificaba si era un archivo lo seleccionado con la ayuda de un objeto estado. Ahora lo primero que se hace es comprobar que sea una carpeta, después se activan las opciones para definir los criterios de la búsqueda de archivos por atributos, entonces el usuario ya podrá escoger el tipo y rango, luego si el usuario presiona el botón buscar, empieza la búsqueda y cada resultado se agregará a la lista de resultados, en este punto el usuario podrá de nuevo verificar la firma, seleccionando un resultado de la lista, si esto sucede, el sistema abre el archivo para ver si se puede leer y entonces comienza a leer los primeros bytes del mismo para sacar su firma y lo guarda para luego compararlo con la base de datos interna. Después se agrega la ruta al visor de datos de archivo y se actualiza el estado final, ahora ya está listo para que el usuario si así lo desea, oprima el botón verificar firma y de nuevo cargue el resultado en la vista del estado final.

Diagrama de Secuencia Buscar Archivo.

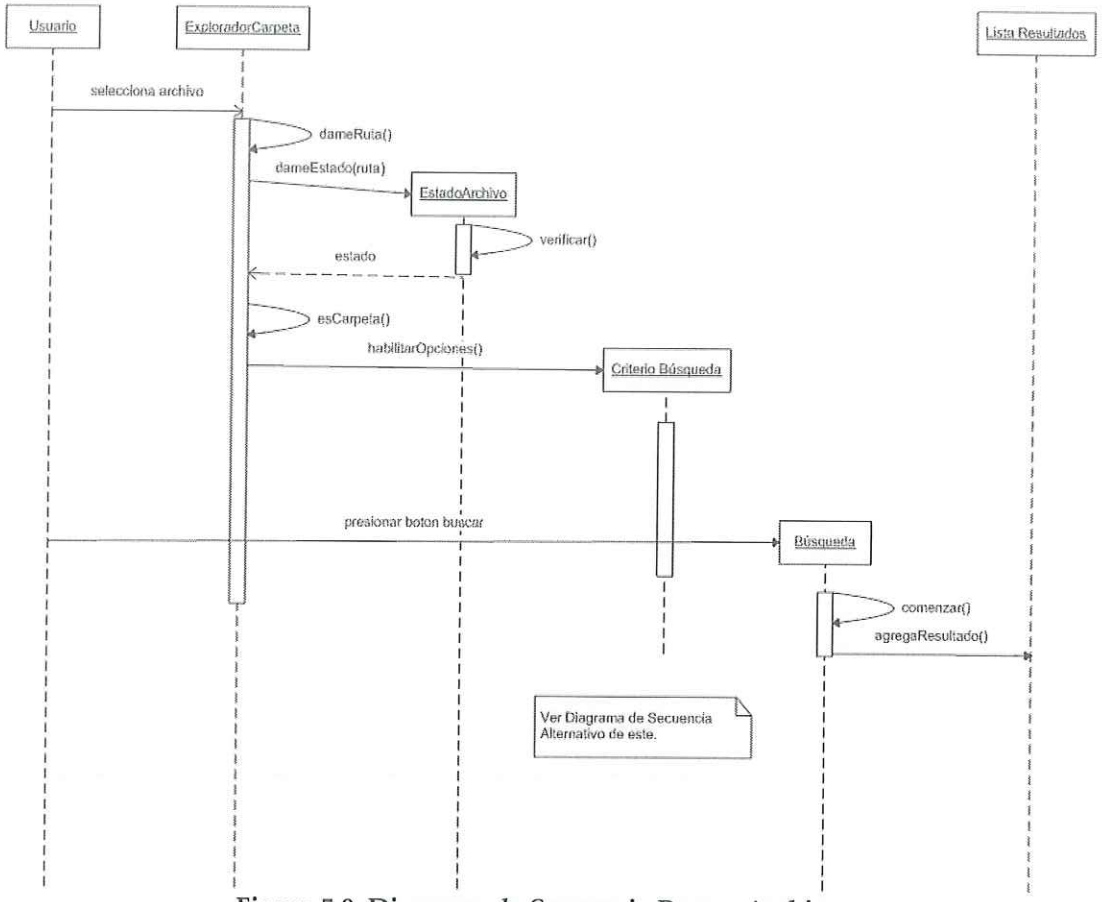

Figura 5.9. Diagrama de Secuencia Buscar Archivo.

Este comportamiento empieza cuando el usuario escoge un archivo del seleccionador/explorador, después este crea un objeto estado para verificar el estado del archivo y determinar si lo seleccionada es una carpeta o un archivo común, si es una carpeta entonces se habilitan las opciones del criterio de búsqueda, entonces el usuario ya puede presionar el botón buscar, una vez hecho esto se inicializa la búsqueda, cada resultado que concuerde con los criterios especificados se agregará a la lista. Hay que señalar que si el usuario no especifica los criterios de búsqueda, entonces se toman los valores que tiene predeterminado para proceder con la misma. Esos criterios son: fecha de modificación y la fecha actual. Por eso que existe un

diagrama de secuencia alternativo donde se muestra esta diferencia con el actual, que es la definición específica de los criterios de búsqueda. A continuación en la figura 5.10, se muestra el diagrama de secuencia alternativo de buscar archivo. Las únicas diferencias con el diagrama anterior, son que cuando se activan las opciones de criterio, el usuario ya puede definir el tipo de parámetro que va a buscar ya sea: fecha de modificación, fecha de acceso y fecha creación; y el rango de fechas donde se define desde que fecha va empezar la búsqueda y hasta cual va a terminar. Los resultados que concuerden con estos criterios serán agregados a la lista de resultados.

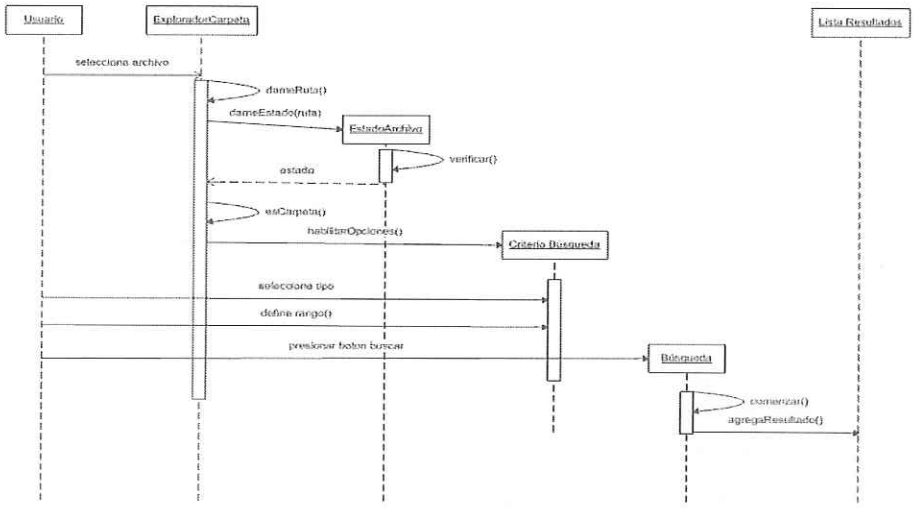

Figura 5.10. Diagrama de Secuencia Buscar Archivo (alternativo)

Diagrama de Secuencia Mostrar Historial de Internet.

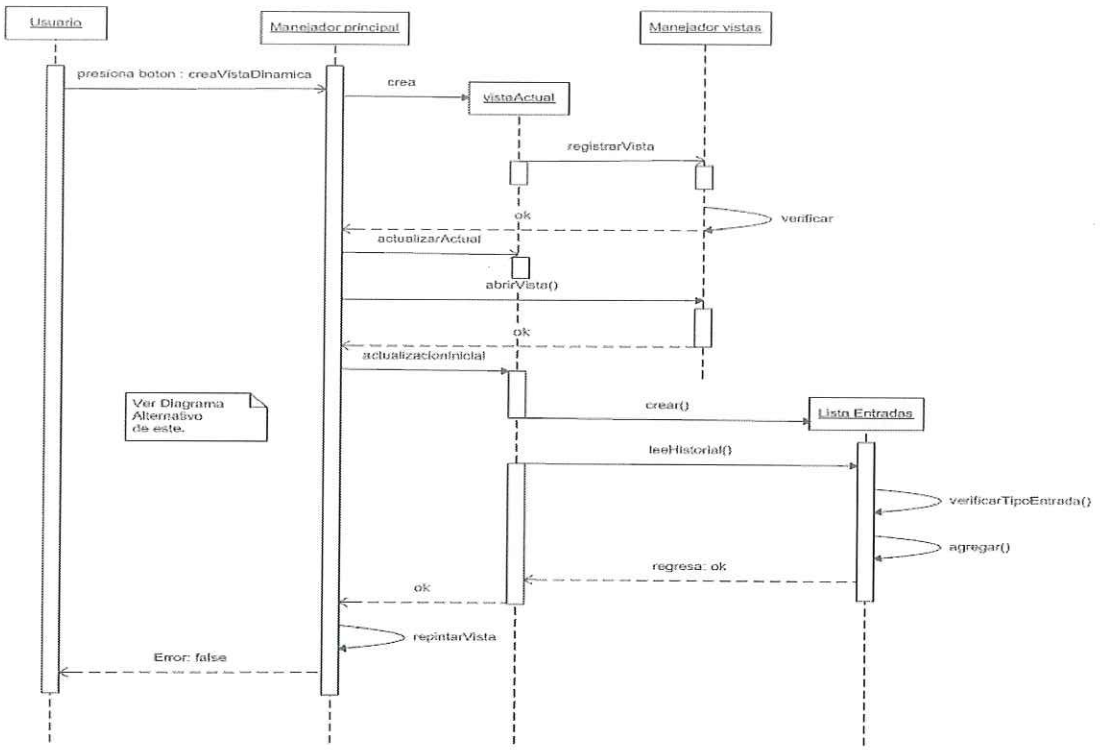

Figura 5.11. Diagrama de Secuencia Mostrar Historial de Internet.

En la figura 5.11 se muestra el diagrama de secuencia de mostrar historial de internet, esta función empieza cuando el usuario presiona el botón historial de internet, entonces el manejador principal crea la vista preliminar, la vista se registra con el administrador de vistas, este verifica y regresa un mensaje de que se ha podido registrar la y crear el objeto, luego ésta a su vez regresa ese mismo mensaje al manejador principal. Este ahora sabe que ya se pudo crear y manda a que se actualice la vista, luego se abre y carga sus elementos entre los cuales se encuentra una objeto lista donde se almacenarán cada entrada del historial, por lo que crea el objeto y manda el mensaje de leer historial, este mensaje es la llamada a ese método el cual empezará a leer el archivo donde se almacena el historial de páginas del navegador Internet Explorer, conforme va leyendo el archivo de historial se va agregando los campos

correspondientes al objeto lista y también va verificando si es válida la entrada para agregarla, por último se regresa un mensaje de terminado cuando se acabo de leer el archivo de historial. La vista entonces manda al manejador principal un mensaje para decir que ya se actualizó y este repinta la vista para que el usuario la visualice.

Diagrama de Secuencia Recuperador de Archivos.

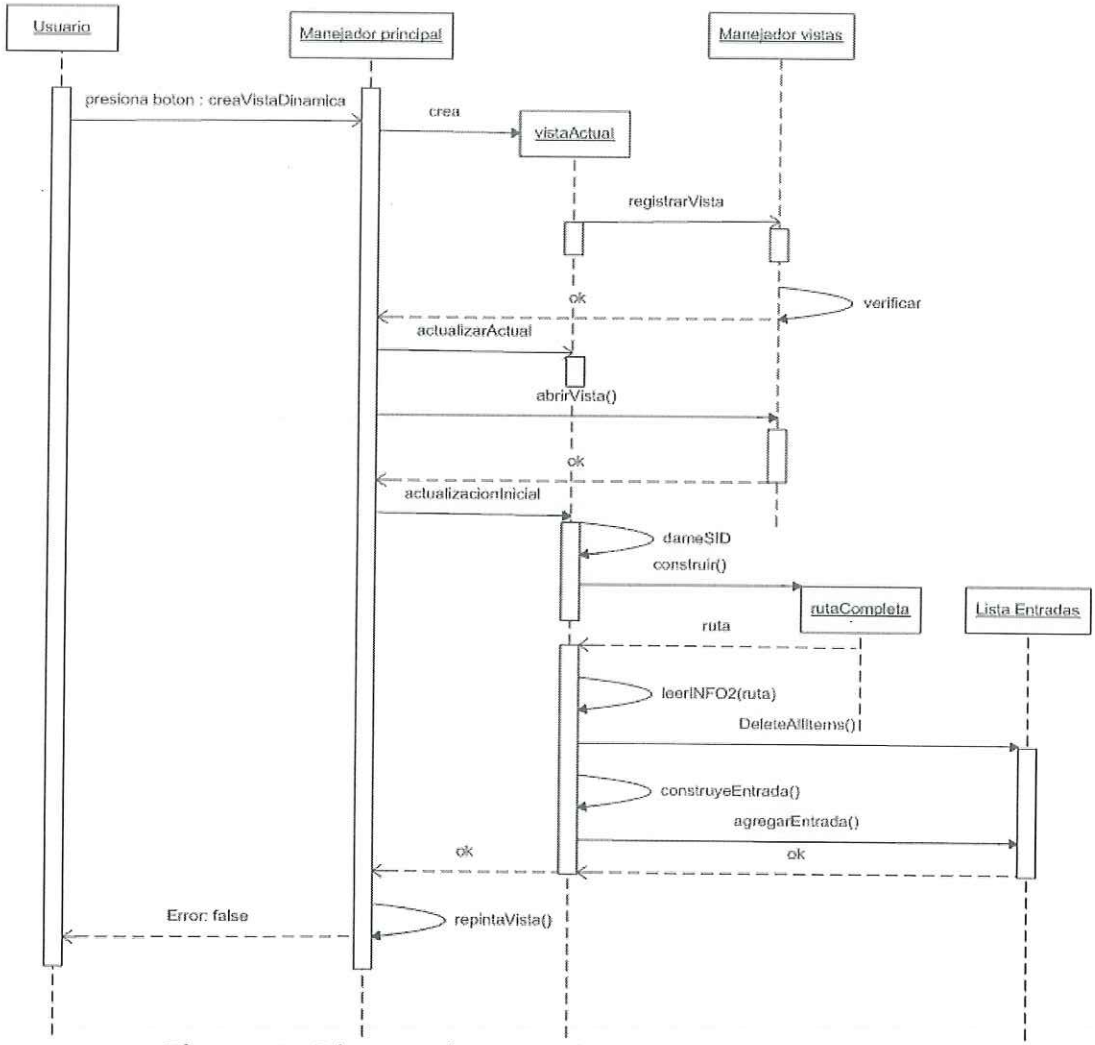

Figura 5.12. Diagrama de Secuencia Recuperador de Archivos.

En la figura anterior se muestra el diagrama de secuencia del Recuperador de archivos, este describe el comportamiento de cómo es que se crea la lista de archivos recuperados o eliminados. Primero éste empieza cuando el investigador o usuario presiona el botón de recuperador de archivos del submenú de búsqueda de Hefoclase, de ahí todo el proceso es modelado en la figura 5.12 donde se pasan varios mensajes entre objetos para construir primero lo que es la ruta donde va a buscar esos archivos, en este caso busca en la papelera de reciclaje, y luego empieza a leer la estructura interna INFO2 del sistema operativo para reunir los datos requeridos como: fecha de eliminación, unidad donde se elimino, peso, una bandera que dice si se puede recuperar o no, el índice en la tabla Interna y la ruta donde se encontraba.

Diagrama de Secuencia Mostrar Lista de cookies.

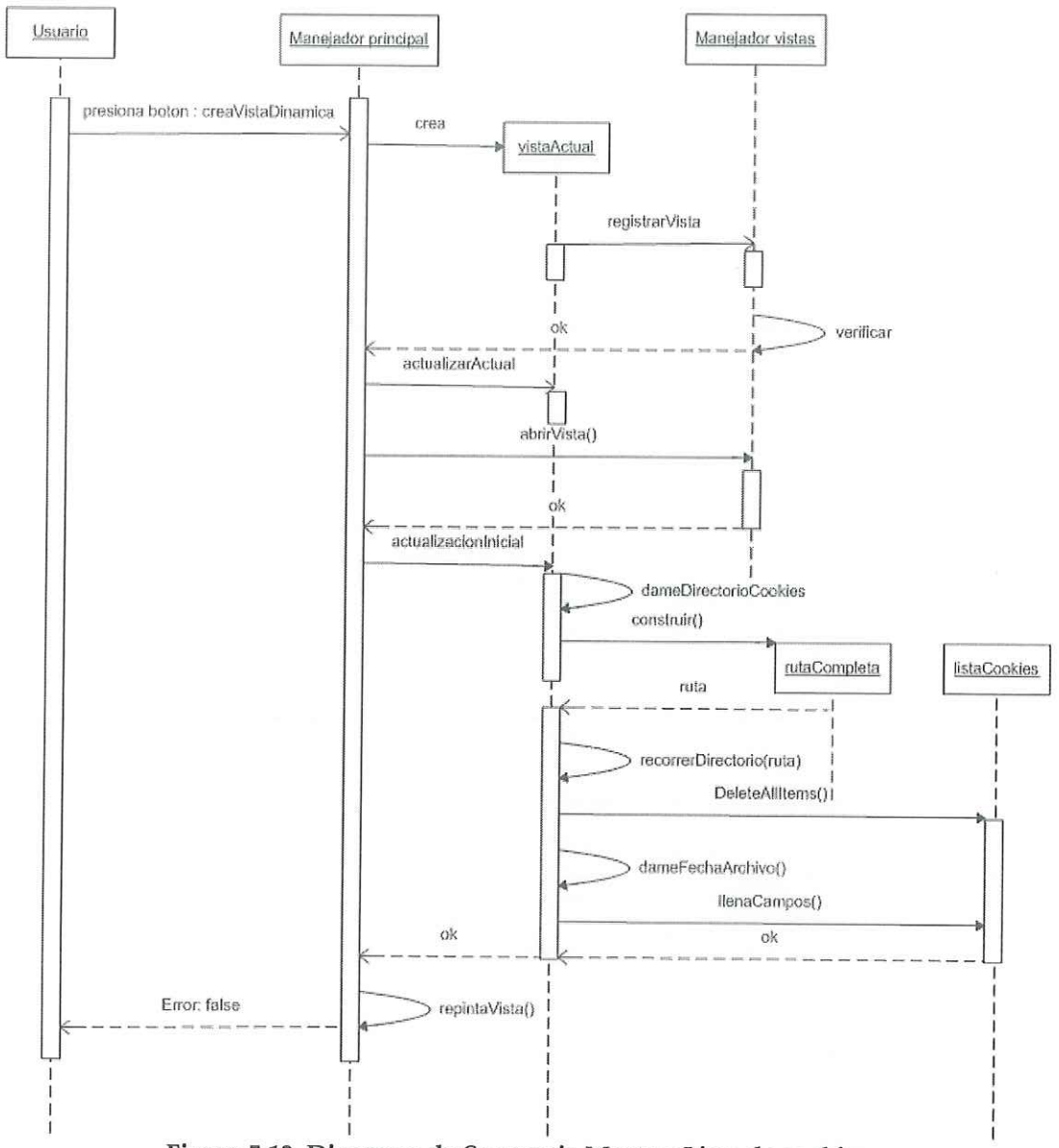

Figura 5.13. Diagrama de Secuencia Mostrar Lista de cookies.

En la figura 5.13 se muestra la secuencia para mostrar la lista de archivos de cookies del sistema, estos archivos se encuentran en un directorio en común en los sistemas Windows, generalmente en c:\documents and settings*nombre de usuario*\cookies para sistemas basados en Windows 2000 o c:\windows\cookies para Windows 95, 98, ME, de ahí que se tiene que

construir la ruta dependiendo del nombre de usuario o sistema. Una vez que se tenga la ruta entonces se procede simplemente a leer el directorio y sacar metadatos respecto a esos archivos, tales como: peso del archivo, fecha creación, fecha de modificación y fecha de acceso. Este proceso es sencillo y simplemente se pasa al objeto *lista de cookies* para agregar estos datos como campos en la lista y desplegar todos los que haya en el directorio. Hay que decir que hay otro diagrama que muestra como se despliega el contenido de estos archivos en el sistema. A continuación el diagrama que muestra como se lleva a cabo este proceso.

Diagrama de Secuencia Mostrar Contenido de Cookies.

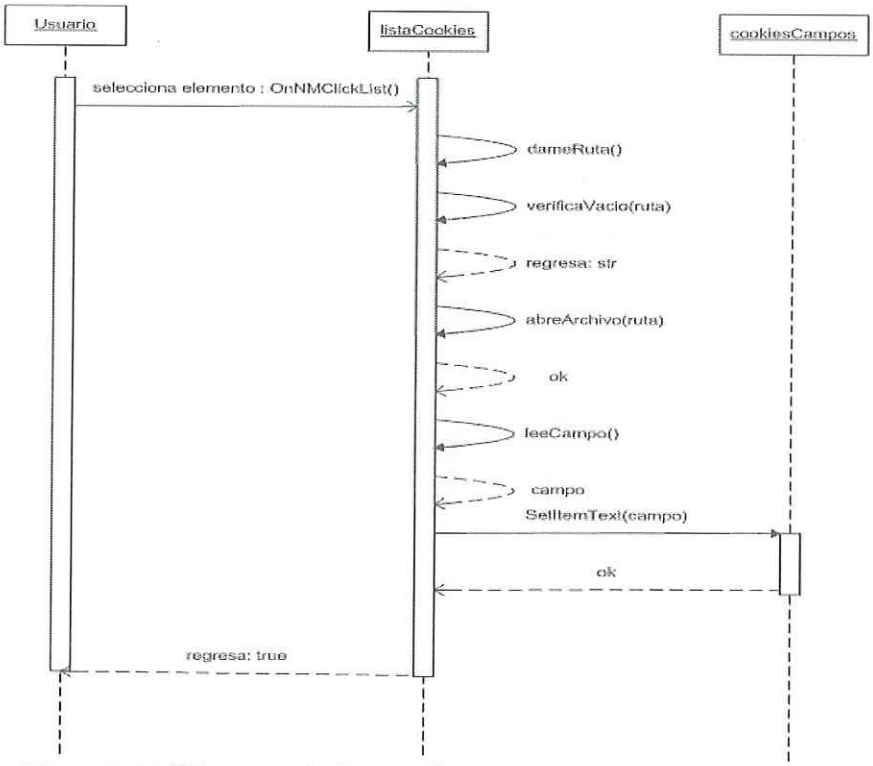

Figura 5.14. Diagrama de Secuencia Mostrar Contenido de cookies.

En la figura 5.14 se muestra el diagrama de secuencia para mostrar el contenido de una cookie, este proceso en sencillo ya que una vez creada la lista de archivos de cookies, el usuario simplemente debe seleccionar alguno de la lista para que pueda ver su contenido.

Dependiendo de lo que contenga se genera una lista de uno o varios elementos, estos elementos son las variables que tienen adentro los archivos de cookie, es decir están formadas por 9 campos como lo son [Jones K., 2003]: nombre la variable, valor de la variable, el nombre del sitio Web, banderas opcionales, los enteros más y menos significativos para los tiempos de expiración y de creación de la cookie, y un delimitador. Todos estos campos son los que componen a una cookie de Internet Explorer de Microsoft.

Diagrama de Secuencia Buscar en Registro.

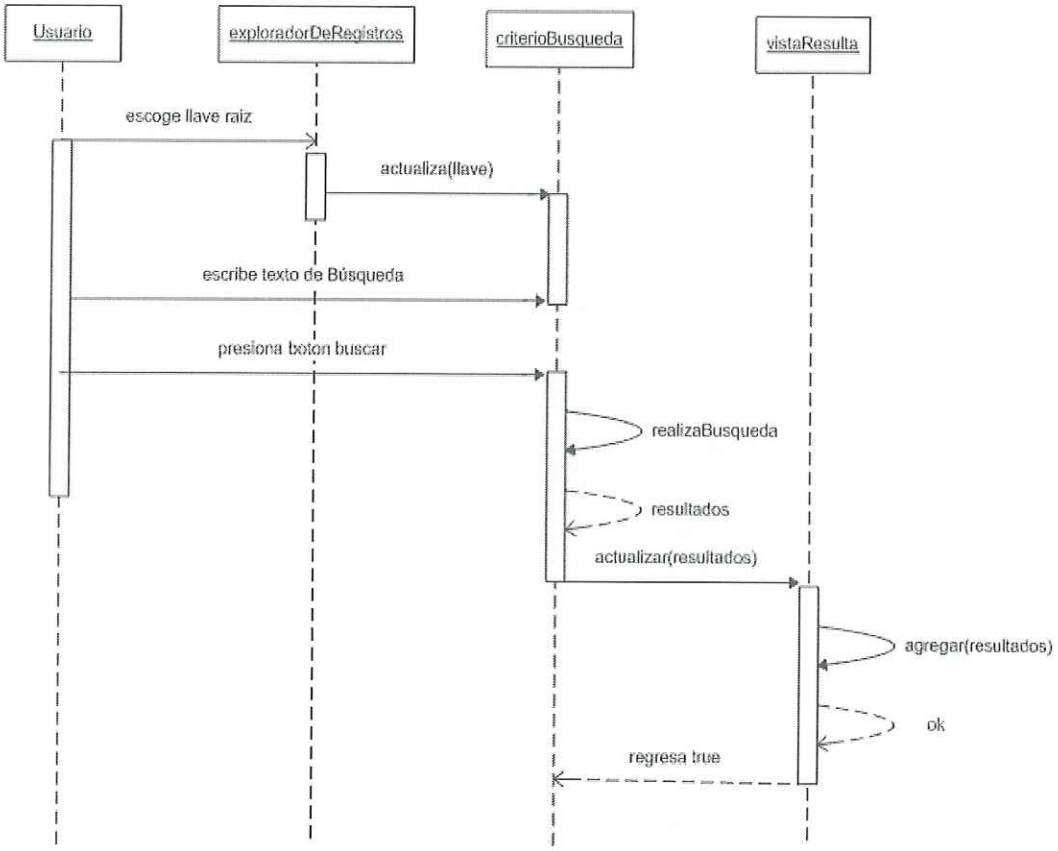

Figura 5.15. Diagrama de Secuencia Buscar en Registro.

En el diagrama 5.15 se muestra el proceso para llevar a cabo una búsqueda en el registro del sistema, aquí el usuario tiene diversas opciones para mejorar los resultados por lo que la

interacción es requerida. Lo primero que se debe hacer es seleccionar alguna llave raíz del explorador de llaves que se despliega en la vista de Explorador de Registro, esta llave raíz es la llave a partir de la cual se va tomar referencia para buscar, es decir el campo desde, y la búsqueda terminara hasta el final del registro. Después se tienen que fijar los criterios, que pueden ser varios, tales como: en donde se va a buscar, ya sea en llaves normales, llaves ocultas o en los valores que tienen las llaves; también se debe definir el término de búsqueda que es la cadena que se va a comparar a ver si concuerda con algún valor o llave, luego el usuario debe presionar el botón buscar para comenzar con la búsqueda. Los resultados se agregaran a la lista de resultados y se podrán visualizar por el usuario una vez terminado el proceso entonces el usuario podrá ir a ratificar la búsqueda pudiendo abrir tal llave para ver que efectivamente el término se encontró ahí.

En los siguientes 2 diagramas se presentan el proceso de abrir una llave de registro del sistema por medio de Hefoclase. Existen 2 formas para abrir una llave y poderla visualizar, en la figura 5.16 se muestra la forma más sencilla y en la figura 5.17 una forma alternativa que requiere más pasos para llevar a cabo tal proceso.

Diagrama de Secuencia Desplegar Contenido de llave del registro.

En el diagrama 5.16 se muestra la manera más fácil de poder abrir una llave para que se despliegue en el visualizador de llaves, este proceso es intuitivo, simplemente con seleccionar alguna rama del explorador de registro se abre su contenido en la parte de la derecha de la vista donde está el visualizador de llaves.

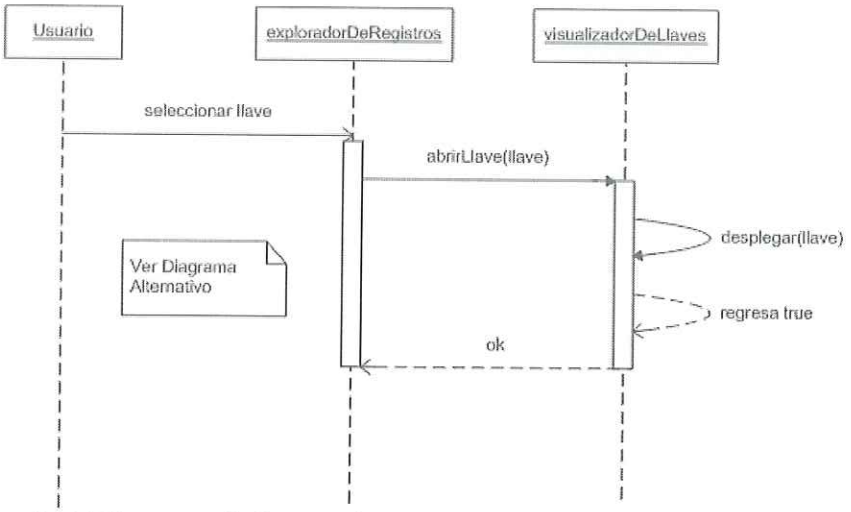

Figura 5.16. Diagrama de Secuencia Desplegar Contenido de llave del registro.

Diagrama de Secuencia Alternativo de Desplegar Contenido de llave del registro.

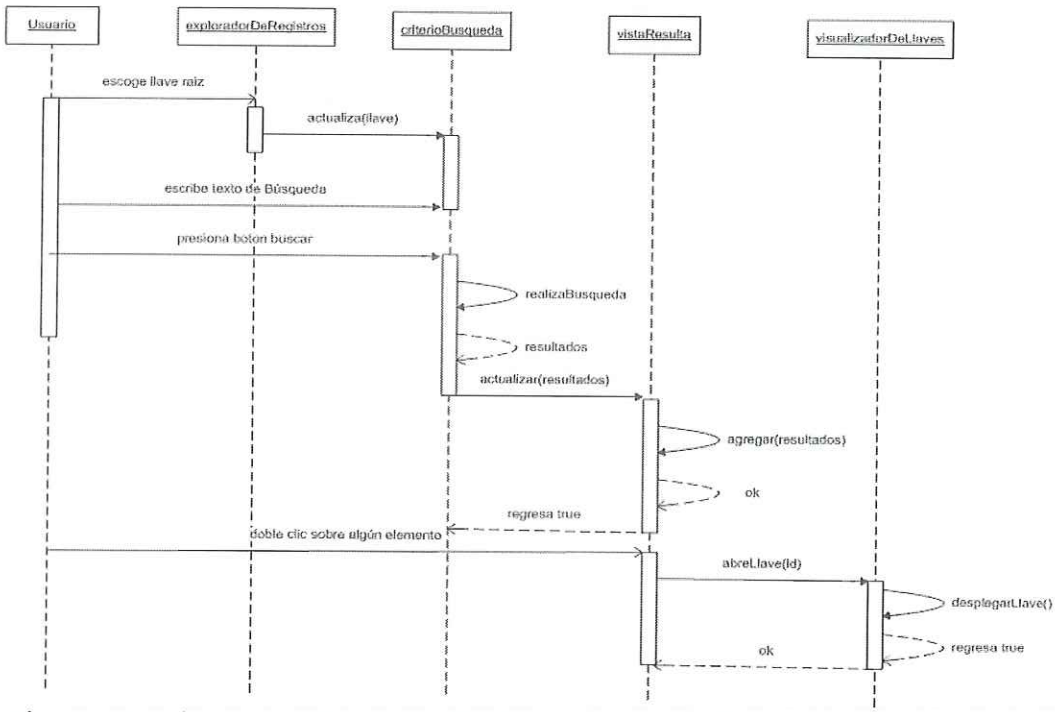

Figura 5.17. Diagrama de Secuencia Desplegar Contenido de llave del registro (alternativo).

En el diagrama 5.17 se muestra como es la otra forma de abrir una llave, este proceso alternativo es similar al de realizar una búsqueda de hecho éste es un previo para poder realizar este segundo proceso que es abrir una llave, esto se hizo con el fin de confirmar los resultados arrojados por la búsqueda, y para abrirla basta con hacer doble clic sobre cualquiera de los resultados que se encuentren en la lista de resultados, automáticamente el sistema hará un recorrido para encontrar esa llave y poder abrirla, al final la desplegará en el visualizador de llaves.

Diagrama de Secuencia Guardar documentación.

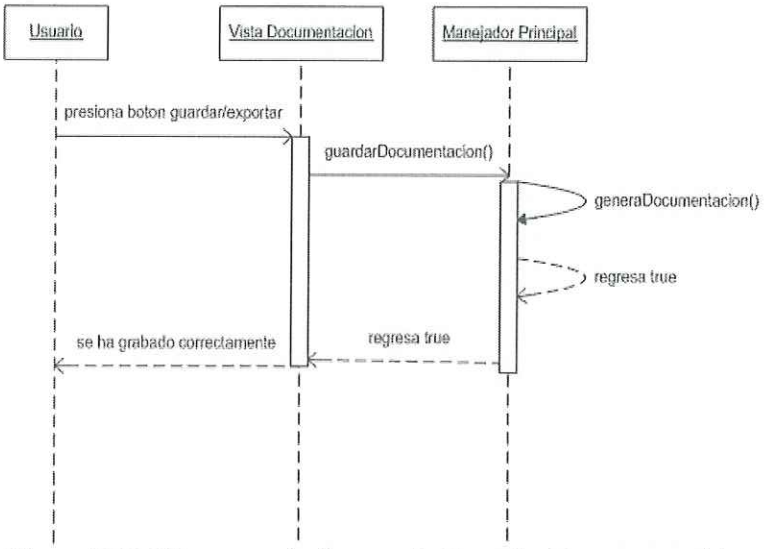

Figura 5.18. Diagrama de Secuencia Guardar Documentación.

En la figura 5.18 se muestra el diagrama de secuencia Guardar Documentación, este describe el proceso para grabar la documentación generada por Hefoclase, sabemos que esta documentación es la salida del sistema, es lo que va a quedar como un reporte para referencia acerca de la investigación que se está llevando a cabo, los datos que contendrá el reporte son los que han sido agregados a lo largo de los pasos que Hefoclase presenta, primero reuniendo

información general del sistema luego pasando por los criterios de búsqueda de evidencia digital y posteriormente al agregado de metadatos y clasificación, a partir de esas fases es de donde se va guardando cada dato que pudiera ser relevante, por ello que una vez pasado a la fase final, se puede proceder a guardar documentación, esto se lleva a cabo solo con presionar el botón guardar/exportar documentación del menú de Documentación, después Hefoclase genera esta documentación a partir de sus estructuras de datos y crea un archivo en formato HTML, se escogió este formato pos su facilidad de creación y flexibilidad; finalmente si se pudo guardar el archivo en la ruta especificada por el usuario el sistema desplegará un mensaje de información diciendo que se pudo crear satisfactoriamente.

5.6 Diagrama de Estado.

Un diagrama de estado muestra una máquina de estado, poniendo énfasis en el flujo de control de un estado a otro. Una máquina de estado es un comportamiento que especifica la secuencia de estados en la que esta o puede pasar un objeto durante su tiempo de vida en respuesta a eventos junto con sus respuestas a esos eventos. Un estado es una condición o situación en la vida de un objeto durante el cual satisface una condición, lleva a cabo una actividad o espera algún evento.

Los siguientes diagramas de estado modelan 2 de los objetos más importantes en el sistema, como lo son el objeto donde se guardan los resultados y el objeto donde guardan los metadatos de estos resultados, son 2 objetos distintos que son los que utiliza Hefoclase para generar la

documentación final o reporte, y que cada módulo por separado lo puede ir alimentando si el usuario lo va actualizando.

Diagrama de estado del objeto Resultados.

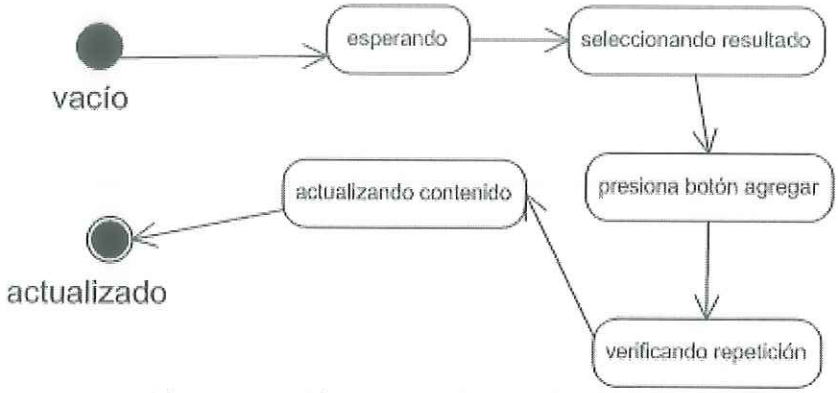

Figura 5.19. Diagrama de Estado Resultados.

Diagrama de estado del objeto Metadatos.

Figura 5.20. Diagrama de Estado Metadatos.

5.7 Diseño.

El proceso de diseñar es una actividad que depende de la persona que la lleve a cabo, es decir pueden existir muchas recetas de cocinas que funcionan o que parece que funcionan pero en el fondo es un actividad abstracta que requiere de una experiencia en el problema que se quiere desarrollar, aunque también ayuda tener una vista clara de las estructuras o componentes que se desean tener. En el desarrollo de software existen varias técnicas y herramientas que permitan llevar de una mejor forma el proceso creativo o de innovación que a veces requiere el diseño de un componente de software. A continuación se presenta una serie de conceptos que nos ayudan a comprender mejor el diseño de Hefoclase.

5.8 Diseño de objetos.

La mayoría de las cosas se componen de varios elementos y no de un elemento único omnipotente que hace todo, por ejemplo el cuerpo humano, que a decir de algunos el cerebro es todo. Sin embargo esto no es así, ya que todo el sistema, está compuesto por muchas otras partes: como los órganos, huesos, tejidos, etc., y a su vez éstos se componen de células, sustancias, y así sucesivamente. Es muy raro ver en la naturaleza y en el mundo de la tecnología, objetos abstractos y reales que lo hagan todo.

En la Ingeniería del software, no es la excepción este tipo de organización, para desarrollar programas que cumplan con objetivos específicos es necesario de la interacción de varios partes de acuerdo a un plan. Una aplicación de software es construida en partes. Estas partes, llamadas a veces objetos, interactúan enviando mensajes para pedir información o acción de otros. A

través de su ciclo de vida, cada objeto se mantiene responsable para responder a un conjunto de peticiones dado, para cumplir con éstas, los objetos encapsulan toda la información necesaria.

Nosotros representamos la información del mundo real tales como procesos, interacciones, relaciones, incluso errores, inventando objetos que a lo mejor no existen en el mundo real. Les damos vida e inteligencia a cosas inanimadas. Tenemos dificultad en comprender los objetos del mundo real y tratamos de dividirlos en cosas más pequeñas, más fáciles de estudiar o analizar, por eso creamos nuevos objetos cada uno con un rol específico dentro de cierto contexto en la aplicación.

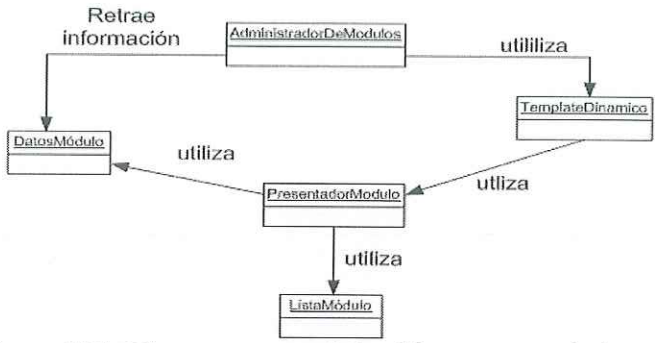

Figura 5.21. Diagrama general de objetos y sus relaciones.

En el diagrama anterior. Observamos que el objeto TemplateDinamico es un objeto contenedor de información que sirve de guía para el administrador para poder canalizar la tarea adecuada, en este caso ya sea presentar el módulo o desplegar los resultados a través del objeto PresentadorModulo; este a su vez utiliza un objeto ListaModulo que es donde se guardan los resultados de cada búsqueda, o el objeto DatosModulo que es donde se guarda la información, metadatos y resultados de cada módulo a través del presentador y el administrador.

5.9 Diseño de datos y Estructuras de Datos.

Es importante conocer el tipo de datos que va a manejar el sistema, cual es su rol en específico y la descripción de lo que es. A continuación se muestra una tabla de los tipos y estructuras de datos generales que utiliza el sistema Hefoclase.

Nombre	Tipo de dato	Descripción	Función
arregloDatos	CStringArray	Arreglo de cadenas	Guardar información relevante a los resultados de cada módulo y utilización en la base de datos de firmas.
bandera	int	Un valor entero para definir un estado.	Determinar algunas precondiciones.
Lista de despliegue	CListCtrl	Lista gráfica donde se guardan datos.	Guardar resultados o preliminares de cada módulo
Lista de imágenes	CImageList	Lista dinámica de imágenes	Guardar los iconos de los ejecutables
arregloChars	TCHAR	Arreglo de caracteres	Guardar caracteres para hacer el mapeo en algunas operaciones.
Comparador	BOOL	Valor booleano para comparar un resultado	Comparar el estado de una operación u objeto.
Cadena de texto	CString	Almacenar cadenas de texto	Guardar datos relevantes de búsqueda o de comparación.
Archivo binario	CFile	Archivo para abrir en modo	Para acceder y verificar el

		binario o texto.	estado de archivos y carpetas.
Archivo de entrada	CStdioFile	Archivo para abrir con opciones especiales extras.	Acceder y verificar archivos.
Tiempos	FILETIME	Estructura que nos permitirá operar sobre fechas.	Guardar información de fechas de archivos o para conversión.
Peso	Long	Un entero largo para guardar cantidades	Nos sirve para calcular el peso de un archivo, además de otras operaciones.
Token	Char *	Apuntador a caracteres.	Guardar la dirección siguiente de memoria de un búfer que utilicemos o para saber la dirección de memoria estamos leyendo
Fecha y hora	CTime	Para guardar fechas y horas en diversos formatos	Guardar los criterios de búsqueda de fechas y horas.
Bufer de bytes	BYTE	Formato en bytes	Guardar un arreglo de bytes.
Opciones	Enum	Una enumeración	Guardar las opciones o criterios para tomar decisiones.
hArchivo	HANDLE	Un manejador del sistema operativo.	Abrir y verificar archivos o procesos del sistema.
Fondo	Estructura declarada	Una estructura con 2 elementos: referencia y color.	Para tomar algunos fondos del sistema operativo.

Mapa	CMap	Colección para guardar relaciones entre un objeto y otro.	Guardar fondos y su relación.
Entrada llave	HKEY	Entrada de una llave al registro del sistema	Guardad una llave principal de referencia.
Entrada subclave	HTREEITEM	Entrada hija al registro del sistema.	Guardar una clave de alguna entrada en el sistema.
EnumeracionProcesos	PROCENUMPROC	Estructura para guardar la lista de procesos corriendo.	Guarda la lista de procesos en ejecución.
Modulo	CModule	Estructura para simular un proceso del sistema operativo.	Simular un proceso del sistema operativo para guardar y operar sobre él.

Tabla 3. Tipos y estructuras de datos generales que nos sirven para operar Hefoclase.

5.10 Diseño Arquitectónico.

A Continuación en la figura 5.22 se muestran los componentes que interactúan en el proceso que utilizamos con el sistema Hefoclase.

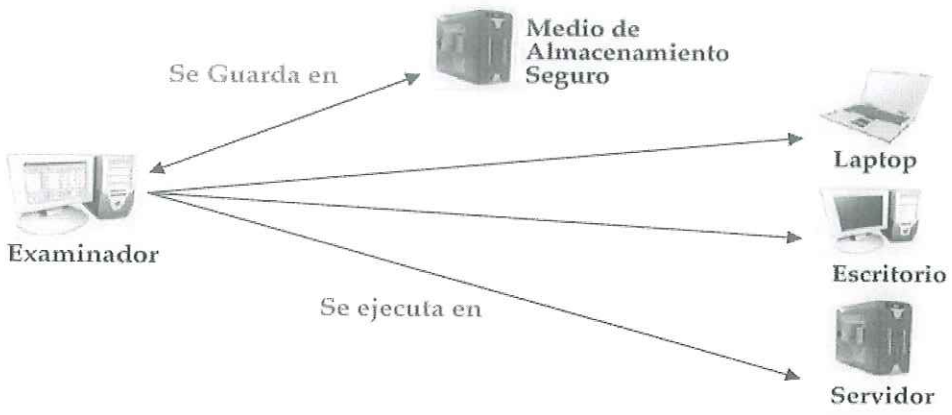

Figura 5.22 Diagrama de componentes de Hefoclase.

El examinador es la parte principal ya que es en sí, el propio sistema, donde están los diferentes módulos de búsqueda de evidencia digital, Hefoclase se puede ejecutar ya sea en un servidor, en una computadora de escritorio o una laptop. Una vez realizada la investigación se podrá guardar los resultados que se generen en un medio de almacenamiento seguro, ya sea un disco removible como un USB o en otro disco duro.

La arquitectura de un sistema es una colección de comportamientos y un conjunto de descripciones acerca de cómo uno impacta al otro [Wirfs-Brock y McKean, 2002].

Hay varios estilos y formas en las que las arquitecturas se pueden figurar, con líneas y cajas es el común para describir la estructura, pero se ignora realmente el comportamiento de inicio a fin, solamente dan guías o palabras clave acerca del funcionamiento. En la figura 5.23 se muestra la arquitectura de Hefoclase, se puede observar una arquitectura muy común en capas, donde en la primera capa esta la presentación, en la segunda los controladores y coordinares y por último en la tercera capa los sistemas y servicios de almacenamiento. Se optó por esta arquitectura porque resulta sencillo de entender, además que va de acuerdo con los componentes que interactúan en el proceso del sistema.

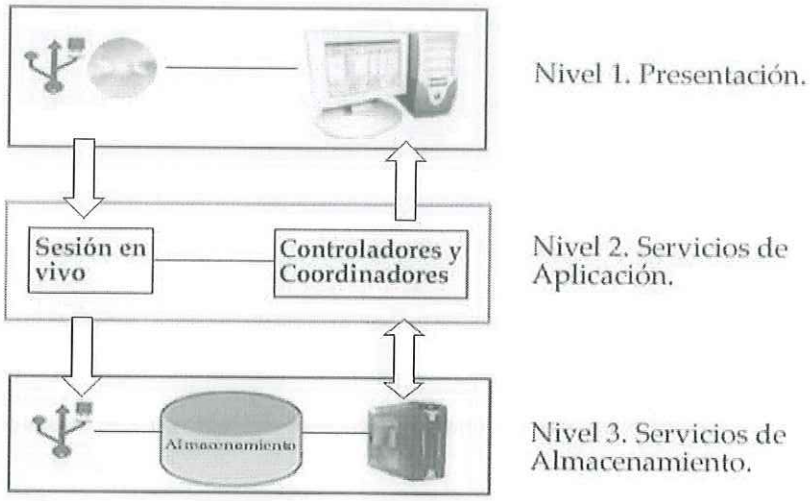

Figura 5.23 Arquitectura del sistema Hefoclase.

Nivel 1. Presentación. Es la que ve el usuario, presenta el sistema al usuario, le comunica con los servicios de aplicación. Está conformada por la interfaz gráfica de usuario del sistema, con menús y botones de funciones.

Nivel 2. Servicios de Aplicación. En este nivel es donde residen los procesos controladores y coordinadores de las tareas a realizar pedidas del nivel 1, se comunica con los 2 niveles de la arquitectura ya que es el corazón del sistema. Aquí se encuentran los módulos de adquisición de evidencia y sus respectivos mecanismos.

Nivel 3. Servicios de Almacenamiento. Es donde se guardan los datos y resultados lanzados por los respectivos módulos de adquisición, a su vez como los sumarios parciales y totales de la categorización de resultados.

5.11 Diseño de Interfaz.

El plano de un edificio no puede estar completo sin que se definan sus ventanas, puertas y demás accesorios que van a estar en primera instancia en contacto con el usuario. El diseño de una interfaz es una parte importante ya que de ahí se define la forma del proceso de interacción entre la funcionalidad (oculta) y el sistema (visible). Una interfaz de usuario es la parte de una computadora y su software que la gente puede ver, escuchar, tocar, hablar con o bien entender [Shneiderman y Plaisant, 2004]. La interfaz principal de Hefoclase es mostrada en la figura 5.24.

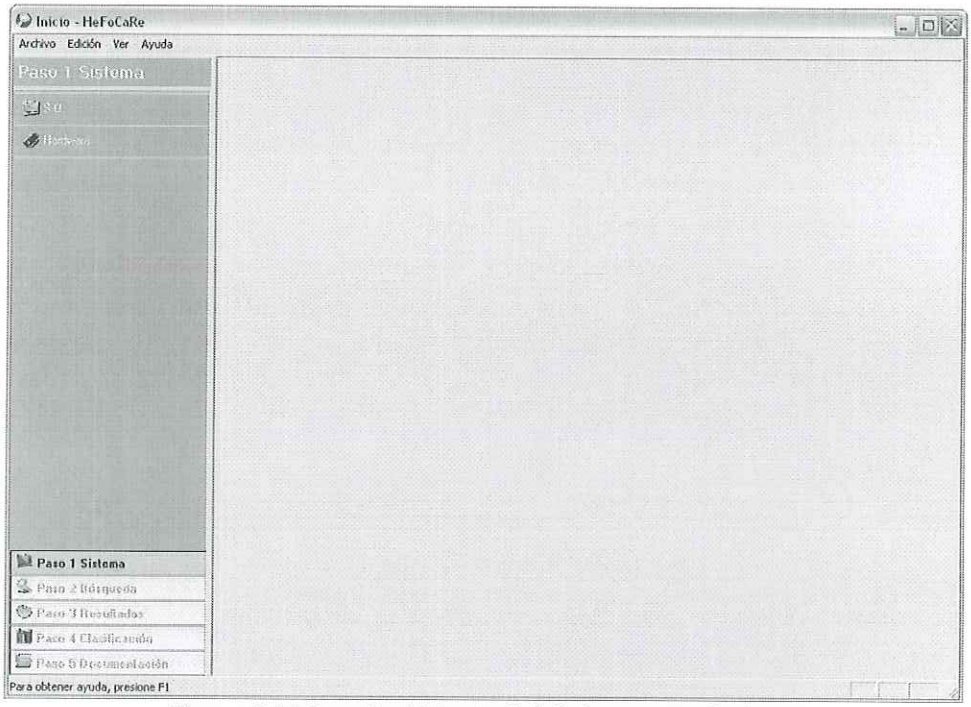

Figura 5.24 Interfaz Principal del sistema Hefoclase.

Como se ve en la imagen anterior, la interfaz principal del sistema está formada por dos menús y un área de trabajo y visualización. Hay un menú en la parte posterior que es el clásico menú para las operaciones generales del sistema como: Archivo, Edición, Ver y Ayuda; y el otro menú dinámico que se ocultan y muestran las opciones dependiendo de lo que se haya

seleccionado. Este menú es el que realmente nos interesa para llevar a cabo la investigación, donde se encuentran los módulos y funcionalidad definida, su organización es intuitiva ya que tiene por etiqueta el nombre y número de pasos en general que se necesitan para poder reunir evidencia digital. Esto es descrito más a fondo en el capítulo 6 en la presentación de la herramienta.

Hay que decir que una interfaz de usuario debe ser lo más amigable posible, es decir que pueda ser utilizada por el mayor número de usuarios que puedan existir para determinado sistema. Se pueden definir elementos concretos que nos ayuden a decidir para diseñar o verificar que una interfaz es buena. Un primer aspecto que se debe tomar en cuenta es que hay que conocer la diversidad de usuarios, existe un amplio rango de situaciones, tareas y frecuencias de uso, entre otros elementos. Comúnmente el usuario percibe la calidad del sistema a través de la interfaz de usuario, si la ventana se difumina, no se ve bien, se mueve u otras cosas el usuario puede rechazar y predisponerse a que la aplicación no es buena. Una causa común de esto puede ser porque el usuario haya tenido experiencias amargas con otras interfaces, por lo que si encuentra alguna que es parecida con la que batalló suele relacionarlo con mala calidad. Por ello que un sistema no solo tiene que ver con lo propiamente funcional sino el mapa mental que se puede generar algún usuario, y por lo general la primera impresión se da a través de la interfaz. Si el usuario mapea una buena imagen desde el principio y lo corrobora con un buen funcionamiento, entonces es muy probable que se sienta satisfecho con lo que le ofrece determinado sistema. Para determinar el nivel de satisfacción se harán una serie de pruebas en el capítulo 8, para verificar algunos puntos acerca de la percepción, uso y eficiencia de la interfaz del sistema. A continuación se muestran las interfaces de los módulos principales del sistema, está dividido en 3 secciones: la sección de búsqueda de evidencia, la sección de clasificación de

la evidencia y la sección de vistas finales y exportación de reporte. Las siguientes imágenes mostrarán la sección de búsqueda, que está dividida en 6 módulos principales, además de otros que apoyan a esta operación.

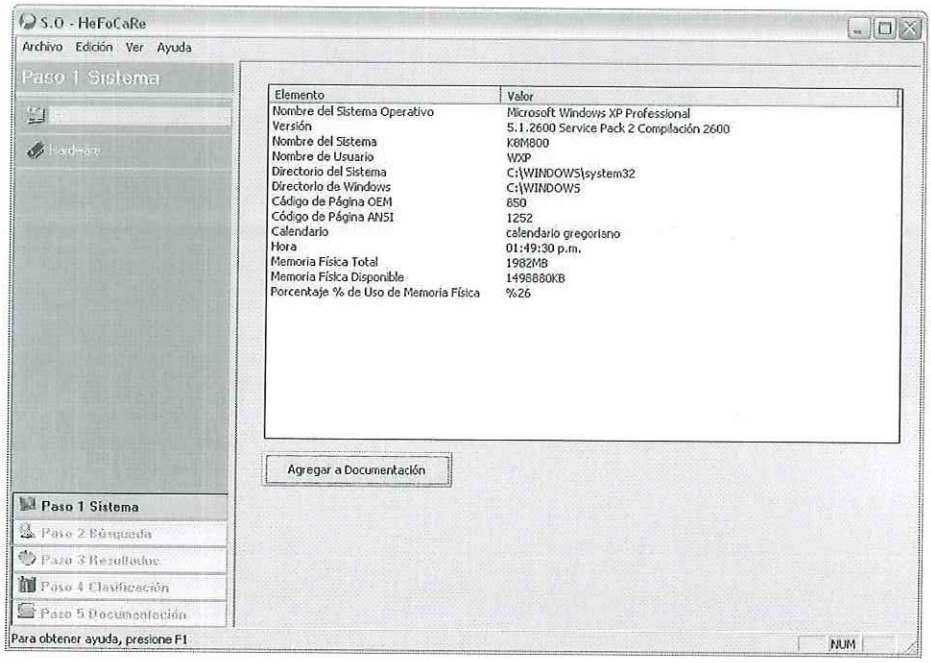

Figura 5.25 Interfaz del módulo Sistema Operativo proporcionado por Hefoclase.

En la figuras 5.25 y 5.26, se muestran la sección de búsqueda, se puede observar el módulo de sistema de archivos y sistema operativo.

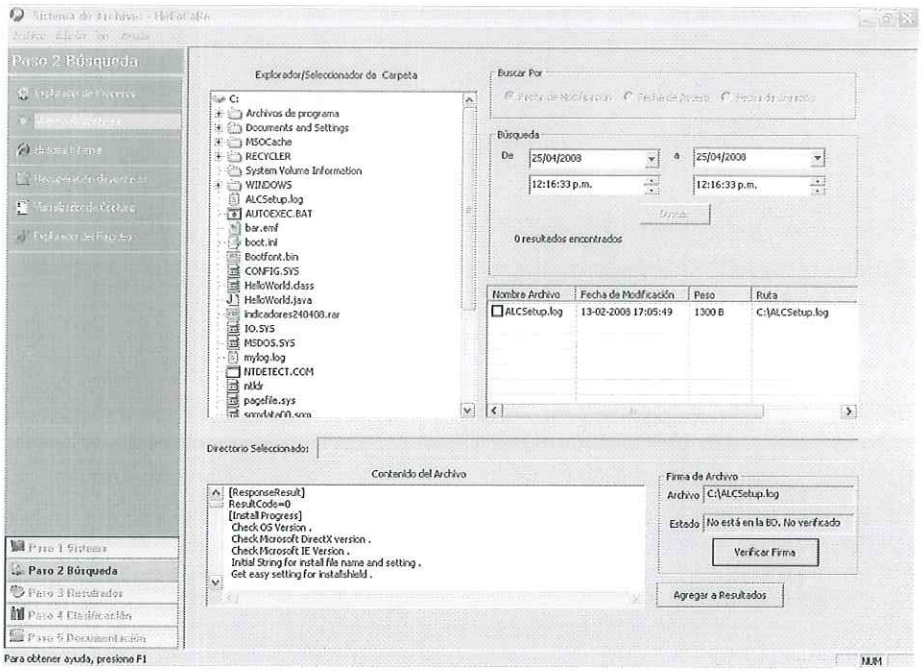

Figura 5.26. Interfaz de la sección de búsqueda, del módulo Sistema de Archivos.

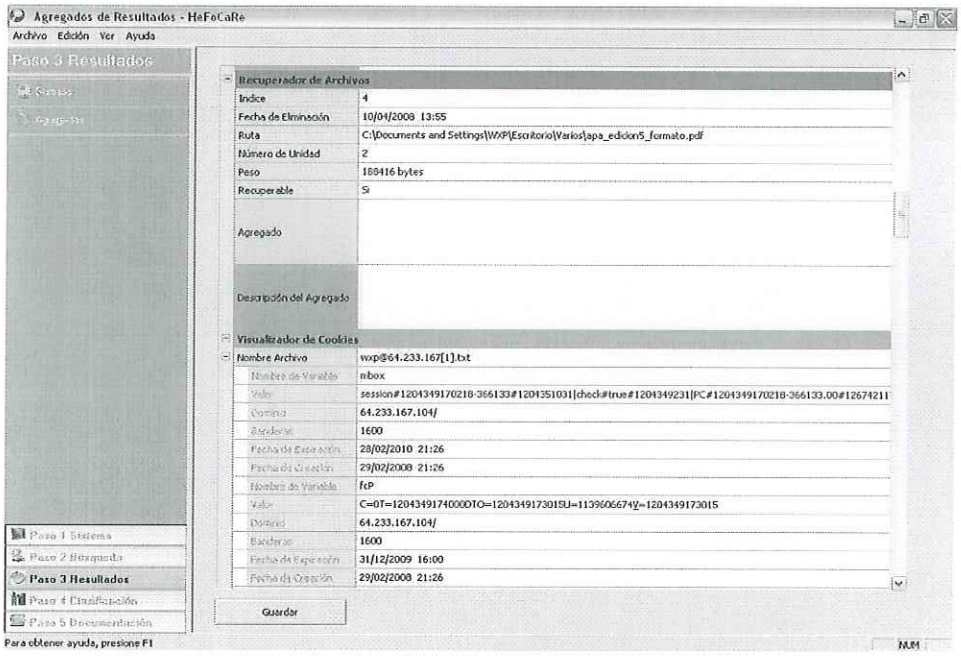

Figura 5.27. Interfaz del módulo de Resultados.

Como vemos en la figura 5.27 se muestra un sumario de resultados que se encontraron en algún equipo al estar realizando la investigación en vivo. Esta sección de visualización es muy

importante que ayude al investigador a que pueda tomar decisiones de acuerdo a la evidencia reunida y que permita ver de una forma organizada esos datos.

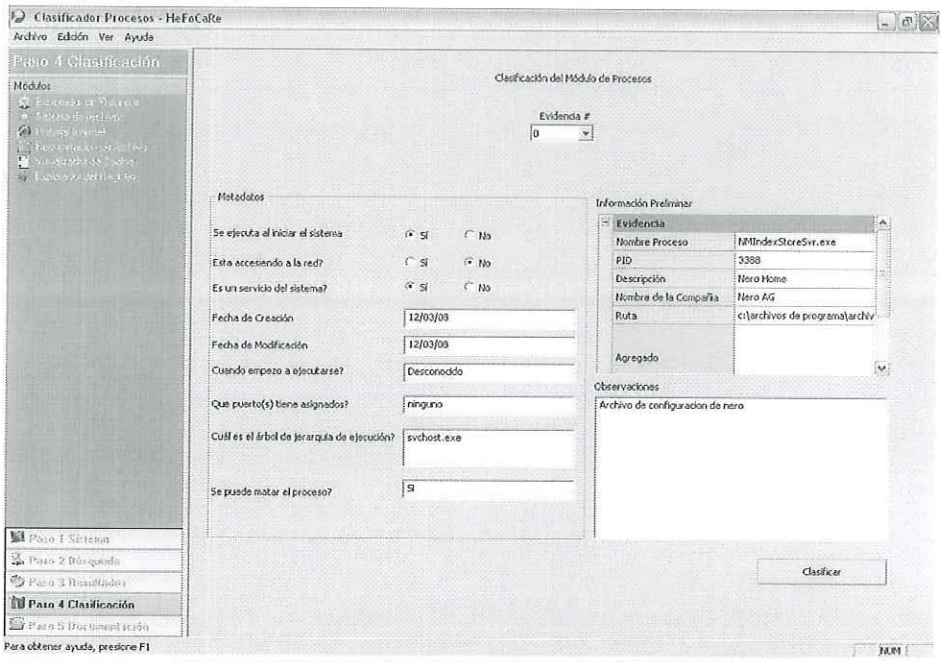

Figura 5.28. Interfaz del módulo de clasificación.

5.12 Algoritmos y Pseudocódigo.

Algunos de los algoritmos más importantes son los siguientes que a continuación muestro en código fuente y otros en combinación de código fuente y algoritmo o pseudocódigo.

Listar Procesos.

```
BOOL   bResultado = TRUE;
InstanciaModuloEjecutable* infoProceso;

if (TRUE == Inicializar())     {
        DWORD pidArreglo[1024];
        DWORD bNecesario;
        DWORD nProcesos;
```

```
if (EnumeraProcesos(pidArreglo, sizeof(pidArreglo), &bNecesario))      {
        // Para saber cuántos procesos hay.
        nProcesos = bNecesario / sizeof(DWORD);
        m_procesos->ReleaseAll();
        for (DWORD i = 0; i < nProcesos; i++)     {
                HMODULE arregloModulo[1024];
                HANDLE hProceso;
                DWORD pid = pidArreglo[i];
                DWORD nModulos;
                hProceso = OpenProcess(         PROCESS_QUERY_INFORMATION |
PROCESS_VM_READ,
                                        FALSE, pid);
                if (!hProceso)
                        continue;
                if (!enumeraModulosDelProceso(hProceso, arregloModulo,
                                                                sizeof(arregloModulo),
&bNecesario))  {
                        CierraHandle(hProceso);
                        continue;
                }
                // Para saber cuántos módulos hay en cada proceso.
                nModulos = bNecesario / sizeof(arregloModulo[0]);

                for (DWORD j = 0; j < nModulos; j++)     {
                        HMODULE hModulo = arregloModulo[j];
                        char nombreModulo[MAX_PATH];
                        dameNombreModuloEjecutable(hProceso, hModulo,

nombreModulo, sizeof(nombreModulo));

                        if (0 == j)  {
                                // El primer modulo  es el ejecutable entonces lo agrego
                                infoProceso = new
                                InstanciaModuloEjecutable(nombreModulo, hModulo,
                                pid);
                                m_procesos->Add(*infoProceso);
                                infoProceso->ListarModulos(this);
                                break;
                        } // if
                } // for
                        CierraHandle(hProceso);
        } // for

        bResultado = TRUE;
} // if
```

```
            else
                bResultado = FALSE;
} // if
else
    bResultado = FALSE;
```

Sacar Icono del Proceso.

```
for (unsigned i = 0; i < administradorTareas.GetProcessCount(); i++)       {
            proceso = administradorTareas.GetProcessByIndex(i);
            path = proceso->dame_Nombre();
            raiz = recurso.dameDirectorioWin();
            path.Replace("?","");
            path.Replace("\\\\","");
            path.Replace("\\SystemRoot",raiz);
            himLst=(HIMAGELIST)SHGetFileInfo(path,FILE_ATTRIBUTE_NORMAL,&psfi,sizeof(
psfi),SHGFI_ICON|SHGFI_SMALLICON|SHGFI_SYSICONINDEX);
            hIcon = ImageList_ExtractIcon(0, himLst, psfi.iIcon);
            m_ListaIconos.Add(hIcon);
}
```

Verificar Firma.

```
int b=666;
int a;
int ind=0;
int tot=0;
int emp=0;
int ban=0;
char coma = ',';
CStringArray hexa;
CStringArray hexa1;
CString todo="";
for(a = 0; a < m_arregloFirmas.GetSize(); a++) {

            if(ban == 0) {
                        todo = m_arregloFirmas.GetAt(a);
                        char *next,*token,*next1,*token1;
                        char sep[]   = ",";
                        char sep2[]   = "|";
                        token = strtok_s( (char*)(LPCTSTR)todo, sep , &next);
                        token1 = strtok_s( (char*)(LPCTSTR)token, sep2 , &next1);
```

```
                hexa.Add(token1);
                while (token1 != NULL) {
                        if (token1 != NULL) {
                                token1 = strtok_s(NULL , sep2, &next1);
                                hexa.Add(token1);
                        }
                }
                int k=0;
                while(k < hexa.GetSize()) {
                        if(hexa.GetAt(k).Compare(ext) == 0) {
                                b=a;
                                ban = 1;
                        }
                        k++;
                }
        } else break;
}

m_arregloFirmas.FreeExtra();
m_arregloFirmas.RemoveAll();
LlenaArregloFirmas();

if(b != 666) {
        CString str="";
        CString rest="";
        int cuentame=0;
        str = m_arregloFirmas.GetAt(b);
         emp = str.ReverseFind(coma);
         tot = str.GetLength();
        int joder = tot-emp;
         rest = str.Mid(emp+1,joder);
        char *token2,*next2;
        char sep3[]  = "|";
        token2 = strtok_s( (char*)(LPCTSTR)rest, sep3 , &next2);
        hexa1.Add(token2);
        while (token2 != NULL) {
                if (token2 != NULL) {
                token2 = strtok_s(NULL , sep3, &next2);
                hexa1.Add(token2);
                }
        }

        cuentame=0;
        for(int j=0; j<hexa1.GetSize()-1; j++) {
                if( val.Find(hexa1.GetAt(j)) != -1)
```

```
                    cuentame++;
            }
            if(cuentame != 0) return 1;
            else return 0;
} else return 2;
```

Si el resultado que devuelve es 1 la firma concuerda, si es 0 entonces hay una incongruencia entre la extensión y la firma. Por último si lo que arroja el algoritmo anterior es 2 quiere decir que no estaba esa firma en la Base de datos.

Leer Historial de Internet.

```
CFile *pFile = new CFile();
 CFileStatus stat;
//La ruta es donde se encuentra el archivo index.dat
 if(pFile->GetStatus(ruta,stat)) {

          try {
                  pFile->Open( ruta, CFile::shareDenyNone
                  |CFile::modeRead);//CFile::shareDenyNone | CFile::typeBinary);
         } catch (CFileException ex) { }
          peso = (int)pFile->GetLength();
         //Me muevo a un Offset en el archivo index.dat, en este caso para encontrar el campo de
         versión.
          pFile->Seek(0x18,CFile::begin);
          pFile->Read(&cuatrobytes,4);
         version = atof( cuatrobytes); //Leo los cuatro bytes donde se guarda el numero de
         version.
         pFile->Seek(0x1C,CFile::begin);
         pFile->Read(cuatrobytes,4);

          tamanoArchivo = hexadecimalaentero( cuatrobytes, 4);

          offsetActual = 0;
          char c = '-';
          int t=0;
          while (offsetActual < peso ) {

                   pFile->Seek(offsetActual,CFile::begin);
                  pFile->Read(&cuatrobytes,4);

                   for (i=0;i < 4;i++) {
```

```
                    tipo[i] = cuatrobytes[i];
                }

                tipo[4] = '\0';
                if (tipo[0] == 'R' && tipo[1] == 'E' && tipo[2] == 'D' && tipo[3] == 'R' ) {
                moverse en el Offset, agarrar campos (url,nombre, cabeceras http,
                directorio entre otros y hacer conversiones a sus tipos para poder manipularlos.
                }
                else if ( (tipo[0] == 'U' && tipo[1] == 'R' && tipo[2] == 'L') || (tipo[0] == 'L' &&
tipo[1] == 'E' && tipo[2] == 'A' && tipo[3] == 'K') ) {
                moverse en el Offset, agarrar campos (url,nombre, cabeceras http,
                directorio entre otros y hacer conversiones a sus tipos para poder manipularlos.
                }
                else {
                puedes hacer algo si te encuentras con datos corruptos en el archivo index.dat, en mi caso
                no hice nada.
                }

        offsetActual = offsetActual + 0x80; //Mas 0x80 que es el tamaño del bloque para
        encontrar la nueva entrada válida y poder leerla.

        }

        pFile->Close();
}
else {
         AfxMessageBox("No existe el archivo de Búsqueda");
         delete pFile;
}
```

Como se mencionó en anteriores capítulos, en la actualidad existen dos tipos de paradigmas para llevar a cabo una investigación de informática forense. El análisis tradicional donde se apaga la máquina y se copia bit a bit el disco duro y por otro lado está la investigación que se hace en vivo, con el sistema corriendo sin apagarlo pero con riesgo de sobrescribir datos que pudieran llevar a pérdida de una posible evidencia digital. La herramienta que se desarrolló en este trabajo fue de este último paradigma, una herramienta para análisis forense informático en vivo.

Hefoclase, se puede separar en tres secciones: primero la parte que me permite buscar datos acerca de la evidencia, es decir los diferentes módulos que realizan búsquedas acerca de algún incidente, tales como: explorador de procesos, historial de internet, sistema de archivos, papelera de reciclaje, explorador de registro y visualizador de cookies. Además recopila datos relacionados del equipo donde se ejecuta la herramienta, como es el hardware y software que lo compone. Podemos decir que esta sección la componen los pasos 1 y 2.

Una segunda parte es donde se agregan metadatos a la evidencia encontrada, esto se hace para fortalecer el contexto en el que está involucrado y tener una mayor eficacia de lo que se está tomando como evidencia. Aquí es donde entra parte del paso 3 y el paso 4. Por último, otra parte importante del sistema es la visualización, es decir una vez que haya encontrado algo y lo haya clasificado entonces se tiene la opción de poder visualizar un sumario de esos resultados o tener la vista final de lo que es la investigación, ya sea guardando el reporte o viendo la vista final. Esta parte la componen el paso 5 y parte del paso 3, así como también todos aquellos

componentes gráficos de cada paso, que nos permite tener una vista de lo que se está buscando o se ha encontrado.

La idea de crear una herramienta intuitiva, fácil de usar, se basa en que no es necesario tener una interfaz muy complicada para poder tener datos que permitan comprobar que algún sistema ha sido comprometido en su seguridad. Por ejemplo hay herramientas nativas de los sistemas operativos que son poderosas para saber y tratar de conectar esas pequeñas pistas que comúnmente no tienen un significado definido. Transformar este conjunto en pequeños elementos interconectados pero con un claro sentido y significado entre sí, de esta forma se ayuda a probar, corroborar o descartar evidencia.

La razón por la que se escogió poner etiquetas y organizar la herramienta en forma de pasos tal como se muestra en la figura 6.1, es porque existe una forma que se apega al desarrollo de ciertas habilidades cognitivas. Y en este caso más relacionadas con lo técnico del área, es decir, una investigación forense tiene una serie de pasos que en su nivel abstracto podemos resumir e identificar. Los pasos que se siguen generalmente para llevar una investigación forense son: primero tener una base o idea del sistema en el cual se va a trabajar, para poder atacarlo de una manera adecuada, luego la atención se centra en el sistema y se comienza a buscar contestar algunas hipótesis o descartar algunas otras, entonces es así cuando inicia el proceso de búsqueda y recolección de pistas. Después se tratan de unir esas pistas, agregando datos que permitan hacer una relación entre varios para tratar de dar un significado coherente y válido. Por último se documenta y guarda toda esa información para posterior análisis y de bitácora acerca de cómo se llevó a cabo el procesos o si se omitió algún detalle en el desarrollo del mismo.

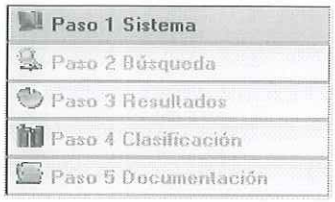

Figura 6.1. Organización del menú principal del sistema Hefoclase.

Como se ha comentado, el sistema que se presenta en este trabajo está organizado para su uso en forma de pasos, esto no necesariamente implica que se debe seguir ese orden sin embargo es el modelo lógico de acción, ya que por ejemplo no se puede clasificar algo sin siquiera tener datos acerca de ello, como características propias. Sin embargo se puede actuar omitiendo alguno de los pasos, por ejemplo un investigador se puede saltar el primer paso, ya que los datos recabados en ese proceso pueden o no ser importante para la investigación. También se puede usar la herramienta alternando el uso del paso 3 y 4.

El primer paso que presentamos, es el paso 1 denominado Sistema. Aquí se refiere a reunir datos acerca del sistema donde se está ejecutando la herramienta. Los tipos de datos que se reúnen están organizados en dos tipos: Hardware y Software. Para el primero su objetivo es reunir información del equipo físico en el que se ejecuta el sistema, tales como tipo de procesador, monitor, discos duros, sistemas de archivos, frecuencia de reloj del CPU, modelo entre otros. Para el segundo que es Software, la información que se maneja es principalmente acerca del sistema operativo Windows, datos como nombre del sistema operativo, versión, nombre del usuario, ruta de instalación, hora, memoria física disponible, memoria física utilizada, porcentaje de uso de la memoria, etc. Todos estos datos relacionados a Software y Hardware del sistema permiten tener más claridad acerca del entorno en el que se trabaja y que

consideraciones son factibles hacer. Pero como se acotó anteriormente este paso puede ser omitido si así lo decide el investigador.

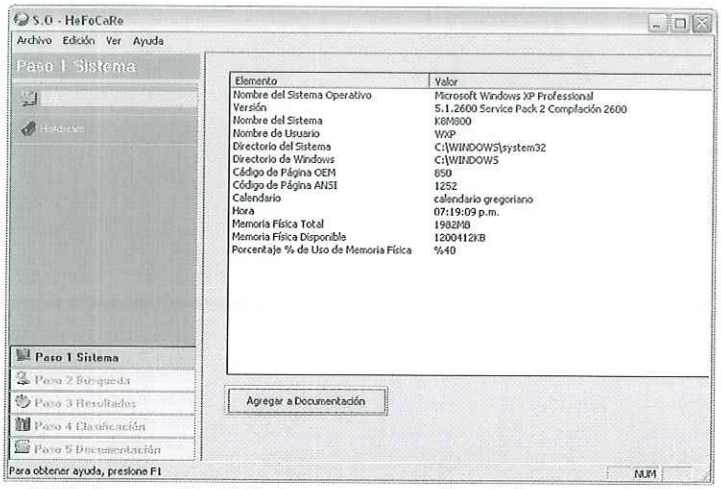

Figura 6.2 Información del sistema operativo del paso 1 de Hefoclase.

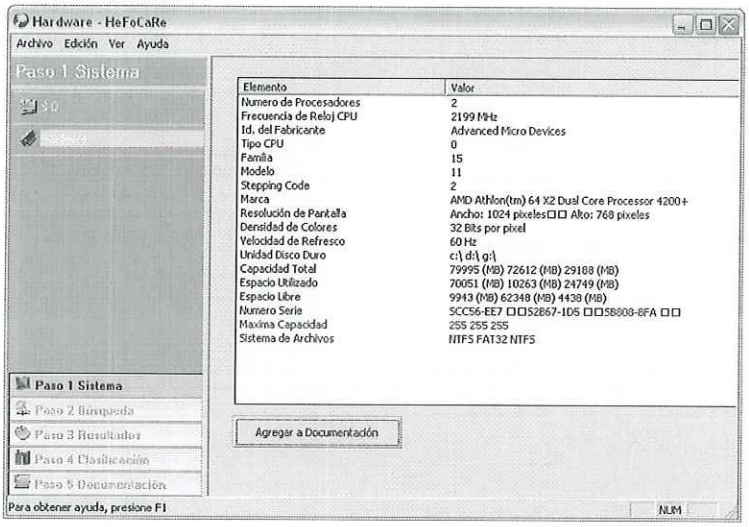

Figura 6.3. Información de Hardware del Paso 1 de Hefoclase.

Se muestra en las figuras 6.2 y 6.3 la vista del paso 1 de Hefoclase, que abarca información del sistema operativo e información de hardware.

Una vez decidido si se recaba información preliminar del entorno o no, se procede a empezar con la búsqueda de pistas que se puedan conectar entre sí. En el paso 2 es donde se lleva a cabo

este proceso. Este módulo es uno de los más importantes, ya que son los requerimientos principales y fuente principal para encontrar información o evidencia. Este paso contiene los criterios que se eligieron para reunir la posible evidencia digital, lugares estratégicos relacionados con la experiencia de usuarios y expertos en seguridad informática en ambientes Windows, los lugares más comunes donde se puede ocultar alguna amenaza. Estos módulos de búsqueda son: Explorador de procesos, Explorador del Sistema de Archivos, Visualizador del Historial de Internet, Visualizador de Cookies, Recuperación de Archivos eliminados y Explorador del Registro del sistema. Aunque estos son los únicos criterios escogidos para el desarrollo de este trabajo, es importante decir que no son los únicos existentes sino los más frecuentes lugares de búsqueda cuando se trata de buscar la causa de algún incidente relacionado con la seguridad en algún sistema Windows. Tal vez sea por características intrínsecas del sistema operativo que lo hacen susceptible a ciertas cosas y por lo tanto tener ciertas debilidades en determinados sitios, pero más que eso se está buscando dentro de las estructuras más importantes internas del mismo. Por ello que no es extraño que el malware se hospede en esas estructuras para debilitar la defensa, tal como lo hace un virus biológico en el cuerpo humano, trata de aprovechar las vulnerabilidades que pueda tener en cierto momento la línea de defensa ante agentes externos, extraños o maliciosos. A continuación se presenta el funcionamiento de cada módulo de búsqueda y como puede ayudarnos a encontrar esas pistas relacionadas para volver en la línea del tiempo acerca de la forma en que pasó determinado incidente.

6.1 Módulo Explorador de Procesos

El primer módulo que se presenta es el Explorador de Procesos, cuya vista se puede observar en la figura siguiente:

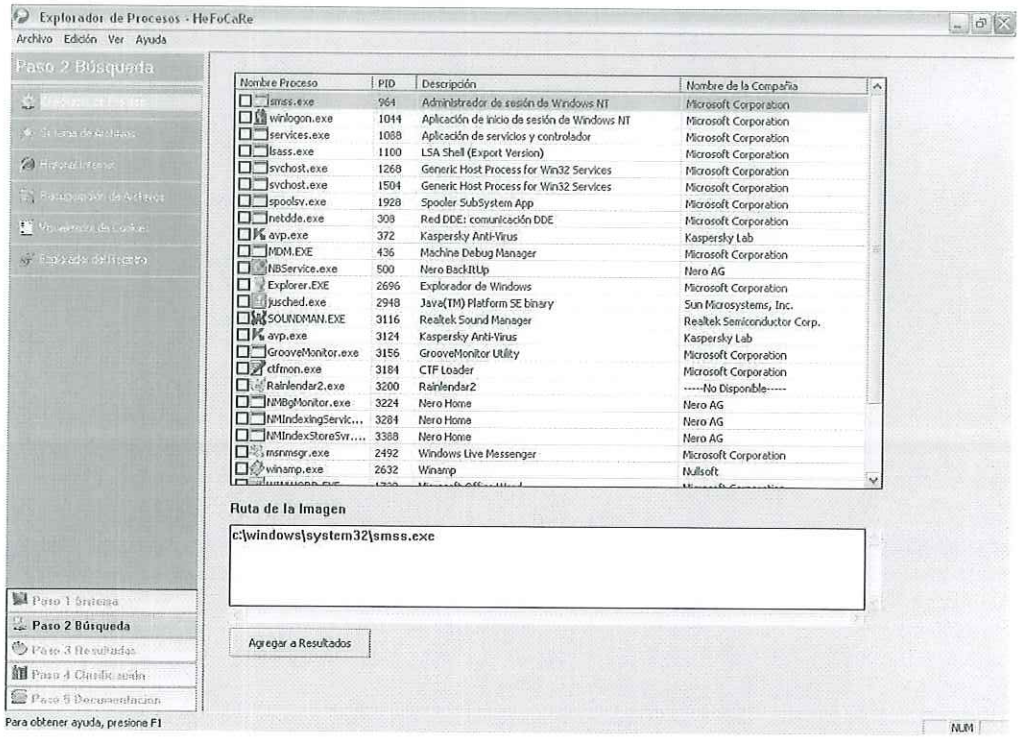

Figura 6.4. Módulo de Explorador de Procesos de Hefoclase.

Hay que decir que el malware comúnmente trata de garantizar su ejecución de cierta forma, y la manera más fácil de ejecutarse es como proceso del sistema. Cuando se abre un archivo .exe lo primero que ocurre es que se realiza una llamada a una estructura del sistema operativo que gestiona esta petición y trata de crear un proceso para ese ejecutable. Para que pueda concretarse tal acción es necesaria una serie de pasos gestionados por el propio sistema operativo. En general de acuerdo a Russinovich [Russinovich y Solomon, 2004] son los siguientes:

1. Abrir el archivo binario ejecutable (.exe) a ser ejecutado dentro del proceso.
2. Crear el objeto de ejecución de proceso de Windows.
3. Crear el hilo inicial (pila, contexto).
4. Notificar al subsistema de Windows del nuevo proceso para que pueda ser puesto como nuevo proceso y en un hilo.

5. Empezar la ejecución del hilo inicial (al menos de que se especifique la bandera CREATE_SUSPENDED).
6. En el contexto del nuevo proceso e hilo, completar la inicialización del espacio de direcciones (como librerías requeridas) y empezar la ejecución del programa.

Como vemos, el flujo para la creación del proceso, es relativamente sencillo, los datos que interesan conocer para determinar si un proceso es legítimo o no, son su nombre, ruta de ejecución, datos relacionados al recurso del binario como: nombre compañía que lo hizo, descripción entre otros. Además es importante tener un mecanismo para saber si tiene actividad de red. El investigador debe analizarlo para ver si este proceso no es malicioso, ya que muchas veces se puede copiar el nombre de un proceso legítimo, que se ejecuta en la misma ruta y tiene el mismo icono y demás datos, sin embargo envía datos a través de internet o captura información sensible relacionada al entorno o uso de aplicaciones del usuario. Por ello que además de toda esa información es deseable conocer si tiene un proceso padre y si es así, determinar cuál es para poder relacionar todas las pistas que se tengan para comprobar o corroborar su legitimidad y validez.

El sistema Hefoclase nos genera la lista que tiene el subsistema encargado de los procesos de Windows y carga cierta información relevante como lo es: icono del proceso, nombre, compañía que lo desarrollo, descripción, pid (process id) y la ruta de ejecución. Esta última se visualiza cuando nosotros seleccionamos algún proceso de la lista de procesos generada, tal como se muestra en la figura 6.4.

6.2 Módulo Sistema de Archivos.

A pesar de todos estos esfuerzos por reconocer que se está ejecutando en nuestro sistema, hay otras amenazas que no saldría ni una pista en una lista de procesos generada por el subsistema de Windows encargado de manejar los procesos. Algunos malware se pueden esconder de eso o combinarse en otro proceso o varios para ejecutarse y dañar la integridad del sistema, pero eso solo es una forma, en la realidad existen otras técnicas que pudieran comprometer al sistema. Por esto que es conveniente combinar herramientas de diagnóstico ya sea nativas o no para comprender que procesos, librerías y archivos están siendo ejecutados o usados en nuestro sistema.

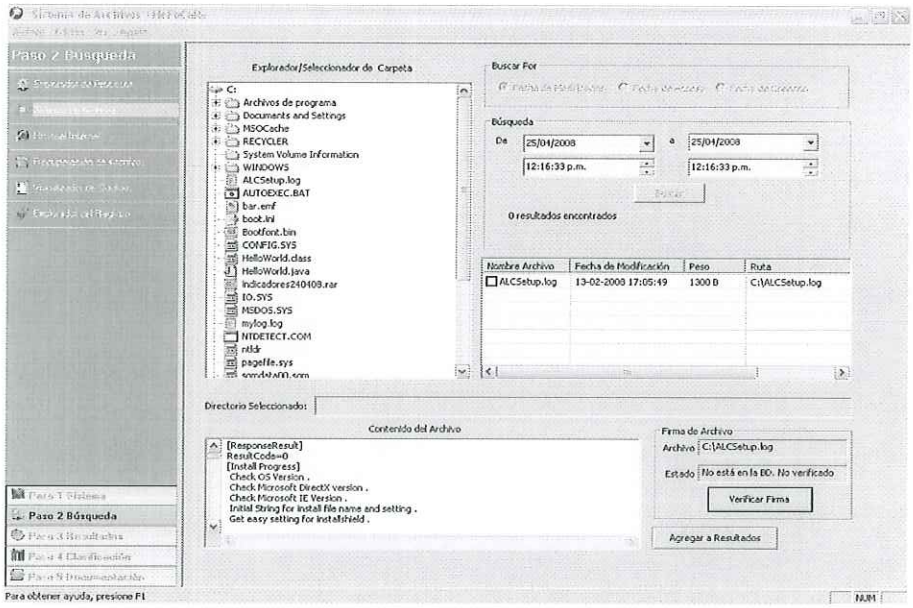

Figura 6.5 Interfaz del Módulo Sistema de Archivos de Hefoclase.

En la figura 6.5 se muestra el módulo de Sistema de Archivos de Hefoclase, integrado por varios elementos gráficos que dividen cierta funcionalidad del módulo. Una parte es donde se puede visualizar todos los archivos y carpetas del sistema, ya sea que tenga los atributos de

oculto, del sistema o sólo lectura. En Hefoclase esta parte que se muestra en la figura 6.6 se llama Explorador/Seleccionador de Carpeta/Archivos.

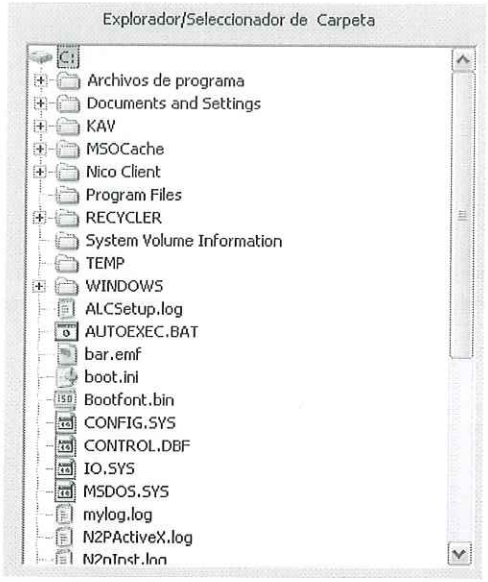

Figura 6.6 Explorador/Seleccionador de Carpeta.

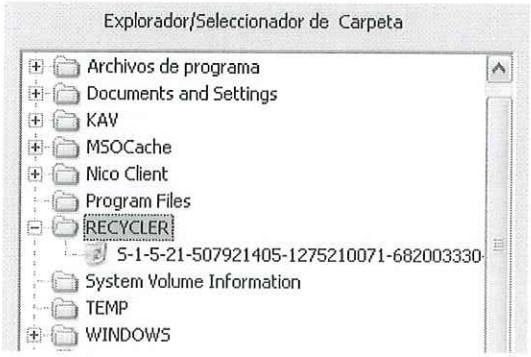

Figura 6.7 Seleccionando un directorio o carpeta desde el Explorador.

Figura 6.8 Visualizador de Directorio Seleccionado.

Si se selecciona un directorio del Explorador/Seleccionador de Carpeta, la ruta de ese directorio se muestra en el visualizador de Directorio Seleccionado tal como se muestra en las figuras 6.7 y 6.8. Cuando se selecciona algún directorio se activa la opción para poder buscar en esa ruta archivos con determinados atributos como los Tiempos de Acceso, Modificación y Creación. Estos atributos nos sirven para conectar en la línea del tiempo algún evento malicioso que queremos comprobar. Aunque esta técnica muchas veces no nos dice mucho cuando se trata de archivos o carpetas que fueron empaquetados o comprimidos con herramientas especificas que permiten modificar esos atributos en cualquier archivo del sistema operativo Windows. En la figura 6.9 se muestra la opción que se activa cuando se selecciona una carpeta, esa opción nos permite definir porque tipo de atributo se realizará la búsqueda, ya sea por fecha de modificación, fecha de acceso o fecha de creación.

Figura 6.9. Tipos de atributo para realizar la búsqueda de archivos en Hefoclase.

Una vez definido el atributo en el que se va a buscar, se definen los tiempos específicos o parámetros que comparará el sistema para traer los resultados. En la figura 6.10 se muestra como se define el parámetro de fecha. Se presiona en el elemento *combo box* y se despliega un calendario para facilitar la especificación del parámetro de fecha.

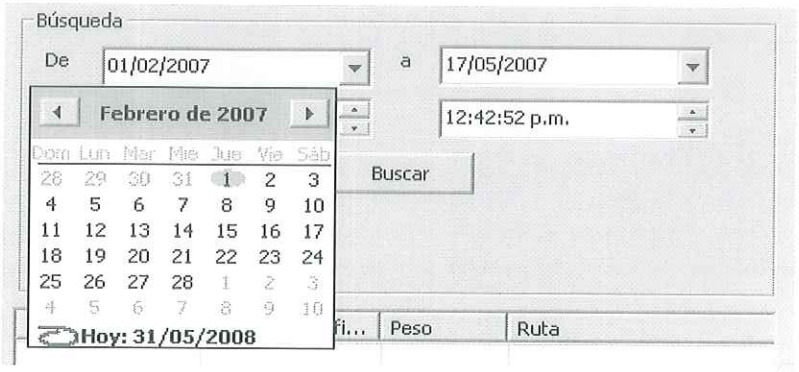

Figura 6.10 Especificando parámetro Fecha en el Módulo Sistema de Archivos.

Para poder realizar una búsqueda más exacta se tiene como opción definir la hora en que fue creado determinado archivo, las herramientas de búsqueda gráficas nativas del sistema operativo no permiten definir este parámetro. En Hefoclase se estableció como imperativo por ello que se adhiere a la funcionalidad del sistema. En la figura 6.11 se observa cómo se define este otro parámetro de la búsqueda.

Figura 6.11. Definiendo la hora de la Búsqueda en el módulo Sistema de Archivos.

Una vez definidos los parámetros de búsqueda, se procede a realizar tal proceso, esto se hace presionando el botón de Buscar, en la figura 6.12 se muestra como se está llevando a cabo la búsqueda, se puede observar que los parámetros están definidos y la etiqueta del botón Buscar cambio a Cancelar, también se puede ver en la parte inferior de la figura la ruta donde se está buscando en tiempo real.

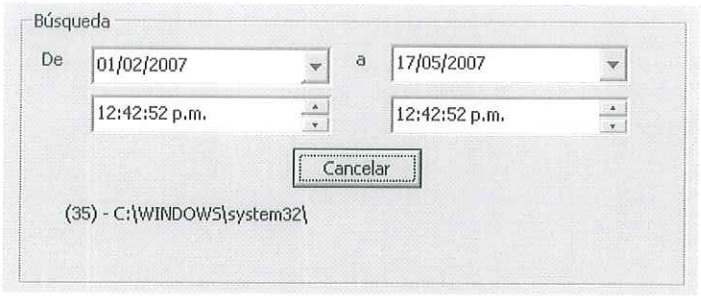

Figura 6.12. Realizando la búsqueda en el sistema de archivos.

Cada archivo que concuerda con los parámetros de búsqueda, es agregado a la lista de resultados, trayendo los atributos para corroborar la exactitud de la búsqueda, también se despliega el nombre del archivo, ruta y peso. A continuación se muestra la figura 6.13 donde se observa la lista de resultados encontrados en determinada búsqueda.

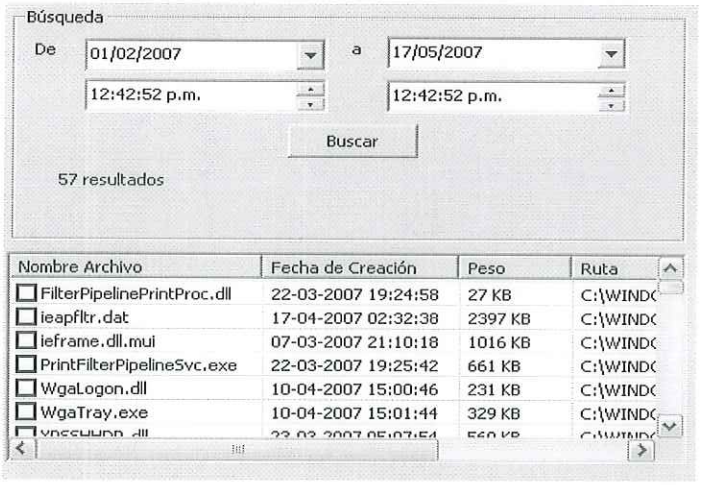

Figura 6.13 Resultados arrojados por una búsqueda en el módulo Sistema de Archivos.

Una vez arrojados los resultados se pueden seleccionar para cotejar su integridad, esto se hace seleccionando el resultado de la lista y después se procede a verificar su contenido y firma digital.

Figura 6.14 Selección de un resultado para su verificación.

Para verificar la integridad o descartar si un archivo es malicioso o no se puede ver su contenido y comprobar su firma digital de acuerdo a su extensión. Para ello la herramienta te permite ver el contenido del archivo y comparar la extensión con su firma digital. En la figura 6.15 se muestra el contenido del archivo seleccionado de la lista de resultados.

Figura 6.15. Contenido del archivo seleccionado de los resultados.

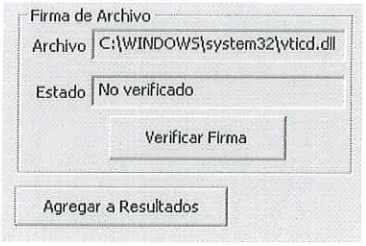

Figura 6.16. Firma del archivo no verificada.

Como se puede observar en la figura 6.16 se muestra la ruta del archivo seleccionado y el estado de la comparación de su firma digital, para realizar la comprobación es necesario presionar el botón de verificar firma, el sistema hará un proceso comparativo con su base de datos interna a partir de la extensión del archivo. Hay que mencionar que la firma no es más que

los primeros 20 bytes de un archivo que caracterizan a determinado tipo de archivo, comúnmente identificado por su extensión. La relación de la firma con el tipo de archivo en Hefoclase, fue creada a partir de varias herramientas comerciales populares, entre ellas está el editor hexadecimal WinHex 13.9 y la herramienta de análisis forense en vivo ProDiscover 4.8 además de información encontrada de las firmas y tipos de archivo en FileSig [FILESIG, 2007]. Hay que decir que es difícil establecer una relación universal, ya que no existe un estándar para esto, ya que cada propietario de algún formato modifica o lanza nuevas versiones que cambien su firma, por ello que es complicado tener una eficacia alta en la detección de esta información y por ello que han sido creados algunos sitios e iniciativas para mitigar este problema.

Figura 6.17. Firma del archivo verificada.

El sistema despliega 3 tipos de mensaje de acuerdo al resultado de la comparación de las firmas con su base de datos, el primero de ellos es "verificado correcto", como se puede apreciar en la figura 6.17, el segundo de ellos es "verificado incongruente" tal como se muestra en la figura 6.18. Este mensaje aparece en el estado de la firma cada vez que encuentre una comparación que no concuerda con el tipo de extensión del archivo. Esto tiene varias causas posibles:

1. Que el archivo haya sido manipulado o comprometido por algún método malicioso, por ejemplo un archivo de una imagen .PNG que en realidad es un ejecutable .EXE pero que alguien lo renombró.

2. Que un proceso legítimo del sistema operativo haya modificado la firma, como lo puede ser el caso en que se actualice el sistema.
3. Que un propietario creador del archivo haya cambiado la firma.
4. Por último que puede existir una extensión de archivo repetida de alguna otra compañía.

Otro mensaje que Hefoclase puede mostrar al momento de verificar la firma es "No está en la bd" tal como se muestra en la figura 6.19. Esto significa que la extensión no ha sido encontrada en la base de datos interna del sistema y que por lo tanto no ha sido evaluada.

Figura 6.18. Verificado incongruente de la firma de un archivo.

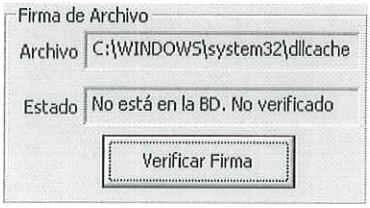

Figura 6.19 Estado que se muestra cuando no reconoce la extensión de un archivo.

El módulo sistema de archivos de Hefoclase nos ayuda a encontrar pistas que nos indiquen cuales fueron las causas en algún posible evento malicioso. Al final del proceso de búsqueda el investigador determinará si es importante añadir esa información respecto a un archivo sospechoso a los resultados. Para ello simplemente se marca el archivo de la lista y se presiona el botón Agregar a Resultados.

6.3 Módulo Historial de Internet.

Cuando se lleva a cabo una investigación forense, en algunos casos relacionados con hostigadores obsesionados, es de suma importancia conocer el ámbito, actividades y hábitos de navegación del sujeto en cuestión, ya que toda esa información puede arrojar información valiosa acerca del comportamiento del individuo, por ejemplo si navega en páginas de citas en línea, si ha visitado páginas pornográficas, entre otros datos que ayuden a los psiquiatras forenses en la investigación acerca de perfil del sospechoso [McGrath y Casey, 2002].

Comúnmente todos los navegadores de internet utilizan un proceso para almacenar el historial de páginas visitadas, además de guardar los archivos de algunos sitios visitados. Esto tiene como fin agilizar el acceso a esos contenidos, estos últimos son llamados archivos de caché del navegador o bien en el caso de Microsoft Internet Explorer, Archivos Temporales de Internet. Estos archivos son una fuente de evidencia digital importante, ya que dan pistas específicas de lo que se ha desplegado en el monitor de algún sospechoso. Sin embargo debido a la diversidad de navegadores que se utilizan hoy en día puede ser una tarea un poco compleja el desarrollar un proceso genérico para todos los navegadores. En la práctica esto no es posible y la solución que se implementa es que se crea un proceso determinado por navegador o conjunto de navegadores que comparten módulos de funcionalidad en su estructura. Además existe otro problema, esos datos se pueden borrar desde la interfaz de cada navegador y también se pueden perder ejecutando programas que van en pro de la privacidad del usuario final. Este tipo de programas pueden complicar a la investigación llevada por el forense. También pueden hacer que se pierda posible evidencia digital, lo que trae consigo un arma de doble filo, por una parte

los reclamos de usuarios íntegros y responsables de tener el derecho de proteger su privacidad y por otro lado aquellos que lo utilizan para borrar comportamientos ilegales o enfermizos.

El módulo de Hefoclase para recuperar el historial y la caché de internet fue desarrollado específicamente para operar sobre el navegador más usado del mundo como lo es Microsoft Internet Explorer [W3C, 2007], ya sea la versión 5, 6 o 7. Para entender el funcionamiento de este proceso es necesario decir que desde la versión 5 de Internet Explorer, el navegador guarda numerosos archivos "index.dat" dentro del directorio del usuario de Windows. Este archivo mapea los sitios webs visitados en archivos caché guardados en directorios con nombres aleatorios de tal manera que cuando el usuario visite de nuevo un sitio Web no tenga que descargar sus contenidos de nuevo. El archivo "index.dat" se puede encontrar en diversas rutas dependiendo del sistema operativo instalado. La tabla 4 muestra la relación de la ruta donde se almacena el archivo index.dat en varios sistemas operativos Windows.

Sistema operativo	Ruta
Windows 95, 98, ME	\Windows\Archivos Temporales de Internet\Content.IE5 \Windows\Cookies \Windows\Historial\History.IE5
Windows NT	\Winnt\Profiles\<nombre de usuario>\Configuración Local\Archivos Temporales de Internet\Content.IE5 \Winnt\Profiles\<nombre de usuario>\Cookies \Winnt\Profiles\<nombre de usuario>\Configuracion Local\Historial\History.IE5
Windows 2000/XP	\Documents and Settings\<nombre de usuario>\Configuracion Local\Archivos Temporales de Internet\Content.IE5 \Documents and Settings\<nombre de usuario>\Cookies \Documents and Settings\<nombre de usuario>\Configuracion Local\Historial\History.IE5
Windows Vista	\Users\<nombre de usuario>\AppData\Local\Microsoft\Windows\Archivos Temporales de Internet\Content.IE5 \Users\<nombre de usuario>\AppData\Roaming\Microsoft\Windows\Cookies \Users\<nombre de usuario>\AppData\Local\Microsoft\Windows\Historial\Baja

Tabla 4. Localización o ruta de los archivos index.dat en varios Sistemas Operativos de Windows.

El archivo index.dat es un almacén de datos acerca de todos los archivos que se pueden abrir en el navegador, esto incluye tanto localmente o en línea. Para entender mejor como es que guarda Internet Explorer estos datos se tiene que investigar cuales son las estructuras internas de este archivo, para ello puede ser beneficioso el uso de herramientas de ingeniería inversa como lo puede ser un editor hexadecimal.

El archivo index.dat es más bien una estructura de datos para almacenar registros, en este caso maneja una tabla de hash, la cual contiene información relacionada con los directorios de la caché, la dirección visitada, el tipo de entrada, la longitud de la tabla hash, un apuntador a la siguiente tabla y algunas otras banderas [Jones18, 2003].

La información más relevante del archivo index.dat son los registros de actividad del navegador, estos son compuestos por 3 elementos básicos: el tipo, la longitud y los datos. El tipo es un campo de 4 bytes de longitud que puede contener 3 palabras clave: REDR, URL y LEAK, que identifican a cada tipo de registro de actividad. La longitud contiene el tamaño de los bloques de los registros expresados en bytes, y los datos son todo lo referente a las direcciones y demás banderas.

Es importante definir qué significado tiene cada tipo de registro, el REDR es un tipo de registro que se inserta cuando el navegador de un usuario fue redirigido hacia otro sitio. El tipo URL es el más importante ya que es un conjunto de datos que representa a la URL o dirección del sitio que se ha visitado. Por último el tipo LEAK es un registro de actividad que tiene la misma estructura interna que el tipo URL y la diferencia es el tipo de acceso con el que se llega a tal dirección, por ejemplo el LEAK podría contener una dirección relacionada con una petición que es redirigida hacia otra URL, como lo puede ser el caso en que se quisiera descargar un archivo de una página pero es redirigido hacia otra y finalmente descargado el archivo.

Comúnmente se puede relacionar un tipo REDR con el tipo LEAK siendo el REDR el inicio y LEAK el final. El tipo LEAK nunca será inicio de una petición de dirección en el navegador, es decir, todas las direcciones escritas desde la barra de direcciones del navegador nunca serán de este tipo. Un registro LEAK nunca estará en el historial del navegador, pero si en el registro de la cache del navegador.

La estructura interna de los tipos LEAK y URL contiene una serie de campos que son mostrados en la tabla 5.

Nombre del campo	Offset desde el inicio del registro	Tamaño en bytes	Descripción
Tipo de Registro	0x0	4	Campo que contiene la cadena del tipo "URL" o "LEAK"
Tamaño del registro	0x4	4	Numero de bloques de 0x80 bytes que contiene el registro
Sello de Tiempo de la última modificación	0x08	8	Sello de tiempo de la última modificación en formato FILETIME
Sello de tiempo del último acceso	0x10	8	Sello de tiempo del último acceso en formato FILETIME
Offset de la URL	0x34	4	Offset de la URL desde el inicio del registro
Offset del nombre del archivo	0x3c	4	Offset del nombre del archivo desde el inicio del registro
Índice del directorio local de la caché	0x38	1	Índice (empezando desde 0) de los directorios locales conteniendo los archivos de la cache.
Offset de la cabecera HTTP	0x44	4	Offset desde el inicio donde están localizadas las cabeceras http

Tabla 5. Campos más importantes del tipo de registro URL y LEAK.

La herramienta Hefoclase genera una lista con todos estos campos de forma que se pueda visualizar correctamente, tal como se puede apreciar en la figura 6.20, donde se muestra un archivo index.dat de la caché del navegador de una computadora.

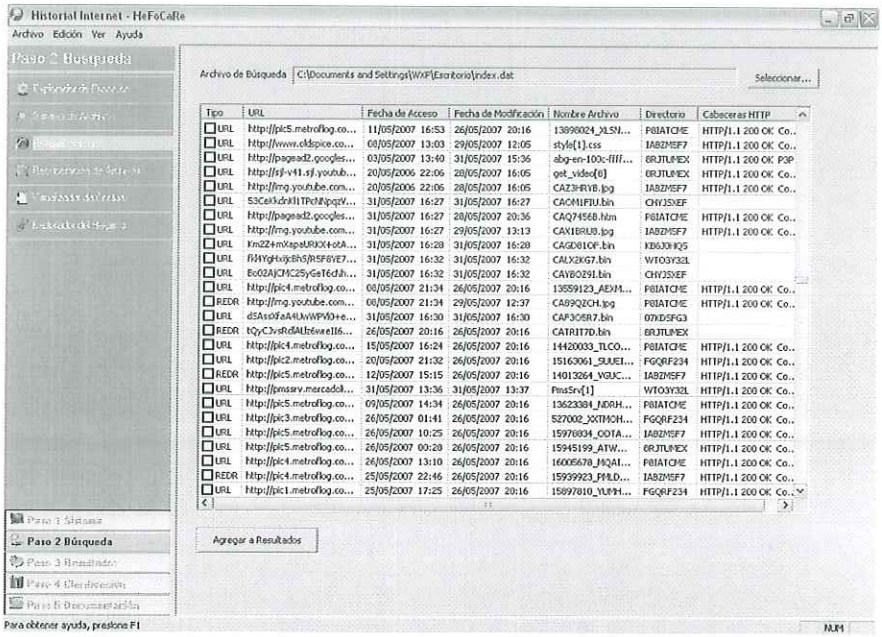

Figura 6.20 Archivo index.dat de la caché de una computadora cargado en la Interfaz del módulo Historial de Internet de Hefoclase.

6.4 Módulo Recuperador de Archivos.

Otro módulo de la herramienta Hefoclase es el Recuperador de Archivos, el objetivo de esta funcionalidad es tratar de encontrar pistas y metadatos acerca de los documentos que han sido "eliminados" de Windows a través de la papelera de reciclaje. Los usuarios que no conocen mucho del sistema operativo Windows tienden a creer que borrando un archivo del explorador de Windows o de alguna otra carpeta, realmente ya lo ha perdido para siempre, sin embargo los que estudian o utilizan un poco más el sistema operativo saben que esto no es así, sino que cada vez que se borra un archivo de alguna carpeta del sistema se va a un lugar llamado papelera de reciclaje. Este lugar es un repositorio de archivos eliminados a través de la interfaz del sistema operativo, a primera vista revela información acerca de los archivos que han sido suprimidos, datos como fecha de eliminación, ubicación original entre otros datos. También nos da la opción

de restaurarlo hacia el lugar de origen de donde fue eliminado. En la actualidad cualquiera puede saber esto, lo que no cualquiera sabe es como trabaja la papelera de reciclaje y que ventajas podría implicar ese conocimiento.

Para que el sistema operativo pueda conocer los metadatos del archivo eliminado es necesario que lo guarde en alguna estructura interna. Esta estructura interna en realidad es un archivo, el nombre de ese archivo que guarda esta información se llama INFO2 y se encuentra localizado en el directorio de la papelera de reciclaje en la raíz de la partición principal de disco duro. La ruta de este archivo varía dependiendo del sistema de archivos en que este instalado, para FAT32 la ruta es \Recycled\INFO2, y para NTFS \Recycler\<SID del usuario>\INFO2, siendo <SID del usuario> una clave única de identificación mantenida por el sistema operativo.

Cuando un archivo es eliminado es renombrado a DC#.EXT donde # es un entero y EXT es la extensión del archivo original. El número # es importante ya que es único, cada vez que se mueve un archivo a la papelera este número entero se incrementa en 1, además de que funciona como el índice de la estructura de dato interna que almacena esos archivos [Jones19, 2003].

Las estructuras de datos internas del archivo INFO2 son mostradas en la tabla 6.

Nombre del campo	Offset desde el inicio del registro	Tamaño en bytes	Descripción
Tamaño del registro de la papelera	0xC	4	Tamaño en bytes del registro
Nombre del archivo de la papelera	0x04	Variables o NULL	Nombre del archivo eliminado
Id. único del registro de la papelera	0x108	4	Identificador concatenado al archivo eliminado (#)
Número de unidad del archivo eliminado	0x10C	4	Entero que representa la unidad de la que fue eliminado el archivo. Siendo 0 A:, 1 B: 2 C:
Fecha de eliminación	0X110	8	Fecha de eliminación del archivo
Peso del archivo eliminado	0x118	4	Peso del archivo eliminado

Tabla 6. Estructuras de datos del archivo INFO2 de la papelera de reciclaje.

El módulo recuperador de archivos de Hefoclase, muestra de una forma ordenada en tabla, las estructuras más importantes del archivo INFO2. Además ofrece la oportunidad de visualizar el contenido del archivo en modo binario. En la figura 6.21 se muestra la interfaz de este módulo.

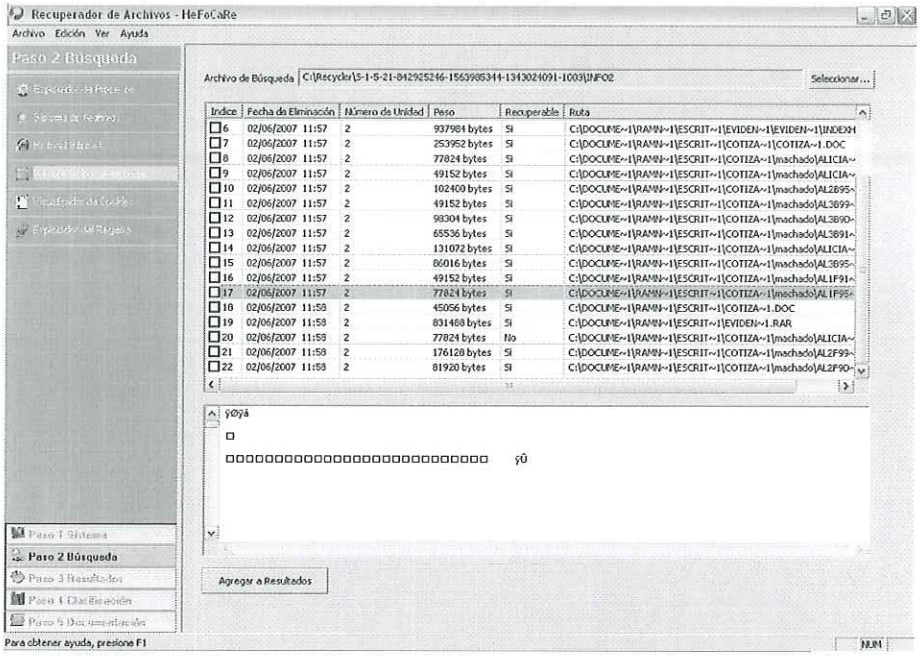

Figura 6.21 Módulo de Recuperador de archivos, se muestra el contenido de un archivo.

Dentro de la tabla que ofrece este módulo, se agrega una columna para identificar si un archivo es recuperable o no. Cuando un archivo se marca como no recuperable significa que alguna vez se borró y que estuvo en la papelera de reciclaje, sin embargo no se puede visualizar su contenido. Para poderlo recuperar existen otras técnicas más avanzadas como buscar en las estructuras internas del propio sistema de archivos, aunque esto no garantiza que se puede recuperar, sin embargo es más probable encontrarlo con ese tipo de técnicas. El mensaje que despliega Hefoclase cuando no se puede recuperar y por lo tanto visualizar el archivo es el que se muestra en la figura 6.22.

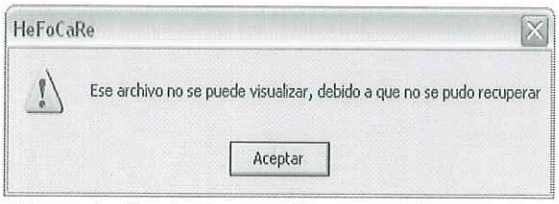

Figura 6.22 Mensaje de alerta cuando un archivo no es recuperable.

6.5 Módulo Visualizador de Cookies.

Comúnmente cuando visitamos un sitio Web, intercambiamos información hacia el servidor donde está alojada la página que estamos visualizando en nuestro navegador. Cuando se trata de manejar la sesión de algún servicio que ofrezca determinado sitio como lo puede ser la sesión del correo o de alguna cuenta en un foro, existe un mecanismo de intercambio para almacenar datos en el cliente que permite recordar al servidor que usuario es y pueda cargar su configuración predeterminada. Las Cookies son esos archivos que guardan información respecto de un servidor Web en la máquina cliente, guarda configuraciones preestablecidas, datos de área geográfica, idioma, entre muchos otros campos o variables que pudiera contener.

Para un investigador resulta útil visualizar este tipo de archivos de una manera legible, de tal forma que permita identificar pistas que lleven a determinar el curso que llevó a cabo algún usuario malicioso en determinada computadora. Los sistemas operativos Windows tienen un directorio preestablecido donde guardan las Cookies, comúnmente se puede ver en las siguientes rutas: \Documents and Settings\<nombre de usuario>\Cookies para los sistemas basados en Windows 2000 y XP o \Windows\Cookies en sistemas más viejos como Windows ME,98 y 95. Éste directorio es leído y gestionado principalmente por el navegador de Microsoft Internet Explorer.

Cuando un usuario visita alguna página que maneja Cookies, dependiendo de la configuración del navegador se graba esa Cookie en los directorios preestablecidos por el sistema.

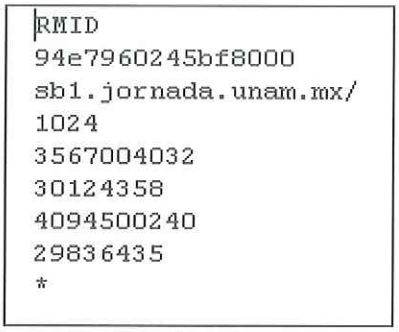

Figura 6.23 Contenido de una cookie vista en un editor de texto.

Como podemos observar en la figura 6.23 el contenido de una cookie es relativamente sencillo de leer ya que está en formato ASCII, sin embargo la interpretación de estos campos no es directamente visible, lo más significativo visiblemente en la figura 6.23 es que hay una dirección de una página web. Observando y analizando más a fondo el contenido de este archivo y con la ayuda de un editor hexadecimal podemos ver que el formato de una cookie del navegador Internet Explorer, se compone de 9 campos que son descritos en la tabla 7. Los campos que más pueden confundir son los valores de las variables ya que puede ser cualquier valor arbitrario dependiendo del servidor que se visite, otro campo que no es claro son los números que aparecen de la línea 5 a la 8, estos números conformados por máximo 10 enteros son las fechas de creación y expiración de la cookie pero en formato UNIX o de la estructura FILETIME. Para visualizar correctamente estos campos se tiene que convertirlos a un formato más legible como lo puede ser el formato de calendario gregoriano.

Línea	Descripción
1	El nombre de la variable
2	El valor de la variable
3	El sitio web dueño de la cookie. El que la creó
4	Banderas opcionales
5	El entero más significativo para la fecha de expiración, en formato FILETIME
6	El entero menos significativo para la fecha de expiración, en formato FILETIME
7	El entero más significativo para la fecha de creación, en formato FILETIME
8	El entero menos significativo para la fecha de creación, en formato FILETIME
9	Delimitador del registro de Cookies (el carácter * para IE)

Tabla 7. Campos que componen a una cookie en Internet Explorer.

Los archivos de las Cookies almacenados en el sistema de archivos pueden estar formados por varias Cookies. Es decir puede contener varios separadores que delimiten una cookie de una o varias más. La herramienta Hefoclase muestra de una forma legible todos estos archivos que se encuentran en el directorio predefinido por el sistema operativo, permite seleccionar determinado archivo y visualizar el contenido de cada cookie de una manera intuitiva, independientemente que este archivo contenga numerosas Cookies. Solamente con seleccionar el archivo de la lista que genera la herramienta, se podrá ver su contenido. La tarea del investigador es determinar que cookie puede ser útil para la investigación que está llevando a cabo, de ahí que la pueda agregar a los resultados de evidencia y posteriormente poderla clasificar o agregar metadatos acerca del contexto que envuelve dicho elemento. En la figura 6.24 se muestra la lista de archivos encontrados en un sistema de prueba.

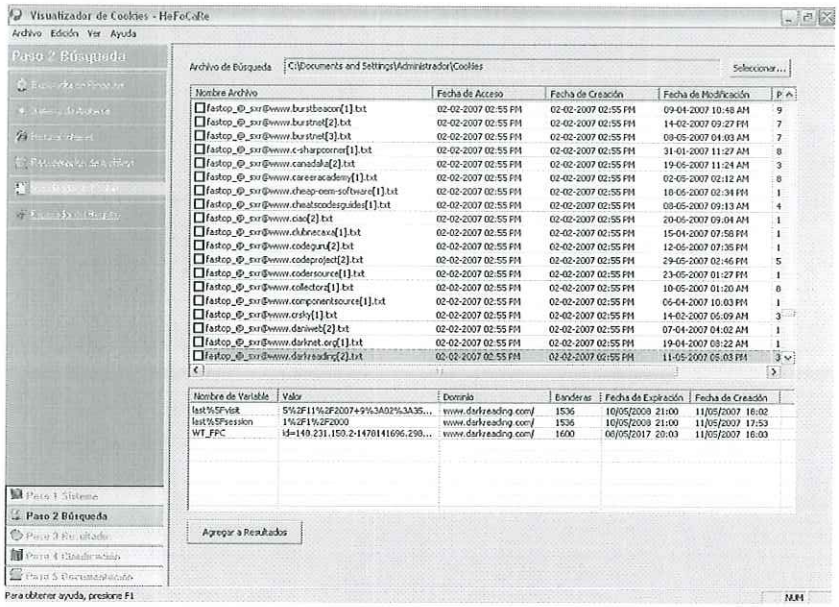

Figura 6.24 Mostrando el módulo de Visualizador de Cookies en un sistema de prueba.

6.6 Módulo Explorador de Registro.

El registro ha sido la base de datos de configuración principal para Windows y las aplicaciones que soporta desde que se introdujo Windows 95. Desde el registro de Windows un examinador forense puede descubrir la configuración de software y hardware, las preferencias de usuario, configuración inicial del sistema e incluso ver el código de algún malware [Russinovich, 2000].

En la actualidad es indispensable comprender la estructura interna del registro, ya que entre más capacitado esté el investigador o examinador forense, más fácil será la tarea de encontrar evidencia digital. Conociendo las partes críticas donde se pueden guardar configuraciones importantes o rutas específicas donde se puede limitar algunas funcionalidades de seguridad

del propio sistema operativo u otras aplicaciones, puede llevar a realizar una lista de lugares de búsqueda críticos en la investigación.

El malware como ya se ha comentado en el presente trabajo, tiende a recurrir a las zonas críticas de un sistema para poderlo vulnerar, en este caso el registro de Windows es un parte clave para el funcionamiento del sistema operativo, si no se toman medidas adecuadas para protegerlo puede ser una puerta abierta fácil de utilizar para propósitos maliciosos.

El comando REGEDIT de Windows ejecuta el binario Regedit.exe que es el Editor de Registro del sistema, cuando se tiene permisos de Administrador en el sistema operativo, esta herramienta puede ser de beneficio si se sabe utilizar, ya que puede permitir controlar y mitigar ciertos riesgos de seguridad que podrían afectar al sistema, por eso es indispensable tener un conocimiento profundo de su funcionamiento.

Cuando un atacante gana permiso de Administrador en un sistema puede realizar todo lo que se le pueda ocurrir y entre ellos está en agregar, eliminar o modificar entradas al registro, aunque esto puede no ser nada comparado con la eliminación permanente de archivos o robo de información sensible. La realidad es que existen diversas formas para ganar acceso de administrador o root en un sistema, lo peor se da cuando un usuario local facilita la tarea del atacante. Por ejemplo la gran mayoría de usuarios conectados a internet navega con una cuenta que tiene permisos de privilegio, esto es una falta de educación en cuanto a temas de seguridad informática, ya que existen numerosos *exploits* que pueden hacer uso de diversas vulnerabilidades de programas propietarios o nativos para ejecutar código con permisos del usuario local. Si un usuario local está con los permisos de administrador y es víctima de una vulnerabilidad, entonces se puede ejecutar código arbitrario y por lo tanto también se puede

estar ingresando datos hacia el registro de Windows, limitando la funcionalidad y estabilidad del sistema o satisfaciendo los requerimientos del atacante.

El registro de Windows se compone de muchas entradas, llamadas claves o llaves. Las llaves pueden contener a su vez otras llaves y éstas al final pueden tener valores asignados u otras llaves, es decir, la estructura del registro es jerárquica en forma de árbol tal como se muestra en la figura 6.25.

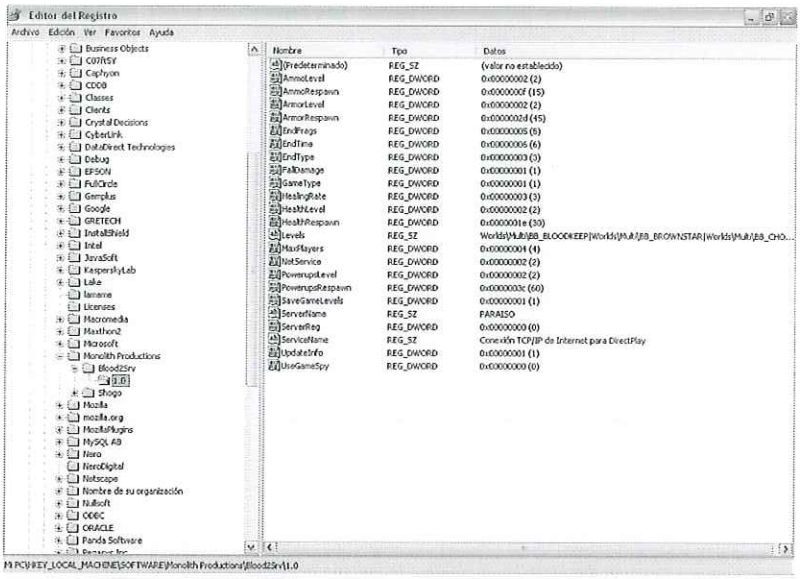

Figura 6.25 Aplicación del Registro de Windows.

El registro está formado por varias claves raíz, dependiendo del sistema operativo pueden ser de 5 a 6 o hasta 7 claves raíz llamados "hives". También es posible que solamente se tenga acceso solo a ciertas áreas del registro o se visualicen solo 2 claves raíz, esto último pasa si se está administrando el registro remotamente. Estas claves raíz sirven como guía de división estructural de la configuración del sistema y su nombre empieza con la cadena "HKEY", cada una contiene una cierta variedad de configuración posible. Las claves más importantes y consistentes en la mayoría de sistemas Windows son las siguientes:

1. HKEY_USERS. Contiene información acerca de todos los usuarios del escritorio, además de tener una descripción genérica del usuario. La información almacenada en esta clave son temas respecto a configuración de aplicación y configuraciones visuales.
2. HKEY_LOCAL_MACHINE. Contiene información específica de la computadora que se relaciona directamente con la máquina en que el sistema operativo esta ejecutándose.
3. HKEY_CLASSES_ROOT. Contiene reglas de arrastre de elementos, accesos directos y de menú contextual.
4. HKEY_CURRENT_USER. Contiene información específica del usuario cuando ingresa al sistema y es construida inicialmente por la información genérica en la clave HKEY_USERS.
5. HKEY_CURRENT_CONFIG. Almacena información respecto a la actual configuración del sistema, además de tener la actual configuración de hardware.
6. HKEY_DYN_DATA. Contiene información del estado dinámico para dispositivos que usan la arquitectura plug and play.
7. HKEY_PERFORMANCE_DATA. Provee soporte para el monitorio de rendimiento sobre los sistemas basados en el núcleo de NT.

Hefoclase permite realizar búsquedas de valores, claves y claves ocultas en el registro. Las claves ocultas son claves que se esconden de la API de Windows, es decir que no se puede visualizar completamente [Cogswell y Russinovich, 2006]. La interfaz del módulo de Explorador de Registro está compuesta por una serie de elementos que nos permiten definir los criterios y términos de búsqueda, así como también visualizar valores de claves y resultados. Para empezar con una búsqueda primero se debe seleccionar el rango donde se quiere buscar, para ello

simplemente se selecciona una llave y la herramienta buscará desde esa llave hasta el final del registro. En la figura 6.26 se muestra como se selecciona la llave raíz de búsqueda.

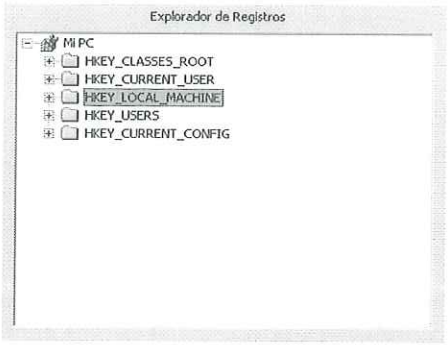

Figura 6.26 Seleccionando la llave raíz de inicio de búsqueda en el Explorador de Registro.

Se puede definir cualquier llave de inicio de búsqueda, una vez que se haya seleccionado se despliega la clave en la etiqueta Buscar desde, tal como se muestra en la figura 6.27.

Figura 6.27 Rango de inicio de la búsqueda en el Explorador de Registro.

Hay que decir que cuando se selecciona una llave, su contenido es mostrado en la parte derecha del módulo de Explorador de Registro, a este elemento de la interfaz del módulo de Explorador de Registro, es llamado Visualizador de Llaves.

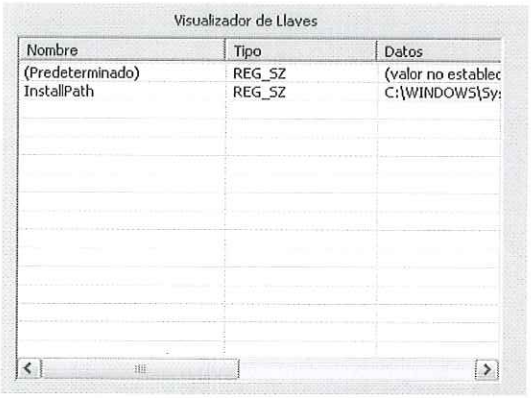

Figura 6.28 Visualizador de llaves del módulo Explorador de Registro.

Después de decidir donde se quiere buscar, se define en qué tipo de valores se va a buscar y el término que se está buscando.

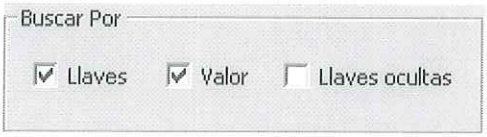

Figura 6.29 Selección del tipo de elemento de búsqueda en el Explorador de Registro.

Figura 6.30 Introduciendo el término de búsqueda en el Explorador de Registro.

El proceso de búsqueda se inicia cuando el usuario presiona el botón buscar, los resultados que encuentra los agrega a la lista de Resultados en la parte inferior del módulo Explorador de Registro.

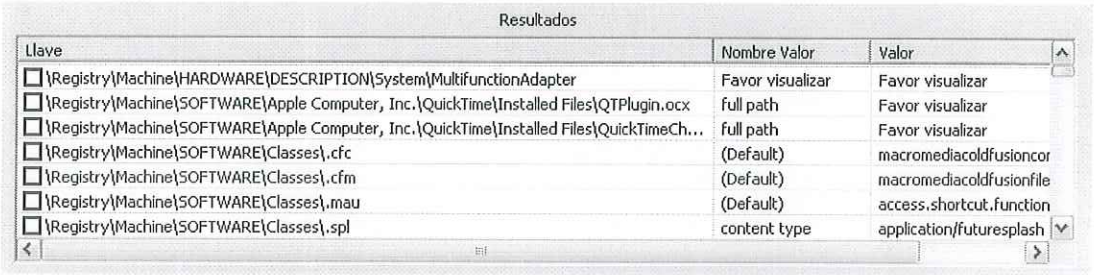

Figura 6.31 Resultados arrojados por la búsqueda del módulo Explorador de Registro.

Para corroborar el resultado se puede dar doble clic en algún resultado de la lista y automáticamente Hefoclase abre esa llave en el Visualizador de llaves, cabe resaltar que dependiendo de los criterios en los que se definió la búsqueda se puede comprobar su exactitud, es decir el término de búsqueda puede estar ya sea en un valor, una llave o una llave oculta.

6.7 Módulo de resultados.

Luego de haber agregado pistas o elementos de evidencia digital es buena idea tener una panorámica general de todo lo que se agregado como resultado de evidencia, para ello se muestra un sumario general de lo que se tiene hasta el momento registrado en la base de datos interna de la herramienta.

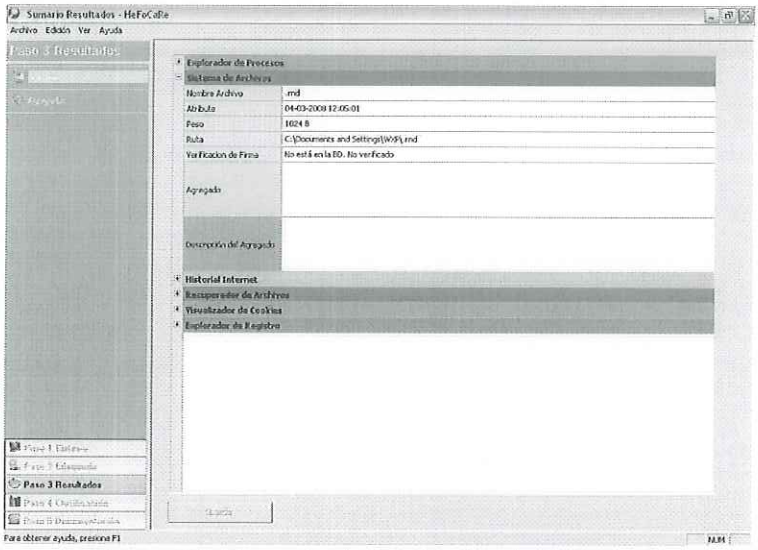

Figura 6.32 Sumario de resultados de Hefoclase.

Después de haber terminado el paso de búsqueda de evidencia, se puede proceder a agregar metadatos a la misma. Para ello hay una sección llamada "Agregados", donde se permite ingresar cualquier descripción, bitácora, ruta o comentario generado ya sea por la misma herramienta o por otras. Con esto se busca flexibilidad para fortalecer el contexto de un posible escenario acerca de ese elemento de evidencia.

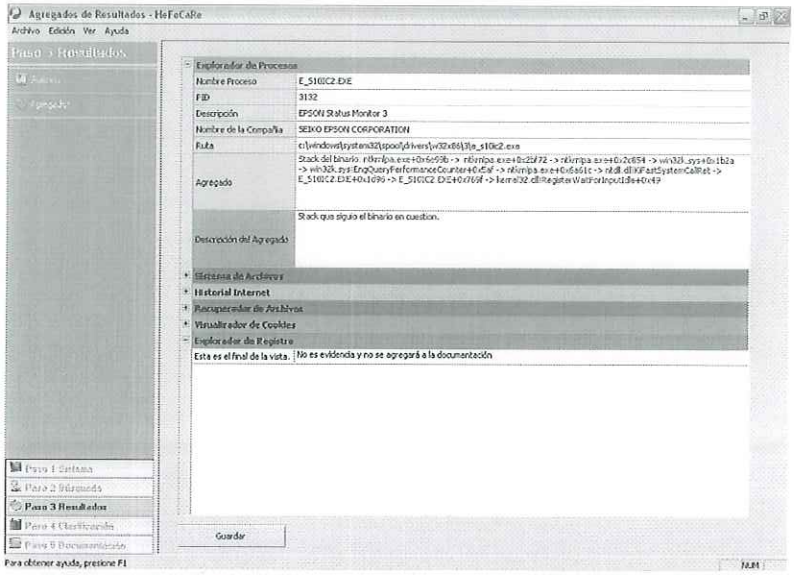

Figura 6.33. Metadatos agregados en el módulo Resultados

6.8 Módulo de Clasificación.

Una vez agregados ciertos metadatos arbitrarios, se pasa al módulo de clasificación, que es otro filtro por el que pasa la evidencia, pero ahora en este caso son conceptos o temas predefinidos que permiten fortalecer el curso de la investigación. En ninguna de las herramientas presentadas en el capítulo 3, se maneja por elemento cada pequeño resultado de evidencia, sin embargo esto es útil a la hora de trata de unir el rompecabezas. Para ello lo primero que se debe hacer es escoger el módulo del que se quieren clasificar la evidencia, después se escoge el número de evidencia o resultado que se agregó desde el módulo de búsqueda respectivo, y después se empieza a seguir agregando metadatos a la evidencia. En las figuras 6.34 6.35 y 6.36 se muestran como se lleva a cabo en Hefoclase estas últimas tareas.

Figura 6.34 Selección del módulo a clasificar.

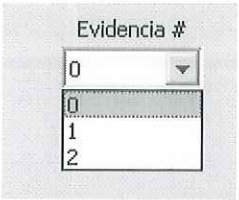

Figura 6.35. Selección del número de evidencia a clasificar

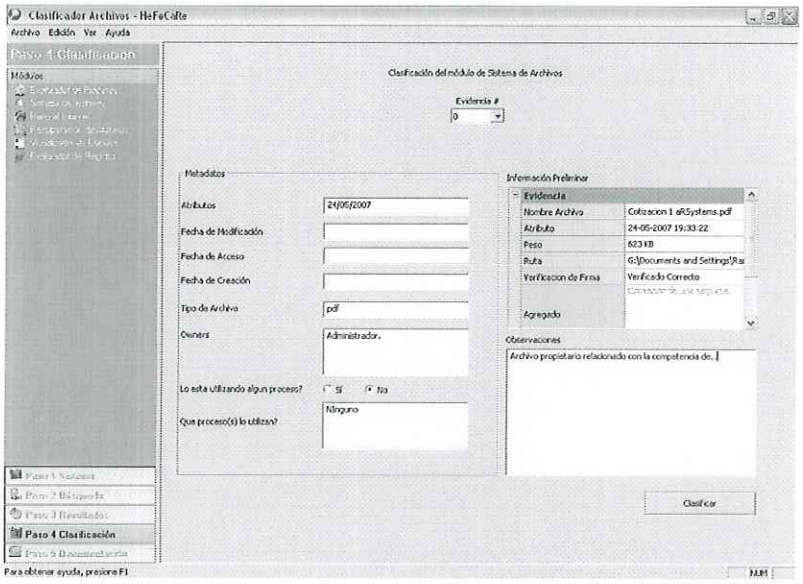

Figura 6.36. Clasificación de un elemento del módulo de sistema de archivos.

Es importante mencionar que cuando se clasifica evidencia se habla acerca de la cadena de custodia y de la manera en que se hace el reporte de una investigación. Los metadatos que se puedan extraer o agregar a algún elemento de evidencia deben de aportar argumentos para no

llegar a conclusiones ambiguas, pero la falta de una visión forense acerca de estos datos componentes de muchas estructuras que utilizan los medios digitales, lleva a considerar no solamente los sistemas de archivos sino cualquier dispositivo que almacene información.

Los metadatos pueden ser vistos como parte de algún objeto que ha sido transmitido o almacenado digitalmente. Muchos de estos metadatos generalmente se generan automáticamente por los mecanismos que los controlan, las fechas de modificación, acceso y creación de un archivo en cualquier sistema de archivos son manejados por el propio sistema operativo, valiéndose de las estructuras del sistema de archivo. Pero muchas veces estos metadatos son fáciles de modificar, la falta de conocimiento del origen de un archivo puede muchas veces hacer perder tiempo en una investigación o simplemente pasar por alto algún detalle relevante.

Se ha discutido cuales serían los elementos deseables que guardara como metadatos algunos sistemas de archivos, estos elementos podrían ser útiles para la investigación de un caso [Buchholz y Spafford, 2004].

CAPÍTULO 7 Pruebas

Una vez terminado con el desarrollo del sistema pasamos a la siguiente etapa, la de pruebas. En ésta se crearon una serie de casos para verificar o consultar cual es el resultado tangible del valor propio de la herramienta. Lo primero que se definió es un plan de pruebas que abarca varios ámbitos o factores en los que se quiere indagar para verificar ciertos aspectos o características del sistema en cuestión.

7.1 Plan de Pruebas.

Un plan de pruebas son una serie de mecanismos artificiales controlados o no que siguen una cierta metodología para probar o ver el comportamiento del objeto que se quiere probar. El plan que se desarrolló fue a partir de los siguientes aspectos de la herramienta:

1. Instalación/Ejecución.
2. Requerimientos.
3. Usabilidad/ Efectividad

Los aspectos seleccionados a evaluar se consideran los más importantes ya que nos permiten conocer las áreas de mejora o críticas que se deben a tomar en cuenta en el desarrollo de este tipo de herramientas.

El objetivo principal de las pruebas es encontrar errores en la herramienta que se desarrolló. También se pueden llevar a cabo una serie de casos de prueba para saber si el sistema cumple

cuáles son las áreas de oportunidad más importantes que se pueden mejorar para agregarle valor a la herramienta y al campo de investigación en el área de informática forense.

7.2 Requisitos para la aplicación de las pruebas.

Para llevar a cabo un plan de pruebas se debe de conocer el ambiente o entorno donde se va a ejecutar el sistema así como identificar los objetos necesarios para completar este proceso, tales como la determinación del tipo de usuarios, logística, control, lugar y requerimientos de hardware. La lista de requisitos para la ejecución de las pruebas del sistema Hefoclase fueron los siguientes:

1. Computadora para ejecutar el programa y/o entrenar al usuario.
2. Personal para documentar y llevar a cabo el caso de prueba.
3. Papel y pluma para escribir los comentarios y observaciones en la ejecución del proceso.
4. Participantes dispuestos a cooperar en la prueba.

En términos generales estos fueron los requisitos primordiales en la evaluación del sistema, otros aspectos de logística como transporte para dirigirse a un lugar no fueron escritos. Sin embargo es importante mencionarlos.

7.3 Casos de pruebas

Como se acotó al principio de este capítulo los aspectos que se quieren evaluar son: instalación/ejecución, requerimientos/efectividad y usabilidad/rendimiento. El primero de ellos tiene como objeto conocer la facilidad en la ejecución e instalación del programa, ver cuáles son

las trabas encontradas y como resolverlas. Los casos de pruebas definidos para este propósito son los siguientes:

1. Se proporcionará al usuario el ejecutable del sistema dentro de una unidad flash para ver si puede hacerlo funcionar en una computadora de escritorio con sistema operativo Windows, se le explicó "Este dispositivo USB que se te entrega contiene un archivo ejecutable o programa de cómputo, ¿podrías ver si puedes ejecutarlo en tu máquina?, y ¿me podrías decir que problemas tuviste al tratar de correrlo?"
2. Se proporcionará al usuario el ejecutable del sistema en un CD para ver si puede hacerlo funcionar en una laptop, se le dijo "Este CD que te entrego contiene un archivo ejecutable o programa de cómputo, ¿podrías ver si puedes ejecutarlo en tu máquina?, y ¿me podrías decir que problemas tuviste al tratar de correrlo?"

Para llevar a cabo este caso de prueba se invitó a diversas personas de diferentes edades y géneros, así como de diferentes niveles de conocimiento en computación (que por lo menos supiera manejar lo básico de Windows). Se les preguntó si querían participar en una prueba de un programa de cómputo a los que la mayoría accedieron. En total fueron 10 personas con las que se ejecutó este caso de prueba. Además se tomó en cuenta que en el proceso de esta prueba hubiera diferentes versiones del sistema operativo Windows, así como la certeza de que se llevara a cabo en diferente máquina y no en una misma computadora.

El siguiente aspecto a evaluar de la herramienta fue la de si cumplía con los requerimientos definidos en la etapa de análisis. Para llevar a cabo este caso de prueba se instaló/ejecutó el programa y se hicieron las pruebas de verificación correspondientes a cada requerimiento funcional definido en la etapa de análisis y diseño. La persona que llevó a cabo estas pruebas fue

un usuario interesado en el sistema que comprende la fase de pruebas y que trabajó en el área de pruebas para una compañía. Al final de este proceso él entrego un reporte acerca de los requerimientos y si la herramienta cumplía o no con esa especificación de requerimientos.

Sin duda otro aspecto importante para evaluar la herramienta es la usabilidad. Los bastiones básicos que prueba son: la utilidad, eficiencia, efectividad, satisfacción y accesibilidad [Rubin et al, 2008]. Todos estos elementos nos permiten ver si el software es bueno para lo que se diseñó. Para ello se idearon una serie de pasos para tratar de evaluar estos puntos.

El primer punto que se evaluó en esta parte, es si la herramienta es fácil de manejar o fácil de recordar cómo se usa. Para este caso se creó el siguiente proceso:

Se pregunta a los candidatos si quieren participar en una evaluación de software, si acceden, se les instala el sistema en su máquina o bien se procede si ya están *in situ* en el lugar donde se lleva a cabo este proceso. A estos usuarios que aceptaron se les da un entrenamiento de cómo usar la herramienta para un fin específico. Estas tareas específicas fueron las siguientes:

1. Obtener la lista de archivos creados de una fecha específica (del día 20 de noviembre de las 12:00 am a las 6:00 pm).
2. Obtener la ruta de una llave oculta en el registro de Windows.
3. Verificar la firma de un archivo específico.
4. Buscar en el registro una determinada clave/llave.
5. Sacar el historial de páginas de una fecha determinada (fijándose en los archivos de la caché).
6. Agregar a los resultados cada dato encontrado.
7. Tener un sumario de resultados de por lo menos una aparición de evidencia de cada módulo.

8. Clasificar un dato encontrado de evidencia.
9. Exportar un reporte generado por la herramienta.

A los participantes como se mencionó anteriormente, se les enseñó cómo hacer cada proceso y después se les dejó que actuaran de forma normal para ver si podían repetir lo que se les había explicado.

Otro de los aspectos que entran en la usabilidad, es medir la eficiencia de la herramienta. Para ello se hizo lo siguiente:

1. Probar lo módulos y documentar que tantos recursos de hardware, llamadas de entrada/salida y cargado de librerías hacían. Medir cual es su "working set" y los bytes privados que cargaba, así como también ver el porcentaje de uso del procesador. Para esto se ejecutó el programa junto con otras herramientas de depuración de procesos.

Después de medir la eficiencia y rendimiento se pasó a medir la efectividad, sin duda el aspecto más importante a evaluar, ya que tiene que ver con que el sistema haga lo que dice poder hacer, que es encontrar evidencia digital. El proceso que se siguió para ello es el siguiente:

1. Ir con las herramientas preparadas para instalación a lugares públicos conocidos como *"ciber o café internet"*. Entrevistar al encargado y decirle que se está probando una herramienta para un trabajo de tesis de licenciatura y comentarle si deseaba participar. Las condiciones eran de que él iba a indicar cuáles son las máquinas que andan fallando y que posiblemente tienen malware, para que su servidor pudiera llegar con la causa con el uso de la herramienta. Además de arreglar el funcionamiento de la computadora. De modo que uno se compromete a arreglar las máquinas que fallen y el participante señala cuales son las computadoras que probablemente tienen malware.

2. Otro escenario definido fue simular un ataque a una computadora. Para ello se le pidió a un usuario que conoce técnicas de pruebas de penetración en sistemas Windows y se le requirió para que comprometiera la seguridad del sistema.

Los resultados de estos casos de pruebas se reportan a continuación.

7.4 Resultados y Evaluación

Lo primero que se evaluó en el sistema es la facilidad de instalación, para ese caso de prueba participaron 10 personas, cuyas edades oscilaron entre los 24 y 35 años, con edad promedio de 26.4 años. Tres mujeres y siete hombres con conocimientos promedio en el uso del sistema operativo Windows. La cantidad de los usuarios que pudieron ejecutar el programa correctamente y de los que no se muestra en la figura 7.1.

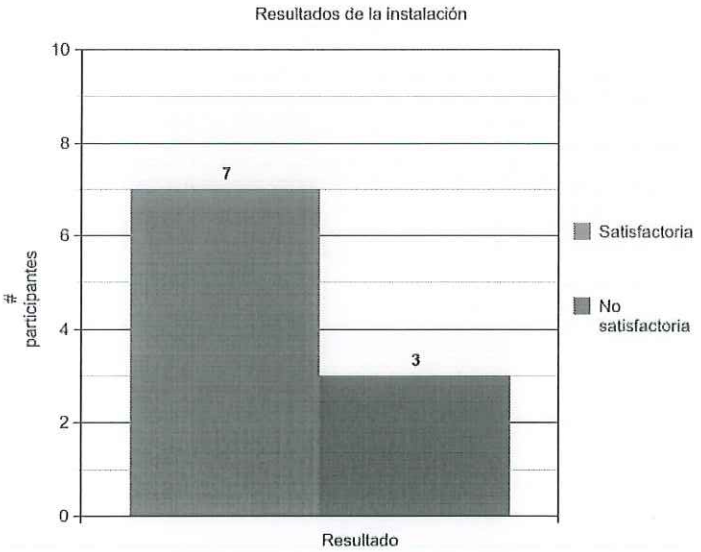

Figura 7.1 Resultados de la instalación en el caso de prueba.

La mayoría de participantes pudo instalar el programa, los que no pudieron fue porque no tenían instalado las dependencias que se ocupaban para la ejecución del programa, como por ejemplo algunas librerías o el entorno .NET 2.0. A continuación se muestran los errores que se marcaron cuando no se pudo instalar.

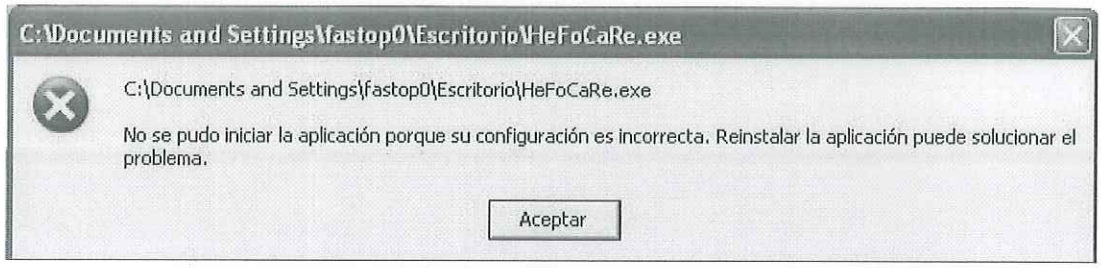

Figura 7.2 Error en Windows XP en la ejecución de la herramienta.

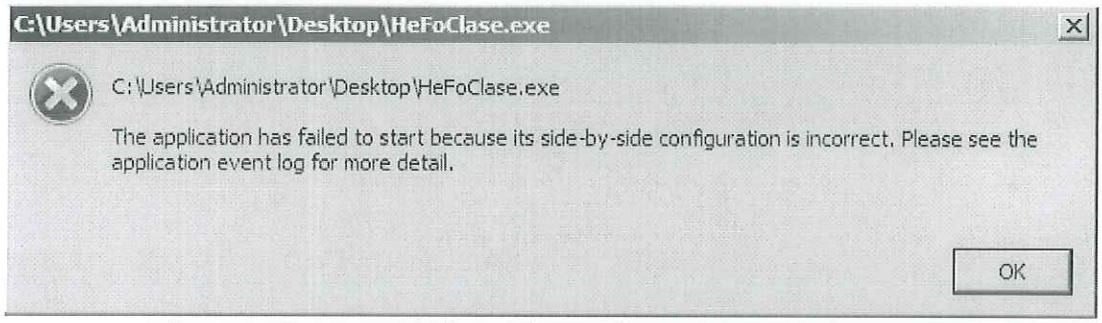

Figura 7.3 Error en Windows Vista en la ejecución de la herramienta.

Hay que decir que el sistema se ejecutó correctamente en Sistemas Windows y Server 2003, los errores que aparecieron fueron corregidos con la creación de un ejecutable portable. Al final de este capítulo se muestra el sumario de corrección de defectos y estado actual del sistema.

El segundo aspecto que se tomó en cuenta para evaluar el sistema fue una simple validación de los requerimientos definidos en el capítulo de Análisis, la tabla 8 muestra la relación de la lista de requerimientos que cumplieron o no según el evaluador que se invito.

Identificador del Requerimiento	Estado final	Anexos
RFT-1	Cumplido/pasó	
RFT-2	Cumplido/pasó	
RFT-3	Cumplido/pasó	
RFT-4	Cumplido/pasó	
RFT-5	Cumplido/pasó	
RFT-6	Cumplido/pasó	
RFT-7	Cumplido/pasó	
RFT-8	Cumplido/pasó	
RFT-9	Cumplido/pasó	
RFT-10	Cumplido/pasó	
RFT-11	Cumplido/pasó	
RFT-12	Cumplido/pasó	
RFT-13	Cumplido/pasó	
RNF-1	Cumplido/pasó	Solo se probó con un dispositivo de almacenamiento no volátil (CD)

Tabla 8. Relación de pruebas de requerimiento del sistema.

El primer aspecto que se evaluó en la parte de usabilidad fue el de facilidad de uso. Los resultados arrojados de los casos de prueba especificados, fueron los que se muestran en la figura 7.4.

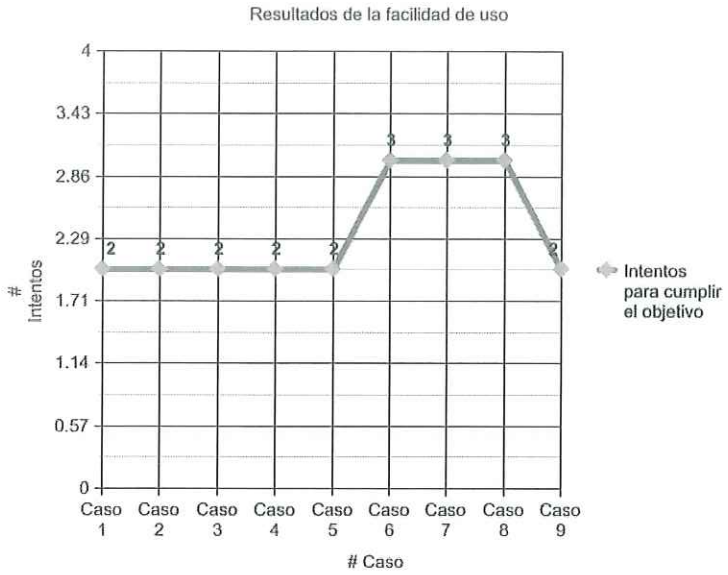

Figura 7.4 Resultados del número de intentos para cumplir con el caso de prueba especificado para determinar la facilidad de uso de la herramienta.

Los primeros 5 casos fueron relativamente sencillo de replicar por parte de los usuarios invitados, los siguientes 3 se observa una ligera dificultad para cumplir con el comportamiento explicado con anterioridad. En resumen hubo 6 casos que fueron al segundo intento y 3 que fueron al tercero. Esto nos da una idea de que hay unos aspectos en la interfaz y funcionalidad que son más difíciles de utilizar que otros. Sin embargo todos los casos se pudieron completar. También hay que decir que el tiempo promedio para realizar cada caso fue de 15 minutos, lo cual no es mucho tiempo comparado con una capacitación acerca del uso de un programa en el mundo real.

Otro de los puntos que se incluyen en la usabilidad y que se trató de probar es la eficiencia de la herramienta. Todo esto con el fin de identificar los módulos más tardados y de conflictos de rendimiento. El módulo que se observó que era más tardado fue el de Historial de Internet, esto es así ya que su programación interna provoca un desfase entre los elementos gráficos y un

proceso de entrada/salida como lo es leer el archivo index.dat de Internet Explorer. Para ver como afectaba el estar leyendo esto se trató de buscar un archivo index.dat que pesara más de 6 MB. Lo que nos interesa es ver cuanta es la cantidad de memoria que se reserva para que el programa pueda ejecutar este proceso. Para ello nos fijamos en el *"working set"* y demás datos relevantes que se muestran en la figura 7.5.

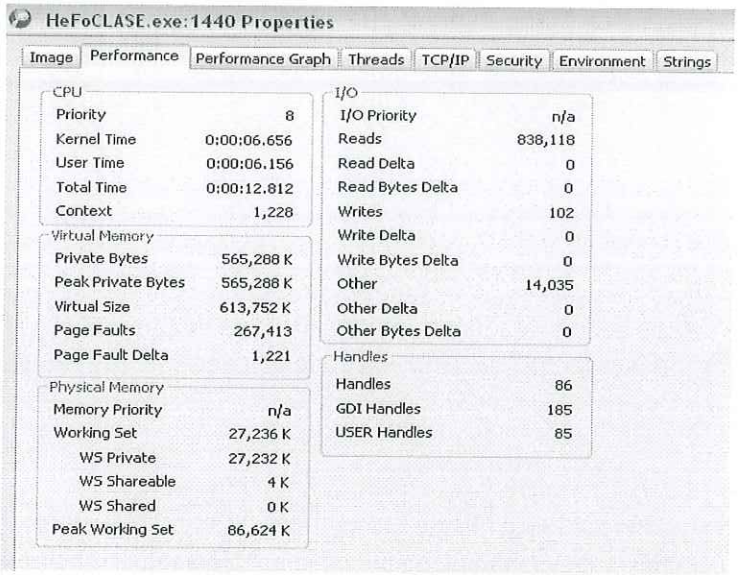

Figura 7.5 Datos acerca del rendimiento de la herramienta mientras ejecuta el módulo de Historial de Internet.

Figura 7.6 Gráfica de rendimiento de la herramienta mientras ejecuta el módulo Historial de Internet.

Como podemos observar los bytes privados que utiliza son de más de 560 MB, es decir, tiene reservado ya esa cantidad de memoria por si la va ocupar. Además el máximo de working set que dio es de 86 MB. Este espacio es la cantidad de páginas en el espacio virtual de direcciones que ya han sido referenciados [MSDN, 2008].

También es importante mencionar las características de hardware de la máquina en que se realizó esta prueba. En la figura 7.7 se muestran estas características.

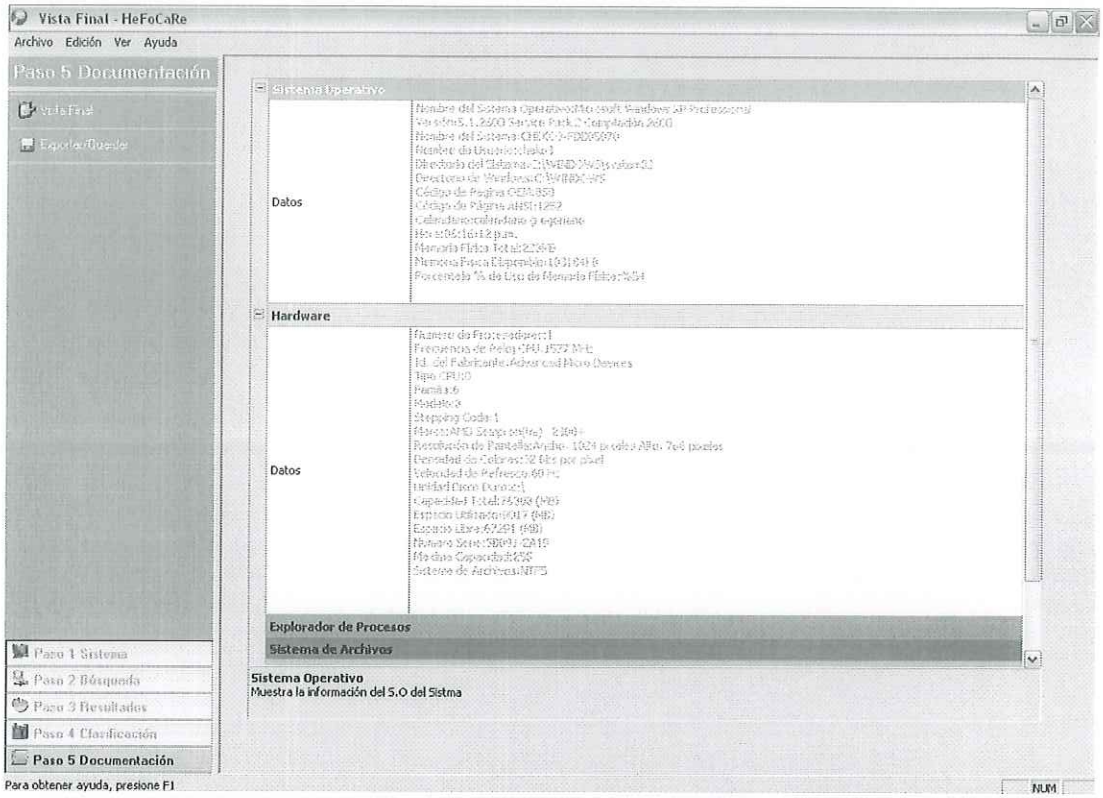

Figura 7.7 Información de hardware de la máquina donde se ejecutó la prueba de Rendimiento.

Como podemos observar la máquina tiene solamente 256 de RAM y el procesador es un Sempron 2300+ de 1.5 GHz, podríamos decir que por la poca cantidad de RAM el sistema operativo tenía que utilizar el espacio de direcciones virtual. Esta operación no consume proceso pero consume peticiones de entrada/salida entre la vista y el control. Esto es debido a que se está añadiendo a la vista el resultado de las lecturas entre *offsets* en tiempo real del archivo index.dat, y además de que el archivo es de más de 6MB. Una solución recomendable para esto es crear otro mecanismo, por ejemplo, primero *parsear* todo el archivo y luego leerlo ya con todos los campos definidos que nos interesan. Así evitaríamos una sobrecarga de reservación de memoria.

Otro aspecto fundamental que se probó de la herramienta es ver si ayudaba a encontrar evidencia digital acerca de un posible incidente de seguridad, es decir su efectividad. Como se describió en el caso de pruebas, para le ejecución de esta prueba se visitó a cafés internet y se platicó con el encargado para ver si quería participar. Si aceptaba entonces se continuaba con la instalación de la herramienta en la máquina en cuestión y se iniciaba el proceso para buscar pistas acerca de que pudiera estar pasando en determinada computadora. La cantidad de personas que accedió fueron 5, arrojando los siguientes resultados.

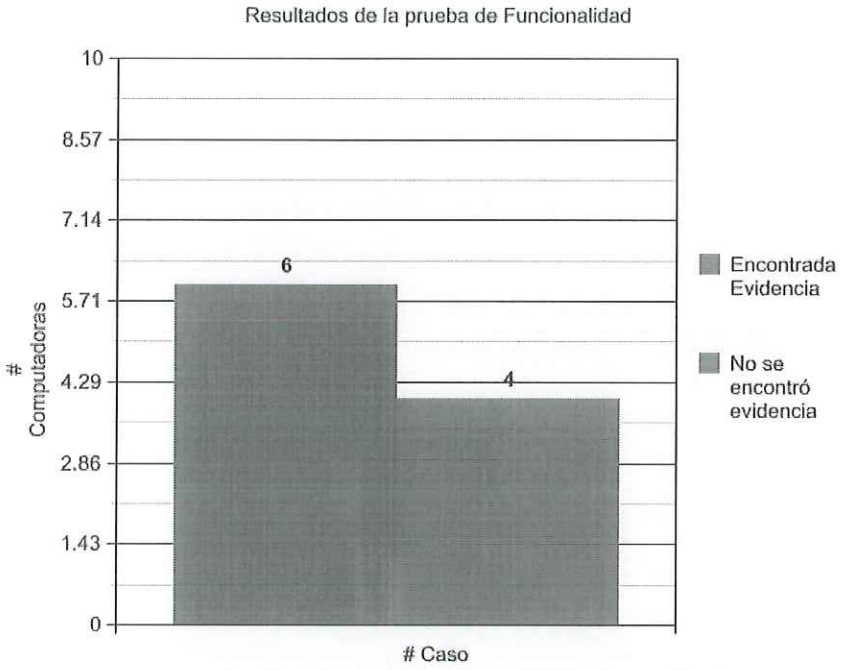

Figura 7.8 Evidencia encontrada en las pruebas que se realizaron dentro de los café internet.

En 6 de 10 de casos, es decir de 10 computadoras que se examinaron, se encontró rastros de evidencia que apuntaba hacia algún malware ya sea un troyano, un *exploit* o pornografía. Las 5 personas que accedieron por lo general decían "pues esa máquina anda fallando, no tiene internet. A ver si la puedes arreglar" otros decían "no la he formateado, está llena de virus". En

la figura 7.9 se muestra un exploit encontrado mientras se indagaba en la caché de internet de una máquina en la que se ejecutó la herramienta.

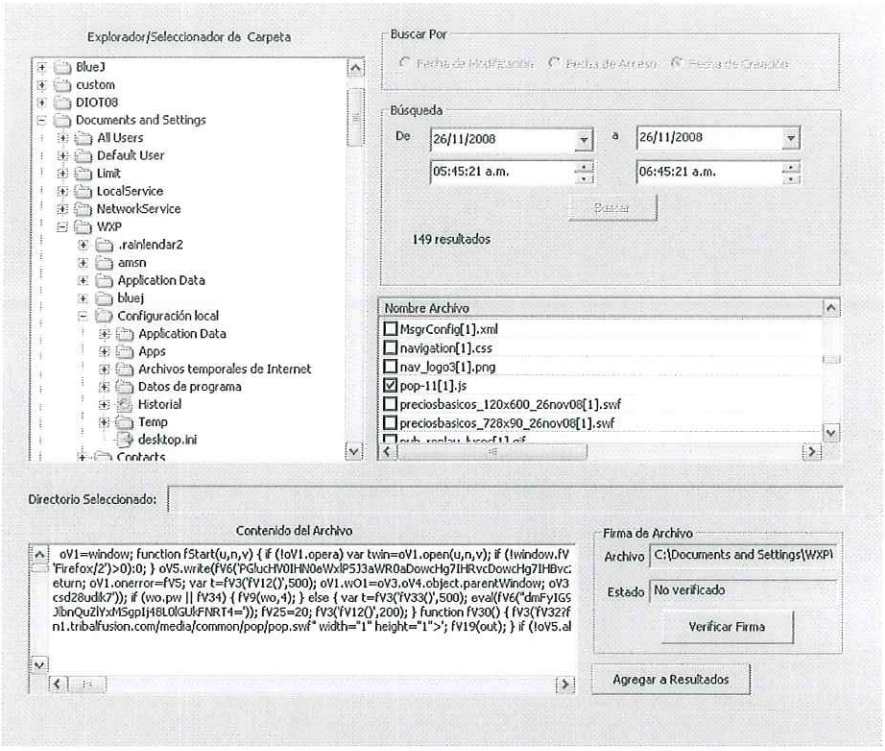

Figura 7.9 Exploit encontrado por Hefoclase en una máquina de prueba.

En la parte del contenido del archivo podemos observar que es código fuente de un javascript. Analizando el código fuente e investigando un poco más, pude llegar con que era un *exploit* que hacía que tu navegador dejara de funcionar, además de que dejaba vulnerable a la máquina en la ejecución de otros malware [Microsoft, 2008].

Otra de las cosas que se pudieron ver en la exanimación de las computadoras, son algunos troyanos corriendo, como por ejemplo, el que se ve en la siguiente figura 7.10. Esa imagen es una vista de la documentación generada por Hefoclase, junto con los metadatos que se agregaron a ese elemento de evidencia.

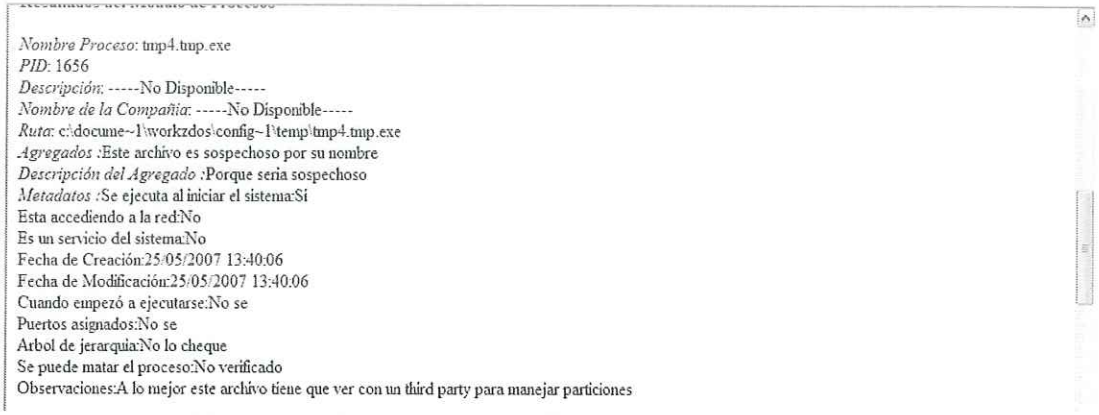

Figura 7.10 Troyano encontrado en una máquina de prueba.

Una de las pruebas más importantes para verificar o ver como se comportaba la herramienta fue la de tener la certeza de un ataque sobre una computadora ya sea mediante una simulación de un ataque o un ataque real. En este caso de prueba, se empezó a buscar evidencia en el sistema sin saber dónde buscar o por donde había ocurrido la violación de la seguridad en el sistema. El perpetrador fue una persona con conocimientos en pruebas de penetración en sistemas Windows, se le dio facilidades de acceso al sistema prueba, con el fin de vulnerar la seguridad como si fuera un ataque normal.

Al realizar la exanimación del sistema, lo primero relevante que se encontró fue ver un proceso sospechoso en el que parecía ser un protector de pantalla del sistema, ya que tenía la extensión .scr. Esta extensión es común en los sistemas operativos Windows. Una investigación más a fondo de este proceso, llevó a la conclusión de que era una puerta trasera remota que se había instalado, esto se corroboró con la ayuda de otra herramienta que permitía ver si el ejecutable enviaba información o mantenía alguna conexión con un servidor. En la figura 7.11 se muestra el binario en cuestión detectado por Hefoclase.

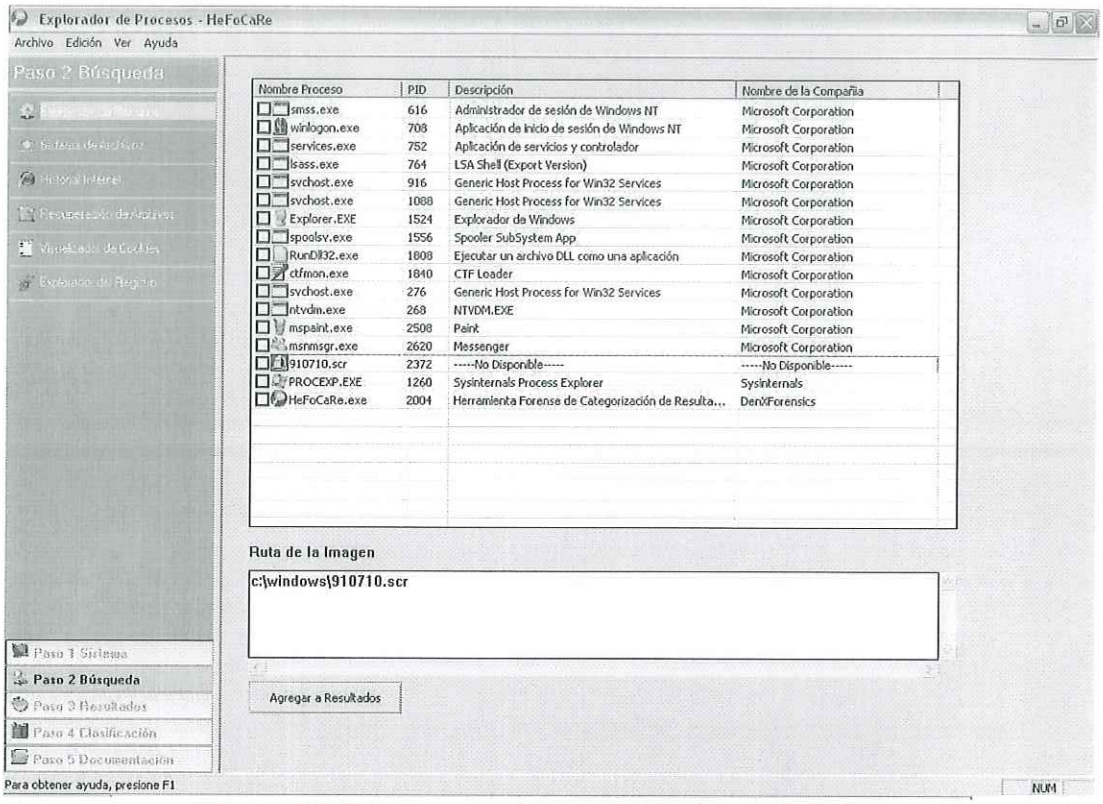

Figura 7.11 Proceso sospechoso encontrado por Hefoclase.

Como se puede apreciar el proceso se encuentra en la ruta c:\windows y el nombre es "910710.scr", pareciendo que es un archivo legítimo del sistema operativo. Comúnmente en el sistema operativo Windows los protectores de pantalla se guardan en la ruta c:\windows\system32 con extensión .scr. Estos archivos se manejan como si fueran ejecutables dentro del sistema, lo que hace propenso a poderse utilizar con otros propósitos. También se encontró una serie de archivos del mismo tamaño dentro de la ruta C:\Documents and Settings\WXP\Configuración local\Datos de programa\Ares\My Shared Folder, en esta carpeta es donde se guardan los archivos descargados por un programa p2p (peer to peer) llamado Ares. Los archivos pesaban 396 KB cada uno y tenían nombres populares para términos de búsqueda en la red p2p, nombres como *vistacrack, smsgratis, sexogratis, videosxxx.avi,*

subseven2008 entre otros. Esto era un indicio claro de que probablemente este malware se propagó a través de las redes peer to peer, sin embargo al final se llego a otra conclusión.

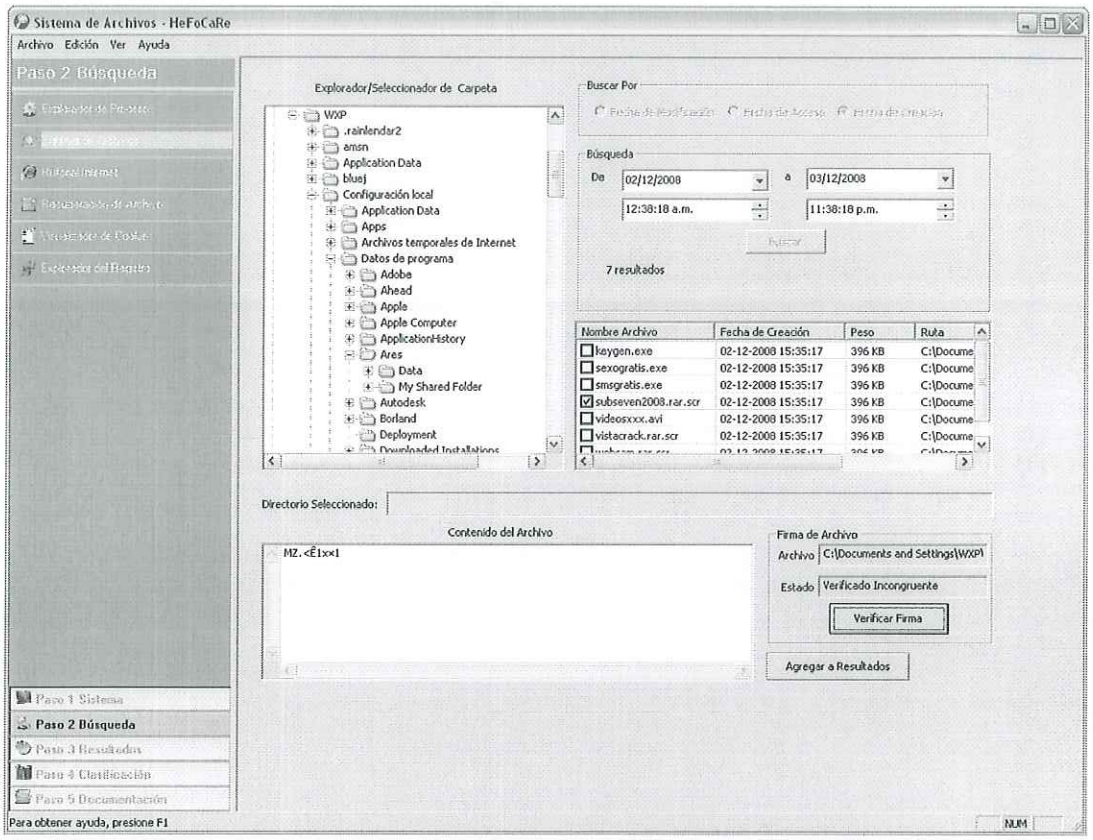

Figura 7.12 Archivos sospechosos con nombres de búsquedas populares encontrados por Hefoclase en el módulo Sistema de Archivos.

Al analizar el ejecutable "910710.scr" se observó que este mantenía una conexión de red hacia un servidor remoto, en este caso apuntaba hacia la dirección *lb2.celeonet.com* por el puerto 2560. En la figura 7.13 se muestra la conexión que tiene establecido el malware con el servidor remoto, un comportamiento común al hablar de un malware. La investigación continuó y se revisó el Historial de Internet de la computadora, se encontraron algunos datos relevantes como por ejemplo que había una entrada donde se había ejecutado un binario mediante el navegador. En la figura 7.13 se muestra el nombre del archivo ejecutado, "navidad.exe", los datos acerca de

cuándo se accedió a esa página dieron como resultado la fecha de acceso 02/12/2008 18:56, esto significa que el 2 de diciembre se ejecutó un archivo .exe mediante el navegador.

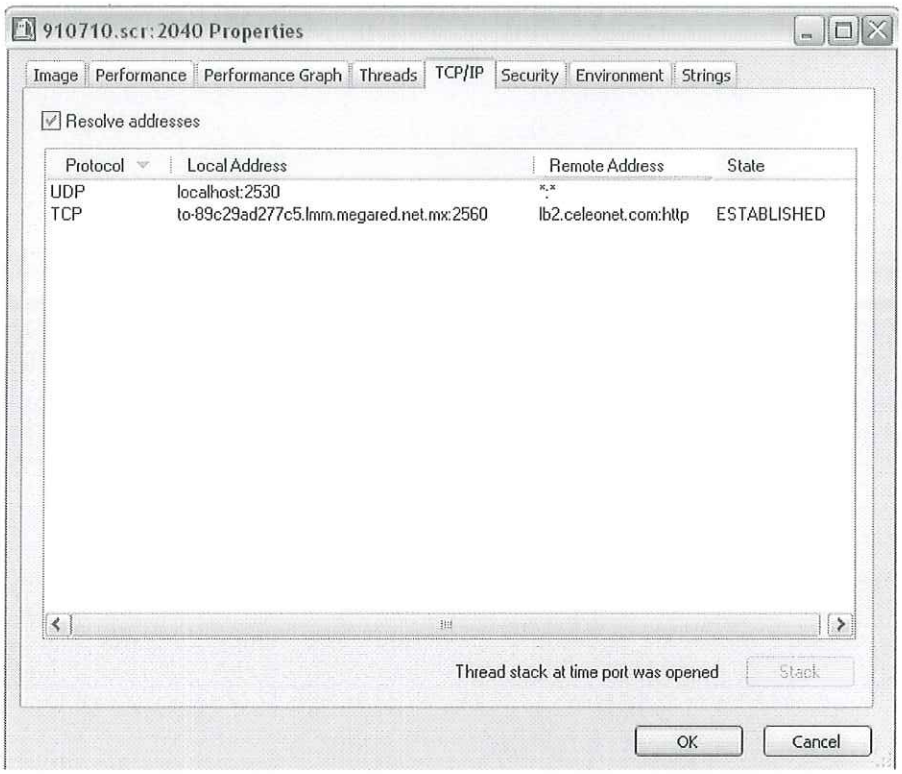

Figura 7.13 Ejecutable sospechoso que se conecta a un servidor remoto.

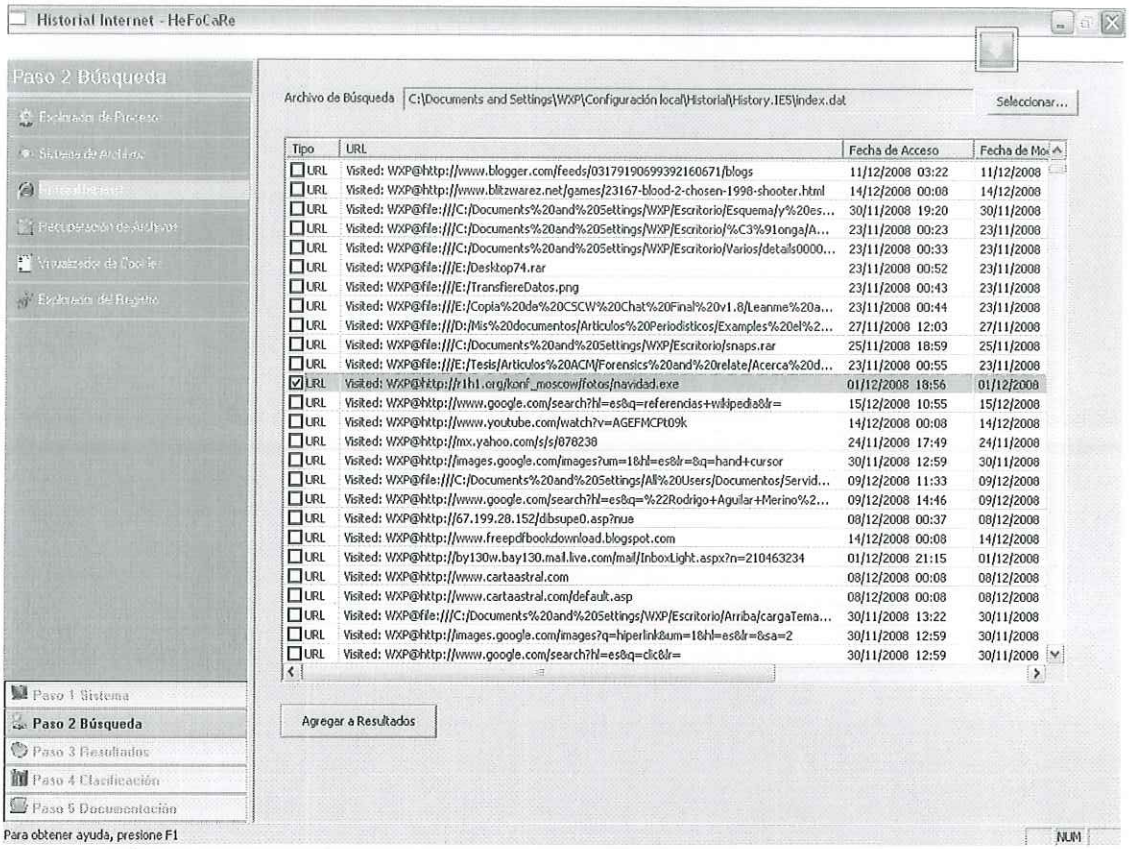

Figura 7.14 Entrada de un malware en el Historial de Internet de Hefoclase.

La URL a la que apunta el malware "navidad.exe" es una página legítima (http://rlhl.org/), sin embargo en ella se hospedó el archivo malicioso. Continuando el análisis se pudo verificar que este troyano se ejecutaba al inicio del sistema, esto se pudo ver claramente mediante el explorador de registro. En la figura 7.15 se muestra la llave encontrada acerca del malware.

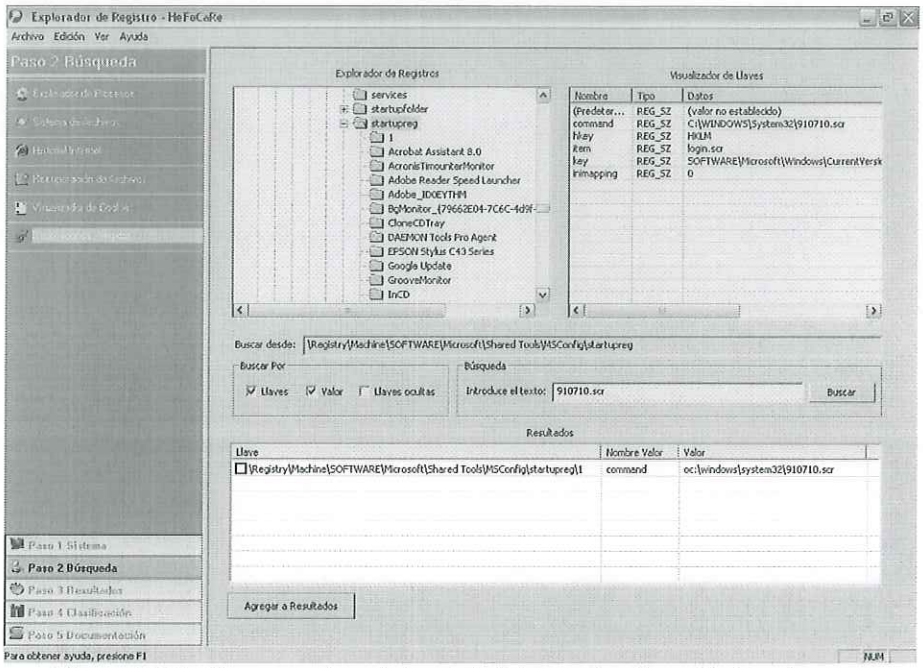

Figura 7.15. Clave en el registro del sistema donde está el malware.

Las conclusiones de esta investigación o análisis forense de la máquina de prueba, fue que el perpetrador invitado (para hacer la simulación de ataque) puso en un servidor remoto el malware y lo ejecutó mediante el navegador. Los archivos encontrados en el directorio del programa p2p Ares fueron infectados y no era propiamente el virus sino que era efecto del malware ejecutado en la máquina. Para este caso si se pudo detectar un ataque, sin embargo el perpetrador invitado señaló que pudo haber borrado esos datos y que además violo la seguridad en otras formas que tenían que ver con los documentos de Microsoft Office, en este caso no se llegó a eso ya que se enfocó en primera instancia en lo más obvio que era el proceso sospechoso "910710.scr". Así como este caso puede haber muchos mas donde se puede encontrar algo pero no necesariamente todo, ya que conforme existe más interés en ganancias financieras mediante el uso del código malicioso, los perpetradores harán más por poder ser indetectables.

Para resumir, hay que mencionar que en los casos que se encontró evidencia se puede decir que los módulos más importantes para este fin fueron: Historial/Caché de internet, sistema de Archivos, módulo de procesos y Explorador de Registro. Una observación importante es que la herramienta puede encontrar evidencia, sin embargo no puede guardarla, es decir, no puede realizar una imagen completa del disco duro o de los sectores importantes donde se encuentra esa evidencia. Aquí se podrían ver las 2 caras de la moneda, ya que es bueno llegar a la causa de un problema, más sin embargo para efectos legales es importante tener una copia bit a bit de eso que se menciona como evidencia digital. En los 4 casos que no se encontró evidencia con la herramienta fue porque probablemente estaba limpia la computadora o realmente no se pudo con la funcionalidad ofrecido por el sistema Hefoclase, sin embargo esto es difícil comprobar debido a que entran otras variables como: experiencia del investigador, tipo de malware entre otros aspectos. Otros de los puntos débiles de la herramienta es que se repite información en algunas partes, por ejemplo en la parte de agregados. Esta sección se hizo con el fin de recabar metadatos relacionados hacia ese elemento de evidencia, sin embargo se observa que este procedimiento se podría omitir y dejar que estos metadatos se agreguen en el módulo de clasificación.

Los reportes generados por Hefoclase son en formato HTML, la página que se crea es muy sencilla. Aquí se puede dar un formato mucho más vistoso para mejorar la forma en que se visualizan los resultados, ya que esto mejoraría el análisis de un investigador.

La tabla 9 que se muestra a continuación hace un sumario de los errores, *bugs* y demás malfuncionamientos que fueron detectados y/o corregidos.

Error	Tipo	Estado final	Anexos
Cuando se hace una búsqueda, si la fecha de inicio es posterior a la fecha final. Por ejemplo INICIO: 30/06/2007 FINAL: 04/06/2007 el motor de búsqueda empieza a buscar.	Moderado	Corregido	No se había validado el rango. Corregido con una validación a partir de la fecha de inicio.
En el módulo Historial de Internet y Recuperador de Archivos, cuando se ejecutaba el proceso predeterminado de búsqueda de archivo, no encontraba el archivo, ya que la ruta no existía.	Moderado	Corregido	Se corrigió con un verificador del estado del archivo antes de empezar a manipularlo
En el módulo de sistema de archivos, cuando se realizaba una búsqueda por MAC times, el sistema no podía leer archivos de más de 500 MB y la búsqueda paraba.	Moderado	Corregido	Se agrego una validación con el peso del archivo que se trataba de leer.
En el módulo de sistema de archivos, cuando se realizaba una búsqueda y se trataba de cambiar a otro módulo, el sistema arrojaba una excepción de violación de escritura de código y se cerraba el programa.	Crítico	Corregido	Se corrigió con una validación dependiendo del módulo que se estaba ejecutando.
En el módulo de Historial de Internet existía una condición de número de operaciones de entradas/salida excesiva entre el elemento gráfico y los datos.	Crítico	Mitigado	Se minimizó el impacto haciendo primero el proceso de lectura de la estructura del archivo y luego agregarlo en vista gráfica y no de forma simultánea
En el módulo Sistema de Archivos, el sistema se colapsaba cuando trataba de leer algunos archivos protegidos o con permisos de escritura/lectura especiales.	Crítico	Corregido	
En el módulo de Explorador de Registro, cuando se trataba de abrir de la lista de resultados algún elemento, el sistema no podía abrir determinada llave.	Crítico	Corregido	Error en el procesamiento de las llaves

Tabla 9. Registro de defectos/error y el estado final.

CAPÍTULO 8 Conclusiones y Recomendaciones

8.1 Conclusiones.

El trabajo realizado y descrito en este documento, tiene que ver con la reconstrucción de un evento. Un evento que posiblemente altera o haya violado las medidas de seguridad en un sistema de cómputo. A lo largo de este trabajo se describió como cambia la forma de hacer delitos conforme evoluciona la tecnología. Éstos no son delitos nuevos sino son los tradicionales, solo que ahora están en un diferente ambiente y se realizan con nuevas herramientas. La informática forense es un área que se está tomando muy en serio por los expertos en seguridad en cómputo, dependencias de gobierno y empresas privadas. La información en este mundo virtual de interacción nos hace tener un riesgo implícito. Ése riesgo es mayor cuando se mueve dinero por este medio, de ahí que existe la necesidad del desarrollo de nuevas tecnologías de software o ambientes integrados de trabajo que permitan a un investigador forense conocer la forma en que se dio un ataque.

En el ámbito educativo es importante reconocer que no abundan los planes de estudio que abarquen este tipo de temas relacionados con la seguridad. Si se revisa los planes de muchas universidades en el país, encontraremos que son contadas las que contemplan este tipo de temáticas. También es relevante llevar a la práctica este tipo de procesos como la informática forense, ya que esto permitirá adquirir conocimientos y experiencia para comenzar a sumar el número de expertos nacionales en el área.

La herramienta que se creó es un buen inicio para poder entender cómo se hace este tipo de software. Además este producto generado tiene cierto enfoque diferente del acostumbrado, tiene

unas variaciones respecto a otras herramientas. Se tomaron las características buenas de algunas herramientas comerciales y se agregaron a la funcionalidad del sistema. Estas variaciones son las siguientes:

1. Enfoque de metadatos. El enfoque de metadatos en una herramienta de informática forense en vivo, es bueno ya que nos permite conocer características de lo que se está encontrando como posible evidencia digital.
2. Cada elemento de evidencia se maneja como un elemento único y no como un todo.
3. Cada elemento de evidencia se puede clasificar de una forma sistemática.

8.2 Recomendaciones.

Algunos problemas o trabas que se tuvieron al crear la herramienta fue por ejemplo la falta de código libre o público que permita a los desarrolladores nuevos indagar un poco más fácil en primera instancia. Algunas decisiones importantes que se deberían tomar antes de empezar con el desarrollo son:

1. Definir cuáles son las características más importantes que los usuarios expertos en este tipo de herramientas recomiendan como básicas y necesarias. Esto permitiría una mejor evaluación del sistema ya que se aprovecha la experiencia en el campo de este tipo de usuarios.
2. Identificar el tipo de sistema que se va a crear, por ejemplo definir si va a trabajar sobre la red, de forma clásica o bien en vivo, como fue nuestro caso. Definiendo en principio sobre que mecanismo principal va a realizar y en qué contexto, entorno y plataforma se

va ejecutar el sistema, puede enfocar de forma más clara los requerimientos que se van a desarrollar.

8.3 Trabajo a futuro.

Sin duda el desarrollo de nueva funcionalidad respecto a temas como eficiencia, eficacia y robustez serían importantes redefinirlos, ya que esto permitiría entrar de lleno en el mercado con las soluciones integrales que existen en la actualidad. Como se describió en el trabajo la mayoría de herramientas de código libre no son una herramienta sino son muchas pequeñas herramientas. Las soluciones integrales comerciales que han probado trabajar bien en casos aceptados por la ley son pocas. Este punto sin duda se debería potenciar con el uso de técnicas que adquieran, guarden o mantengan la evidencia digital de forma íntegra. Un paso fundamental para la admisión y aceptación de evidencia en las cortes de la ley. La herramienta que se desarrolló se vería muy beneficiada si se agrega todos estos aspectos y funcionalidad.

Literatura Citada y Referencias

[IWS, 2007] Internet World Stats. (2007) extraído el 26 de Noviembre de 2007 desde http://www.internetworldstats.com/stats10.htm

[FBI- IC3R, 2007] Internet Crime Compliance Center (2007). *Internet Crime Report 2007.* Bureau of Justice Assistance. Federal Bureau of Investigation.

[AMIPCI, 2008] Asociación Mexicana de Internet (2008). *Estudio de Comercio Electrónico 2008.* Vicepresidencia de Investigación de Mercados, Pedro Menéndez Dirección General Elogia.

[Kshetri Nir, 2007] Kshetri, N. *The simple economics of cybercrime.* Security & Privacy, IEEE (2007). Publication Date: Jan.-Feb. 2006. Volume 4, Issue 1. Pág. 33-39.

[Anthony Reyes et al, 2007] *Cyber Crime Investigations: Bridging the gaps Between Security Professionals, Law Enforcement, and Prosecutors* (2007). Anthony Reyes, Kevin O'Shea, Jim Steele, Jon R. Hansen, Captain Benjamin R. Jean. Thomas Ralph.

[Quinion, 2002] Quinion Michael. *Cyberplague. Help! A prefix out of control*, World Wide Words, extraído el 20 de Junio del 2007.

[Carvey, 2004] Harlan Carvey (2004). *Windows Forensics and Incident Recovery*. Addison Wesley 2004. ISBN 0-321-20098-5.

[Skoudis, 2003] Ed Skoudis, Lenny Zeltser (2003). *Malware: Fighting Malicious Code.* Prentice Hall PTR, 2003. ISBN 0-13-101405-6.

[Mohay et al, 2003] George Mohay, Alison Anderson, Byron Collie, Olivier De Vel, Rod McKemmish (2003). *Computer and Intrusion Forensics.* Artech House 2003. ISBN 1-58053-369-8.

[SWGD, 1999] Scientific Working Group on Digital Evidence (SWGDE) (1999). *Proposed Standards for the Exchange of Digital Evidence.*

[Schultz et al, 2001] Eugene Schultz, Rusell Shumway, Terry Gudaitis (2001). *Incident Response: A Strategic Guide to Handling System and Network Security Breaches.* New Riders Publishing First Edition 2001.

[Wikipedia, 2008] Wikipedia. La enciclopedia libre. (2008). *Steganography*. Extraído el 2 de Junio de 2008 desde http://en.wikipedia.org/w/index.php?title=Steganography&oldid=265138216

[PC History, 2007] Stan. *The history of computer forensics* extraído el 15 de Julio del 2007 desde http://www.pc-history.org/forensics.htm

[Kabay, 2008] Kabay M. E. (2008). *A Brief History of Computer Crime: An Introduction for Students.*

[Adelstein, 2006] Frank Adelstein (2006). *Live forensics: Diagnosing your system without killing it first.* Communications of the ACM, Febraury 2006/Vol 49 No 2.

[US-CERT, 2007, 1] United States Computer Emergency Readiness Team. *Vulnerability Note VU#310057, Guidance EnCase fails to detect more than 25 partitions.* 2007.

[US-CERT, 2007, 2] United States Computer Emergency Readiness Team. *Vulnerability Note VU#912593, Guidance EnCase Enterprise uses weak authentication to indetify target machines.* 2007.

[Pressman, 2001] Roger S. Pressman (2001). *Ingeniería del Software: Un enfoque práctico.* Quinta Edición.

[Wikipedia, 2007] Wikipedia. La enciclopedia libre. (2008). *Waterfall model.* Extraído el 7 de Julio de 2008 desde http://en.wikipedia.org/w/index.php?title=Waterfall_model&oldid=263673379

[Royce, 1970] Winston. W. Royce (1970). *Managing the Development of Large Software Systems.*

[Galitz, 2007] The Essential Guide to User Interface Design 3rd Edition. An introduction to GUI Design Principles and Techniques. Wilbert O. Galitz. Wiley Publishing 2007.

[Jones, 2003] Keith Jones (2003). *Forensic Analysis of Microsoft Internet Explorer*

[Wirfs-Brock et al, 2002] Rebecca Wirfs-Brock, Alan McKean. (2002). *Object Design: Roles, Responsabilities and Collaborations.* Addison Wesley 2002. ISBN 0-201-37943-0.

[Schneiderman y Plaisant, 2004] Ben Shneiderman y Catherine Plaisant. (2004). *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (4th Edition). Pearson Addison-Wesley2004.

[Russinovich, 2004] Mark Russinovich, David A. Solomon. *Flujo de la creación de un proceso.* Capitulo 6. *Microsoft Windows Internals*, Fourth Edition: Microsoft Windows Server 2003, Windows XP and Windows 2000. Microsoft Press 2004. ISBN 0735619174.

[FILESIG, 2007] File Signatures, FileSig desde http://www.filesig.co.uk.

[McGrath y Casey, 2002] Michael G. McGrath, and Eoghan Casey (2002). *Forensic Psychiatry and the Internet: Practical Perspectives on Sexual Predators and Obsessional Harassers in Cyberspace.* The Journal of the American Academy of Psychiatry and the Law

[W3C, 2007] World Wide Web Schools. *Browser Statistics* extraído el 20 de junio del 2007 desde http://www.w3schools.com/browsers/browsers_stats.asp

[MSDN, 2008] Microsoft Developer Network. *Process Working Set* extraído el 22 de febrero 2008 desde http://msdn.microsoft.com/en-us/library/ms684891(VS.85).aspx

[MICROSOFT, 2008] Microsoft TechNet Security. *Microsoft Security Advisory (925568) Vulnerability in Vector Markup Language Could Allow Remote Code Execution* extraído el 22 de Febrero del 2008 desde http://www.microsoft.com/technet/security/advisory/925568.mspx

[Jones, 2003] Keith Jones (2003). *Forensic Analysis of Internet Explorer Activity Files.* 3/19/2003.

[Jones, 2003] Keith Jones (2003). *Forensic Analysis of Microsoft Windows Recycle Bin Records.* 4/1/03.

[Russinovich, 1997] Mark Russinovich (1997). *Inside the Windows NT Registry*. April 1997.

[Cogswell y Russinovich, 2006] Bryce Cogswell and Mark Russinovich (2006) *Rootkit Revealer v.1.71.* http://technet.microsoft.com/en-us/sysinternals/bb897445.aspx#hid

[Buchholz y Spafford, 2004] Florian Buchholz and Eugene Spafford (2004). *On the role of file system metadata in digital forensics.* CERIAS.

[Jeffrey Rubin et al, 2008] Jeffrey Rubin, Dana Chisnell y Jared Spool (2008). *Handbook of Usability Testing: Howto Plan, Design, and Conduct Effective Tests*. 2nd Edition. John Wiley & Sons.

Apéndice A Lista de Figuras

Orden de aparición.

Apéndice B Lista de Tablas