Instituto Politécnico Nacional

Centro de Investigación en Computación

Secretaría de Investigación y Posgrado

Sistema Forense de captura aleatoria de eventos de red

T E S I S

QUE PARA OBTENER EL GRADO DE

MAESTRA EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A

ING. MARÍA TERESA CANSECO RODRÍGUEZ

DIRECTORES DE TESIS: Dr. Rolando Quintero Téllez
M. en C. Sergio Sandoval Reyes

México, D.F. a 17 de Diciembre de 2012

SIP-14-Bis

INSTITUTO POLITÉCNICO NACIONAL
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

ACTA DE REVISIÓN DE TESIS

En la Ciudad de México, D.F. siendo las 9:30 horas del día 28 del mes de noviembre de 2012 se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Estudios de Posgrado e Investigación del:

Centro de Investigación en Computación

para examinar la tesis titulada:

"Sistema forense de captura aleatoria de eventos de red"

Presentada por el alumno:

CANSECO	**RODRÍGUEZ**	**MARÍA TERESA**
Apellido paterno	Apellido materno	Nombre(s)

Con registro:

B	1	0	1	6	3	7

aspirante de: **MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN**

Después de intercambiar opiniones los miembros de la Comisión manifestaron ***APROBAR LA TESIS***, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Directores de Tesis

Dr. Rolando Quintero Téllez

M. en C. Sergio Sandoval Reyes

Dra. Nareli Cruz Cortés

Dr. Rolando Menchaca Méndez

M. en C. Sandra Dinora Orantes Jiménez

PRESIDENTE DEL COLEGIO DE PROFESORES

INSTITUTO POLITECNICO NACIONAL
CENTRO DE INVESTIGACION EN COMPUTACION

Dr. Luis Alfonso Villa Vargas

INSTITUTO POLITÉCNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

CARTA CESIÓN DE DERECHOS

En la Ciudad de México el día 30 del mes de Noviembre del año 2010, el (la) que suscribe María Teresa Canseco Rodríguez alumno (a) del Programa de Maestría en ciencias de la computación con número de registro B101637, adscrito a Sistemas computacionales de control y embebidos, manifiesta que es autor (a) intelectual del presente trabajo de Tesis bajo la dirección de Dr. Rolando Quintero Téllez y M. en C. Sergio Sandoval Reyes y cede los derechos del trabajo titulado Sistema forense de captura aleatoria de eventos de red, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del autor y/o director del trabajo. Este puede ser obtenido escribiendo a la siguiente dirección mcansecor@gmial.com. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Nombre y firma

Resumen

En esta tesis se presenta un estudio documental sobre los principales ataques a las redes de computadoras, así como una descripción detallada de las especificaciones e implementaciones de los principales protocolos que soportan el Internet actual. Es importante recalcar que los ataques a los sistemas en red están basados en explotar las debilidades tanto de las especificaciones como de las implementaciones de los protocolos de comunicación por lo que es importante tener un conocimiento profundo de ambas.

Uno de los retos más importantes en el cómputo forense es recopilar la información mínima necesaria que sirva para analizar, modelar y entender la forma en cómo se llevan a cabo los ataques informáticos y para eventualmente llevar a los perpetradores ante la justicia. En esta tesis se propone el uso de Muestreo Montecarlo para recopilar información relacionada con los eventos de red. El muestreo Montecarlo, y en particular el muestreo de tipo aceptación/rechazo, tiene la ventaja de que las muestras obtenidas por este método preservan las distribuciones originales de la población completa (de todos los flujos de datos) sin la necesidad de conocer la distribución real de la población. Para implementar la propuesta se desarrolló un sistema de software, ADA**, que utiliza Libpcap para obtener paquetes de red. ADA** permite la definición de filtros que relacionan conjuntos de protocolos con una probabilidad de aceptación, es decir, con la probabilidad de que un paquete que contiene encabezados de ese conjunto de protocolos sea almacenado.

Abstract

In this thesis we present a documentary study about the main attacks that can be mounted against a computer network as well as a detailed description of the specifications and implementations of the main Internet protocols. It is important to point out that the attacks mounted against the computer networks exploit the weakness in either the specifications or the implementations of these protocols.

One of the most important challenges in forensic computing deals with selecting the minimum amount of information needed to analyze, model and understand the way in which the attacks to computer networks are carried out and eventually, to bring the perpetrators to justice. As a way to tackle this problem, we propose the use of Montecarlo Sampling, and in particular, of accept/reject sampling which has the advantage that samples obtained by this method preserve the original distributions of the whole populations (all the data flows traversing the network) without requiring to know the actual population's distribution. We implemented our proposal as a software system named ADA** which uses Libpcap to obtain the data packets sent and received by the network card. ADA** allows the definition of filters that establish relations between sets of protocols and an acceptance probability which is the probability of storing a packet that contains headers of the set of protocols defined in the filter.

Agradecimientos

Al llegar al final de mi arduo trabajo, lleno de dificultades para el desarrollo de esta tesis, considero necesario agradecer a todos aquellos que de alguna manera contribuyeron a que finamente pueda concluir de manera satisfactoria con esta etapa de mi vida.

Debo agradecer de manera especial y sincera al Dr. Rolando Menchaca Méndez por aceptar realizar esta tesis de maestría bajo su supervisión. Su apoyo y confianza en mi trabajo y su capacidad para guiar mis ideas ha sido un aporte invaluable, no solamente en el desarrollo de esta tesis, sino también en mi formación como profesionista.

Debo agradecer también a mis directores de tesis: Dr. Rolando Quintero Téllez y M. en C. Sergio Sandoval Reyes que me han acompañado a lo largo del camino con su retroalimentación y participación activa. Debo destacar, por encima de todo, su disponibilidad y paciencia que hizo que, aunque algunas veces nos enfrascáramos en acaloradas discusiones, redundarán benéficamente tanto a nivel científico como a nivel personal.

Un agradecimiento especial para aquellos amigos que han formado parte de mi vida compartiendo momentos o intercambiado experiencias. Para aquellos que en algún momento han representado un obstáculo en mi camino, gracias mil, por haberme mostrado mis debilidades. Conocerlas me ha permitido finalmente trabajar en ellas para que algún día sean parte de mis fortalezas.

Y, por supuesto, el agradecimiento más profundo y sentido va para mi familia. Sin su apoyo y colaboración e inspiración habría sido imposible llevar a cabo esta dura empresa. A mi madre Teresa Rodríguez por su ejemplo de lucha y actitud incansable. A mis hermanos: Edgar Canseco y Claudia Cristina Canseco por siempre mostrarme su apoyo. A mi gran amigo José Antonio García Andrade por sus palabras de aliento y su apoyo incondicional.

Finalmente, debo agradecer al Centro de Investigación en Computación por darme la oportunidad de desenvolverme como investigadora, al CONACYT por haber financiado

gran parte de mis estudios de maestría otorgándome una beca a partir de la convocatoria del año 2010. También agradezco al Instituto Politécnico Nacional por el financiamiento de la finalización de esta tesis de maestría mediante la concesión de la beca Institucional.

La presente tesis se desarrolló como parte del proyecto de investigación ICyTDF-IPN titulado "Sentient City: Algoritmos y Arquitecturas Distribuidas de Monitoreo Colaborativo Multiescala".

Índice de figuras

Figura 1.1 Porcentaje por tipo de delito. 2
Figura 1.2 Las 10 ciudades con más quejas realizadas. 3
Figura 2.1 Metodología de análisis forense. 10
Figura 2.2 Arquitectura de un analizador de protocolo. 19
Figura 2.3 Módulo analizador de protocolos con Libpcap. 20
Figura 2.4 Diagrama de flujo del analizador de protocolos. 21
Figura 2.5 Diagrama de flujo del analizador de protocolos. 26
Figura 2.6 Información Forense de la capa de aplicación basada en el análisis de paquetes. 33
Figura 3.1 Capas del modelo OSI. 42
Figura 3.2 Mensaje con los siete encabezados de control de la pila OSI. 43
Figura 3.3 Capas de protocolo de Internet. 44
Figura 3.4 LLC y MAC, subcapas de la capa de enlace. 46
Figura 3.5 Encabezado de la capa de enlace. 47
Figura 3.6 Formato de la trama Ethernet en la capa de enlace. 48
Figura 3.7 Formatos de trama a) Ethernet Dix b) IEEE 802.3. 49
Figura 4.1 Encabezado TCP. 56
Figura 4.2 Saludo TCP. 57
Figura 4.3 Encabezado IP. 60
Figura 4.4 Mensaje echo o echo replay. 66
Figura 4.5 Mensaje de destino inalcanzable. 66
Figura 4.6 Mensaje de enfriamiento fuente. 67
Figura 4.7 Mensaje Redireccionar. 67
Figura 4.8 Parte del encabezado IP. 68
Figura 5.1 Capas de red de Linux. 72
Figura 5.2 Arquitectura Libpcap. 74
Figura 5.3 Arquitectura del sistema 75
Figura 5.4 Administración: Agregar, eliminar y mostrar 76
Figura 5.5 Módulo de filtrado 77
Figura 5.6 Diagrama de flujo Agregar protocolo. 79
Figura 5.7 Dagrama de flujo Borrar protocolo. 79
Figura 5.8 Diagrama de flujo Mostrar protocolo. 80
Figura 5.9 Diagrama de flujo pedir datos de protocolo. 80
Figura 5.10 Diagrama de flujo leer protocolos de memoria secundaria. 81

Figura 5.11 Diagrama de flujo escribir protocolos en memoria secundaria. 82
Figura 5.12 Diagrama de flujo mostrar protocolos. 82
Figura 5.13 Diagrama de flujo eliminar protocolo de la lista. 83
Figura 5.14 Diagrama de flujo para solicitar datos del nuevo filtro. 84
Figura 5.15 Diagrama de flujo: Copiar datos a campos especiales. 85
Figura 5.16 Diagrama de flujo: Definir datos de cada campo. 86
Figura 5.17 Diagrama de flujo: Comparar filtros. 86
Figura 5.18 Diagrama de flujo: Escribir datos. 87
Figura 5.19 Diagrama de flujo: Leer filtros. 87
Figura 5.20 Diagrama de flujo: Mostrar protocolo. 88
Figura 5.21 Diagrama de flujo: borrar filtros. 88
Figura 5.22 Diagrama de flujo de filtrado de paquetes. 89
Figura 5.23 Encabezado Ethernet II. 92
Figura 5.24 Administración de protocolos. 96
Figura 5.25 Campos generales del protocolo. 96
Figura 5.26 Campos de encabezado del protocolo. 97
Figura 5.27 Mostrar protocolos. 97
Figura 5.28 Borrar protocolo. 98
Figura 5.29 Administrador de filtros. 98
Figura 5.30 Campos generales para el filtro. 99
Figura 5.31 Campos especiales para el filtro. 100
Figura 5.32 Mostrar filtros. 101
Figura 5.33 Borrar filtro. 101
Figura 5.34 Datos generales para la estructura de la caca de enlace. 102
Figura 5.35 Campos especiales. 102
Figura 5.36 Datos para Libpcap. 103
Figura 6.1 Total de paquetes en ADA** vs Wireshark 106
Figura 6.2 Análisis de paquete por hora 106
Figura 6.3 Total de paquetes 109
Figura 6.4 Información en hexadecimal. 110
Figura 6.5 Cantidad de paquetes por hora. 112
Figura 6.6 Total de paquetes por segundo. 112

Índice de tablas

Tabla 1.1 Número de reportes por tipo de fraude. 3
Tabla 2.1 Herramientas utilizadas en el cómputo forense. 13
Tabla 2.2 Ventajas y desventajas del contenido completo. 16
Tabla 3.1 Clasificación de redes basada en su escala. 40
Tabla 4.1 Clasificación de los atacantes según sus conocimientos. 52
Tabla 4.2 Puertos bien conocidos. 56
Tabla 4.3 Elementos de un ataque DDoS. 62
Tabla 4.4 Pasos para llevar a cabo un DDoS. 62
Tabla 4.5 Combinaciones invalidas de banderas TCP. 69
Tabla 4.6 Resumen de ataques y los campos utilizados. 69
Tabla 6.1 Protocolos por capa 107
Tabla 6.2 Filtros para la capa tres 107
Tabla 6.3 Filtros para la capa cuatro 108
Tabla 6.4 Filtros definidos. 111

Tabla de contenido

Resumen _____ *i*

Abstract _____ *ii*

Agradecimientos _____ *iii*

Índice de figuras _____ *v*

Índice de tablas _____ *vii*

Tabla de contenido _____ *viii*

CAPÍTULO 1. INTRODUCCIÓN _____ *1*

1.1 Antecedentes _____ **1**

1.2 Planteamiento del Problema _____ **4**

1.3 Objetivos _____ **5**

1.3.1 Objetivo General _____ 5

1.3.2 Objetivos Específicos _____ 5

1.4 Justificación _____ **6**

1.5 Contribuciones _____ **7**

1.6 Alcances y Límites _____ **7**

1.7 Organización de la Tesis _____ **8**

CAPÍTULO 2. CÓMPUTO FORENSE _____ *9*

2.1 Definición de cómputo Forense _____ **9**

2.2 Tipos de tecnología de cómputo forense _____ **11**

2.2.1 Análisis forense _____ 12

2.2.2 Analizador de protocolo _____ 17

2.2.2.1 Organización general de una red de computadoras _____ 17

2.2.2.2 Módulo de captura de paquetes*:* _____ 19

2.2.2.3 Módulo analizador de protocolo: _____ 21

2.2.3 Captura de datos _____ 23

2.2.3.1 Implementación de la captura de datos _____ 23

2.2.3.2 Motores de captura de datos _____ 25

2.2.4 Ksensor _____ 27

2.2.5 Henpa _____ 29

2.2.5.1 Tecnología de intersección _____ 29

2.2.5.2 Operación de captura de paquetes _____ 30

2.2.5.3 Arquitectura _____ 31

2.2.6 Información Forense de la capa de aplicación obtenida en el análisis de paquetes _____ 32

2.2.6.1 Modelo forense _____ 33

2.2.6.2 Captura de paquetes 34
2.2.6.3 Aplicación de regeneración de datos 35
2.3 Propuesta de solución 37
2.4 Resumen 37
CAPÍTULO 3. ARQUITECTURA DE INTERNET 39
3.1 Tipos de redes 39
3.2 Modelo de referencia 41
3.2.1 Capas del modelo OSI 41
3.2.2 Capas de protocolos de Internet TCP/IP 44
3.3 Ethernet 48
3.4 Resumen 49
CAPÍTULO 4. PRINCIPALES TIPOS DE ATAQUE 51
4.1 Introducción 51
4.2 Ataque cibernético 51
4.3 Vulnerabilidades en la pila de protocolos TCP/IP 55
4.3.1 Ataques de monitoreo 55
4.3.2 Ataques de validación 59
4.3.3 Ataques de denegación de servicios 61
4.3.4 Otros tipos de ataque 66
4.4 Resumen 69
CAPÍTULO 5. ANÁLISIS Y DISEÑO DEL SISTEMA DE CAPTURA ALEATORIA ADA** 71
5.1 Introducción 71
5.2 Plataforma de desarrollo 71
5.3 Captura de paquetes 73
5.4 Arquitectura de la aplicación 75
5.5 Descomposición en subsistemas 76
5.6 Diagramas de flujo 77
5.6.1 Administrador de protocolos 78
5.6.2 Administrador de filtros 83
5.6.3 Módulo de filtrado 89
5.6.4 Muestreo 90
5.6.5 Captura de tráfico de red 92
5.7 Interfaz Gráfica 96
5.7.1 Administrador de protocolos 96
5.7.2 Administrador de filtros 98
5.7.3 Captura de paquetes 101
5.8 Resumen 103
CAPÍTULO 6. PRUEBAS Y RESULTADOS 105

6.1 Captura de paquetes ADA vs Wireshark** 105
6.2 Análisis de paquetes 107
6.3 Estadísticas 111
6.4 Resumen 113
CAPÍTULO 7. CONCLUSIONES ***115***
7.1 Logros Alcanzados 115
7.2 Trabajo a futuro 116
BIBLIOGRAFÍA ***119***
Anexos ***122***
ANEXO A. Glosario 122
ANEXO B. Legislación en México 129
ANEXO C. Símbolos de soporte de información o dispositivos físicos: 132
ANEXO D. Manual de operación del sistema 136

CAPÍTULO 1. INTRODUCCIÓN

El cómputo forense estudia arquitecturas, algoritmos, técnicas y procedimientos que tienen como objetivo almacenar, recuperar y analizar información relacionada con las actividades tanto de programas de cómputo como de los usuarios humanos. Uno de los principales objetivos del cómputo forense es recopilar la información necesaria para poder recrear y analizar los posibles ataques a un sistema de cómputo (computadora individual o red de computadoras) y de esta forma desarrollar contramedidas para los mismos, pero sobre todo para poder obtener pruebas útiles en una investigación criminal. Debido a que una de las principales motivaciones del cómputo forense es contrarrestar los crímenes informáticos. En este apartado se presentan algunos datos del crimen en Internet. Así mismo, se describe el planteamiento del problema, el objetivo general, se enumeran los objetivos específicos, se desarrolla la justificación y se enlistan los beneficios esperados, además de definir los alcances y límites.

1.1 Antecedentes

Los pequeños negocios e incluso las grades organizaciones no saben proteger la información sensible de sus sistemas informáticos, esta falta de conocimiento provoca que las puertas estén abiertas para que los delincuentes puedan cometer sus delitos [1].

En fechas recientes, el IC^3 (*Internet Crime Complaint Center* por sus siglas en inglés) menciona en su publicación 2011 Internet Crime Report [2], que en el año citado se incrementó el número de quejas como puede apreciarse en la Figura 1.1 y que el ajuste de pérdidas fue de 485.3 millones de dólares. La tendencia desde el año 2008 hasta 2011 muestra que en 2012 el total de quejas llegará a 328.

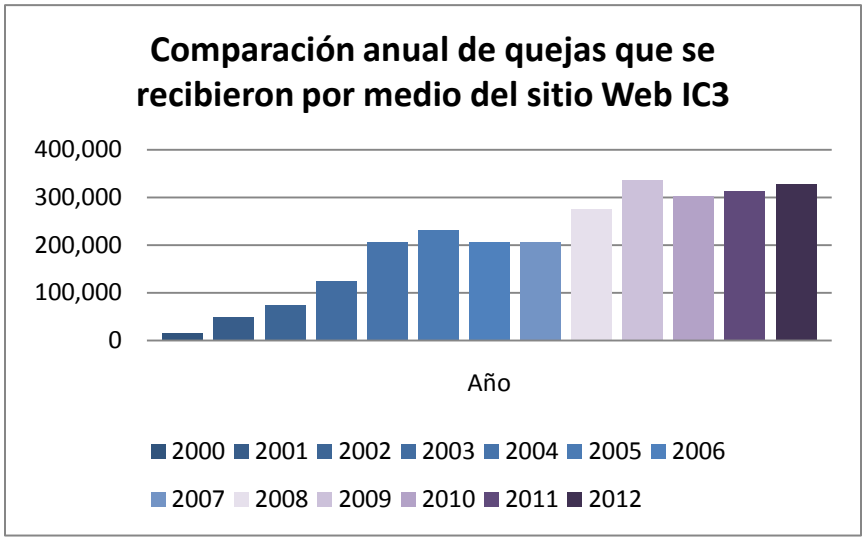


Figura 1 Porcentaje por tipo de delito.

IC^3 recibe en promedio 25,000 quejas cada mes y, en el año 2010 las más comunes fueron: no entrega de mercancía o pago, fraude haciéndose pasar por el FBI y robo de identidad. Las víctimas de estos fraudes reportaron pérdidas por cientos de millones de dólares [3]. En el reporte de ese mismo año México ocupó el lugar ocho con un 0.2% de quejas en el periodo (ver Figura 1.2), sin embargo, para el 2011, México no aparece en la lista, pero es importante hacer notar que esto no significa una disminución en la actividad criminal, pues recientemente el IFAI (Instituto Federal de Acceso a la Información y Protección de datos) menciona que cada 18 segundos una persona adulta es víctima de algún delito informático en el mundo [4]. En esta misma publicación se cita que *Norton Symantec* calcula que en México más de 14.8 millones de personas han sido víctimas de delitos informáticos entre los 12 meses anteriores, reportando pérdidas de hasta 2.2 billones de dólares al año.

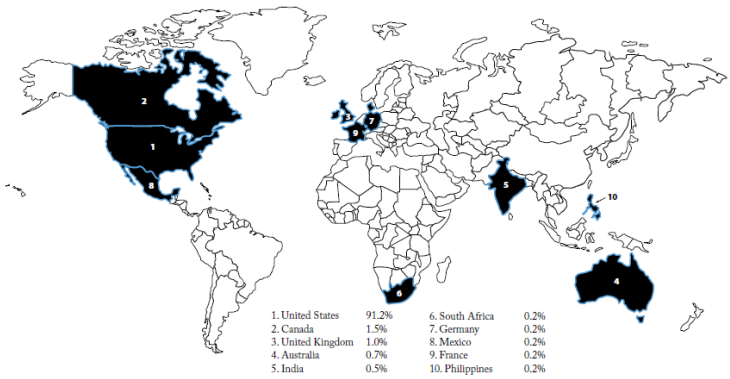


Figura 2 Las 10 ciudades con más quejas realizadas.

En 2011 IC[3] reporta que la ofensa más común fue la estafa, en segundo lugar estuvo el robo de identidad, seguido del fraude en el que los criminales convencen a las víctimas de pagar una cantidad de dinero para recibir algo a cambio pero no se hace la entrega. La tabla 1.1 muestra la cantidad de reportes por tipo de fraude reportado en IC[3].

Tabla 1.1 Número de reportes por tipo de fraude.

Tipo	Cantidad
Estafas relacionadas con el FBI	35,764
Robo de identidad	28,915
Fraude de pagos por adelantado	27,892
No subasta, no entrega de mercancía	22,404
Fraude de pago en exceso	18,511

Bajo este escenario de aumento de crimen en Internet y sus consecuentes pérdidas, se refleja la importancia de los siguientes puntos a tratar.

- Proceder en contra de los delincuentes para que este tipo de delito no quede impune. Para cumplir con este propósito, es necesario encontrar las pruebas necesarias que

cumplan con el requisito de ser confiables para que se puedan presentar en una corte y demostrar el delito.

- Tener un registro de los mecanismos utilizados por el delincuente para cometer el crimen, proporciona una herramienta para reconstruir los hechos, encontrar los tipos de ataque más comunes y buscar patrones de comportamiento para poder evitarlos en el futuro.

Para tratar con estos puntos existe el Cómputo Forense, cuyo objetivo es recuperar, analizar y presentar material basado en la computadora de tal manera que pueda ser utilizado como evidencia en una corte de ley [1]. Con este fin, existe una serie de procedimientos bien definidos encaminados a preservar de manera segura los datos encontrados, sin el peligro de que sean modificados. En el siguiente capítulo revisaremos de manera más detallada estos conceptos.

1.2 Planteamiento del Problema

Como podrá observarse, existen diversas herramientas utilizadas en el cómputo forense, algunas de ellas se utilizan para recolectar datos que llegan a través de la red. En este tipo de herramientas, uno de los problemas a resolver para llevar a cabo esta recolección y detección del tráfico de la red, es identificar los eventos útiles y almacenar lo mínimo con la mayor cantidad de evidencia [4]. En este trabajo se plantea el desarrollo de un sistema que facilite la tarea de recolección de tráfico de red. Además, la muestra obtenida debe contar con la información suficiente para que sea útil en la investigación de un delito y sea posible rastrear los mecanismos de ataque.

La recolección de datos se hace a nivel del *kernel* del sistema operativo con la finalidad de restringir el acceso de los usuarios comunes al control de la aplicación. Cada dispositivo se encarga de almacenar el tráfico que recibe haciendo uso de una función de probabilidad que define el usuario del sistema; la probabilidad depende del tipo de protocolo y la posibilidad de que sea dañino. Se hace una investigación de los protocolos más utilizados en crímenes cibernéticos para la definición de estas funciones.

El hecho de que cada computadora sea responsable de monitorear parte del tráfico de red, puede contribuir a mejorar el rendimiento en la recolección de paquetes debido a que se vuelve

un trabajo colaborativo donde el tráfico se distribuye entre todas las computadoras y cada una tiene información de lo que ocurre en la red; al mismo tiempo, se elimina la necesidad de un servidor especializado en la recolección de evidencia y por lo tanto el sistema tiende a ser más robusto y escalable.

Cuando se tiene un servidor especializado, se requiere que todo el tráfico de red pase por ese servidor, como consecuencia se deben considerar las diversas situaciones que se presentan en una Red de área local (LAN): LAN de difusión, switch LAN y LAN inalámbricas en la que cada una representa una manera diferente de hacer la recolección de datos [5]. Se espera que esto ya no sea un problema en el sistema propuesto.

Después de cumplir con el requisito de que la información sea mínima y útil, es de suma importancia desarrollar una serie de herramientas que permitan su presentación de manera que sea entendible y útil para los usuarios humanos.

1.3 Objetivos

En esta sección se describe la finalidad del sistema y lo que se espera obtener con el desarrollo de este trabajo.

1.3.1 Objetivo General

Diseñar, desarrollar y probar un sistema forense de recolección de datos para redes de computadora, con la capacidad de muestrear información de acuerdo a una distribución de probabilidad que dependerá del tipo de protocolo que esté siendo usado.

1.3.2 Objetivos Específicos

- Realizar un estudio documental para caracterizar los principales ataques a las redes de computadoras.

- Determinar qué características deben cumplir los datos recopilados por un sistema forense para que éstos sean considerados como evidencia válida, ya sea en un juicio público o en una investigación privada.
- Diseñar, implementar y probar un módulo para Linux encargado de recopilar información que fluya a través de la interfaz de red.
- Diseñar, implementar y probar un módulo de almacenamiento de la información recopilada.
- Verificar que los datos almacenados por el sistema forense propuesto, cumplen con los requisitos básicos para ser considerados como evidencia válida, ya sea en un juicio público o en una investigación privada.
- Desarrollar una herramienta de análisis estadístico de los datos recopilados.

1.4 Justificación

Internet ha abierto un mundo nuevo de comunicaciones para consumidores, pero a su vez se ha convertido en un lugar con una actividad criminal creciente como se mostró en la Figura 1.1. El uso de dinero electrónico y bancos virtuales, intercambios y tiendas web ha comenzado a ser uno de los factores para que se desarrolle un nuevo tipo de crimen conocido como crímenes transnacionales. El crimen puede ser cometido a miles de kilómetros de la escena real, en otras palabras, un criminal cibernético no necesita salir de su casa o cruzar una frontera para cometer un acto criminal en muchas ciudades alrededor del mundo. La comunicación podría ser enrutada a través de compañías de teléfono locales, operadores de larga distancia, proveedores de servicio de Internet y redes inalámbricas y redes satelitales; podría ir a través de computadoras localizadas en todo el mundo antes de atacar a los sistemas destino. La evidencia del crimen cibernético podría entonces ser almacenada sobre una computadora (o conjunto de computadoras) en una ciudad diferente de donde el criminal ejecuta el acto criminal.

La proliferación de la actividad criminal tecnológicamente asistida y el crimen cibernético, merece atención adicional de la comunidad mundial mediante la promulgación de leyes y la implementación tecnológica efectiva de herramientas que reduzcan la vulnerabilidad de los sistemas a la actividad criminal.

Al desarrollar un sistema de seguridad forense se puede contribuir al desarrollo de mejores sistemas de seguridad. Al hacer un análisis del tráfico de la red se puede tener una mejor idea de cuáles son las constantes o variables presentes en un delito informático que lleven a un posible rastreo de los intrusos [6].

1.5 Contribuciones

El presente sistema aborda principalmente el problema de la recolección distribuida de datos forenses. Las principales ventajas de este esquema contra su contraparte centralizada son:

- Debido a que no existe un único punto der recolección, el sistema es tolerante a ataques. Bajo el esquema distribuido, es necesario comprometer un gran número de máquinas para que un atacante pueda evitar que el sistema forense recopile evidencias de dicho ataque.
- No es necesario tener un servidor de altas prestaciones para realizar la recopilación de datos. Lo anterior se debe a que la sobrecarga impuesta a las máquinas que componen el sistema de recolección es relativamente bajo, ya que no capturan todos los datos que pasan por su interfaz de red, sino que realizan un muestreo probabilístico de los datos relevantes.

Por otra parte, se espera que la puesta en operación del sistema propuesto sirva para recopilar datos reales que puedan servir a investigaciones futuras en el área del análisis de datos.

1.6 Alcances y Límites

El software desarrollado llevará a cabo la recolección de datos del tráfico de red y realizará el almacenamiento de los datos que el usuario solicite. Estos datos serán filtrados a través de la especificación de los protocolos sobre los que se tenga interés, aplicando un muestreo probabilístico. El sistema permitirá que el usuario agregue nuevos protocolos a través de la especificación de filtros.

Dentro de las limitaciones del sistema se encuentra que no hará la traducción de los datos a formato legible, éstos sólo se presentarán en formato hexadecimal. Sin embargo, el sistema presenta datos estadísticos en forma gráfica lo cual facilita su interpretación

1.7 Organización de la Tesis

La tesis se divide en siete capítulos, el primero es una breve introducción al tema forense y su importancia. El capítulo dos contiene la definición de conceptos relacionados con el cómputo forense, así como algunos trabajos que han abordado el tema desde diferentes perspectivas. En el capítulo tres se revisan los fundamentos de la arquitectura de Internet que son de utilidad para llevar a cabo la construcción del sistema de software. Un tema sobresaliente en el análisis forense son las vulnerabilidades en los protocolos de comunicación. Este tema se desarrolla en el capítulo cuatro. El diseño del sistema con sus respectivos diagramas se presenta en el capítulo cinco. Las pruebas y los resultados se explican en el capítulo seis. Finalmente, en el capítulo siete se pueden ver las conclusiones del trabajo desarrollado, trabajo a futuro y las recomendaciones para mejorarlo Además se incluyen la bibliografía y los anexos.

CAPÍTULO 2. CÓMPUTO FORENSE

En este capítulo se hace una revisión general de lo que significa cómputo forense; y los pasos que se deben seguir para la obtención de datos confiables. Se profundiza en el concepto de Red forense y las tecnologías que son utilizadas en ellas. Así mismo, Se hace una revisión general de los posibles tipos de ataque (este tema se desarrolla en el capítulo cuatro) y las técnicas de recolección de los datos.

2.1 Definición de cómputo Forense

El cómputo forense involucra una serie de tareas enfocadas a la recolección de evidencia confiable; es la recolección, preservación, análisis y presentación de evidencia relacionada con las computadoras que puede ser útil en casos criminales, y disputas.

Una de las características en el cómputo forense, es que la evidencia en computadoras es con frecuencia creada de manera transparente por el sistema operativo de la misma computadora sin el conocimiento del usuario. La información podría, de hecho, no estar a la vista. Para encontrarla se requiere de herramientas forenses de *software* y técnicas especiales [1].

Si se encuentra una computadora como premisa en la escena de un crimen, la oportunidad de que exista evidencia valiosa es grande, si la computadora y su contenido se examina, aunque sea brevemente por alguna persona que no sea un especialista forense, la credibilidad de la evidencia se contamina y deja de ser útil [1].

Para salvaguardar la veracidad de la información es necesario seguir una metodología que comienza con la verificación del incidente, la descripción del sistema y la adquisición de evidencia [7] como se muestra en la Figura 2.1.

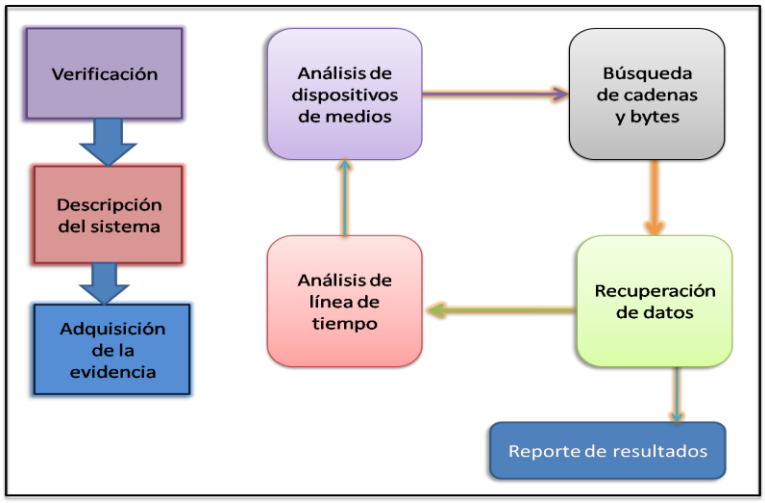


Figura 3 Metodología de análisis forense.

Además de seguir una metodología el procedimiento de análisis forense se debe de llevar a cabo en presencia de los directivos de la organización, personal de tecnologías de la información y abogados [7].

Existen diversos casos en los que el cómputo forense es útil. Una computadora puede ser la principal herramienta para la investigación de un crimen o la investigación en una demanda de abuso en el trabajo:

Uso del cómputo forense para aplicar la ley: Si hay una computadora como parte inicial de una escena del crimen, hay muchas probabilidades de que exista evidencia valiosa en ella; si se examina el contenido de la computadora (aunque sea brevemente) por cualquier otra persona que no es un especialista en cómputo forense entrenado y experimentado, la credibilidad y utilidad de la evidencia se contamina.

Asistencia del cómputo forense para los procedimientos de recursos/empleo humanos: El análisis de cómputo forense está comenzando a ser cada vez más útil en

los negocios. Las computadoras pueden contener evidencia en muchos tipos de procedimientos de recursos humanos, incluyendo demandas por acoso sexual, denuncias por discriminación y demandas por despido injustificado. La evidencia se puede encontrar en sistemas de correo electrónico, servidores de red, o computadoras personales de los empleados; sin embargo, debido a la facilidad con la que los datos de la computadora pueden ser manipulados, si la búsqueda y el análisis no se realizan por un especialista entrenado en cómputo forense, probablemente no sean válidos en la corte.

Servicios de cómputo forense: No importa lo cuidadosas que sean las personas, cuando intentan robar información electrónica (cualquier cosa desde la bases de datos de los clientes hasta las huellas digitales), dejan rastros de sus actividades; igualmente, cuando intentan destruir evidencia incriminatoria contenida en su computadora (desde notas de acoso hasta tecnología robada), dejan pistas vitales.

Un profesional del cómputo forense no hace más que encender la computadora, hacer una lista de directorios y buscar a través de los archivos.

Algunos de los servicios forenses que se pueden hacer son:

- Confiscación de datos
- Duplicado y preservación de datos
- Recuperación de datos
- Búsqueda de documentos
- Conversión de medios

2.2 Tipos de tecnología de cómputo forense

Como se ha mencionado, la cibernética forense es el descubrimiento, análisis y reconstrucción de evidencia extraída de cualquier elemento de un sistema de computadora, redes de computadora, medios informáticos y periféricos de computadora que permite a los investigadores resolver un crimen.

Existen dos componentes en el campo de la tecnología de la cibernética forense. El primero es el cómputo forense, que se ha revisando con anterioridad y el segundo componente, redes forenses, técnicamente involucra un mayor desafío en la cibernética

forense ya que tiene que ver con la obtención de evidencia digital que es distribuida a gran escala, a través de redes complejas. Con frecuencia esta evidencia es transitoria de manera natural y no se preserva dentro del medio de almacenamiento permanente. Una red forense trata principalmente con el análisis a fondo de la evidencia de intrusión en redes de computadora. Las herramientas comerciales de análisis de intrusión son inadecuadas para tratar con la redes de entornos distribuidos de la actualidad. En la batalla en contra de hackers maliciosos, se deben realizar funciones de cibernética forense que soporten objetivos como, contención de un ataque cibernético, localización e identificación de los perpetradores y mitigación de los daños.

Una red forense como la definen Daniels, Wei y Thomas [8] es el uso de técnicas científicamente probadas para coleccionar, fusionar, identificar, examinar, correlacionar, analizar y documentar evidencia digital de múltiples fuentes digitales, activamente procesando y transmitiendo con el propósito de descubrir hechos allegados al intento planeado, o medir los éxitos de actividades no autorizadas que tienen como objetivo interrumpir, corromper y/o comprometer los componentes de los sistemas, así como también proveer información que ayude a la asistencia o recuperación de estas actividades.

Más adelante se hace una descripción de las herramientas que actualmente se utilizan en el cómputo y redes forenses así como el análisis de algunos trabajos que se han desarrollado en investigación.

2.2.1 Análisis forense

Unido al concepto de red forense se tiene el de análisis forense que es definido por Farmer y Wietse [9], como la obtención de datos empleando métodos que distorsionen lo menos posible la información con la intención de reconstruir todos los datos y/o eventos que ocurrieron sobre un sistema en el pasado.

En la Tabla 2.1 se describen diversas herramientas de seguridad y monitoreo en redes que se utilizan en el cómputo forense.

Tabla 2.1 Herramientas utilizadas en el cómputo forense.

Nombre	Características
TCPDump	Sniffer y analizador que se ejecuta en línea de comandos, intercepta y muestra los paquetes que se están transmitiendo en la red. Esta herramienta captura, muestra y almacena todas las formas de tráfico de red en diversos formatos de salida. Imprime paquetes datos de paquete como timestamp, protocolo, dirección fuente y dirección destino, puertos, banderas, opciones y números de secuencia.
Wireshark	Es el analizador de protocolo más popular. Realiza captura en tiempo real en formato Libpcap, inspecciona cientos de protocolos, hace análisis fuera de línea y trabaja en múltiples plataformas. Puede leer y escribir archivos en diferentes formatos de otras herramientas.
TCPFlow	Captura los datos transmitidos en conexiones *TCP* y los almacena para el análisis de protocolo. Reconstruye el flujo de datos actual y lo almacena en un archivo diferente. TCPFlow entiende el número de secuencia y reconstruye correctamente sin tomar en cuenta retransmisiones o entregas fuera de orden.
Flow-tools	Librería para recolectar, enviar, procesar y generar reportes del flujo de datos de la red. Las principales herramientas de la suite son: – flow capture, recolecta y almacena los flujos que son exportados desde un router, -flow-cat, concatena el flujo de archivos, – flow report, genera reportes del conjuntos de datos de la red y – flow filters filtra el flujo basado en campos exportados.
NfDump	Una suite de herramientas que trabaja con formatos de flujo de datos: – Nfcapd, El demonio de captura de datos de la red lee y almacena el flujo de datos de la red. – NfDump, descarga los datos de la red leídos de estos archivos, muestra y crea estadísticas de los datos como dirección *IP*, *puertos*, etc. – Nfprofile, perfila los filtros de flujo de datos de la red de acuerdo al conjunto de filtros especificados y almacena los datos filtrados. – Nfreplay, reenvía el flujo de datos a otro *host*.
PADS	Es un sniffer portable, ligero e inteligente. Está basado en motores de detección utilizados para hacer la detección en modo pasivo. Puede monitorear tráfico de paquetes *TCP (Transmission Control Protocol)*, *ARP* (Address Resolution Protocol) e *ICMP (Internet Control Message Protocol)*.
Argus	Procesa paquetes en archivos de captura y genera reportes de estado detallados del flujo detectado en el flujo de paquetes. El reporte de flujo captura la semántica de todo el flujo teniendo cuidado de la reducción de datos. Los datos de auditoría son buenos para redes forenses, no repudio, detección baja de escaneo y soporte para eventos *zero-day.*
Nessus	Descubre vulnerabilidades de escaneo a gran velocidad, configuración de auditoría, perfiles activos, descubre datos vulnerables y análisis de vulnerabilidades.

Sebek	*Kernel* basado en herramienta de captura de datos diseñado para capturar toda la actividad sobre un *Honeypot*. Éste almacena las pulsaciones del teclado de una sesión que está utilizando encriptación, recupera archivos copiados con *SCP*, captura contraseñas utilizadas para acceder a un sistema remoto y realiza muchas otras tareas relacionadas a lo forense.
TCPTtrace	Produce diferentes tipos de salida que contiene información, tal como el tiempo transcurrido, segmentos que son enviados y recibidos, retransmisiones, tiempos de ida y vuelta, ventanas de anuncio y rendimiento.
Ntop	Utilizado para medición de tráfico, monitoreo de tráfico de red, optimización, planeación, detección de violaciones. Esta herramienta provee soporte para ambos casos, rastreo de ataques en tiempo real e identificación de agujeros potenciales incluyendo suplantación de identidad, tarjetas de red en modo promiscuo, ataque de denegación de servicios, caballos de troya y ataques de escaneo de puertos
TCPStat	Reporte de las estadísticas de la interfaz de red como la banda ancha, número de paquetes, paquetes por segundo, promedio del tamaño de paquetes, desviación estándar de medidas de paquetes y carga de interfaz para monitorear una interfaz o leer de un archivo.
IOS NetFlow	Colecciona y mide atributos de los paquetes *IP*, de cada paquete enviado a través de *routers* y *switches,* grupos similares en un flujo para ayudar a entender ¿Quién? ¿Qué? ¿Cuándo? y ¿Cómo? está fluyendo el tráfico. También detecta las anomalías de la red y vulnerabilidades de seguridad.
TCPDstat	Produce desglose de tráfico por protocolo para un archivo *libpcap*, como número de paquetes, promedio y desviación estándar, número de un par único de dirección destino y fuente. Esta herramienta es útil para obtener una vista a alto nivel de patrones de tráfico.
Ngrep	Una herramienta pcap-aware que permite especificar expresiones regulares o hexadecimales extendidas para comparar con los datos de carga útil. Puede depurar interacción de protocolos en texto plano para identificar y analizar comunicación de red anómala y almacenarla, leer y reprocesar archivos de vaciado mientras buscamos patrones específicos de datos.
TCPXtract	Extrae archivos de tráfico de red basados en firmas de archivo. Esta herramienta también se puede utilizar para interceptar archivos que se transmiten a través de la red.
SilK	Soporta captura eficiente, almacenamiento y análisis de flujo de datos de la red basado en Cisco NetFlow. La suite de herramientas consiste en herramientas de recolección y análisis. Entrega análisis con los medios para entender, consultar y resumir las dos cosas, datos de tráfico reciente e histórico en registros de tráfico de red. Silk soporta redes forenses en identificar artefactos de intrusión, explotación de vulnerabilidades, comportamiento de gusano, etc. Silk tiene un elemento clave y administra el gran volumen de tráfico de red administrando sólo la información relacionada a la seguridad.
TCPReplay	Conjunto de herramientas con la habilidad de clasificar previamente el tráfico capturado como cliente o servidor, reescribe los encabezados de la capa 2,3 y 4, finalmente reproduce el tráfico de regreso sobre la red. TCPPrep es un archivo pcap de pasadas múltiples que determina paquetes como cliente o servidor, TCPRewrite es un archivo editor pcap que reescribe los encabezados de paquetes, TCPReplay reproduce archivos pcap a velocidades arbitrarias sobre la red y TCP Brridge puentea dos segmentos de red.

POf	Passive OS por sus siglas en inglés, toma huellas dactilares para capturar el tráfico de red que viene de un host a la red. También puede detectar la presencia de un firewall, uso de NAT, existencia de una herramienta balanceadora de carga, distancia al sistema remoto y su tiempo de actividad.
Nmap	Herramienta para exploración de redes y auditoria de seguridad. Soporta varios tipos de escaneo de puertos. Utiliza flujo de paquetes *IP* en formas novedosas para determinar la disponibilidad de un *host* en la red, sus servicios, sistema operativo y versión que utiliza, firewalls en uso y otras características.
Bro	NIDS que monitorea en forma pasiva el tráfico de red buscando actividad sospechosa. Detecta intrusión ya que primero analiza el tráfico de la red para extraer su semántica a nivel aplicación y después ejecuta analizadores orientados a eventos que comparan su actividad con patrones que considera problemáticos. Esta herramienta es en primer lugar una plataforma de búsqueda para IDS, analizador de tráfico y redes forenses.
Snort	Intrusión de redes sistema de prevención y detención capaz de realizar registro de paquetes, *sniffing* y análisis de tráfico en tiempo real. Puede realizar análisis de protocolo búsqueda de contenido, comparación y análisis a nivel aplicación, capturar el tráfico en formato Libpcap.

Las bitácoras de *firewall*, *routers* e *IDS (Intrusion Detection System por sus siglas en inglés)* unidas a las herramientas de seguridad y monitoreo de redes se utilizan para facilitar el análisis de tráfico de red.

Puede haber diferentes tipos de evidencia basada en tráfico de red (Network Based Evidence por sus siglas en inglés) y cada una tiene relación con otra como se indica a continuación.

- Datos de contenido completo.
- Datos de sesión.
- Datos de alerta.
- Datos estadísticos.

Los datos de contenido completo involucra la captura de cada elemento de una conversación de datos. Esto quiere decir que incluye cabeceras y carga útil que se observa en el medio en cualquiera de sus dos formas, cableado o inalámbrico. Cada bit del paquete de datos es capturado. VoIP (Voz sobre IP) ha complicado las cosas, pero incluso la captura y decodificación de este tipo de comunicaciones es posible.

La mejor forma de NBE (Network Based Evidence) es la que es más completa, es decir, tiene el registro de cada paquete que viaja a través de la red. No obstante, existen algunas desventajas. La Tabla 2.2 contiene la lista de ventajas y desventajas de este tipo de contenido.

Tabla 2.2 Ventajas y desventajas del contenido completo.

Ventajas	Desventajas
Identificación de naturaleza y propósito de cada registro.	Exige recursos de almacenamiento.
Identificación de técnicas de reconocimiento.	Un indicador puede estar entre cientos de miles de paquetes.
Información intercambiada entre dos aplicaciones.	La mayoría de la información almacenada pude no ser útil.

Los datos de sesión terminan con el problema de almacenar y analizar cientos de datos de tráfico de red. En lugar de almacenar cada bit del paquete de datos, se almacena un resumen. El resumen muestra el intercambio de paquetes en una conversación agrupados por flujos o intercambio de datos. Cada conversación es un registro diferente en los datos de contenido completo.

Los datos de alerta se crean al analizar otras formas de NBE en busca de patrones que indiquen elementos de interés. Los datos de alerta generalmente se crean por medios detectores de intrusos en red (NIDS por sus siglas en inglés). Un NIDS es programado para reconocer patrones de bits, ciertos patrones de actividad maliciosos, y otras características de tráfico en busca de violaciones de políticas. Cuando el NIDS observa tráfico que concuerda con el patrón (firma o regla), informa al administrador por medio de una alerta enviada a una base de datos, a la consola o en un mensaje de correo electrónico. Una alerta puede ser inofensiva pero es sospechosa y se debe de examinar.

Los datos estadísticos se enfocan en dar una visión muy general del tráfico de red y pocas características de los paquetes y flujos. Los datos estadísticos son utilizados comúnmente como una medida del desempeño de una red. Estadísticas similares pueden ser utilizadas para la seguridad y análisis forense, aunque las técnicas actuales son muy pobres e inmaduras. Un ejemplo sencillo puede ser un listado de servicios o protocolos

por los que se transfirió más información en una red. Tal vez una organización intenta descubrir los equipos que participan en un intercambio ilegal de archivos por redes *P2P*. Los equipos probablemente aparezcan en la lista de equipos más activos.

La captura de datos se puede hacer antes o después de un ataque. Cuando ocurre antes se conoce como método de monitoreo de seguridad de red proactivo y es monitoreo de seguridad de red reactivo si la captura se hace después.

El monitoreo proactivo puede ayudar a identificar intrusiones antes de que ocurran, sin embargo, la experiencia muestra que la mayoría de las organizaciones no captura información de esta forma [7].

La definición de red forense dice que es la ciencia que trata con la captura, recolección y análisis de tráfico de la red [10]. Al llevarse a cabo el análisis de datos de este tipo de redes se espera resolver uno de los siguientes problemas: identificación del grupo atacante o reconstrucción del escenario del ataque [11].

Para llevar a cabo la recolección de los datos para el análisis se han propuesto diferentes soluciones que abordan alguno de los retos de esta tarea y se revisan a continuación.

2.2.2 Analizador de protocolo

En este trabajo Zhin [12] propone la construcción de un analizador de protocolos con el fin de explicar los conceptos básicos y arquitectura de una red TCP/IP. Se inicia con este trabajo a manera de introducción porque muestra de manera general cómo funciona un analizador de protocolos. Como se hará evidente, un analizador de protocolo es la base de la mayoría de las herramientas forenses.

Pero, ¿Qué es un analizador de protocolo? Un analizador de protocolo que también se conoce como *sniffer*, es un programa o dispositivo que escucha a escondidas el tráfico de la red para grabar la información que viaja en ella [13]. En las siguientes subsecciones se realiza un recorrido por el contenido del trabajo presentado por Zhin.

2.2.2.1 Organización general de una red de computadoras

Existen dos modelos de referencia principales, el modelo OSI y el modelo TCP/IP. El modelo OSI consta de siete capas [14]: física, enlace de datos, red, transporte, sesión, presentación y aplicación. Este modelo se explica de manera más detallada en el siguiente capítulo.

Por otro lado la arquitectura del modelo TCP/IP descrita por Zhin está conformado por cinco capas.

- La capa física: La capa física define la manera en que se transmite al más bajo nivel, es decir, secuencias de bits codificadas en alguna forma de señal (generalmente electromagnética) en lugar de paquetes de datos, a través de un enlace físico que conecta los nodos en la red.
- La capa de enlace: Transfiere datos entre nodos de red adyacentes en una red de área amplia o entre nodos el mismo segmento de red. Se asegura de que se hizo la conexión, divide la salida en tramas de datos y maneja la confirmación de que los datos llegaron correctamente. Los protocolos de la capa de enlace incluyen protocolo punto a punto (*PPP*), *Ethernet v2* para redes de área local, *HDLC* son los protocolos principales de la capa de enlace.
- La capa de red: La capa de red es responsable de la entrega de datos de la fuente al destino e incluye ruteo a través de *host* intermediarios. Además de *IP*, para TCP/IP se encuentran *ARP*, *RARP*, *ICMP* e *IGMP*.
- La capa de transporte: La capa de transporte es responsable del encapsulamiento de los bloques de datos en segmentos de datos adecuados para su transmisión al *host* destino. En la capa de transporte se encuentra el protocolo *TCP* que brinda transmisión de datos confiable y control de la congestión; y el protocolo *UDP* que brinda semántica de mejor esfuerzo en la entrega de paquetes.
- La capa de aplicación: La capa de aplicación implementa la lógica de las aplicaciones de usuario, es decir, es la capa que genera la información útil que es intercambiada por medio de la red.

Las redes típicas no se desarrollan normalmente a partir del protocolo OSI, éste sólo se utiliza como guía y en la realidad se utiliza el modelo TCP/IP.

Para poder observar el comportamiento de los diferentes protocolos que componen un sistema de comunicaciones se utiliza un analizador de protocolo que como se definió

anteriormente, es un dispositivo que escucha el tráfico de red. Un analizador de protocolo simple (ver Figura 2.2) está conformado por tres partes: módulo de captura de datos, módulo de análisis de protocolo e Interfaz de usuario. El módulo de captura de paquetes toma la responsabilidad de capturar el tráfico de red que puede ser escuchado por la tarjeta de red Ethernet, éste alimenta al módulo analizador de protocolo con los paquetes capturados para el análisis. Finalmente, la interfaz de usuario despliega los resultados.

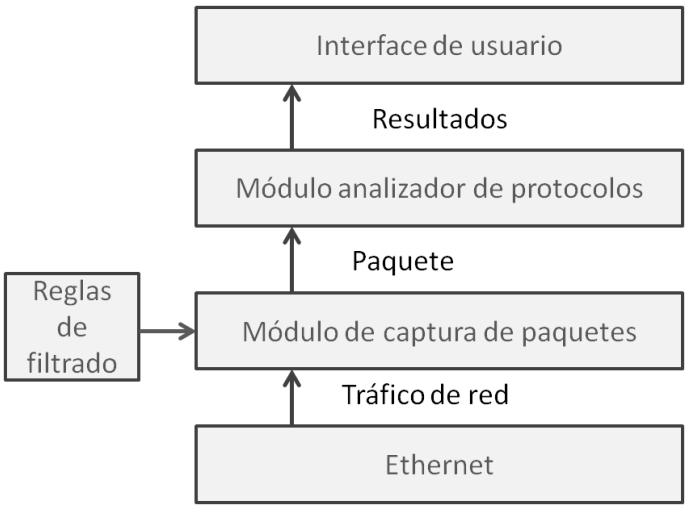


Figura 4 Arquitectura de un analizador de protocolo.

Este modelo es similar en su arquitectura a las funciones básicas de un sistema de red forense (captura, recolección y análisis). A continuación se describe el funcionamiento de cada módulo con el fin de dar una idea general de lo que significa llevar a cabo estas tareas.

2.2.2.2 Módulo de captura de paquetes*:*

El módulo de captura de paquetes es la parte más importante del analizador de protocolo. La librería Libpcap provee una implementación de acceso independiente a la captura de paquetes subyacente que provee el sistema operativo. Esta librería se implementaba originalmente en plataformas Linux/Unix y después se exportó a la

plataforma Windows como WinPcap. Con estas librerías se lleva a cabo la captura de paquetes (ver Figura 2.3).

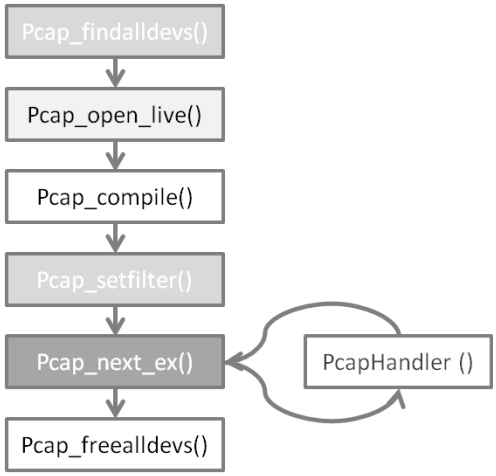


Figura 5 Módulo analizador de protocolos con Libpcap.

A continuación se describe cada paso del módulo analizador de protocolos.

- La función pcap_findalldevs, obtiene todos los dispositivos de red disponibles en el S.O. y de los cuales se puede elegir para escuchar el tráfico de red. .
- Pcap_open_live, se utiliza para abrir el dispositivo elegido en el paso anterior, en modo promiscuo, para poder escuchar todo el tráfico de red. .
- Después de elegir el tipo de filtrado: IP, protocolo, etc. se hace uso de la función pcapcompile y setfilter para probar que los filtros sean correctos y establecerlos antes de que comience la captura.

Las funciones pcap_next_ex y packetHandler, se encargan de llevar a cabo la captura y proceso del paquete. Se toman uno a uno los paquetes que llegan a la tarjeta de red, se aplican los filtros, si pasan el filtro, se llama a la función packetHandler que tiene el procedimiento que se le aplica a cada paquete.

Finalmente la función pcap_freealldev, cierra el o los dispositivo de red. .

2.2.2.3 Módulo analizador de protocolo:

Este módulo está a cargo de analizar los paquetes capturados, se puede ver como una imitación del manejador de procesos de la pila de red Figura 2.4.

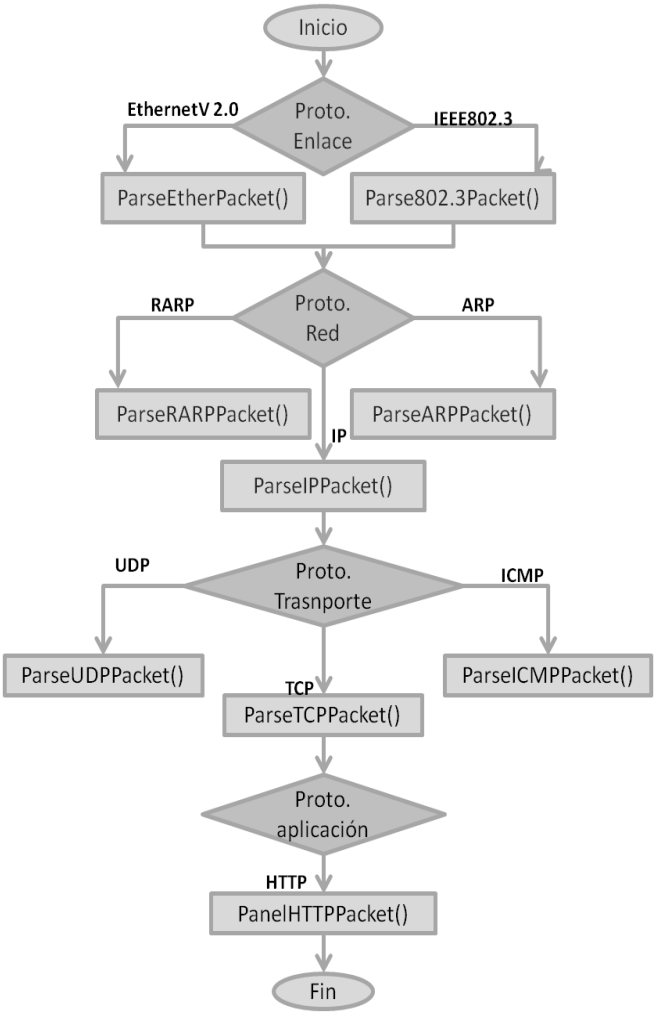


Figura 6 Diagrama de flujo del analizador de protocolos.

El proceso de análisis inicia desde la trama de Ethernet en la capa dos del modelo OSI (capa de enlace). Hay dos estándares para el formato de la trama Ethernet: Ethernet versión 2 y IEEE802.3. Se distinguen por el valor del campo Tipo en la trama de cabecera. De acuerdo al valor del campo Tipo, se libera la carga útil de la trama Ethernet al manejador de protocolo apropiado en la capa tres (capa de red), tal como IP, ARP y RARP por nombrar algunos.

En la etapa final, se extrae e interpreta el valor de cada campo en el encabezado del protocolo respectivo, en el paquete capturado y se llena el campo correspondiente en la estructura del programa diseñada para este fin. La estructura con los datos recolectados se pasa al módulo de interfaz para hacer el desplegado de la información.

Este tipo de herramienta se utiliza estableciendo la tarjeta de red en modo promiscuo para monitorear la actividad en la red. Con el incremento en el tráfico de red, aumenta la tasa de paquetes que se recibe en el nodo [15]. Cuando los paquetes llegan a la tarjeta de red (NIC), tienen que ser transferidos a la memoria principal para procesamiento. Un solo paquete se transfiere por el bus. Se sabe que el bus PCI (Interconexión de componentes periférico) tiene actualmente una trasferencia de no más de 40 a 50 Mbps porque un dispositivo puede tener control a través del bus por cierta cantidad de tiempo, después de esto se transfiere el control del bus. Otra cosa que se conoce es que el componente más lento de una PC es el *driver* del disco, por lo tanto, el cuello de botella se presenta al escribir los paquetes al disco en redes sensibles al tráfico. Por lo anterior, es de suma importancia desarrollar herramientas sumamente eficientes que sean capaces de seleccionar los paquetes relevantes para que solamente esos sean almacenados en el disco.

Por otro lado, es importante remarcar que el tipo de análisis que hacen estas herramientas es identificar el protocolo al cual pertenecen los datos enviados, pero no proporcionan información acerca de actividades sospechosas en la red. Para obtener datos que proporcionen información acerca de actividades delictivas, se necesitan llevar a cabo mecanismos de inferencia sobre la información que entrega un analizador de protocolos.

2.2.3 Captura de datos

La captura de datos tiene algunas variantes que se abordan en el trabajo titulado: “An Introduction to Data Capturing” [16], esta es la razón por la que se revisan los puntos de interés para el desarrollo de la tesis actual.

La captura de datos es de gran utilidad para la administración y el análisis de redes.

Administración de redes: Se pueden obtener los parámetros para la administración de la red analizando los datos capturados.

Análisis de la red: Se pueden hacer gran cantidad de análisis sobre los datos capturados. Análisis de protocolos, análisis de problemas de la red, así como sistemas de detección de intrusos.

2.2.3.1 Implementación de la captura de datos

Por omisión, el modo de trabajar del adaptador de red es que desecha todos los paquetes que tienen una dirección IP destino diferente a la suya. Si cambiamos el modo por omisión del adaptador, entonces se establece en modo promiscuo y se puede implementar la captura de datos. En algunos casos este método es suficiente; sin embargo, existen algunos entornos como veremos a continuación, en que cambiar el modo del adaptador no basta.

- LAN de difusión: Una LAN de difusión usualmente está conformada por un concentrador (*hub)* al que se conectan varios nodos. Cada paquete es difundido en la LAN de difusión. Cuando hay una difusión, el adaptador de red compara la dirección destino del paquete con la suya, si no es igual, el paquete es ignorado. En la captura de paquetes se busca recibir todos los paquetes de la red, para conseguirlo el adaptador se cambia a modo promiscuo. Cuando está en modo promiscuo, el adaptador pasa a capas superiores todos los paquetes que decodifica.
- Switch LAN: Cuando hablamos de un conmutador LAN (switch LAN), el método que se describe arriba, no funciona. Los paquetes que se transfieren sobre un

conmutador LAN no lo hacen por difusión, sino que están limitados por un puerto. Cuando un paquete se envía de un nodo a un conmutador, el conmutador revisa la dirección destino del paquete y lo envía directamente a un puerto específico de acuerdo a la dirección destino del paquete. Basándose en el buffer ARP, el conmutador decide hacia dónde enviar el paquete, por lo que no se puede hacer la capturar de datos aunque el adaptador de red esté en modo promiscuo. Para resolver este problema existen dos métodos que son comunes: el primero es usar un puerto espejo, que es un puerto especial que puede duplicar uno o más datos de otro puerto para él. La mayoría de los conmutadores tienen un puerto espejo que se puede utilizar para hacer la captura de datos. La segunda forma es atacando el conmutador con alguno de los métodos más conocidos: MAC Flooding o Fraud ARP.

  - MAC Flooding: El conmutador mantiene un buffer de MAC dinámico que es una tabla de relaciones entre los puertos del conmutador y la dirección MAC de los adaptadores de red. Después de la inicialización la tabla y los registros que son consultados por el conmutador están vacíos. Este es el buffer en el que el conmutador puede unir el puerto con la dirección MAC. Pero la memoria del conmutador es limitada. Por lo tanto, si se envían al conmutador los paquetes suficientes con la dirección MAC por defecto, entonces el buffer se va a desbordar y el conmutador deja de cumplir con su tarea principal y regresa a la forma de difusión. Ahora tenemos el primer caso, una red de difusión y se pueden capturar los datos.
  - ARP Fraud: De acuerdo al protocolo TCP/IP y Ethernet, un remitente busca su tabla ARP para obtener la dirección MAC destino de la dirección IP destino antes de que de que envié el paquete. Si no puede encontrar el registro localmente, envía una difusión del paquete de solicitud ARP a la LAN y la tabla ARP local se actualiza con información ARP de respuesta. Entonces, se captura el paquete de solicitud ARP y se regresa un paquete de respuesta ARP que reemplaza la dirección MAC del Gateway por la dirección MAC actual y se pueden obtener los datos.

- Wireless LAN: IEEE802.11b WLAN usa el método CSMA/CA el cual se parece a CSMA/CD usado en Ethernet. El adaptador de red inalámbrico se puede fijar en modo promiscuo pero existe una pequeña diferencia, el adaptador de red

inalámbrico puede capturar las tramas Ethernet de IEEE802.11b, pero no puede obtener las cabeceras de la trama 802.11b. Sin embargo el adaptador de red inalámbrico tiene otra forma de trabajar diferente del modo normal y el modo promiscuo, el cual se conoce como modo monitor radio frecuencia (RF). Cuando el adaptador de red trabaja en este modo, puede capturar todos los datos en el BSS (*Basic Service Set*, conjunto de servicios básicos). Por lo tanto, si se quiere capturar todos los datos, se pone el adaptador de red inalámbrico en modo RF monitor.

- LAN remota: La captura de datos en una LAN remota se puede llevar a cabo utilizando los métodos que se han descrito con anterioridad, respaldarlos y traerlos al destino requerido. Si los resultados son muy grandes para transferirlos, se puede llevar a cabo un análisis antes para seleccionar los datos útiles y transferir solo los resultados después de analizados. Se debe tener un nodo remoto para realizar la captura de datos que por lo general es una computadora controlada a la que se fija el adaptador de red a modo promiscuo. En el peor de los casos se tendrán algunos ataques para lo que se pueden implementar medidas de cifrado para evitar la vulnerabilidad de datos. La importancia de la seguridad en Intranet es más rígida para hacer la captura de datos en LAN remotas.

2.2.3.2 Motores de captura de datos

Se pueden desarrollar aplicaciones de captura de datos basados en los motores de captura de datos soportados por el sistema operativo. Los procesos y la estructura de los motores de captura de datos se parecen pero hay pequeñas diferencias entre ellas. Los procesos de motor incluyen proceso de adaptador de red, proceso de filtrado y proceso de aplicación de usuario. Esto es a nivel *kernel* y a nivel usuario.

Los usuarios podrían estar interesados en ciertos datos del paquete, tal vez sólo requieren conocer la actividad de ciertas IP's o la actividad de un protocolo particular. Para obtener este tipo de información de entre todos los paquetes recibidos, se pueden escribir reglas de filtrado que cambien las instrucciones de filtrado del *kernel* para filtrar los datos a nivel *kernel*. El proceso de filtrado sucede después de que el adaptador de red obtuvo los datos y antes de que los datos sean transferidos a la aplicación de usuario.

Generalmente el filtrado y los motores de captura constituyen los motores de captura de datos. Los más famosos son BPF [16] (Berkeley Packet Filter) y NPF [17] (Network Packet Filter) que se utilizan en sistemas Unix y Windows.

Sistemas Unix y Linux: La estructura de BPF tiene dos componentes principales [16] Figura 2.5: the *network tap*, y los filtros. *The network tap*, obtiene copias de los paquetes que pasan por el dispositivo de red y los entrega a las aplicaciones que se encuentran escuchando. Los filtros deciden si un paquete se acepta o no y cuantos bytes del paquete se van a copiar a las aplicaciones escuchando.

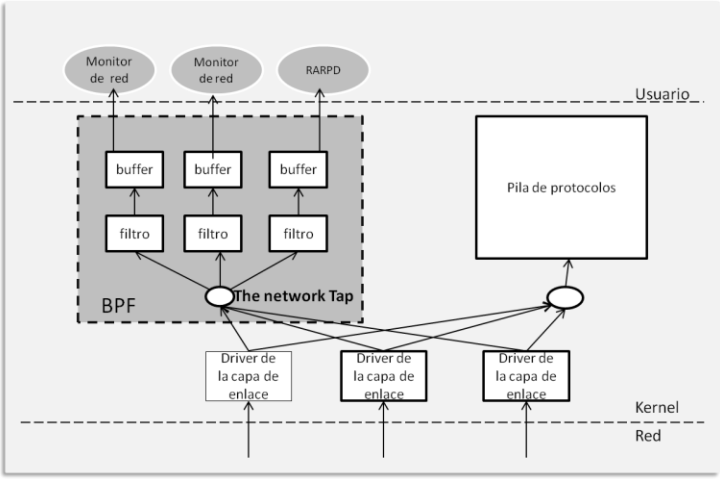


Figura 7 Diagrama de flujo del analizador de protocolos.

Cuando un paquete llega a la interfaz de red, el controlador (*driver)* del dispositivo de nivel de enlace lo envía a la pila de protocolos pero en este caso, BPF primero hace una copia a cada filtro definido. Por cada filtro que acepta el paquete, BPF copia la cantidad de bytes solicitada, después, el controlador recupera el control. Si el paquete no está direccionado al *host* local, el hilo de ejecución vuelve al controlador de la interrupción.

Libpcap provee una interfaz entre los motores de captura de datos y las aplicaciones de usuario. BPF está basado en el modelo de filtro de red del sistema BSD (Berkeley Software Distribution es un sistema operativo Unix). Éste utiliza aritmética que está basada en registros, es decir, no utiliza la pila de memoria si no un buffer, por lo que

su eficiencia está muy optimizada [20]. El programa de filtrado consiste principalmente en las reglas de filtro definidas por el usuario y son cargadas en la maquinaria de captura. Si los datos de los atributos se igualan con la regla, entonces son transferidos a la aplicación de usuario, de otra manera se desechan.

BPF está embebido en el sistema operativo y exporta una interfaz por medio de Libpcap para que los usuarios desarrollen librerías de enlace dinámico. Libpcap esconde detalles de operación interna entre el sistema operativo y la aplicación de usuario.

Sistema Windows: NPF es una migración del sistema Unix al sistema Windows. Éste hereda los filtros, dos niveles de *buffer* (*buffe*r del *kernel* y *buffer* de usuario) y las librerías a nivel usuario.

NPF proporciona más funciones que BPF. Tiene cinco partes: *network tap*, filtros, motores estáticos, descarga de motores e interfaz a nivel usuario. NPF se utiliza principalmente en sistemas Windows pero no está embebido en el sistema Windows por lo que se debe de instalar el paquete del sistema NPF.

La captura de datos actualmente enfrenta muchos desafíos que no se abordan en el trabajo de investigación de este documento. Sin embargo, se hace uso de los motores de captura en Linux, la estructura BPF, Libpcap es la herramienta que se utiliza para llevar a cabo la comunicación entre éstos y las aplicaciones desarrolladas.

Otro elemento importante analizado en esta sección es el tipo de red y los requerimientos para llevar a cabo la captura.

2.2.4 Ksensor

En 2007 se propuso Ksensor [18] un marco de trabajo (*framework)* a nivel *kernel* que tiene como objetivo mejorar el monitoreo en la red tomando en cuenta el monitoreo de redes a alta velocidad y explotando el paralelismo que hoy en día está disponible en las estaciones de trabajo. La primera característica del marco de trabajo es que el módulo de procesamiento está a nivel *kernel* del sistema operativo, creando ejecución específica de hilos. El sistema de configuración y el módulo de administración de resultados se encuentran a nivel usuario. La comunicación entre el espacio de usuario y *kernel* se lleva a cabo por un módulo diferente llamado *driver*. Se definen tantas instancias de ejecución

como números de procesadores sobre la arquitectura de *hardware*. Cada hilo pertenece a una instancia de ejecución del sistema (captura y análisis) y siempre está ligado al mismo procesador.

La captura de datos se realiza dejando una instancia de captura de paquetes de una NIC, ejecutándose en un procesador (afinidad de interrupción), mientras el otro procesador analiza paquetes. Esta alternativa brinda mayor eficiencia que si todo el trabajo se llevara a cabo de la misma manera en los dos procesadores, es decir, que todas las instancias capturen paquetes y la etapa de análisis se lleve a cabo en el tiempo restante.

El paradigma para procesamiento paralelo que se usa es a nivel paquete. Este paradigma sugiere una cola de paquetes compartida por todas las instancias de análisis. Esta forma se eligió por dos razones. Por un lado la cola protege al sistema de las ráfagas de tráfico (tráfico muy intenso, la mayoría de los paquetes pueden ser almacenados temporalmente durante la ráfaga). Por el otro lado, esto permite al sistema controlarse a sí mismo, es decir, una instancia puede analizar más o menos paquetes que otra, porque simplemente los toma de la cola global. Para evitar la congestión, se hace uso de un umbral máximo y un umbral mínimo. El umbral máximo es el tamaño de la cola. El umbral mínimo no es muy grande porque si una ráfaga de tráfico permanece por mucho tiempo, la cola se podría llenar rápidamente. Tampoco es muy pequeño porque si la cola se llena con una ráfaga de tráfico pequeña, la captura podría reactivarse más tarde perdiendo todos los paquetes que llegaron en ese intervalo.

Cuando se llega al umbral máximo, el mecanismo desactiva las funciones de análisis con el objetivo de reducir el consumo de recursos y lograr una situación estacionaria tan pronto como sea posible, mientras la cola de paquetes se vacía rápidamente. Un controlador (*driver*) ofrece la interfaz de acceso a memoria que permite a los módulos operar sin cambio, da un conjunto de funciones para obtener los resultados, envía órdenes o verifica el estado del sensor.

En Ksensor se propone un diseño que mejora el rendimiento en el sistema de tráfico de red para la calidad de servicio de monitoreo, se reduce el impacto de recibir situaciones de *livelock* (El sistema utiliza todas las interrupciones de tiempo de procesamiento excluyendo otras tareas necesarias), [22] en redes de alta velocidad y permite mantener un rendimiento razonable.

2.2.5 Henpa

El objetivo de HENPA [19] es tomar la salida de un sniffer (herramienta de captura de tráfico de red) y procesar los datos para extraer evidencia forense potencial. El marco de trabajo propuesto tiene las siguientes propiedades:

1. Enfoque forense.
2. Marco de trabajo extensible.
3. Identificar todo el contenido de los paquetes HTTP dentro de los paquetes TCP.
4. Modelo de limitada habilidad compartida.

El marco de trabajo HENPA da a los desarrolladores de *plug-ins* el control completo sobre el formato de entrada y salida para la aplicación. En el alto nivel, se necesita presentar diferentes tipos de información en diferentes formas. La información directa es la información contenida dentro del paquete. Información tal como la IP del remitente y el destinatario, número de puerto, etc., se puede usar como fuente de evidencia.

Agregada a esta información directa que se encuentra dentro del paquete, hay información de inferencia. Esta información está definida como información que no está contenida actualmente dentro del paquete pero se puede encontrar buscando entre muchos paquetes por patrones interesantes. Este es el tipo de información que promete mucho en la recolección de evidencia en contra del sospechoso. Es el tipo de información más difícil de extraer ya que puede involucrar información combinada entre múltiples protocolos no relacionados de paquete y búsqueda de patrones para identificar acciones complejas de usuario. La flexibilidad de la arquitectura que provee el marco de trabajo HENPA facilita el desarrollo de algoritmos complejos para extraer evidencia significativa.

2.2.5.1 Tecnología de intersección

Existen cientos de protocolos usados en Internet. Ethereal (un analizador de protocolos conocido actualmente como Wireshark) pudo identificar más de 750

protocolos. Para poder reconstruir los datos recolectados, se debe tener un alto conocimiento de cómo se comunican las computadoras y de los protocolos que maneja. El beneficio de automatizar el análisis es que el conocimiento solo se aplica en el desarrollo del sistema que va a automatizar el proceso. Después de que el sistema se desarrolla, se puede utilizar repetidamente.

La secuencia de pasos para extraer la información de paquetes en forma general para HENPA son:

1. Identificar paquetes de interés.
2. Organizar los paquetes dentro de grupos lógicos secuenciales.
3. Extraer información deseada.

Este proceso esencialmente sigue el procedimiento estándar forense de colección, organización, análisis y presentación de evidencia forense.

La gran cantidad de datos que pueden estar involucrados exige almacenar los datos en una forma que es lógica y minimiza el tiempo de procesamiento. Con este fin en HENPA se propone manejar una estructura de dos listas doblemente ligadas bidimensionales. Cuando llega un paquete se extrae la dirección IP fuente y destino, así como, el número de puerto. Entonces, se escanea el nodo cabeza del flujo de paquetes, si se encuentra con el mismo valor fuente y destino, entonces el paquete se pone en la lista ligada bajo el nodo cabeza. Si uno no se encuentra, entonces un nodo cabeza nuevo se crea y se pone en la lista. Esto asegura que se utiliza el mínimo procesamiento de paquetes de datos. Una vez que todos los paquetes son almacenados en este formato el flujo de datos individual se puede analizar.

2.2.5.2 Operación de captura de paquetes

Cuando la tarjeta de red está en modo promiscuo, el *host* recibe todos los paquetes de la red y los debe almacenar en el disco para hacer el procesamiento más tarde. Una de las dificultades de utilizar un *sniffer* es la pérdida de paquetes. Un sniffer debe de poder recibir un paquete, guardarlo y regresar para escuchar antes de que llegue otro paquete. Cuando la cantidad de paquetes es muy grande el *sniffer* se esfuerza, para

recibir y guardar todos los datos de los paquetes que recibe, pero algunas veces hay pérdida de paquetes. La cantidad de paquetes que pierde se relaciona con el sistema operativo sobre el cual está instalado el *sniffer*, la velocidad del procesador y la cantidad de tráfico de la red. Existen varias propuestas para resolver este problema y se ha llegado a disminuir la pérdida de paquetes, sin embargo, es un problema que no se ha resuelto totalmente.

2.2.5.3 Arquitectura

HENPA se puede dividir en cuatro subsistemas principales:

- Núcleo: Contiene el controlador HEPNA y cualquier otra clase asociada con la carga de *plug-in* externos dentro del *framework*.
- Interfaz grafica de usuario (GUI por sus siglas en inglés): Contiene los controladores de GUI y cualquier otra clase requerida para generar la GUI para el usuario.
- Almacenamiento: Contiene todas las clases que manejan el almacenamiento interno de paquetes de datos y relacionados a la información. Incluye: PacketSet, Packet and Stream.
- Analizador: Contiene todo de las clases que manejan el aspecto extensible del *framework*. Este incluye el PacketParserController y todas las clases que implementan el PAcketPArserController y todas las clases que lo implementan y clases abstractas FileParserTemplate.

En el diseño del *software* se utilizó un número de patrones de *software*. Un patrón de *software* es una descripción de un problema y una solución genérica que se puede utilizar de diferentes formas. En el diseño del GUIcontrolador, PacketParserController y las clases PacketSet se utilizó el patrón fachada. Cada una de estas clases actúa como puntos de entrada a sus respectivos subsistemas. El patrón de induración se implementó en la clase HEPNAController. Todas las interacciones iníciales entre los diferentes subsistemas tienen lugar a través de HEPNAController. De esta forma se reduce la unión entre los diferentes subsistemas y por lo tanto hace más fácil agregar, modificar y eliminar clases del *framework*.

Los aspectos extensibles de la arquitectura se hicieron utilizando el Polimorfismo y estrategias de diseño de patrones de *software*. El uso de estos patrones permite al análisis de paquetes y archivos compartir alguna funcionalidad básica que todos los análisis requieren, pero todavía le permite al *framework* reconocer la diferencia entre diferentes análisis y utiliza la funcionalidad individual codificada en sus clases.

Las clases abstractas PAcketPArserTemplate y FileParserTemplate son las dos clases que se utilizan para implementar las propiedades de extensibilidad del *framework*. La clase Packet representa un solo paquete de red extraído del archivo dump. Éste almacena los paquetes de datos y cualquier funcionalidad asociada. El objeto Stream contiene listas de los paquetes relacionados. La clase PacketSet es el punto de acceso a la estructura de almacenamiento de datos interno del subsistema. Éste mantiene la lista de todos los paquetes y flujos y da el plug-in al desarrollador de estos objetos.

2.2.6 Información Forense de la capa de aplicación obtenida en el análisis de paquetes

El trabajo presentado por Luo [21] et. al., se enfoca en obtener información de la transmisión de datos basada en servicios Web. En su trabajo, Luo refiere que el departamento de monitoreo de redes de la oficina de seguridad pública de Xuzhou intercepta decenas de miles de paquetes de datos sobre la entrada y salida de redes externas todos los días, pero sólo una pequeña cantidad de información puede ser extraída. Para los datos en la capa de aplicación tal como *e-mail* mensajes de texto plano, contenido de video específico, *software* y descarga de documentos, no se puede extraer la información. Gran cantidad de paquetes de datos ocupan una gran cantidad de espacio, pero la utilidad de esos datos es muy poca, lo cual hace difícil el trabajo para el departamento de monitoreo de redes. Para ayudar a la resolución de este problema se propone un modelo forense para la capa de aplicación de la pila de protocolos de Internet.

2.2.6.1 Modelo forense

Con miras en la sesión de aplicación de servicio de red basado en el protocolo de Internet, se proponen una variedad de tecnologías tal como captura de paquetes de datos, filtrado de paquetes y aplicación para regeneración de datos. Estas tecnologías se utilizan para proponer e implementar un modelo forense de información en la capa de aplicación basado en el análisis de paquetes de datos de la red (ver Figura 2.6).

Cuando se monitorea el tráfico de la red, se utiliza el mecanismo de captura de paquetes de datos basado en WinPcap y se extrae la información básica de los paquetes irregulares tal como dirección IP fuente, dirección IP destino y número de puerto. Se diseñan las correspondientes reglas de filtrado y se filtran intencionalmente los paquetes irregulares. Después, se hace el análisis de protocolo y regeneración de datos para los paquetes recibidos. De manera recurrente la sesión de aplicación, extrae y reporta la información en texto plano de la capa de aplicación al modelo de análisis forense, para hacer comprensivo el análisis junto con el estatus del tráfico de la red, dirección IP, número de puerto y otra información. Finalmente, genera la evidencia digital de la red.

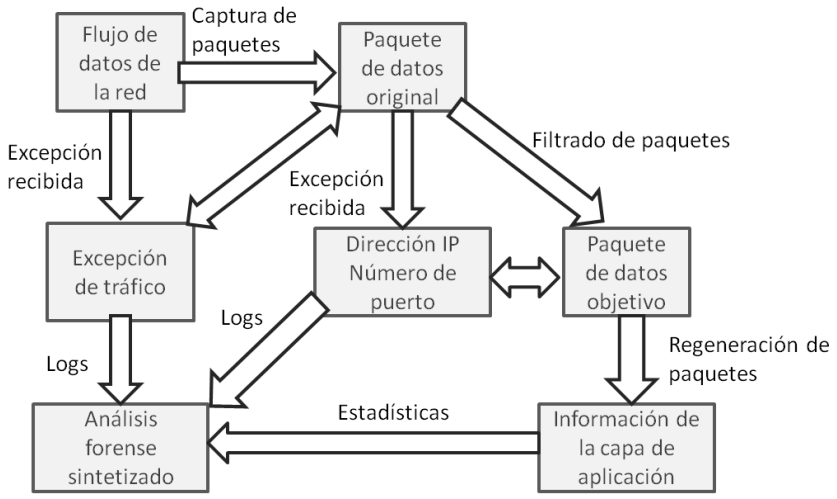


Figura 8 Información Forense de la capa de aplicación basada en el análisis de paquetes.

2.2.6.2 Captura de paquetes

El programa de captura se logra por medio del controlador de captura de paquetes que se ejecuta en el *kernel* de Windows, el cual no utiliza un sistema recolector especial, pero captura paquetes de datos con la tarjeta de red. En esta propuesta eligieron la arquitectura WinPcap como se ha mencionado antes.

WinPcap consta de un *driver*, que se extiende del sistema operativo para proveer acceso a la red en bajo nivel, y una librería que se utiliza para acceder fácilmente a las capas de bajo nivel de la red. Se utiliza para acceder a la capa de enlace de datos con estructuras extendidas.

WinPcap esconde los detalles de interacción entre el programa de usuario y el *kernel* del sistema operativo, que no tiene nada que hacer con el adaptador y es independiente de la forma concreta de la red.

La implementación del programa de captura de paquetes de datos es muy simple, se pone la tarjeta en modo promiscuo, para obtener una copia de todo el tráfico de que pasa por la tarjeta de red. Esta tarjeta solamente necesita monitorear los paquetes de datos con el puerto destino en la red, se determina si la IP del paquete es la misma que el *host* necesita monitorear más rigurosamente, si es así, verifica si hay archivos nombrados después de especificar la IP, para que dinámicamente dé nombre a los archivos y almacene los paquetes de datos. Si la IP no es la que busca, almacena los paquetes de datos o escoge desecharlos.

En la práctica, un mecanismo de información eficiente es un componente importante del sistema, que le permita al usuario especificar un segmento de red particular y un tipo de protocolo. Solamente los datos concernientes al usuario se van a reportar a las capas superiores a fin de mejorar la eficiencia del trabajo del sistema.

Con el uso de WinPcap lo más importante es definir correctamente las reglas de filtrado. La identificación del paquete archivado indica la posición del paquete de datos en el flujo de datos, lo cual también muestra la localización del paquete en los datos de la aplicación. Por lo tanto, de acuerdo a la identificación del paquete de datos archivado, una colección de paquetes se puede recombinar para obtener los datos intercambiados por los procesos que implementan la aplicación.

2.2.6.3 Aplicación de regeneración de datos

De acuerdo al campo de identificación de los paquetes, una colección de paquetes se puede recombinar para mostrar los datos en una aplicación.

La capa de enlace tiene una característica llamada máxima unidad de transmisión, que limita la longitud máxima de la trama de datos. Si el paquete es más grande que esta cantidad, es necesario para la capa IP fragmentar el paquete de datos de tal manera que el tamaño de cada fragmento sea menor o igual que la cantidad máxima establecida.

2.2.6.3.1 Ordenación de paquetes Out-of-order

Cuando ordenamos los paquetes, el sistema establece una cola de buffer, que tiene un tamaño máximo igual a la ventana deslizante (El número de segmentos que se espera recibir antes de que se espere confirmación).

El sistema compara el número de secuencia del paquete p con el número de secuencia del paquete que debería ser recibido q. Si p = q, el paquete p se introduce en la cola de paquetes recibidos. Después, el sistema buscará el paquete p + 1 el cual conoce las condiciones en las que se lleva a cabo la separación de la cola del buffer. Si el paquete p + 1 existe se retira, si no, el sistema continuará recibiendo paquetes de datos nuevos. Si p ≠ q el paquete p se introduce en la cola del buffer. Entonces el sistema comparará el número de los paquetes de datos recibidos con el número de los paquetes de datos que podrían ser recibidos y determina el número de secuencia necesario del siguiente paquete.

2.2.6.3.2 Iniciativa de fijación de paquetes

Con el objetivo de determinar el punto de inicio y el punto de terminación de aplicación de datos, se debe de examinar la capa de aplicación de datos. Si se toma como ejemplo HTTP, ésta tiene dos tipos de contenido de datos. Uno es de *request data* y el otro tipo es de *response data*. Para el *request data* si al inicio el fragmento de datos contiene "GET", "POST", "HEAD" o "HTTP", éste se puede determinar como el fragmento

de "initiative data", pero para el fragmento *data response*, solo se determina por el "HTTP".

El *request data* y *response data* tienen cada uno dos formas de determinar el final. Uno aplica a los dos: Si el contenido de los datos contiene un campo "Content-length", el sistema puede sacar la cantidad contenida y el final del fragmento de datos se puede determinar. Si el *request data* no contiene el campo "Content-Length" y dos marcas "CRLF" se pueden considerar como marcas de fin. Mientras que en *data response* el final se puede determinar si el fragmento de datos contiene al final el símbolo "FIN".

Todo el tiempo que se obtiene un paquete, WinPcap reportará su información usando un puntero "unsigned char *" apuntando a la sección de la zona de *buffer*. Posteriormente lee los primeros 14 bytes del *buffer* y estructura un objeto del encabezado de la trama de Ethernet basado en el formato de la trama de Ethernet. De acuerdo al campo "Type" del encabezado de la trama de Ethernet, decide cuál es el siguiente protocolo. Entonces, construye un objeto que define el siguiente protocolo. Este proceso se repite para cada capa del modelo TCP/IP, construyendo objetos en concordancia con el formato del paquete.

La realización del programa de recombinación es principalmente el reemsamble del paquete TCP. Primero, selecciona el paquete TCP y produce un objeto IP y un objeto TCP para éste y después revisa todos los paquetes en turno. Generalmente, cada puerto de archivo es diferente en la transmisión TCP. Por lo tanto, si encuentra un paquete fuente, la dirección destino, dirección fuente y puerto son los mismos al paquete seleccionado y la longitud de datos es más grande que 0, determina si el paquete seleccionado y el paquete encontrado pertenecen al mismo archivo, produciendo un objeto "Found" para éste y lo une dentro de una cola "Found" (el índice de este paquete determina la posición actual en la cola; "start" es el desplazamiento desde la posición donde inician los datos hasta el encabezado del paquete; "len" es la longitud de los datos). Después de que se revisaron todos los paquetes, los toma uno por turno en la cola "Found", descarta los encabezados, solamente deja la parte TCP del paquete. Finalmente, conecta todos los datos en un buffer *zone*.

2.3 Propuesta de solución

Dentro del marco de soluciones se propone ADA**, un sistema que hace uso de filtros muy específicos para el encabezado del protocolo de interés, además de aplicar una función de probabilidad para cada filtro. Esta función determina si la información debe de ser almacenada.

El entorno de red pensado para el sistema propuesto en este documento es una LAN de difusión ya que es el escenario en el que cada host puede estar disponible para recibir todos los datos de red. Si no es así, es decir, si el sistema se distribuye en una switch LAN, nos encontramos con el problema de que cada host sólo recibe lo que está dirigido a la dirección IP que posee, sin tener la posibilidad de obtener toda la información.

El sistema que se propone en este trabajo de tesis no pretende mejorar el rendimiento como objetivo principal, sin embargo se espera que el realizar la captura de los paquetes en un entorno distribuido en lugar de uno centralizado, distribuya la tarea de captura mejorando el rendimiento. A diferencia de Ksensor el trabajo está dirigido al área forense por lo que la captura de paquetes se hará llevando a cabo el establecimiento de filtros muy detallados que permita recolectar la información más útil para una investigación en la escena de un delito.

2.4 Resumen

El cómputo forense involucra una serie de tareas enfocadas a la recolección de evidencia confiable. Para salvaguardar la veracidad de esta información es necesario seguir una metodología que comienza con la verificación del incidente, la descripción del sistema y la adquisición de la evidencia.

Existe otro campo de la tecnología de la cibernética forense conocida como: redes forenses, que trata principalmente con el análisis a fondo de la evidencia de intrusión en redes de computadora.

El concepto de análisis forense está unido al de red forense y para llevarlo a cabo existen diferentes tipos de herramientas. En la Tabla 2.1 se muestra algunas de ellas con sus principales características.

Los datos recolectados para llevar a cabo el análisis pueden ser de diferentes tipos: datos de contenido completo, sesión, alerta o estadísticos. Cada tipo tiene su enfoque, ventajas y desventajas.

El objetivo principal de un análisis de red forense abarca dos problemas fundamentales: identificación del grupo atacante y la reconstrucción del escenario de ataque.

Como podemos ver, el análisis de paquetes de datos es la parte más importante en un sistema de red forense. Los paquetes llegan a la tarjeta de red y son copiados a alguna aplicación para su posterior análisis. El analizador de protocolo sólo nos dice cuáles fueron los protocolos que se utilizaron para enviar el paquete. Existen diversos problemas que se enfrentan al llevar a cabo la captura de datos: el tipo de red es uno de ellos. Para llevar a cabo la captura de datos se mencionan dos tipos de motores de búsqueda: Libpcap para Linux y su versión para Windows, Winpcap.

Ksensor es una herramienta enfocada a mejorar la eficiencia en la captura de datos en redes de alta velocidad y Hepna es un marco de trabajo teórico que podría resolver algunos de los problemas de las herramientas de tráfico de red forense como lo son el presentar la información de manera legible para alguien que no es experto en protocolos de comunicación y también el de extraer la información útil dentro de una gran cantidad de datos, pero aún no se ha implementado.

En el siguiente capítulo se revisan los fundamentos de la arquitectura de red.

CAPÍTULO 3. ARQUITECTURA DE INTERNET

En el capítulo anterior se profundizó en el tema del cómputo forense que es fundamental para el desarrollo de la tesis; en este capítulo se definen otros conceptos que son necesarios para el desarrollo del sistema propuesto.

3.1 Tipos de redes

Existen diferentes tipos de red y los recursos que requiere dependen de la naturaleza de la aplicación, del número de computadores implicados y de su separación física. Existen varias formas de clasificar a las redes de computadoras, sin embargo, hay dos clasificaciones que son las más destacadas. La primera es la tecnología de transmisión la cual divide a las redes en redes de difusión y redes punto a punto.

- Las **redes de difusión** tienen un solo canal de comunicación, por lo que todas las máquinas de la red lo comparten. Si una máquina envía un mensaje corto, en algunos contextos conocido como paquete, todas las demás lo reciben.
- Las **redes de punto a punto** constan de muchas conexiones entre pares individuales de máquinas. Para ir del origen al destino, un paquete en este tipo de red podría tener que visitar primero una o más máquinas intermedias.

La segunda clasificación se basa en su escala, es decir, en el número de nodos que componen a la red, así como en su distribución geográfica. La Tabla 3.1 muestra esta clasificación.

Tabla 3.1 Clasificación de redes basada en su escala.

Distancia entre procesadores	Procesadores ubicados en el mismo	Ejemplo
1 m	Metro cuadrado	Red de área personal
10 m	cubículo	Red de área local
100 m	Edificio	
1 km	Campus	
10 km	Ciudad	Red de área metropolitana
100 km	País	Red de área amplia
1000 km	Continente	
10,000 km	Planeta	Internet

- Las redes de área personal, son redes que están destinadas para una sola persona. Un ejemplo de este tipo de red es una computadora con un ratón, teclado e impresora.
- Las redes de área local (LAN) son redes de propiedad privada que se encuentran en un solo edificio o en un campus de muy pocos kilómetros de longitud, operan a una velocidad de 10 a 100 Mbps tienen un retardo bajo (microsegundos o nanosegundos) y son propensas a errores. Las LAN más nuevas funcionan hasta a 10 Gbps.
- Una red de área metropolitana (MAN) abarca una ciudad.
- Una red de área amplia (WAN), abarca una gran área geográfica, con frecuencia un país o un continente. Los host están conectados por una subred de comunicación.

3.2 Modelo de referencia

La arquitectura de red descompone un problema común dentro de un conjunto de sub problemas que se resuelve de manera independiente por un componente individual de la red. Para llevar a cabo la comunicación se necesita un mensaje, una manera de codificar el mensaje y un medio por el que transmitir dicho mensaje. Evidentemente, la manera de codificar el mensaje ha de ser estándar y ambos interlocutores deben ser capaces de entender y decodificar. Uno de los elementos más importantes que hace posible la interacción en red es el protocolo de comunicación.

Un protocolo es un acuerdo entre las partes en comunicación sobre cómo se debe llevar a cabo la comunicación. En una comunicación en red, se utiliza un protocolo por capa y se conoce como pila de protocolos.

Existen dos arquitecturas importantes: los modelos de referencia OSI y TCP/IP que analizaremos detalladamente a continuación.

3.2.1 Capas del modelo OSI

Los componentes dependientes de la red y los orientados a las aplicaciones (independientes de la red) del modelo OSI vienen implementados en varias capas (ver Figura 3.1)

Cada capa desempeña una función bien definida en el contexto del subsistema de comunicación global, y opera según un protocolo definido intercambiando mensajes, tanto datos de usuario como información de control adicional, con una capa igual correspondiente en un sistema remoto. Cada capa tiene una interfaz bien definida con las capas adyacentes superior e inferior. Este modelo se divide en siete capas:

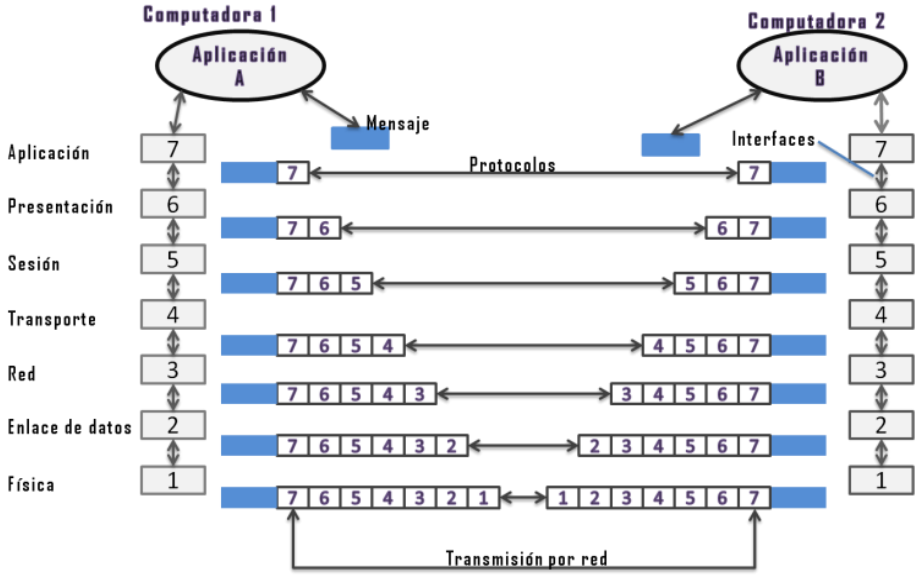


Figura 9 Capas del modelo OSI.

- **La capa física**: En la capa física se lleva la transmisión de bits puros a través de un canal de comunicación. Los aspectos de diseño implican asegurarse de que cuando un lado envía un bit 1, éste se reciba en el otro lado como tal.
- **La capa de enlace de datos:** La tarea principal de la capa de enlace de datos es transformar un medio de transmisión puro en una línea de comunicación que al llegar a la capa de red, aparezca libre de errores de transmisión. Logra esta tarea haciendo que el emisor fragmente los datos de entrada en tramas de datos (típicamente, de algunos cientos o miles de bytes) y transmitiendo las tramas de manera secuencial.
- **La capa de red:** La capa de red controla las operaciones de la subred. Un aspecto clave del diseño es determinar cómo se enrutan los paquetes desde su origen a su destino.
- **La capa de transporte:** La función básica de la capa de transporte es aceptar los datos provenientes de las capas superiores, dividirlos en unidades más pequeñas si es necesario, pasar éstas a la capa de red y asegurarse de que todas las piezas lleguen correctamente al otro extremo.
- **La capa de sesión:** La capa de sesión permite que los usuarios de máquinas diferentes establezcan sesiones entre ellos. Las sesiones ofrecen varios servicios,

como el control de diálogo (dar seguimiento de a quién le toca transmitir), administración de *token* y sincronización.

- **La capa de presentación:** A la capa de presentación le corresponde la sintaxis y la semántica de la información transmitida.
- **La capa de aplicación:** La capa de aplicación contiene todos los protocolos del más alto nivel. Un ejemplo de protocolo en esta capa es http que es la base de World Wide Web.

Los protocolos asociados a este modelo no se implementaron, pero el modelo es ampliamente utilizado en forma teórica [14].

Si una aplicación en una computadora A, se quiere comunicar con una aplicación en una computadora B, hace una solicitud a los servicios de la capa de aplicación. La capa de aplicación forma un mensaje con un encabezado en formato estándar. Después de que la capa de aplicación tiene el mensaje lo pasa a la capa siguiente. La capa de presentación recibe, agrega su información en forma de encabezado y lo pasa. Este comportamiento continúa hasta terminar con la pila OSI en donde se tiene un paquete formado con la información a enviar y todos los encabezados de control (ver Figura 3.2)

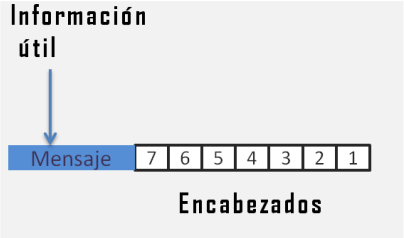


Figura 10 Mensaje con los siete encabezados de control de la pila OSI.

Finalmente, el paquete con la información y los encabezados llega al medio físico de transmisión de red y viaja hasta la capa de red del host destino. El *host* destino recibe el paquete y comienza a leer los encabezados, quitarlos y pasar la información a la siguiente capa de más arriba hasta llegar a la capa de aplicación.

3.2.2 Capas de protocolos de Internet TCP/IP

En un sistema real, el modelo más utilizado es el modelo TCP/IP. Esta pila de protocolos fue desarrollada para Internet, tiene varias ventajas sobre otros protocolos especialmente cuando se comienzan a desarrollar redes que incluyen enlaces WAN. La capacidad de fragmentación de paquetes es una de las más útiles y hace posible utilizar esta pila de protocolos en redes de gran escala.

Como se define en el *RFC* 1122 [21], una computadora o *host*, es el último consumidor de los servicios de comunicación. Un sistema de comunicación en Internet está constituido de redes de paquetes interconectados que soportan comunicación entre computadoras que utilizan el protocolo de Internet. Las redes están interconectadas por puertas de enlace o *gateways,* o por *routers IP.*

Para comunicarse utilizando el sistema de Internet, un *host* debe de implementar el conjunto de capas de los protocolos que comprende la suite de protocolos de Internet. Un *host* debe de implementar al menos un protocolo en cada capa, Figura 3.3.

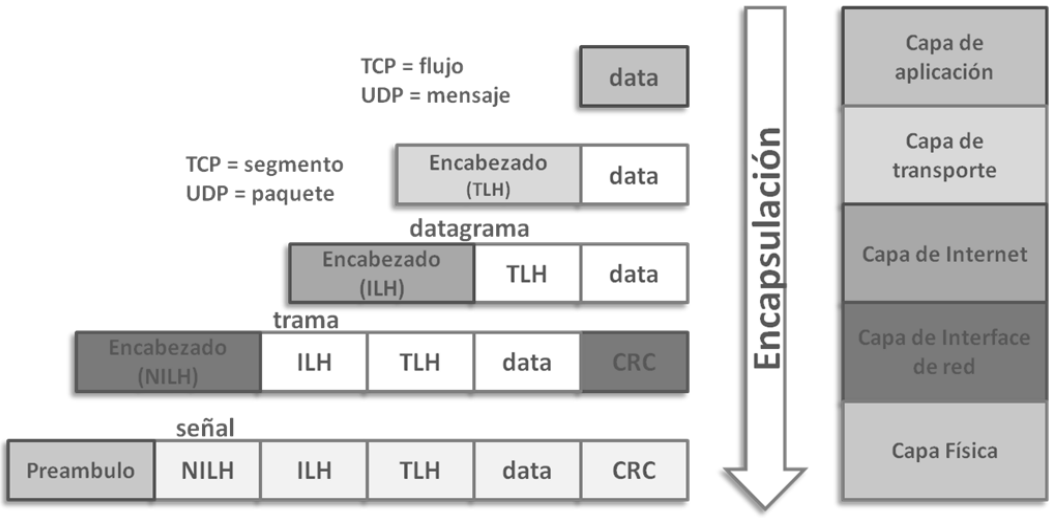


Figura 11 Capas de protocolo de Internet.

La capa de aplicación es la capa de más alto nivel de la suite de protocolos de Internet. Esta capa combina las funciones de las dos capas de más arriba del modelo OSI, presentación y aplicación.

Hay dos categorías de los protocolos de la capa de aplicación: **protocolos de usuario** que proveen servicios a usuarios y **protocolos de soporte** que proveen funciones del sistema comunes.

Los protocolos de usuario más comunes son:

- Telnet (abrir sesión remotamente)
- FTP (Transferencia de archivos)
- SMTP (Entrega de mensajes electrónicos)

Pero los protocolos listados no son los únicos, existen otros protocolos estandarizados y muchos protocolos de usuario privados.

Los protocolos de soporte utilizados para mapeo de nombres de *host*, arranque y administración, incluyen SNMP, BOOTP, RARP y DNS.

La **capa de transporte** proporciona servicios de comunicación *host* a *host* para las aplicaciones. Existen dos protocolos principales en la capa de aplicación, TCP y UDP.

TCP o Protocolo de Control de Transmisión es una interfaz entre los procesos de comunicación por un lado, y por el otro, los protocolos de la capa de nivel más bajo tal como IP. Este protocolo pretende ser utilizado para dar un servicio de conexión confiable y segura entre pares de procesos en un entorno multired [22].

UDP o Protocolo de Datagrama de Usuario asume que IP se utiliza como protocolo fundamental. Este protocolo proporciona un procedimiento para programas de aplicación para enviar mensajes a otros programas con un mínimo de mecanismos de protocolo, es orientado a transacción y no garantiza la entrega o los duplicados de los datos [23].

Capa de red. Todos los protocolos de transporte de Internet utilizan el protocolo de Internet (IP) para transportar los datos de un *host* fuente a uno destino. IP es un servicio de datagramas de entre redes que no da garantía en la conexión. Por lo que los datagramas podrían llegar dañados, duplicados, fuera de orden o no llegar al *host*

destino. El protocolo IP incluye direccionamiento, especificación de tipo de servicio, fragmentación y reensamblado, e información de seguridad.

El *datagrama* o su naturaleza sin conexión, del protocolo de Internet es una característica fundamental de la arquitectura de Internet.

IP, ICMP e IGMP son protocolos de la capa de red.

El protocolo ICMP está basado en mensajes de control que tienen como propósito dar aviso de los problemas del entorno de comunicación, no para hacer IP confiable. [24].

IGMP es un protocolo de la capa de red que se utiliza para establecer grupos de *host* dinámicos para *multicasting*.

En la **capa de enlace** para comunicarse de manera directa dentro una red, un host debe implementar el protocolo utilizado para la interfaz de esa red. En esta capa se considera necesario mencionar el RFC 1042 que es un estándar para la transmisión de datagramas IP sobre redes IEEE 802. La especificación IEEE 802 define una familia de estándares para LANs que trata con la capa física y la capa de enlace de datos. Se definieron los estándares 802.3, 802.4 y 802.5 para la capa física, para la capa de enlace sólo se definió el 802.2. Los estándares para la capa de enlace especifican la capa física del modelo OSI y la subcapa MAC de la capa de enlace. El estándar 802.2 especifica la subcapa LLC de la capa de enlace (ver Figura 3.4).

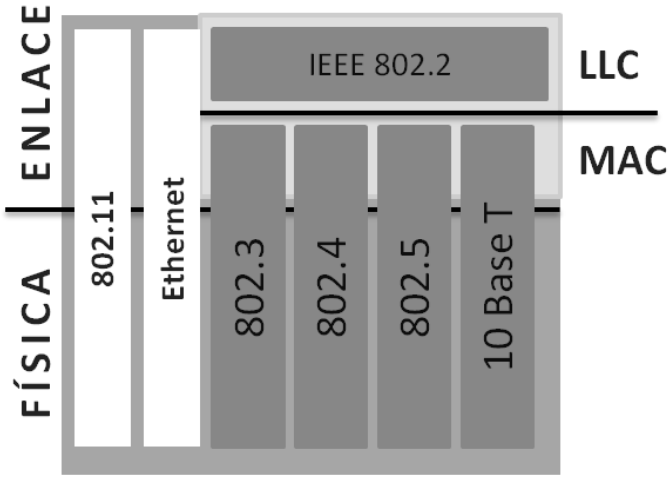


Figura 12 LLC y MAC, subcapas de la capa de enlace.

Este sistema utiliza dos campos LSAP (Link Service Acces Point) del encabezado LLC de manera muy parecida a la manera en que ARPANET utiliza el campo enlace. Más aún, es una extensión del encabezado LLC llamado SNAP (Sub-Network Access Protocol). El formato de encabezado para la capa de enlace es como el que se muestra en la Figura 3.5.

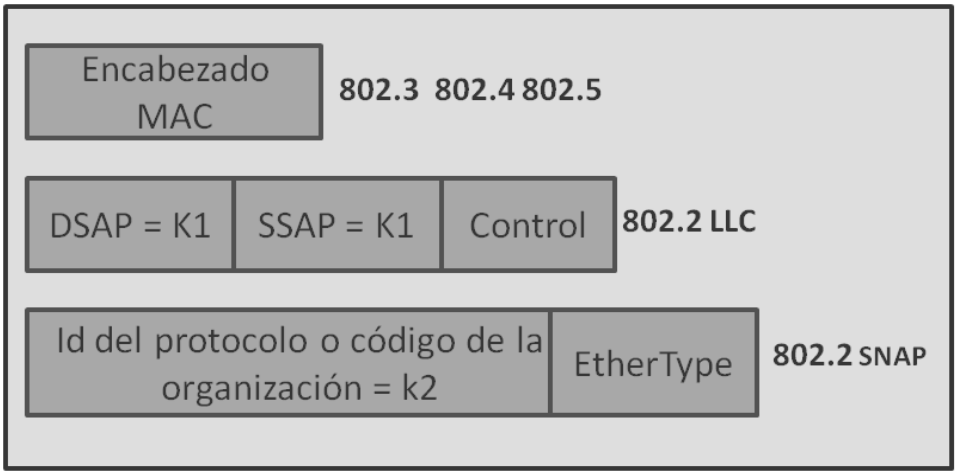


Figura 13 Encabezado de la capa de enlace.

La longitud total del encabezado LLC y SNAP es de 8 octetos. Los datagramas que se envían con el estándar IEEE 802 están encapsulados dentro del LLC 802.2 con la capa de enlace SNAP y la capa física de red 802.3, 802.4, 802.5. El SNAP se utiliza con un código de organización que indica que los 16 bits siguientes especifican un código ethertype de los que se especifican en “Assigned Numbers” en el RFC 1010.

La subcapa MAC lleva la dirección física de cada dispositivo de red, la dirección MAC. La dirección MAC es una dirección de 48 bits que es codificado sobre cada dispositivo de red por su fabricante. Esta es la dirección MAC que la capa física usa para mover los datos entre los nodos de la red.

3.3 Ethernet

Bob Metcalfe y su colega David Boggs, diseñaron la primera red de área local (LAN) en 1976. Llamaron Ethernet al sistema; esto debido a “luminiferous ether”, que alguna vez se pensó, que a través de éste se propagaba la radiación electromagnética [14].

El medio de transmisión en esta red era un cable coaxial grueso (el éter) de más de 2.5 Km de largo y contaba con repetidoras cada 500 metros. En este entorno, una computadora tenía que escuchar el cable para ver si había alguien más transmitiendo. En caso de que así fuera, la computadora se mantenía en espera de que la transmisión actual terminara. Todas las computadoras escuchan, aún la que está transmitiendo para detectar una posible colisión ocasionada cuando dos computadoras transmiten al mismo tiempo. Si la computadora que envía detecta interferencia, manda una señal para poner en alerta a todos los transmisores y .espera un tiempo aleatorio para volver a intentarlo.

El Ethernet Xerox fue tan exitoso que DEC, Intel y Xerox diseñaron un estándar en 1978 para una Ethernet de 10 Mbps, llamado estándar DIX. Con dos cambios menores, en 1983 el estándar DIX se convirtió en el estándar IEEE 802.3. Además de éste, están estandarizados Token Bus (802.4) y Token Ring (802.5).

En las especificaciones de Ethernet [25] se presenta el formato de la trama en la capa de enlace como en la Figura 3.6.

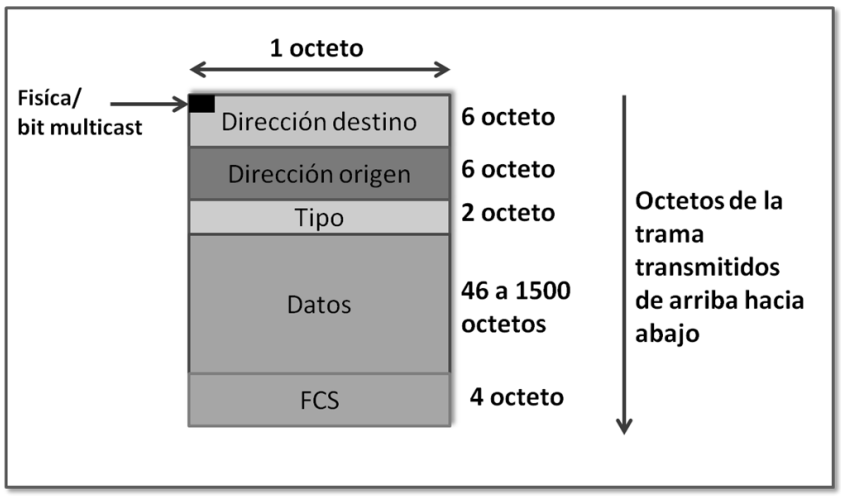


Figura 14 Formato de la trama Ethernet en la capa de enlace.

Cuando la IEEE estandarizó Ethernet, se realizaron dos cambios al formato DIX. El primero fue reducir el preámbulo a 7 bytes y utilizar el último byte para un delimitador de inicio de trama, por compatibilidad con 802.4 y 802.5. El segundo fue cambiar el campo de Tipo en un campo de Longitud. Para saber cuál es la trama siguiente se agregó un pequeño encabezado a la porción de datos para proporcionar esta información. Para el campo Tipo, si el número es menor o igual que 1500, se interpreta como longitud y si es mayor, como Tipo.

Tanenbaum presenta la comparativa de ambas tramas como se muestra en la Figura 3.7.

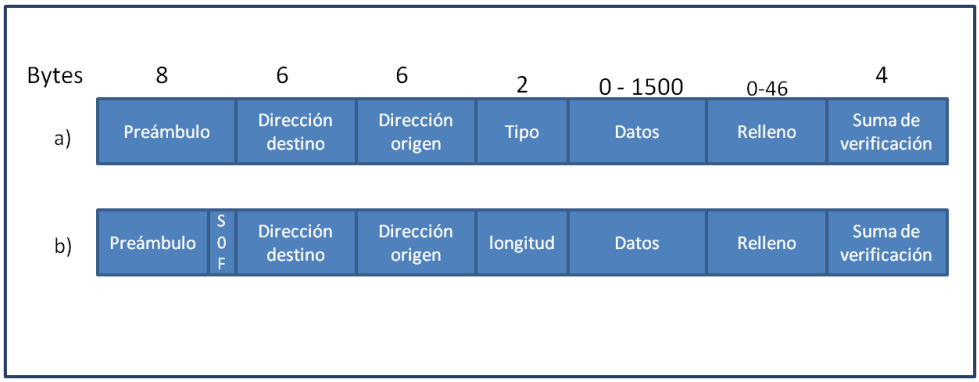


Figura 15 Formatos de trama a) Ethernet Dix b) IEEE 802.3.

3.4 Resumen

Conocer el formato de los encabezados de cada protocolo que se desea monitorear es de suma importancia como se ve en cada propuesta que se ha revisado en los trabajos del capítulo 2, sesión 2.2. Además, en el sistema propuesto es un punto fundamental porque al conocer el formato se puede extraer la información que es relevante en el momento de buscar evidencia

En el siguiente capítulo se hace una introducción a los principales tipos de ataque que existen y como utilizan los campos de los encabezados del protocolo para aprovechar las vulnerabilidades y consumar su delito.

CAPÍTULO 4. PRINCIPALES TIPOS DE ATAQUE

En este capítulo se lleva a cabo la definición de conceptos y la revisión de algunas vulnerabilidades que tienen los protocolos de la pila TCP/IP. Así mismo, se hace una revisión de cuáles son los ataques más conocidos y qué protocolos se utilizan para llevar a cabo un delito.

4.1 Introducción

A través de los años Internet se ha desarrollado cada vez más, ahora es posible realizar transacciones bancarias, declaraciones de impuestos, pago de facturas y compra y venta de bienes e inmuebles. Este gran desarrollo ha traído consigo la vulnerabilidad de información privada que puede ser utilizada por otras personas con fines delictivos. En México, se agrega también el lento desarrollo en la legislación para dar seguimiento a este tipo de delitos. Existen sin embargo, algunas leyes [ANEXO B] que pueden ser mejor aplicadas si se cuenta con herramientas capaces de proporcionar información que sea útil en una investigación. Para tener un mayor entendimiento de la información que podría ser de utilidad en una investigación de un delito cibernético, se revisan las características de los ataques a sistemas más comunes.

4.2 Ataque cibernético

Comenzamos con la definición de algunos términos útiles en la materia. El primer elemento necesario para que se realice un ataque cibernético es precisamente el

atacante. El atacante será toda persona que intentan acceder o accede sin autorización a un sistema, ya sea una computadora personal o una red con decenas de servidores. Hoy en día, cualquier persona con conocimientos mínimos en computación puede ser un atacante con la ayuda de la gran cantidad de herramientas que existen en Internet. Sin embargo, los ataques a sistemas computacionales existen desde hace ya varios años. Una de las primeras clasificaciones de los atacantes se hizo tomando en cuenta su nivel de conocimiento informático [26]. La Tabla 4.1 muestra esta clasificación.

Tabla 4.1 Clasificación de los atacantes según sus conocimientos.

Tipo	%	Características
1	85	Nuevos intrusos, usuarios de computadora, generalmente adolescentes que encuentran documentación y herramientas en Internet y las ponen en uso, sin plena conciencia de lo que están haciendo.
2	10	Son un poco más sofisticados, no han aprendido a programar pero tienen los conocimientos básicos para compilar algunos códigos fuentes de exploits para reventar diversas computadoras que se encuentran por la red. Además de que tienen un buen entendimiento de las herramientas que están manejando.
3	4	Tienen conocimientos profundos, definen muy bien sus objetivos, cuentan con la destreza necesaria para ingresar a los sistemas, observar y ver lo que quieren o necesitan, y efectuar borrado de huellas para evitar su descubrimiento por herramientas de detección de intrusos o por los administradores de red.
4	1	En este grupo se encuentran los cibercriminales, son individuos con las habilidades del grupo tres que cometen acciones delictivas a sueldo para terceras personas u organizaciones o beneficio propio.

El objetivo de un atacante, es llevar a cabo un ataque que puede ser de diferentes tipos dependiendo de la información que desea obtener. En el RFC 2828 se hacen dos clasificaciones:

La primera los divide en activos y pasivos: los ataques de tipo **pasivo** están destinados aprender y hacer uso de la información del sistema pero no alteran sus recursos. Por otro lado, ataques de tipo **activo** tienen como finalidad alterar los recursos del sistema y afectar sus operaciones. Este último traspasa definitivamente la barrera de

la legalidad en su más estricto sentido, y en La biblia del hacker [26] se hace una sub clasificación:

- Interrupción. Si el atacante hace que algún servicio del sistema informático se interrumpa y quede inutilizable durante un periodo de tiempo.
- Modificación: Si el atacante consigue acceso a determinado flujo de datos o partes del sistema y además modifica la información en ellos contenida.
- Suplantación: Si el atacante consigue acceso y suplanta a la identidad de origen frente a un receptor de información de la entidad de origen, de manera que este no sea capaz de distinguir entre el verdadero o falso origen de la información, datos o comunicación.
- Destrucción: Si el atacante consigue acceso y los suficientes privilegios para destruir o borrar información destinada a los usuarios del sistema, produciendo una interrupción del servicio entre emisor y receptor.

La segunda clasificación hace la división entre ataques internos y externos: esta clasificación señala que un ataqu**e interno** se encuentra dentro de los perímetros de seguridad, es una entidad que cuenta con los permisos para acceder a los recursos del sistema y que hace uso inadecuado de estos. El ataque de tip**o externo** inicia fuera del perímetro de seguridad por un usuario no autorizado o ilegitimo del sistema. Este tipo de ataque involucra un conjunto de acciones que se llevan a cabo para obtener acceso al sistema de forma no autorizada.

- **Reconocimiento**: El reconocimiento es la fase en la que el intruso validó la disponibilidad del objetivo, enumera servicios y busca aplicaciones vulnerables.
- **Explotación:** La explotación es el proceso de abusar, manipular o vulnerar los servicios en la víctima. Se llama abuso al hecho de utilizar un acceso legítimo de manera no legítima.
- **Obtención de privilegios**: La obtención de privilegios es la fase en la que los intrusos toman ventaja del acceso no autorizado que obtuvieron en la fase anterior. Los intrusos más sofisticados instalan los medios necesarios para comunicarse con el exterior. Desde canales encubiertos, canales cifrados hasta complicados mecanismos de señalización para lo cual hacen uso de tráfico que no ocupa mucho ancho de banda. Esto se denomina puerta trasera o back door.

- **Aseguramiento de plan central del intruso**: La puerta trasera funciona como un medio por el cual el intruso se comunica con la víctima. Ésta puede parecer un servicio a la escucha por el cual se conecta el intruso. Otra manera es que la víctima se conecta con el intruso y no al revés. Una tercera forma es que la víctima se conecta a un servidor remoto *IRC (Internet Relay Chat por sus siglas en inglés)*.
- **Ejecución del plan**: Finalmente el atacante comete el delito.

Debido a que varias organizaciones destinan la mayoría de sus recursos para detectar y prevenir ataque externos, cuando un intruso ha comprometido un sistema al interior de la organización, puede pasar desapercibido por mucho tiempo. Por lo general lo que hacen las organizaciones es llevar a cabo el monitoreo de seguridad de red, el cual se basa en identificar amenazas y no vulnerabilidades [7].

En esta parte nos encontramos con el siguiente concepto a analizar, la amenaza. Un elemento que tiene la capacidad y la intención de explotar una vulnerabilidad de un servicio es una amenaza, se dividen en estructuradas y no estructuradas:

- La amenaza estructurada involucra espías, crimen organizado, terroristas, y agencias de inteligencia extranjeras entre otras. Este tipo de amenaza se caracteriza por contar con una metodología de ataque, tiene los recursos económicos para llevarlo a cabo y su objetivo es bien definido.
- La amenaza no estructurada engloba a *Crackers* y *malware*, infiltrados con la intención de aprovechar su posición. A diferencia del estructurado, esta amenaza carece de metodología, dinero y objetivo. Sus motivos pueden ser diversos y se preocupan poco por prevenir que sus actividades sean monitoreadas.

Como se ha visto, la mayor parte de los atacantes son personas que no están organizadas, no tienen una metodología específica y por lo tanto no son cuidadosos al cubrir sus huellas. Los ataques pueden ser de tipo externo o interno y pueden ser pasivos o activos pero todos los ataques implican una comunicación a través de la red.

En el monitoreo de seguridad se trata de detener al atacante, el cómputo forense va más enfocado a analizar cómo se llevo a cabo el delito y quién fue el atacante. Los ataques en una red tienen como objetivo dañar, sabotear o robar la propiedad de una organización con el propósito de obtener información o ventaja competitiva. Para este fin

son muy utilizadas las vulnerabilidades que existen en los sistemas [27] y se revisan algunas de ellas en la siguiente sección.

4.3 Vulnerabilidades en la pila de protocolos TCP/IP

Se define vulnerabilidad como una falla o debilidad en el diseño, implementación u operación y administración de un sistema que se puede explotar para violar las políticas de seguridad del mismo (RFC 2828). La mayoría de los sistemas tiene vulnerabilidades pero no todas las amenazas resultan en un ataque y no todos los ataques son exitosos. Se dice que un ataque es exitoso cuando logra evadir los servicios de seguridad y viola las políticas de seguridad de un sistema.

A continuación se revisan algunos tipos de ataque que hacen uso de las vulnerabilidades de la pila de protocolos TCP/IP. Los ataques pueden ser realizados sobre cualquier tipo de red, sistema operativo o conjunto de computadoras.

4.3.1 Ataques de monitoreo

El protocolo más utilizado en el ataque de monitoreo es el TCP (Transfer Control Protocol) por lo que se hace una breve descripción de los campos que conforman el encabezado. El protocolo TCP corresponde a la capa de transporte de la pila de protocolos TCP/IP. Éste utiliza puertos o número de sockets para pasar la información a las capas superiores. En la Figura 4.1 se muestra el encabezado que presenta el RFC 793. Los puertos que asigna el sistema operativo también se conocen como sockets.

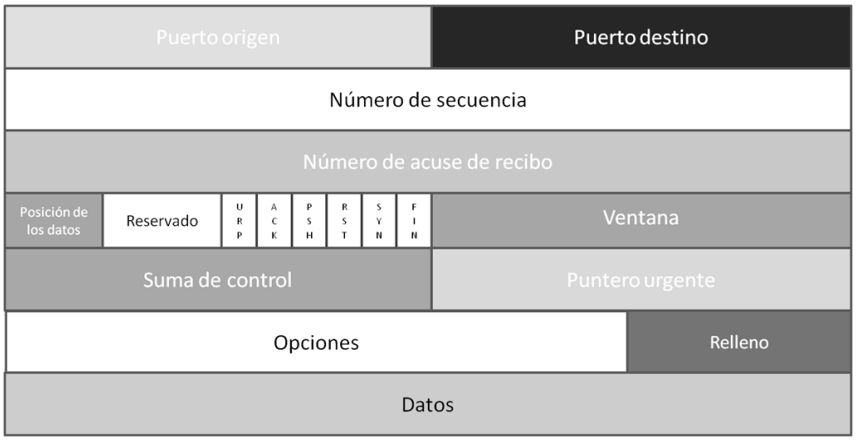


Figura 16 Encabezado TCP.

Los números de puerto se dividen en tres categorías: puertos bien conocidos, puertos registrados y puertos privados. En la tabla 4.2 se muestran algunos de los puertos bien conocidos que aparecen en el RFC 1060.

Tabla 4.2 Puertos bien conocidos.

Protocolo de la capa de aplicación	Número de puerto
ftp	21
telnet	23
SMTP	25
HTTP	80
HTTPS	443

En una conexión, los nodos finales se pueden identificar de forma única por cuatro parámetros: puerto origen, puerto destino, dirección IP destino, dirección IP fuente. Con el protocolo IP tenemos las direcciones y con TCP en este caso, obtenemos los puertos.

Cuando se utiliza el protocolo TCP, el intercambio de datos sucede hasta que el saludo (Figura 4.2) se lleva a cabo de manera exitosa. TCP utiliza diferentes banderas para determinar la forma en la que los segmentos van a ser procesados. Las banderas que forman parte de esta comunicación son las siguientes:

- URG (Urgent Pointer Field)
- ACK (Acknowledgment Field)
- PSH (Push Function)
- RST (Reset the connection)
- SYN (Synchronize Sequence Number)
- FIN (No more data from sender)

El proceso del saludo implica tres pasos y comienza cuando la máquina A desea comunicarse con la máquina B (Figura 4.2). Primero A le pregunta a B enviando la bandera SYN establecida en uno en el primer segmento, B responde con un SYN igual a uno y además la bandera ACK establecida en uno. Si B no desea hablar con A, entonces responde con la bandera RST igual a uno. Si B envío un SYN-ACK, entonces A envía otro ACK y B lo recibe con lo que están listos para comenzar la comunicación

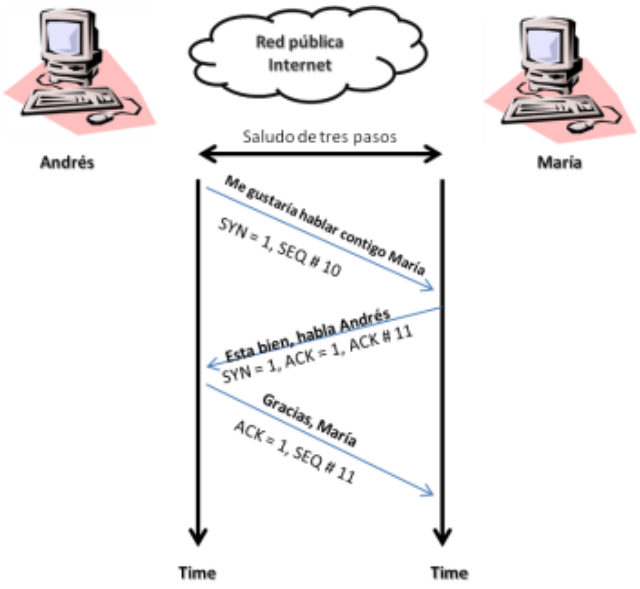


Figura 17 Saludo TCP.

Es en esta parte de la comunicación que el protocolo es vulnerable como veremos enseguida.

La finalidad de un ataque de monitoreo es observar a la víctima y su sistema con el objetivo de obtener información para enumerar sus debilidades y obtener posibles formas de acceso al sistema monitoreado. En este tipo de ataque se encuentran los escaneos.

- **Escaneo TCP Connect:** Es la forma básica de escaneo de puertos TCP. Si el puerto está escuchando, se devuelve una respuesta de éxito y se establece la conexión con el puerto de la máquina víctima, cualquier otro evento significa que el puerto no está abierto o que no podemos establecer conexión con él, este método es conocido también como el protocolo de acuerdo a tres vías.

  Esta técnica es rápida y no precisan de ningún permiso de usuario sobre la máquina víctima, por otro lado es fácilmente detectable por el administrador de la máquina, pues el procedimiento se observa con una dirección IP tratando de hacer conexión a todos los puertos en tiempos consecutivos.
- **Escaneo TCP SYN:** Lo que realiza el escaneo TCP SYN es la técnica de “la media apertura”. El intruso no pretende establecer una conexión completa. Se envía un paquete SYN (para iniciar la conexión), se espera la respuesta y, al recibir el ACK de la máquina víctima, se responde con un RST, con esto decimos que no se quiere establecer la conexión pero se va haciendo un registro de los puertos que están abiertos en la víctima.
- **Escaneo TCP FIN**: Este tipo de escaneo se lleva a cabo cuando se desea ser más sutil y no ser detectado tan fácilmente, (en la actualidad determinados firewall y programas de detección de intrusos están preparados para detectar los escaneos del tipo SYN). Esta técnica se basa en el principio de que los puertos cerrados suelen responder a los paquetes FIN, con un RST; los abiertos, por el contrario, no les hacen caso y no responden, ése es el momento de registrar el puerto como abierto. Esta técnica tiene nula utilidad en sistemas Microsoft Windows pues siempre envía paquetes RST ante un FIN, esté el puerto abierto o no, pero puede ser de gran utilidad en sistemas UNIX/LINUX.
- **Escaneo fragmentado:** Aumentado la sutileza de nuestros paquetes enviados a los puertos de la máquina víctima, este procedimiento consiste en partir los paquetes completos de sondeo, dirigidos a la máquina víctima, en pequeños fragmentos de paquetes, con lo cual estos pequeños paquetes provocan menos ruido en las posibles herramientas firewall o de detección del administrador de la máquina objetivo.

Para obtener información de los patrones de este ataque, se puede pensar en hacer un filtro que guarde los paquetes TCP en la etapa del saludo y se guarde el encabezado para posteriormente hacer un análisis de las banderas utilizadas en la secuencia de paquetes.

4.3.2 Ataques de validación

Tiene como objetivo suplantar al dueño o usuario de la máquina, mediante la consecución de sus cuentas de acceso y contraseñas, o suplantando la identidad de la máquina en la red.

- **Suplantación de identidad en saltos**: Se conoce también con el nombre de *Spoofing-Looping*, consiste en obtener el nombre y contraseña de un usuario del sistema, entrar en su máquina para obtener información de la red y del resto de equipos e ir suplantando nuevas identidades de usuarios de computadora en computadora (esto se aplica a computadoras conectadas a la red de Internet). Cuando nos encontramos en el salto cuatro por ejemplo, si realizamos un ataque de escaneo u otra técnica sobre otra máquina, la IP origen del ataque quedará registrada como la del usuario de la computadora cuatro, y el intruso en realidad entró desde la computadora uno para llevar a cabo su ataque, sólo que fue saltando de máquina en máquina, suplantando identidades. Seguir el rastro a un intruso que ha realizado este tipo de ataque es muy complicado, porque requiere de la colaboración del administrador del sistema de cada uno de los host por donde ha ido saltando.
- **Suplantación de IP**: Este ataque también se conoce como IP *Spoofing*, en este ataque debemos generar paquetes de información, con una dirección IP falsa, en la parte del paquete TCP/IP que identifica el host origen de la llamada. Si el intruso conoce la dirección IP de la computadora de una persona cualquiera en la red, será suficiente alterar el paquete para incluir esta dirección, con lo cual si el host atacado y el administrador registran esos paquetes donde aparece la IP origen, la persona culpada será otra y no el intruso, incluso se pensará que el ataque proviene de una red distinta a la original, dependiendo de la IP falsa que decidamos incluir. El encabezado IP que está definido en el RFC 791 se presenta en la Figura 4.3.

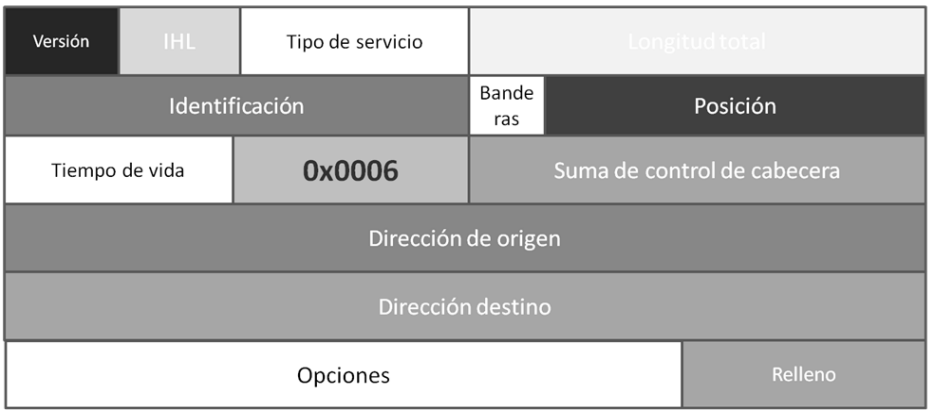


Figura 18 Encabezado IP.

- **Suplantación DNS**: Otro nombre que se le da a este tipo de ataque es DNS *Spoofing*, consiste en la manipulación de paquetes del protocolo UDP, para engañar al servidor de dominio y que éste entregue información de sus tablas de registros, a una petición del atacante. Esto es posible si el servidor DNS tiene el método recursivo en funcionamiento. Éste es el método de funcionamiento por defecto en los servidores de dominio.
- **Suplantación IP *Splicing-Hijacking***: El atacante ha interceptado una conexión establecida recientemente, observa el intercambio de paquetes entre el usuario original y el servidor (La computadora víctima), entonces suplanta al usuario emitiendo un paquete casi idéntico al que el servidor espera recibir del usuario conectado, lo suficientemente idéntico para que el servidor no note nada; en este momento el intruso recibe la respuesta del servidor y puede seguir enviando paquetes de información habiendo suplantado al usuario original. El usuario original se queda esperando datos.

Este tipo de ataque podría ser analizado si todas las máquinas en la red están llevando a cabo un muestreo de los paquetes con protocolos vulnerables.

4.3.3 Ataques de denegación de servicios

La comunicación entre computadoras se basa en protocolo. Esta comunicación no es inspeccionada, se da por hecho que los protocolos se van a utilizar para la correcta comunicación entre dispositivos de red. Como nadie está verificando que esta suposición es correcta, cualquier tipo de atacante podría llevar a cabo un ataque con o sin intención, traspasando alguno de los límites de la comunicación. Los ataques de denegación de servicios pretenden desbordar los recursos de la computadora víctima de forma tal que se interrumpen los servicios suministrados por ella. El objetivo principal es bloquear los servicios ofrecidos por la computadora a sus clientes (conjunto de computadoras que solicitan de éste un servicio).

Este tipo de ataque se puede dividir en cinco categorías:

- Ataque DoS en el nivel de dispositivo de red. En esta categoría se lleva el ataque de cualquiera de las siguientes maneras: tomando ventaja de los errores o vulnerabilidades del software, o agotando los recursos del hardware de los dispositivos de red.
- Ataque DoS a nivel de sistema operativo. Se aprovecha de la manera en la que el sistema operativo maneja los protocolos, un ejemplo de este es el ping de la muerte que se explica más abajo. En este tipo de ataque el ICMP *echo request* tiene una medida de datos más grande del que está permitido para el protocolo IP.
- Ataques basados en aplicación. Intenta poner una máquina o servicio fuera de orden de dos maneras, explotando errores de las aplicaciones de red que se encuentran ejecutando en el host destino o utilizando tales aplicaciones para drenar los recursos de sus víctimas. También es probable que el atacante tenga puntos encontrados de alta complejidad algorítmica y explotarlos con el objetivo de consumir todos los recursos disponibles de un host remoto. Un ejemplo de este tipo de ataque es *finger bomb*.
- En un ataque de inundación de datos, un atacante intenta utilizar la banda ancha disponible en la red, host o dispositivo en el grado más alto enviando una cantidad masiva de procesos de datos. Un ejemplo de este tipo de ataque es la inundación de ping. La inundación simple se caracteriza como un ataque DDoS (Ataque de denegación de servicios distribuido).

- Ataque DDoS basado en las características del protocolo. Toma ventaja de ciertas características estándar de los protocolos. Por ejemplo muchos ataques utilizan el hecho de que la dirección IP fuente puede ser suplantada. Más aún muchos tipos de ataques DoS intentan atacar el cache DNS sobre el nombre del servidor.

Los ataques DDoS están compuestos por cuatro elementos en la red y el atacante real que lleva a cabo todo el plan. La Tabla 4.3 muestra esta clasificación.

Tabla 4.3 Elementos de un ataque DDoS.

El atacante real	**Persona que lleva a cabo el ataque**
El manejador o maestro	Es un *hosts* comprometido con un programa especial que se está ejecutando en él que lo hace capaz de controlar múltiples agentes.
zombies	Son *hosts* comprometidos en el que se está ejecutando un programa especial y son responsables de generar un flujo de paquetes destinado a la víctima. Este elemento es usualmente externo a la red de la víctima y a la red del atacante con el objetivo de evitar una respuesta eficiente que podría parar el ataque y rastrearlo.
hosts comprometidos	Son hosts en los que se está ejecutando un programa especial y son responsables de generar un flujo de paquetes destinado a la víctima. Este elemento es usualmente externo a la red de la víctima y a la red del atacante con el objetivo de evitar una respuesta eficiente que podría parar el ataque y rastrearlo.
Una víctima o host objetivo	Es el *hosts* al que va dirigido el ataque, por lo general un servidor

Un ataque DDoS se lleva a cabo en cuatro pasos. Estos pasos y su explicación se encuentran en la Tabla 4.4.

Tabla 4.4 Pasos para llevar a cabo un DDoS.

Selección de agentes	El atacante elige a los hosts que llevarán a cabo el ataque. A estas computadoras se les conoce como agentes.
Compromiso	El atacante explota los huecos de seguridad y vulnerabilidades de los agentes seleccionados e instala el código atacante.

Comunicación	Antes de que el atacante inicie el ataque, se lleva a cabo una comunicación con los manejadores para determinar cuáles agentes pueden utilizarse en el ataque, si es necesario actualizarlos y el mejor momento para hacerlo. Los agentes se pueden comunicar con uno o varios manejadores dependiendo de la configuración del ataque. Los protocolos utilizados en la comunicación entre manejadores y agentes son: TCP, UDP y ICMP.
Ejecución	En este paso el atacante comanda el comienzo del ataque. En esta etapa se pueden ajustar detalles como la víctima, la duración del ataque y las características especiales del ataque como son el tipo, longitud, tiempo de vida (TTL), así como el número de puertos.

Existen diversos protocolos utilizados para un ataque DoS o DDoS y algunos de los ataques más conocidos y el protocolo que utilizan están numerados a continuación.

- ***Flooding***: Este tipo de ataque desborda los recursos del sistema. El hacker malicioso satura el host víctima de mensajes que requieren establecer conexión y dar respuesta. Como la dirección IP del mensaje puede ser falsa (técnicas de *Spoofing*), la máquina atacada intenta dar respuesta a la solicitud de cada mensaje, saturando su buffer con información de conexiones abiertas en espera de respuestas: esto impide que oiga las solicitudes de otros usuarios que sí necesitan realmente conectarse. El ping de la muerte es un ejemplo de este tipo de ataque pero las defensas están implementadas desde hace tiempo. Una manera más rudimentaria de *Flooding* puede ser el envío masivo, y repetidas veces, de un mail a la cuenta de correo de todos los usuarios existentes, terminando por saturar su servicio de correo.
- **SYN Flood**: Este tipo de ataque hace uso de los *timers* que se utilizan mientras se lleva a cabo una comunicación que utiliza la pila de protocolos TCP/IP. Por ejemplo, los *timers* involucrados en el saludo de la Figura 4.2 son tres:
    - Connection Establisment *Timer*: Se inicializa después de que se envía el SYN cuando se establece la conexión inicial (Paso 1 del saludo).
    - FIN_WAIT *Timer*: Se inicializa después de que se establece la bandera FIN y el que la envía está esperando por la confirmación para dar por terminada la conexión.
    - KEEP_ALIVE *Timer:* El contador se reinicia después de que se transmitieron todos los segmentos de datos. Se utiliza periódicamente para comprobar el final remoto.

Todos los *timers* son críticos para una trasmisión de datos apropiada y precisa. Estos *timers* o la carencia de ellos es utilizado por los atacantes para desactivar los servicios del sistema o incluso, para entrar en este.

Utilizando el saludo de ejemplo de la Figura 4.2, después del paso 2 de la comunicación, no se establece un tiempo límite de espera después de que se recibe la bandera SYN. El atacante puede iniciar muchas solicitudes de conexión al servidor web, casi con toda seguridad utilizando otro ataque: La suplantación de dirección IP. El servidor envía la contestación: SYN + ACK a la dirección fuente del cliente y no recibe respuesta.

Esta situación deja la sesión TCP del servidor abierta, por lo que muchas sesiones TCP están abiertas en el servidor, pero de acuerdo a las limitaciones del *hardware* en el servidor, hay un número limitado de sesiones TCP que pueden estar abiertas. En estas circunstancias, el servidor web comienza a rechazar solicitudes de conexión de cualquier *host* en el momento que alcanza su límite.

Las conexiones que quedaron abiertas se deben de completar o agotarse el tiempo de espera antes de que se puedan establecer nuevas conexiones. Esta vulnerabilidad se utiliza por los atacantes para dejar el servidor inaccesible por varios segundos, mientras la plataforma esta desactivada temporalmente, el atacante puede depositar otro *exploit* o instalar una puerta trasera.

Existe otro tipo de ataques que utiliza el protocolo TCP: cerrar conexión utilizando la bandera FIN y secuestro de conexión.

Cerrar conexión utilizando la bandera FIN: este tipo de ataque puede llamarse asesino de conexiones. En una comunicación normal, la fuente establece en uno la bandera FIN para indicar que no enviará más paquetes y que la conexión se puede cerrar. Este es un saludo de cuatro pasos en el que la fuente y el destino esperan para enviar una confirmación sobre un paquete recibido con la bandera FIN. En un ataque en donde se desea asesinar la conexión, se construye un paquete falso con la bandera FIN en uno, este paquete tiene la secuencia de números correcta por lo que el *host* lo toma como válido. La secuencia de números es fácil de predecir y al hecho de hacerlo se le denomina predicción de número de secuencia. Este se puede hacer de dos maneras, en el primero el

atacante pueden espiar para obtener el número de secuencia y confirmación (SEQ/ACK). En el segundo se hace la predicción algorítmicamente.

Cuando el paquete construido se envía, el *host* destino cree que el fuente falso no tiene más datos para ser transmitidos. Cualquier otro paquete recibido es ignorado por falso y se desecha. Los paquetes restantes para completar el saludo de los cuatro pasos son previstos por el fuente falso. Este ataque también se lleva a cabo utilizando la bandera RST.

- ***Conection Flood***: El desbordamiento de conexiones tiene el mismo principio de saturar a la máquina atacada con peticiones maliciosas. Por ejemplo, si un servicio Web admite 5000 conexiones, es suficiente que un atacante realice 5000 conexiones sin nada que solicitar a la máquina víctima, con lo cual otros usuarios no pueden acceder al servicio que ésta suministra. Aunque transcurrido un tiempo las conexiones se van liberando, por inactividad, tan solo se debe seguir enviando peticiones de conexión par ir reemplazando las liberadas y dejar el servicio inoperativo.
- **Tormenta *Broadcast***: Este ataque pude ser destructivo, radica en mandar una petición ICMP a una dirección broadcast de la red. La petición ICMP lleva, como IP de origen, la IP del host que se desea atacar. Ante esta petición, el dispositivo *broadcast* diseminará la solicitud a todas las máquinas de la red y simultáneamente empezarán a responder a la víctima objetivo saturándola de respuestas que no puede procesar y produciendo un colapso en su sistema.
- **Desbordamiento NET**: Consiste en lanzar a la red, en la que se ha incursionado, paquetes y paquetes sin sentido que empiecen a colapsar el tráfico que circula por ella, de manera que no dejamos que las transmisiones útiles circulen o reducimos drásticamente su velocidad de transmisión.
- **Teardrop one y teardrop two:** Este ataque se realiza enviando paquetes fragmentados con datos que se traslapen entre sí. Algunas implementaciones de colas IP no pueden ensamblar correctamente este tipo de paquetes y el host se queda sin saber qué hacer. Por ejemplo, enviar un paquete con los datos del 1 al 100 y después enviar otro paquete con los datos del 50 al 150, produciría este efecto en la máquina víctima. Este ataque afecta a diversas versiones de sistemas Unix, Linux y Windows.

4.3.4 Otros tipos de ataque

CANALES ENCUBIERTOS: Para comenzar con la explicación de este tipo de ataque es necesario definir el concepto. Un canal encubierto es un pipe o canal de comunicación entre dos entidades que pueden ser utilizadas por un proceso o aplicación que envía información quebrantando las especificaciones de seguridad del sistema. De forma más específica en TCP/IP los canales encubiertos se establecen y los datos se envían de manera secreta entre dos sistemas finales. En algunas circunstancias como las que se listan a continuación los mensajes ICMP se envían para proporcionar mecanismos de error y control.

- Para probar la conectividad o alcance empleando datagramas, en esta situación se utiliza Mensajes echo y *echo-Reply* (ver Figura 4.4) del protocolo ICMP.

Figura 19 Mensaje echo o echo replay.

- Para reportar destinos inalcanzables para datagramas, se hace uso del Mensaje de destino inalcanzable (ver Figura 4.5) del protocolo ICMP.

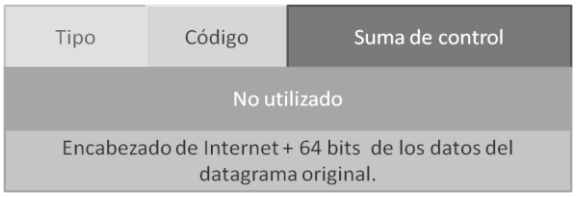


Figura 20 Mensaje de destino inalcanzable.

- Reportar problemas con la capacidad del buffer para enviar datagramas, utiliza Mensaje de enfriamiento fuente (source quench message) (ver Figura 4.6) del protocolo ICMP.

Figura 21 Mensaje de enfriamiento fuente.

- Reportar cambios en la ruta del camino del datagrama, con el mensaje del protocolo ICMP Redireccionar mensaje (ver Figura 4.7).

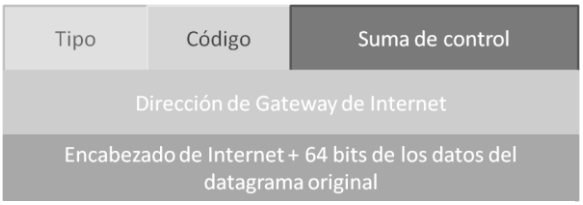


Figura 22 Mensaje Redireccionar.

Un mensaje ICMP *Echo Request* podría tener un encabezado de 8 bytes y 56 bytes de datos. Puede ser que el paquete no contenga datos, sin embargo, estos paquetes se utilizan con frecuencia para transportar información secreta. Esto con frecuencia aumenta el tamaño del paquete, sin embargo, no existe algún tipo de control en la pila de protocolos para este comportamiento. Esta vulnerabilidad permite que los intrusos envíen programas cliente-servidor especializados que exportan información confidencial sin alertar a los administradores de red. La solución que se da es bloquear los paquetes ICMP que exceden cierto límite de medida.

ATAQUE DE FRAGMENTOS IP: Primero se van a revisar algunos conceptos relacionados a la fragmentación de paquetes con la finalidad de dar claridad a este tipo de ataque.

La fragmentación tiene lugar cuando un datagrama que va a pasar a través de la red es más grande que el MTU (Unidad máxima de transmisión) el cual representa la mayor cantidad de datos que puede transportar la trama física de un tipo de red. El MTU de las redes Ethernet es 1500 bytes lo que significa que una red Ethernet nunca podrá transportar un datagrama con una cantidad de bytes mayor, sin fragmentarlo.

Cuando sucede el caso de que el datagrama a transportar es más grande que el MTU, la capa IP divide el datagrama en varios datagramas más pequeños. En general, la fragmentación permite que un *host* o *router* divida un paquete en fragmentos más pequeños y los envíe a través de la red [28]. En la 4.8 se muestra la parte del encabezado IP en la que se encuentra la información de fragmentación del datagrama y está ubicada a partir del byte 5 hasta el byte 8.

Figura 23 Parte del encabezado IP.

Todos los fragmentos de un mismo paquete tienen el mismo valor en el campo identificación que es un identificador del datagrama actual. El campo posición indica la posición que ocupa el fragmento en el contexto del paquete original, es útil para dar seguimiento a las diferentes partes del datagrama. Los primeros tres bits del tercer byte son banderas individuales para varios aspectos de la fragmentación: el primer bit siempre es cero y nunca se utiliza, el segundo bit sirve para indicar que el paquete no debe fragmentarse, el tercer bit es el bit de más fragmentos, cuando el sistema fragmenta un datagrama, pone a uno el bit en todos los fragmentos excepto en el último.

Un atacante puede aprovechar que muchos *routers* de acceso y *firewalls* no llevan a cabo el ensamble de paquetes y crear paquetes fragmentados artificialmente para engañarlos. En operaciones normales los fragmentos IP no se superponen, pero si los paquetes están malformados se lleva a cabo este comportamiento.

En este tipo de ataque se modifica el valor del campo posición; cuando llega el segundo fragmento de un datagrama en una trasmisión de datos, este contiene el tamaño del fragmento anterior en el campo posición, pero un atacante puede modificar este campo poniendo una cantidad inferior al tamaño del fragmento anterior. De esta forma, cuando el destino ensambla los fragmentos, se sobrescriben los datos del primer fragmento, lo que da lugar a un paquete malformado que causa que el sistema operativo no funcione apropiadamente o incluso que el sistema se bloquee.

BANDERAS TCP: Este procedimiento puede ser usado para provocar un ataque DoS que cause la caída o cuelgue del servidor, este tipo de ataque se lleva a cabo utilizando combinaciones de las banderas de comunicación con valores inválidos (ver Tabla 4.5).

Tabla 4.5 Combinaciones invalidas de banderas TCP.

SYN	FIN	PSH	RST	Validez
1	1	0	0	Combinación ilegal
1	1	1	0	Combinación ilegal
1	1	0	1	Combinación ilegal
1	1	1	1	Combinación ilegal

4.4 Resumen

Un atacante es la persona que intenta acceder sin autorización a un sistema y que puede tener mínimos conocimientos de computación o ser alguien con conocimientos profundos en la materia. Un ataque puede iniciar dentro del perímetro de seguridad (interno) o fuera de estos (externo). Los ataques pueden ser de tipo pasivo como los monitoreos o activo como los DDoS. En este capítulo se ha visto que las vulnerabilidades en los protocolos son una herramienta muy utilizada para llevar a cabo ataques en la red. Entre los protocolos utilizados de manera más diversa están el protocolo TCP e ICMP. En la tabla 4.7 se hace un resumen de tipos de ataque y los campos de encabezado utilizados.

Tabla 4.6 Resumen de ataques y los campos utilizados.

Nombre del ataque	Objetivo	Protocolo	Campos utilizados
Suplantación de dirección IP	Poner en marcha la denegación de servicio	IP	Dirección fuente
Canales encubiertos	Intercambio de datos de manera secreta	ICMP	56 bytes de datos, después de los 8 bytes de encabezado
Ataque de fragmentos IP	Que el sistema operativo no funcione o se bloquee.	IP	Posición

Banderas TCP	Provocar un DDoS que cause la caída del servidor.	TCP	Banderas URG, ACK, PSH, RST, SYN, FIN
ARP Spoofing		ARP	Dirección hardware fuente, dirección de protocolo fuente, dirección hardware destino, dirección protocolo destino.

En el siguiente capítulo se lleva a cabo el diseño del sistema y se presentan los diagramas de flujo propios de la programación estructurada.

CAPÍTULO 5. ANÁLISIS Y DISEÑO DEL SISTEMA DE CAPTURA ALEATORIA ADA**

En este capítulo se lleva a cabo el diseño de la aplicación, en primer lugar se revisan los elementos por los que está constituido el sistema, posteriormente se desarrollan los diagramas de flujo.

5.1 Introducción

ADA** es un sistema escrito en su mayor parte en C, pero se complementa con otros elementos. Está diseñado y construido tomando en cuenta las características de una plataforma Linux, con Fedora 14. Uno de los elementos base del sistema es Libpcap 4.2.1, que lleva a cabo la recolección de datos. Finalmente, la interfaz de usuario se escribió en Python 2.7 por muchas razones: forma parte de las herramientas que Fedora 14 tiene integradas, tiene integración con los componentes de C, por la gran cantidad de interfaces y utilidades ya codificada facilita la velocidad en el desarrollo de nuevas aplicaciones. Además se hace uso matplotlib que es una librería de Python que se va a utilizar para generar las gráficas de los datos almacenados. A continuación se hace una descripción más detallada de cada parte del sistema.

5.2 Plataforma de desarrollo

Como se vio antes, el sistema operativo que se va a utilizar para la implementación de ADA** es Linux, a continuación se describe como está soportada la pila de protocolos TCP/IP en este sistema operativo. En Linux se implementa la familia de protocolos de

Internet como una serie de capas conectadas de software (ver Figura 5.1). La capa de más alto nivel es la que se encarga de manejar los sockets BSD (Berkeley Software Distribution). A continuación se encuentra la capa INET, ésta maneja el punto final de la comunicación para los protocolos basados en IP, tal como, TCP y UDP. Linux tiene las estructuras de estos protocolos predefinidas para poder llevar a cabo la comunicación. UDP es un protocolo sin garantía de conexión, al contrario TCP es un protocolo que garantiza esta conexión. Cuando los paquetes UDP son transmitidos, Linux no sabe si llegaron seguros a su destino. Por otro lado, los paquetes TCP son numerados y en ambos extremos de la conexión TCP se aseguran de que los datos fueron recibidos correctamente. La capa IP contiene la implementación de código del protocolo de Internet, este código antepone los encabezados IP para transmitir datos y entender cómo rutear paquetes IP entrantes en cualquiera de los dos casos: *UDP* o *TCP*.

Debajo de la capa *IP*, están los dispositivos de red, por ejemplo *PPP* y *Ethernet*.

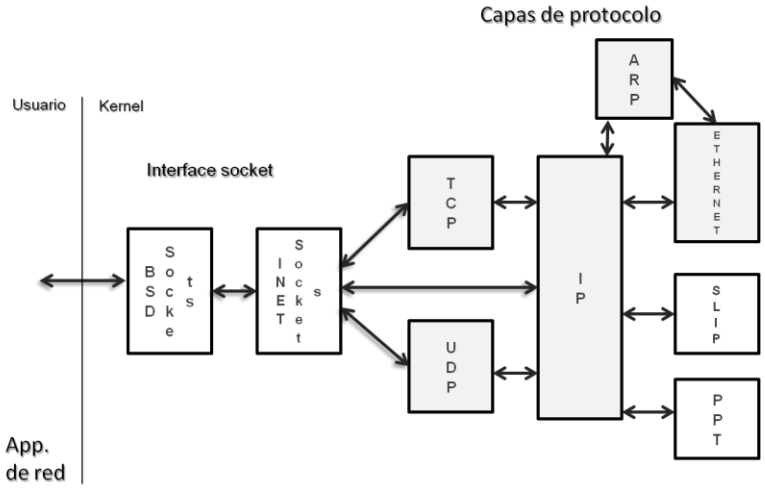


Figura 24 Capas de red de Linux.

En Linux, un dispositivo de red no siempre representa un dispositivo físico, existen algunos como el loopback (localhost) que son dispositivos de software puros. Estos dispositivos facilitan la comunicación en la capa de enlace. Los protocolos utilizados por estos, son el punto de partida para el analizador ADA**. A diferencia de los dispositivos estándar de Linux que se crean por medio del comando mknod, los dispositivos de red

aparecen sólo si el software del que se trate lo ha encontrado e inicializado. Por ejemplo, sólo se ve /dev/eth0 cuando se tiene un kernel con el driver del dispositivo Ethernet en él. Esta es la interface con la que se llevan a cabo las pruebas en el sistema y se utiliza así: eth0. El protocolo ARP se sitúa entre la capa IP y los protocolos que soportan ARPing para dirección. Fedora 14 tiene el kernel versión 2.6.35.12

Cuando un paquete llega al dispositivo de red los drivers se hacen cargo de la recepción y el paquete es analizado utilizando la estructura del dispositivo de red correspondiente. Cada interface que existe en un servidor tiene un archivo de este tipo que proporciona datos acerca de la interface. Por ejemplo para la interfaz de red Ethernet, se encuentra el archivo /etc/sysconfig/network-scripts/ifcfg-eth0

Las cabeceras de los protocolos más conocidos están en /usr/include/ehter.h

5.3 Captura de paquetes

Libpcap 4.2.1 es la librería de funciones que lleva a cabo la captura. Los procesos que ejecutan las funciones de Libpcap lo hacen a nivel usuario, pero la captura de paquetes se hace a nivel del *kernel de Linux*. Los filtros se establecen en la zona *Kernel* para disminuir el trabajo de transportar cada paquete que transita por la tarjeta de red, al espacio de usuario. Prácticamente cada sistema operativo tiene su sistema de filtrado, BPF es un socket de la capa de enlace que permite el filtrado de paquetes de manera granular, su funcionamiento se basa en dos grandes componentes:

El Network Tap: Es el encargado de recopilar los paquetes desde el *driver* del dispositivo de red y entregárselos a aquellas aplicaciones a las que vaya destinado.

Packet Filter: Es el encargado de decidir si el paquete debe ser aceptado (coincidencia direcciones Ethernet) y en caso afirmativo, cuánto de este paquete debe de ser entregado a la aplicación.

En la Figura 5.2 se puede ver el funcionamiento de BPF dentro del sistema. Cuando llega un paquete a la interfaz de red, el *driver* de red lo manda hacia la pila de protocolos, siempre y cuando no esté activo BPF, si es así, el paquete es procesado primero por él.

BPF es el encargado de comparar el paquete con cada uno de los filtros establecidos, pasando una copia de dicho paquete a cada uno de los buffers de las aplicaciones cuyo filtro se ajuste a los contenidos del paquete. En caso de que no exista ninguna coincidencia, se devuelve el paquete al *driver*, el cual actuará normalmente, es decir, si el paquete está dirigido a la propia máquina se lo pasará a la pila de protocolos y en caso contrario lo descartará sin que el paquete haya salido en ningún momento de la zona *kernel*.

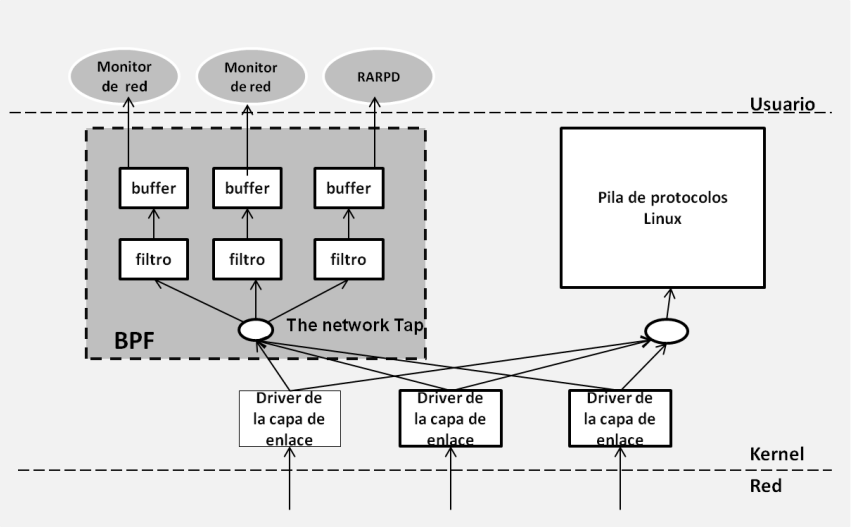


Figura 25 Arquitectura Libpcap.

Para evitar que se traspase la frontera por cada paquete, BPF agrupa la información de varios paquetes para luego pasarlos todos de una vez. Para mantener la secuencialidad en que se recibieron los paquetes, BPF añade una marca de tiempo, tamaño y *offset* a cada uno de los paquetes del grupo.

Esta característica es importante a la hora de almacenar los datos y después para presentar las gráficas en el sistema.

Es importante mencionar que ADA** no hace uso de los filtros que proporciona Libpcap debido a que se diseñaron otros que pueden realizar el filtrado de manera más específica. Libcap solo lleva a cabo la comunicación con el kernel y le entrega el paquete a ADA**, este último aplica los filtros de su propiedad y aplica la función de muestreo.

5.4 Arquitectura de la aplicación

ADA** está conformado por dos módulos de administración (ver Figura 5.3). El módulo de administración de protocolos y el módulo de administración de filtros. La información que manejan estos módulos es utilizada por el módulo de captura para hacer el filtrado y el almacenamiento de la información si es el caso.

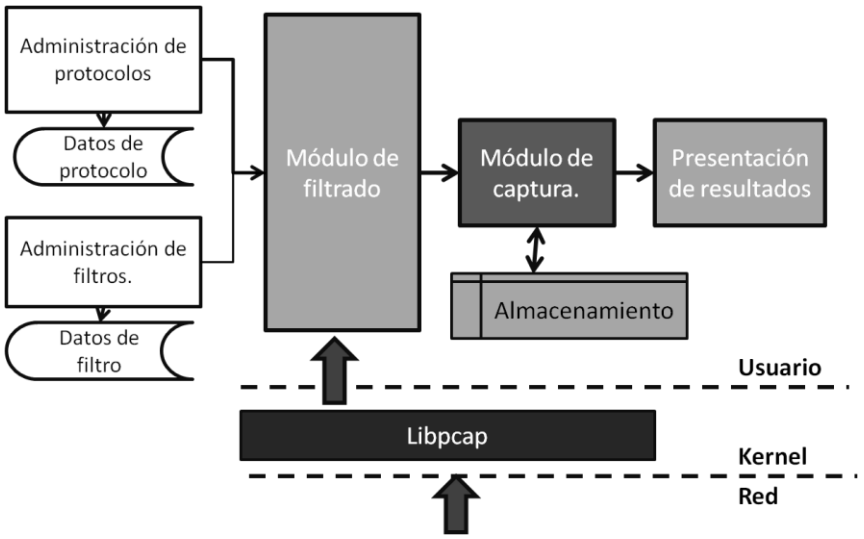


Figura 26 Arquitectura del sistema

Libpcap proporciona los paquetes a ADA** como se vio en el apartado anterior, éste último comienza a revisar la información. Los paquetes que van a formar parte de la muestra se seleccionan aplicando los filtros definidos en el módulo de administración de filtros. Cuando se sabe que el paquete tiene la información solicitada por el filtro, se aplica una función de probabilidad, esta parte se revisa al final de capítulo. Si el paquete cumple con las reglas de filtrado y la aplicación de la función nos dice que debe formar parte del muestreo, se almacena. El almacenamiento puede ser de dos maneras: Sólo encabezados o el paquete completo. A continuación se explica de forma más detallada cada una de las funciones que conforman el sistema ADA**.

5.5 Descomposición en subsistemas

Para definir los filtros es necesario definir primero los protocolos que lo van a conformar. Los filtros y protocolos deben de ser administrados por el sistema de tal manera que se puedan agregar, eliminar y mostrar elementos cada vez que es necesario. Las tres funcionalidades de los módulos de administración se muestran en la Figura 5.4.

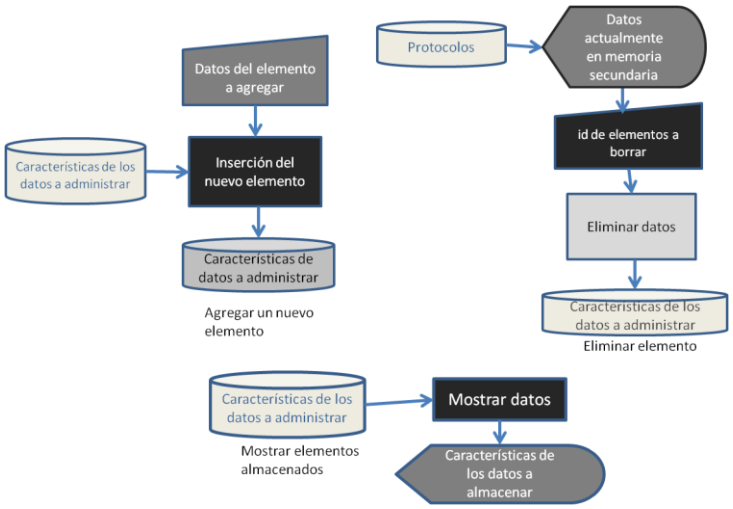


Figura 27 Administración: Agregar, eliminar y mostrar

De manera general, el método utilizado para administrar los protocolos y filtros es muy similar, sin embargo, como veremos más adelante, se diferencian en que la información que maneja cada módulo es de una complejidad diferente.

Administración de protocolos. Esta parte del sistema está diseñada para agregar un protocolo, borrarlo o mostrar los protocolos que están agregados. La parte de los protocolos se maneja de forma diferente para cada capa y se divide de acuerdo al modelo de capas de protocolos de Internet (ver Figura 3.3).

Administración de filtros. Definir un filtro de manera correcta es fundamental para llevar a cabo la selección de la información que será almacenada por ADA**. Este integrante del sistema permite agregar, eliminar y mostrar los filtros que se han dado de alta y está organizado por el número de capa al que pertenece el protocolo que se va a monitorear. Puede tomar los datos del filtro desde la definición del protocolo realizada en

el administrador de protocolos, o pedir los datos al usuario para determinar los campos del encabezado que son de interés para el filtro. Un filtro está conformado por la información que es útil para encontrar y analizar los datos que determinan el comportamiento de ADA** con respecto al paquete actual.

La parte del sistema que hace la revisión del paquete entrante (Figura 5.5), utiliza los filtros definidos para cada capa de la pila de protocolos TCP/IP. Toma el paquete y va comparando encabezado por encabezado. Si el paquete pasa el filtro, se aplica la función de probabilidad, que determina si la información debe de ser almacenada.

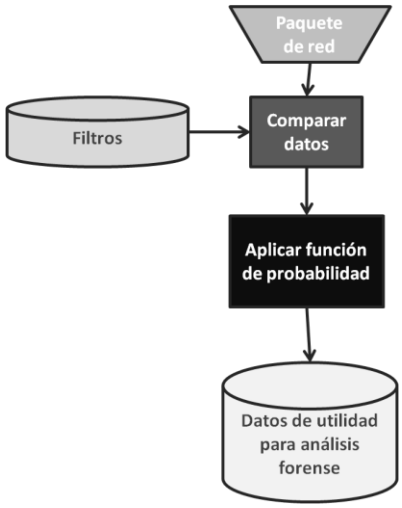


Figura 28 Módulo de filtrado

El módulo de captura se encarga de decidir cuánta información del paquete se va a almacenar en el disco duro. Sólo encabezados que fueron revisados o el paquete completo. Finalmente, la información que ha sido almacenada se presenta por medio de gráficas dependiendo de los datos que se desea conocer.

5.6 Diagramas de flujo

A continuación se muestra el diseño más detallado del sistema utilizando diagramas de flujo. Recordemos que los diagramas de flujo es una técnica de representación de un algoritmo y se dividen en dos grupos [31].

Organigramas: Estos engloban a los diagramas de flujo de configuración en los que se muestra el flujo de datos e información entre los periféricos o soportes físicos (de entrada/salida) que maneja el programa (Ver Anexo C).

Ordinogramas: Se conocen también como diagramas de flujo de programas se utilizan para mostrar la secuencia lógica y detallada de las operaciones que se van a realizar en la ejecución de un programa a través de símbolos (Ver Anexo D).

Los diagramas de flujo permiten mostrar una representación gráfica de la secuencia lógica de las operaciones o acciones que debe de seguir la computadora. Esta característica ofrece una forma didáctica de mostrar el diseño del sistema.

5.6.1 Administrador de protocolos

En primer lugar vamos a revisar el módulo administración de protocolos. Para agregar un elemento (ver Figura 5.6), el sistema pide los datos del nuevo protocolo, para esto utiliza la siguiente estructura.

```
struct protocol{
   unsigned char name[30];      /*nombre del protocolo      */
   unsigned int id;             /*Identificador del protocolo */
   unsigned int end;            /*Si el protocolo es fina 1 */
   unsigned int option;         /* Tamaño del encabezado, '0' si es variable */
   unsigned int position;       /* Posición del campo que contiene el tamaño del encabezado*/
   unsigned int wordSize;       /*Tamaño del encabezado o '0' si es variable*/
   unsigned int size;           /*Tamaño de palabra del campo tamaño del encabezado*/
   int totalItem;               /*Total de campos que conforman el encabezado*/
   struct item *itm;            /*Apuntador a la estructura que contiene los datos de cada campo*/
   struct protocol *next;
   struct protocol *past;
};
```

La mejor característica al definir un nuevo protocolo es que se puede definir el formato que se desea, no se limita a los protocolos conocidos. El usuario define el número de campos que va a tener el encabezado y después el sistema pide que proporcione el

nombre, posición y la medida de cada campo. Esta información es utilizada posteriormente al definir filtros.

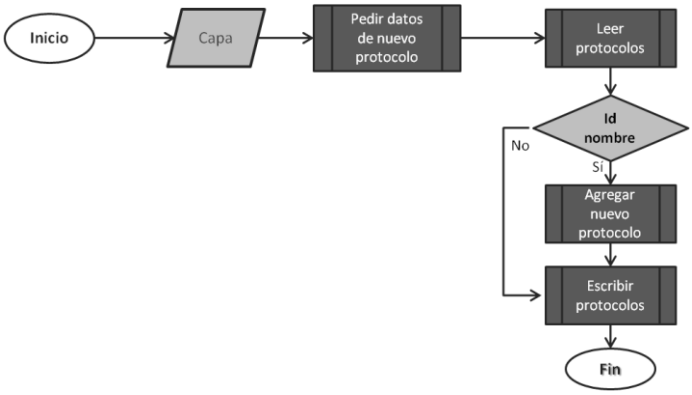


Figura 29 Diagrama de flujo Agregar protocolo.

Después lee los datos que han sido definidos anteriormente y que están guardados en el disco duro, compara el id y nombre del nuevo protocolo con sus contrapartes en cada uno de los protocolos almacenados. Al hacer la comparación el sistema determina si el protocolo no ha sido definido y lo guarda. En esta parte se toma en consideración que id y nombre del protocolo es una relación uno a uno y que no existen dos id con un mismo nombre o dos nombres con un mismo id. Si se encuentra alguna inconsistencia en esta regla, el sistema informa y el nuevo elemento no se agrega.

Para llevar a cabo la funcionalidad borrar un protocolo (ver Figura 5.7), ADA** lee los protocolos que tiene almacenados en la capa especificada por el usuario, los muestra al usuario, pide el id del protocolo que debe eliminar, lo elimina y guarda de nuevo los datos que no se borraron.

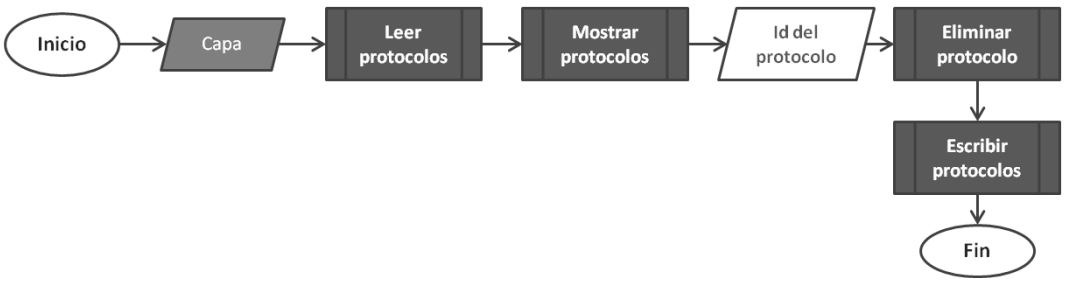


Figura 30Dagrama de flujo Borrar protocolo.

ADA** puede mostrar los protocolos definidos con anterioridad (ver Figura 5.8). Sólo se muestra el nombre e id de cada protocolo, ya que sólo se necesita saber que elementos existen.

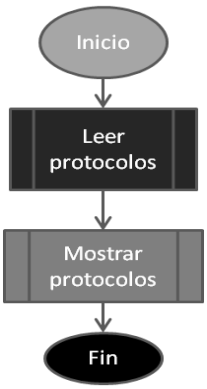


Figura 31Diagrama de flujo Mostrar protocolo.

A continuación se describen las funciones que se utilizan al agregar un nuevo protocolo. Para cada protocolo, ADA** debe solicitar el nombre del protocolo, el número que lo identifica y si es un protocolo final o no; además, se define el tamaño del encabezado en el caso de que sea fijo o la posición en bits y el tamaño del campo que contiene esta información cuando es variable. Otro dato solicitado es el número de campos que conforman el encabezado. Se pone cero si no se van a dar los campos del encabezado. En la Figura 5.9 se muestra el diagrama de flujo de esta última parte.

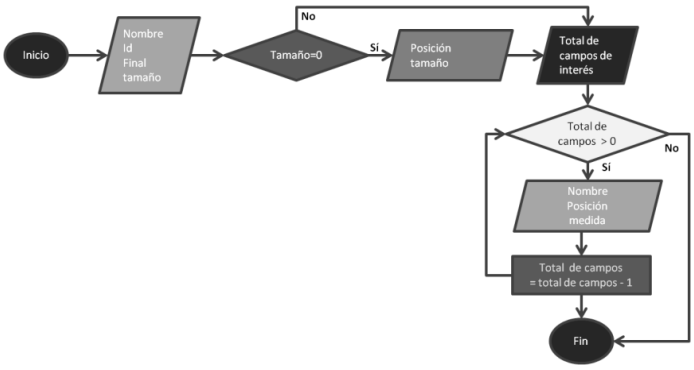


Figura 32 Diagrama de flujo pedir datos de protocolo.

Cada campo del encabezado está definido por cuatro características: nombre, identificador del protocolo, número de bit en donde comienza el campo y tamaño en bits.

Para leer la lista de protocolos de la memoria secundaria (disco duro), se toman en cuenta las mismas consideraciones que se hacen al pedir los datos para un nuevo protocolo (ver Figura 5.10). Se leen todos los protocolos que están almacenados en la lista.

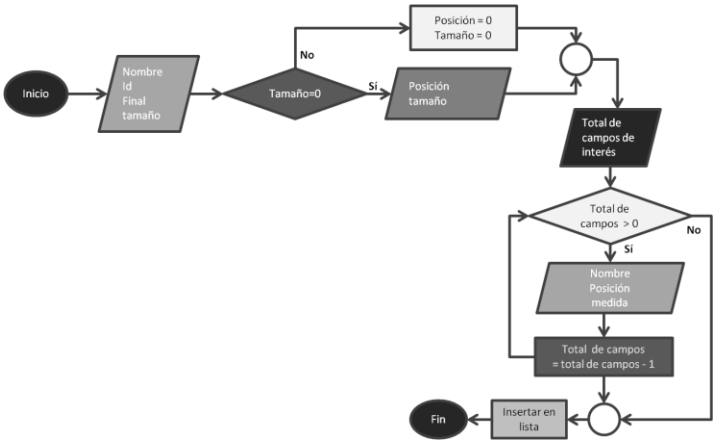


Figura 33 Diagrama de flujo leer protocolos de memoria secundaria.

Después de haber agregado o borrado algún protocolo, se escriben los datos al archivo que guarda los protocolos (ver Figura 5.11). Los archivos disponibles son 3 y 4, representan el número de la capa en la pila de protocolos TCP/IP. Los protocolos para las otras capas se manejan de otra manera como se verá más adelante.

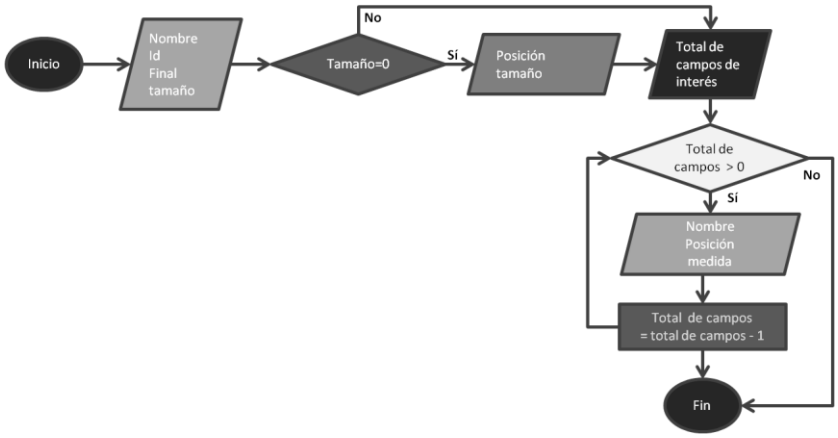


Figura 34 Diagrama de flujo escribir protocolos en memoria secundaria.

La función que se utiliza para mostrar la lista de protocolos que se encuentran actualmente almacenados realiza los siguientes procedimientos (ver Figura 5.12): leer del archivo y mostrar datos. Cabe mencionar que sólo se muestran los campos que definen el protocolo, esto es: nombre e id.

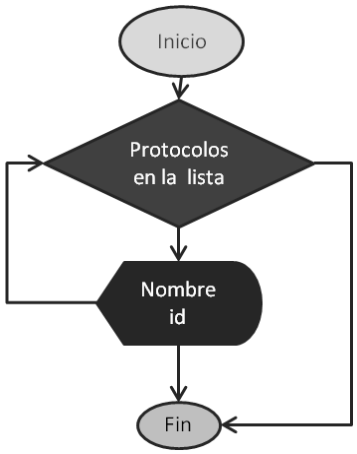


Figura 35 Diagrama de flujo mostrar protocolos.

Para eliminar un protocolo que ya no es útil o que está definido de manera incorrecta se encuentra la opción borrar (ver Figura 5.7). Para eliminar un protocolo del sistema, se leen datos, se elimina de la lista el protocolo elegido y se escriben los datos restantes. Las funciones de leer y escribir son las mismas que se utilizan cuando se agrega un dato. La función que falta analizar es precisamente la que hace el borrado del protocolo y su diagrama de flujo se muestra en la Figura 5.13.

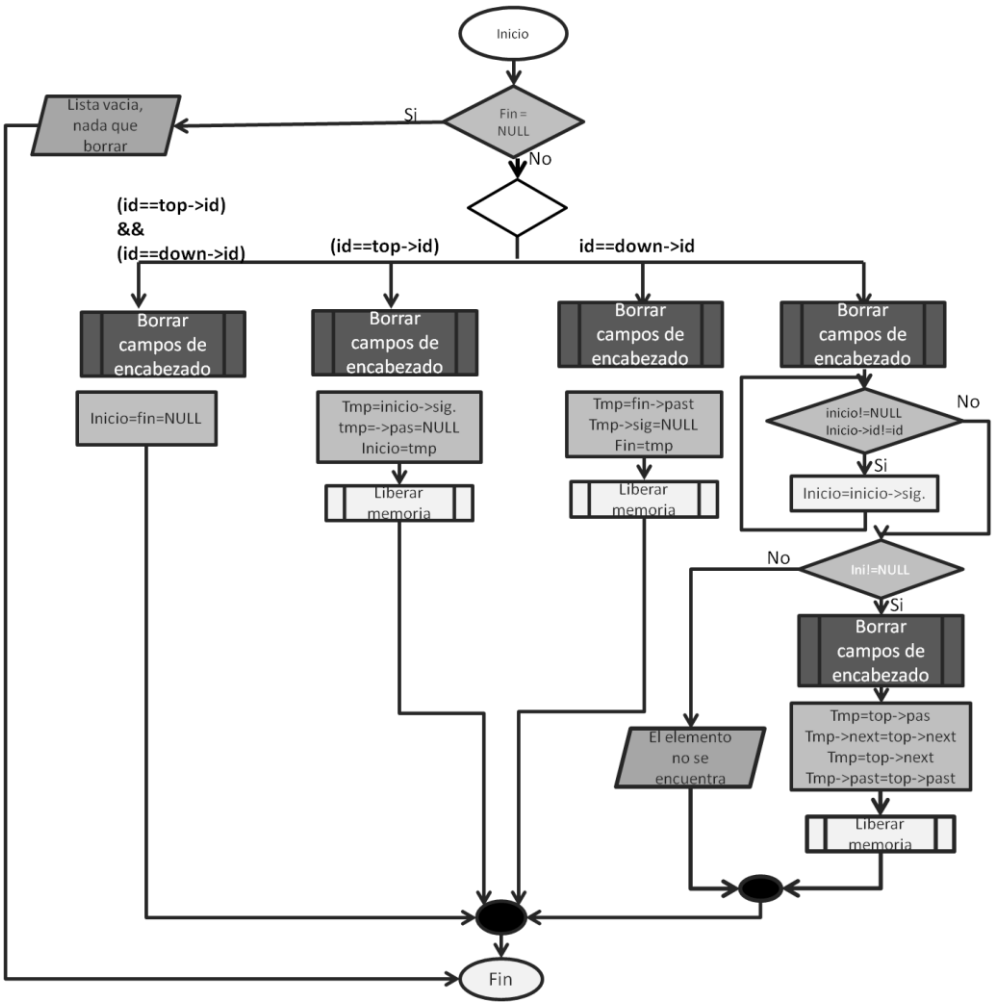


Figura 36 Diagrama de flujo eliminar protocolo de la lista.

5.6.2 Administrador de filtros

El administrador de filtros es otro elemento del sistema, su función es muy parecida al administrador de protocolos; sin embargo, en el administrador de protocolos cada

elemento que maneja el software tiene un identificador único que lo hace diferente a los demás. Esta característica permite que la administración de los protocolos sea sencilla. Por el contrario, el administrador de filtros debe considerar elementos con el mismo identificador, por lo que debe revisar y comparar más minuciosamente cada elemento.

El administrador de filtros agrega, borra y muestra los filtros almacenados.

Para agregar un nuevo filtro lo primero es establecer el protocolo a monitorear y los campos que se van revisar. El sistema muestra los protocolos que se encuentran disponibles, para esto utiliza las funciones de leer protocolo y mostrar protocolo que fueron estudiadas en la sección anterior.

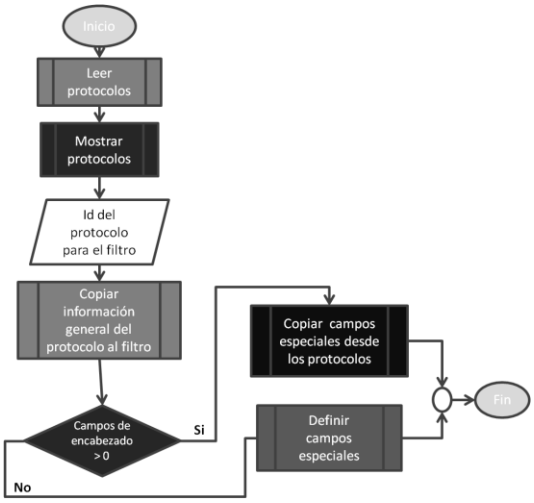


Figura 37 Diagrama de flujo para solicitar datos del nuevo filtro.

El sistema solicita el id del protocolo y copia la información general del protocolo a los datos del filtro (ver Figura 5.14). Para recolectar los datos del filtro se utiliza la siguiente estructura:

```
struct filter{
   unsigned int id;                 /*El id del protocolo actual*/
   unsigned int sizeHdr;            /*Tamaño del encabezado del protocolo*/
   unsigned short positionHdr;/*Posición del campo,tamaño del encabezado*/
   unsigned short sizeWord;   /*Tamaño de palabra el tamaño del encabezado*/
   unsigned int sizeFieldHdr; /*Tamaño de palabra del camp tamaño del encabezado*/
   unsigned short end;              /* Si el filtro es final contiene un uno*/
   unsigned int func;               /*Porcentaje de probabilidad entre 0 y 100%
   unsigned char fileName[20];  /*Archivo de almacenamiento del filtro*/
   unsigned int ttlFields;           /*Número de campos de interés*/
   struct field *fld;                /*Lista de campos de interés    */
   struct filter *next;
   struct filter *past;
};
```

Se revisa si se definieron los datos del encabezado, si es así, se copia la información desde allí para los campos especiales (ver Figura 5.15), si no, se definen los datos de cada campo (ver Figura 5.16).

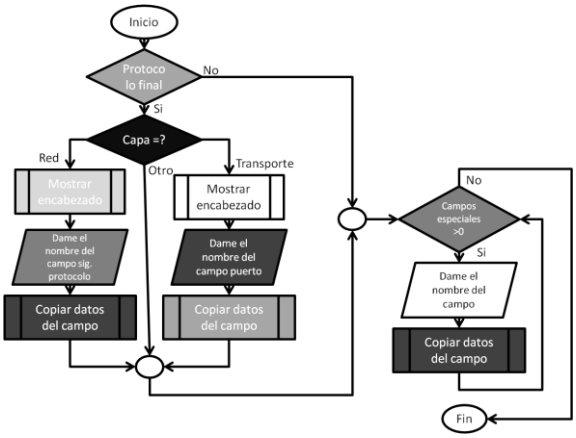


Figura 38 Diagrama de flujo: Copiar datos a campos especiales.

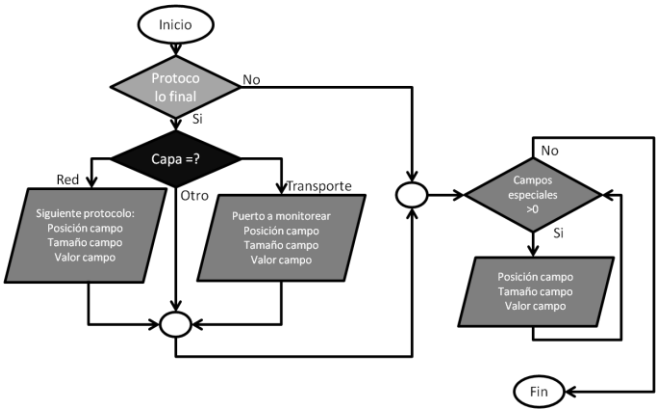


Figura 39 Diagrama de flujo: Definir datos de cada campo.

El siguiente paso es comparar el filtro definido recientemente con los que están guardados en el sistema (ver Figura 5.17). Los elementos están organizados con respecto a su id de mayor a menor, en segundo lugar se organizan con respecto al número de campos de interés, si esto datos son iguales, entonces se comparan los valores de cada campo de interés. La función regresa un uno si encuentra un elemento igual al actual con respecto a los campos comparados, de lo contrario regresa un cero.

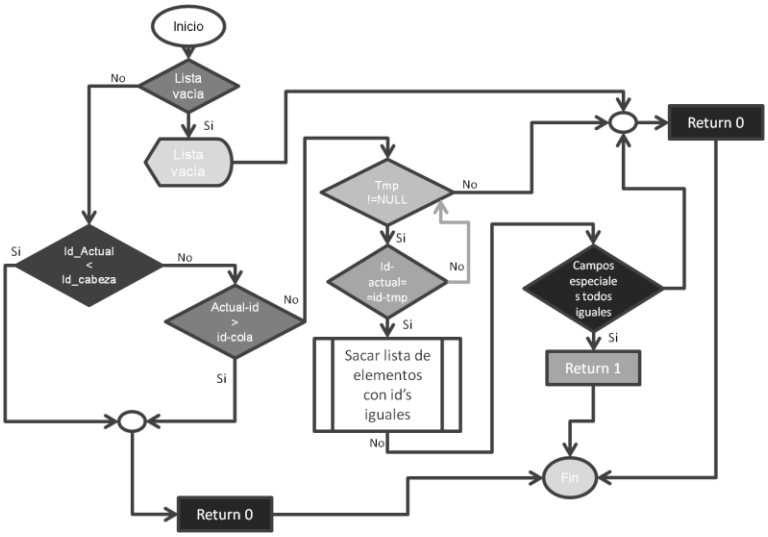


Figura 40 Diagrama de flujo: Comparar filtros.

Si el filtro no se encuentra, se guarda en la lista de filtros utilizando las funciones de insertar en la lista y escribir a archivo. La Figura 5.18 muestra la función de escribir a archivo.

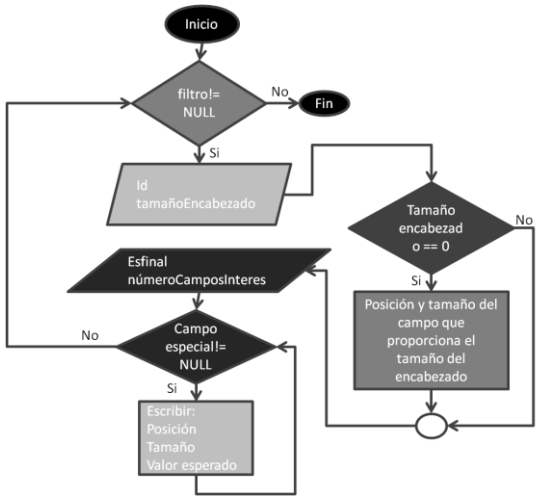


Figura 41 Diagrama de flujo: Escribir datos.

Para ver los filtros que están guardados en el sistema se leen los datos de la memoria secundaria (ver Figura 5.19) y después se muestran en pantalla (ver Figura 5.20).

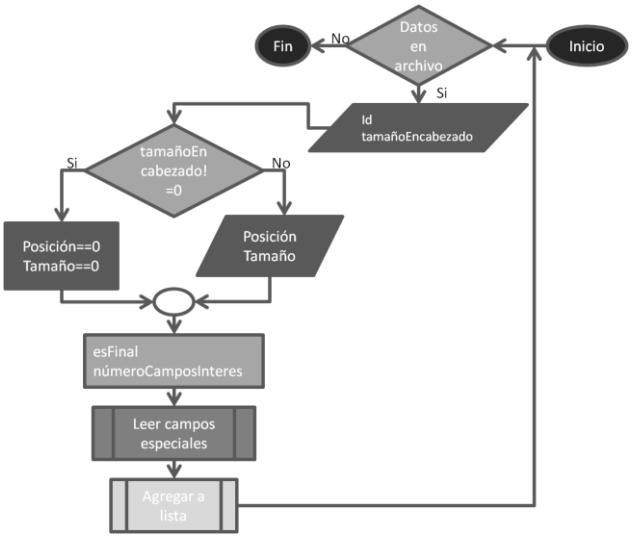


Figura 42 Diagrama de flujo: Leer filtros.

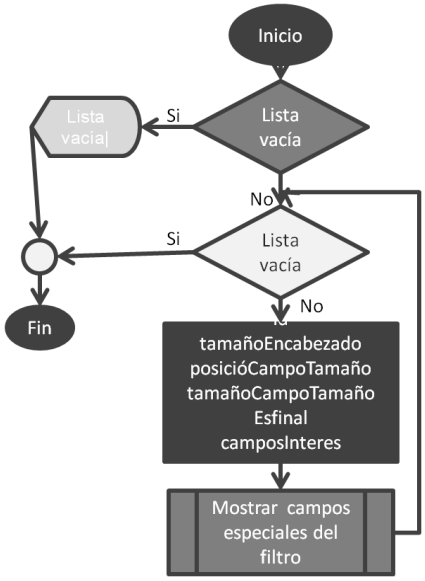


Figura 43 Diagrama de flujo: Mostrar protocolo.

Para actualizar los filtros, se pueden borrar los filtros que están definidos. La Figura 5.21, muestra el diagrama de flujo de esta funcionalidad.

Esta función pide el número de capa del protocolo, lee los filtros definidos en esa capa y los muestra al usuario. La función muestra los filtros ordenados y les da un número. El usuario le da al sistema el número que le fue asignado al filtro para que lo borre.

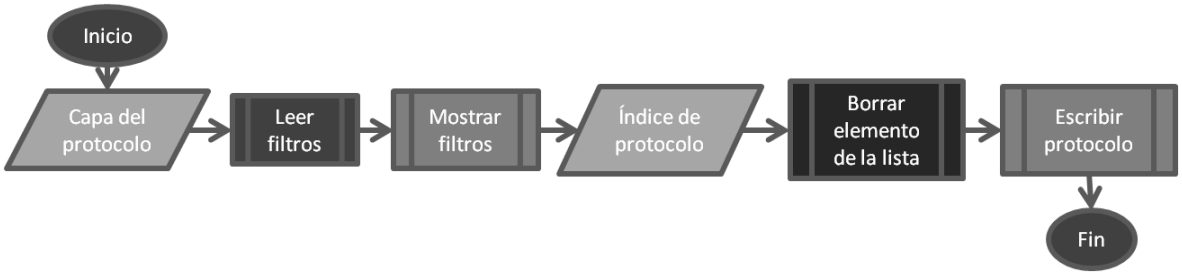


Figura 44 Diagrama de flujo: borrar filtros.

5.6.3 Módulo de filtrado

El módulo de filtrado es una de las partes medulares de ADA**. Aquí se hace uso de toda la información que se ha almacenado hasta el momento para llevar a cabo el filtrado de paquetes (ver Figura 5.22)

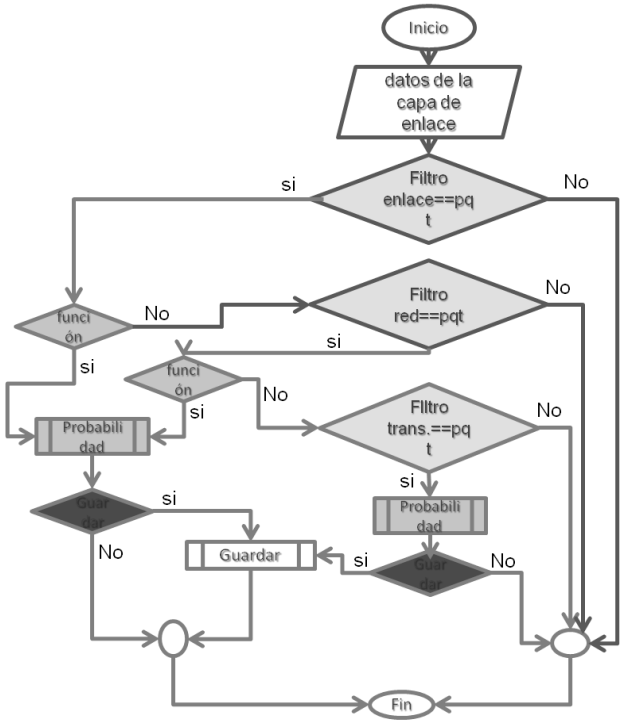


Figura 45 Diagrama de flujo de filtrado de paquetes.

En esta parte se recibe el paquete de red y se aplican los filtros establecidos. Para llevar a cabo la comparación de datos, se utiliza una función especial que traduce los datos que trae cada paquete y se hace una comparación entre los valores obtenidos en el paquete y los valores esperados definidos en el filtro. Si alguno de los pares comparados no coincide, la copia que se obtuvo del paquete, se desecha. Si todos los campos coinciden, entonces se aplica la función para llevar a cabo el muestreo por medio de números aleatorios.

5.6.4 Muestreo

Hasta ahora, ADA** hizo una selección minuciosa del tráfico de red para obtener una población bien definida que engloba todos los paquetes que cumplen con las reglas de un filtro determinado. Sin embargo, la cantidad de datos puede requerir aún un gasto excesivo en recursos de almacenamiento. Con el fin de reducir la cantidad de datos a almacenar sin perder la calidad de la información almacenada, se lleva a cabo un muestreo sobre la población. Un punto que se desea mencionar es que las mejores muestras para estudiar se seleccionan en una forma que proporcione a todo miembro de la población una oportunidad para convertirse en parte de la muestra [34]. Los paquetes que se capturan del tráfico de red resultan difíciles de muestrear directamente si se desea tener la certeza de que el muestreo es representativo de la población, sin embargo existen algunos métodos que resuelven esta problemática.

El muestreo que se lleva a cabo en este trabajo está fundamentado en el muestreo Monte Carlo de manera general y de manera particular en el muestreo de aceptación rechazo. Para fines prácticos, sólo se describen las características del método de aceptación rechazo de manera detallada. Este método hace uso de funciones de distribución sencillas para encontrar otras más complicadas de la siguiente manera: sí f(x) es la función de la que necesitamos muestrear y existe otra función h(x) que es mucho más fácil de muestrear, utilizamos esta última como la propuesta de distribución en el muestreo aceptar rechazar, siempre y cuando satisfaga las siguientes condiciones.

- Existe una constante c >0 tal que, $f(x) \leq ch(x)$ para cualquier x en el dominio de f(x). Esta condición implica que si $f(x) > 0, h(x)$ necesariamente debe de ser mayor que cero. Las bases de h(x) deben contener las bases de f(x).

Se hace el muestreo x* de h(x) y se acepta con la probabilidad de $\frac{f(x*)}{ch(x*)}$ cuando x* es aceptada dado un número aleatorio. El numero aleatorio se encuentra en el intervalo (0,1) y es el asistente para decidir entre aceptar o rechazar una muestra.

Entre mayor es la probabilidad de aceptación $\alpha(x*) = \frac{f(x*)}{ch(x*)}$, la posibilidad de aceptar también es mayor. Cuando $ch(x) = f(x)$ la probabilidad de aceptar siempre es uno, lo que significa que siempre se aceptan las muestras de x.

Para el caso de ADA** El muestreo se realiza primero aplicando una función generadora de un número aleatorio X en el rango [0,1]. El número aleatorio encontrado se mapea al evento actual como variable aleatoria con la siguiente función de distribución.

$$f(x_i) = porcentaje\ de\ peligrosidad$$

El valor de f(x) se basa en el conocimiento adquirido al llevar a cabo la investigación sobre los tipos de ataque más frecuentes. Esta investigación se hizo para tener un mayor conocimiento de las vulnerabilidades de los protocolos, cuáles son los protocolos más utilizados en los ataques y cuáles son los campos del encabezado afectados.

La función h(x) utilizada en este caso es la distribución uniforme y su función de densidad de probabilidad pdf es:

$$h(x) = \begin{cases} \frac{1}{b-a} para\ a \leq x \leq b \\ 0\ para\ x < a\ o\ x > b \end{cases}$$

Como el numero aleatorio se encuentra en el intervalo [0,1], entonces, h(x)=1 en todos los casos y, si c=1, tenemos que el porcentaje de peligrosidad del filtro particular va a determinar la probabilidad de aceptación $\alpha(x_i)$ y el número aleatorio determina si el evento debe de almacenarse o no.

El muestreo se basa en la afirmación de que no importa la probabilidad de distribución verdadera con la que se generen los datos, si se muestrea utilizando una distribución uniforme con una probabilidad lo suficientemente alta, los datos almacenados se comportan de acuerdo a la distribución original.

De esta manera, un paquete que cumple con los requerimientos del filtro es almacenado cuando el valor aleatorio que se genera, se encuentra en el rango de 0 y $\alpha(x_i)$. Por ejemplo, si se establece un filtro para una combinación de protocolo Ethernet-IP-TCP con los campos especiales de una de las combinaciones invalidas (ver Tabla 4.5) en el protocolo TCP: Syn=1, Fin=1, PSH=0 RST=0 y le damos una probabilidad de 100 de que sea un paquete dañino, cada que llegue un paquete se genera una variable aleatoria

entre cero y uno y cómo cien por ciento es igual a 1, tenemos el caso de que f(x) es igual a cg(x) y por lo tanto todo paquete que entra a la tarjeta de red con estas características es almacenado.

5.6.5 Captura de tráfico de red

Para iniciar el proceso de muestreo ADA** pide al usuario la interfaz de la cuál tomará los paquetes. Puede ser eht0, eth1, entre otros. Además, pide el tamaño número máximo de bytes a capturar, si se elige una cantidad de 1518 bytes será suficiente para una interfaz Ethernet v.2, 14 bytes para el encabezado, 1500 bytes de datos [31]. Si no se está seguro del tamaño de paquete, un valor de 6500 es suficiente para cualquier tipo de paquete [32]. Aunque la captura de paquetes con Libpcap puede llevarse a cabo para tamaños de paquete más grandes, dependiendo del tipo de red, ADA** está dirigido a una red de difusión y por lo tanto un tamaño de 1518 es suficiente. Se debe definir el tiempo de espera que es el tiempo que Libpcap esperara antes de pasar los paquetes capturados del espacio Kernel al espacio Usuario. El siguiente paso es definir el tamaño del encabezado, en este caso el encabezado Ethernet mide 14 bytes (ver Figura 6.23) y los campos donde debe buscar el valor del siguiente protocolo es el campo Tipo que mide 2 bytes y comienza en el byte 14.

Figura 46 Encabezado Ethernet II.

En esta parte del filtro pueden definir uno o varios campos especiales para la capa de enlace, cada uno cuenta con un valor esperado y si es el caso, se puede aplicar la función de muestreo desde aquí. Al definir un valor mayor que cero para el campo func (probabilidad), el sistema aplicará la función de probabilidad para llevar a cabo el muestreo y continuará con el siguiente paquete. La estructura que recibe los datos para el filtrado en la capa de enlace es:

```
struct linkLayer{
   unsigned int sizehdr;      /*Tamaño del encabezado*/
   unsigned int position;     /*Posición del campo: siguiente protocolo*/
   unsigned int size;         /*Tamaño del campo: siguiente protocolo*/
   unsigned int ttlField;     /*Número total de protocolos a monitorear*/
   struct value *fld;
};
```

```
struct value{
  unsigned int value;
  unsigned int func;
  struct value *next;
};
```

Después de definir los datos de la capa de enlace se arranca el análisis del tráfico de red. El sistema toma el encabezado Ethernet, en este caso, y traduce el campo tipo con la función que traduce los datos.

Esta función va a tomar el tamaño y la posición del campo de datos que va a traducir. Recibe el paquete, hace un desplazamiento a la posición, toma el número de bits indicado en tamaño y lleva a cabo la traducción a formato hexadecimal.

La función es capaz de traducir datos de tamaño de un bit, dos, ocho, dieciséis, treinta y dos, etc. Es importante que se pueda traducir cualquier parte del encabezado, incluso si sólo se desea filtrar por el valor de una bandera del encabezado que puede ser de tamaño de un bit. Si el dato traducido no concuerda con alguno de los valores definidos para la capa de enlace, se desecha y se comienza a analizar el siguiente paquete. Pero si el dato concuerda, entonces se busca el siguiente encabezado del paquete. Este procedimiento continua hasta la capa de transporte o cuando el protocolo es final.

```
unsigned int translateData(unsigned char **packet, unsigned int size, unsigned int position){
    unsigned int l, m=0, min,max, offset=0;
    unsigned char *p;
    p=NULL;
    p=*packet;
    if(!(size%8)){
        if(size==EIGHT){
            m=(unsigned int)p[0];
        }else{
            max=size-EIGHT;
            for(min=0;min<size/EIGHT;min++){
                l=(unsigned int)p[min] & 0xff;
                l=l<<max;
                max=max-EIGHT;
                            m=m|l;
                }
            }
    }else{
        if(!(position%8)){
            if(size<EIGHT){
                m=(unsigned int)p[0];
                m=m>>(EIGHT-size);
                    }else{
                max=size-EIGHT;
                for(min=0;min<size/EIGHT;min++){
```

```
                l=(unsigned int)p[min] & 0xff;
                l=l<<max;
                max=max-EIGHT;
                m=m|l;
            }

l=(unsigned int)p[min+1];
            l=l>>(EIGHT-(EIGHT-(size%EIGHT)));
                m=m|l;
            }
        }else{
            if(size<EIGHT){
                unsigned int pvt=255;
            m=(unsigned int)p[0];
                pvt=pvt>>(EIGHT-size);
                m=m&pvt;
            }else{
                unsigned int pvt=255;
            m=(unsigned int)p[0];
            max=size-EIGHT;
                pvt=pvt>>(EIGHT-(size%EIGHT));
                m=m&pvt;
            max-=EIGHT;
            for(min=1;min<size/EIGHT;min++){
                l=(unsigned int)p[min] & 0xff;
                l=l<<max;
                max-=EIGHT;
                m=m|l;
            }
          }
        }
    }
    return m;
}
```

5.7 Interfaz Gráfica

La interfaz gráfica de ADA** tiene un concepto minimalista y no ocupa más espacio del necesario para llevar a cabo su funcionamiento. Las pantallas de funcionalidad del sistema son.

5.7.1 Administrador de protocolos

Administrador de protocolo: La pantalla principal del administrador de protocolos (ver Figura 5.24) es la parte central para acceder a las funciones de mostrar, insertar y borrar. En primer lugar el administrador solicita la capa de protocolo.

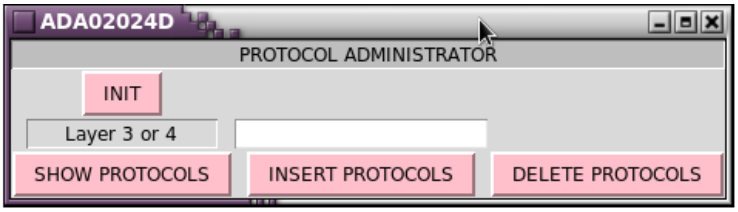


Figura 47 Administración de protocolos.

Al presionar alguno de los botones se cambia de pantalla. Para **agrega**r un nuevo protocolo el sistema muestra una pantalla con las opciones generales del protocolo (ver Figura 5.25).

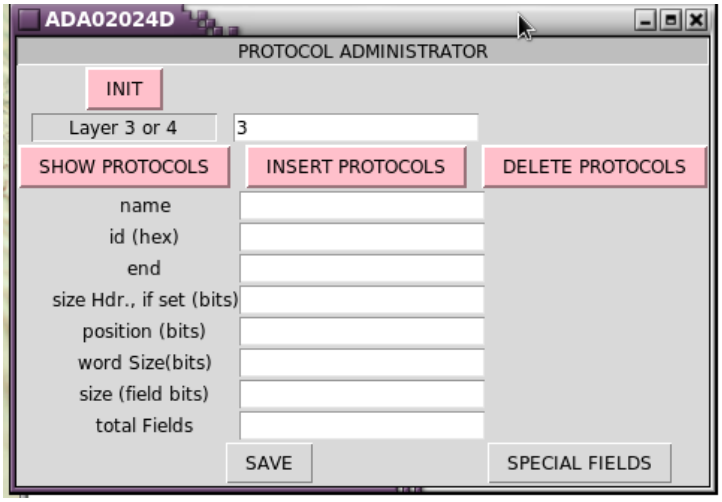


Figura 48 Campos generales del protocolo.

Los datos se guardan (SAVE) y se llama a los campo especiales, en donde se van a definir los valores que tiene cada campo del encabezado del protocolo (ver Figura 5.26).

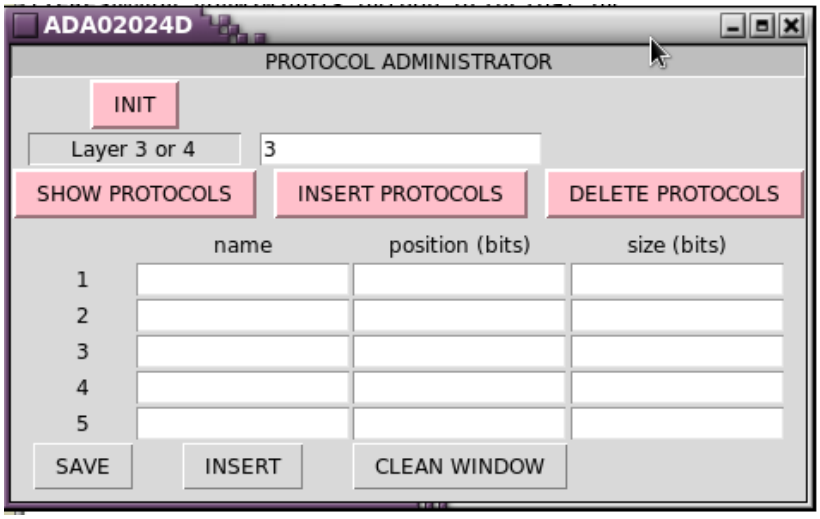


Figura 49 Campos de encabezado del protocolo.

Existen tres campos por cada campo de encabezado. El primero es una cadena y los otros dos son enteros para indicar la posición donde comienza el campo y el tamaño de este.

Los datos se guardan (SAVE) y se agregan (INSERT).

La opción **mostrar** protocolos lee los datos del archivo de almacenamiento y muestra los campos Id y Nombre (ver Figura 6.27).

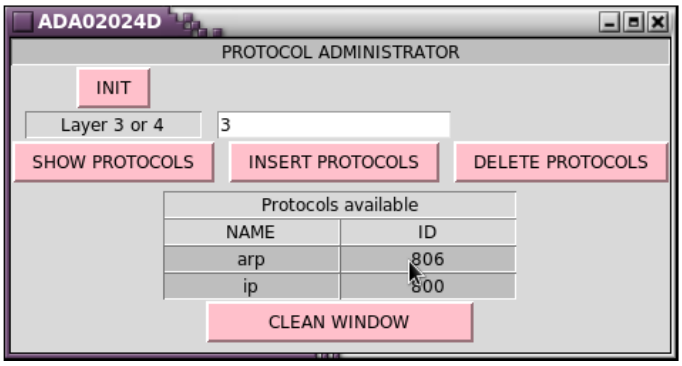


Figura 50 Mostrar protocolos.

Para **borrar** un protocolo, ADA** muestra los que están actualmente almacenados y el usuario debe proporcionar el nombre del que va a borrar (ver Figura 6.28).

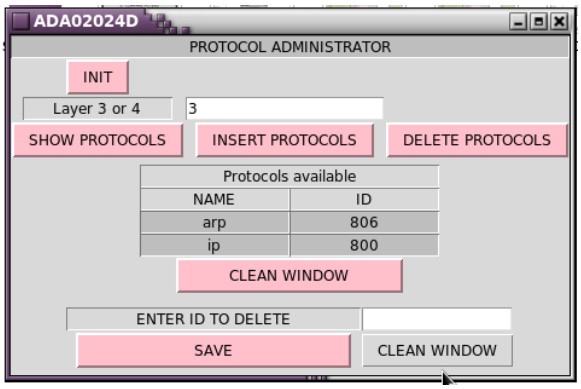


Figura 51 Borrar protocolo.

5.7.2 Administrador de filtros

Administrador de filtros: La pantalla principal del administrador (ver Figura 5.29) de filtros es la parte central para acceder a las funciones de mostrar, insertar y borrar.

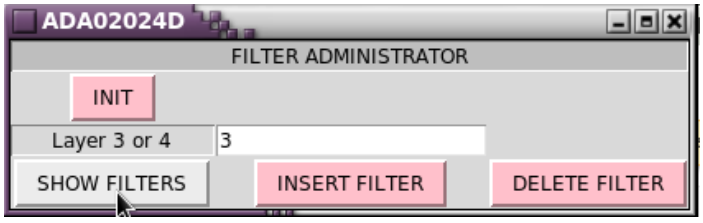


Figura 52 Administrador de filtros.

Las funciones de mostrar, borrar e insertar filtros, tiene el mismo funcionamiento que las funciones para los protocolos. Sin embargo, son datos más complejos de manejar.

Para **agregar** un nuevo filtro el sistema pide los datos generales que ya tiene (ver Figura 5.30). Muestra los protocolos disponibles para construir el filtro. Los datos agregados se guardan y se cambia a la pantalla de los campos especiales.

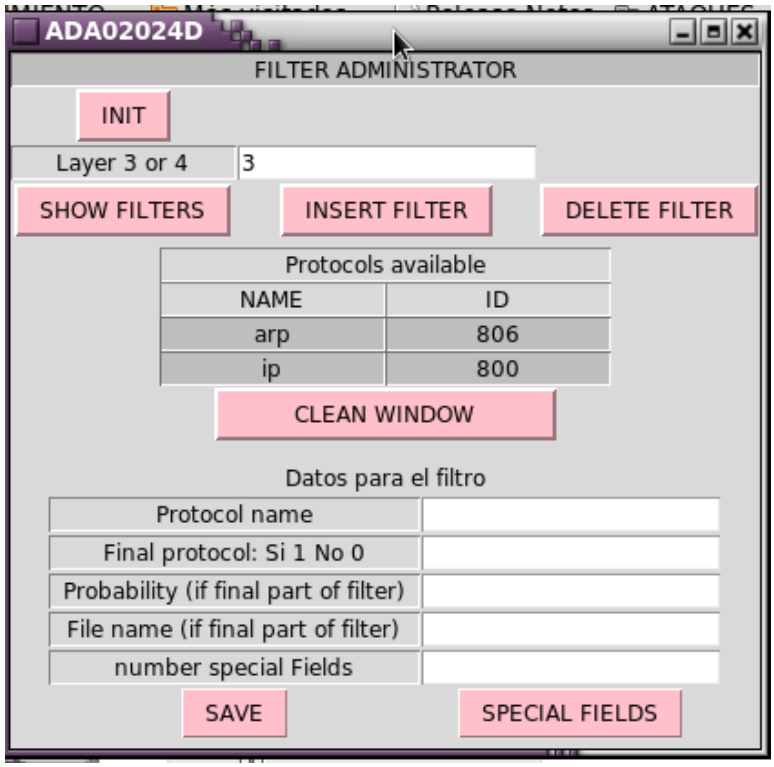


Figura 53 Campos generales para el filtro.

Para agregar los campos hay dos posibilidades: que el encabezado del protocolo tenga los campos definidos o que no los tenga. En el primer caso, ADA** presenta las opciones disponibles para los campos especiales (ver Figura 5.31) y en el segundo el usuario tendrá que agregar todos los datos.

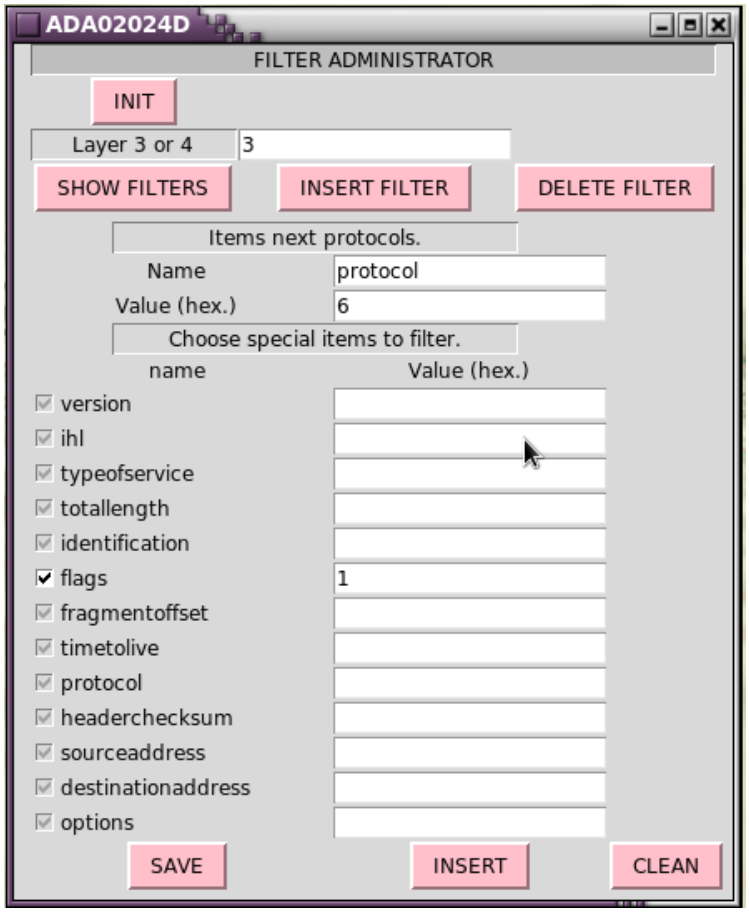


Figura 54 Campos especiales para el filtro.

Los dos primeros campos aparecen cuando el protocolo no es final, es decir, debe conectarse con el siguiente encabezado. Entonces, se solicita al usuario que proporcione el nombre del campo que contiene el valor del siguiente encabezado. Todos los campos definidos aparecen a continuación con una entrada para asignar el valor esperado en hexadecimal.

Cuando se muestran los filtros almacenados se muestran todos los campos almacenados para cada filtro (ver Figura 5.32). Esto es con la finalidad de detectar algún dato incorrecto y poder redefinir el filtro.

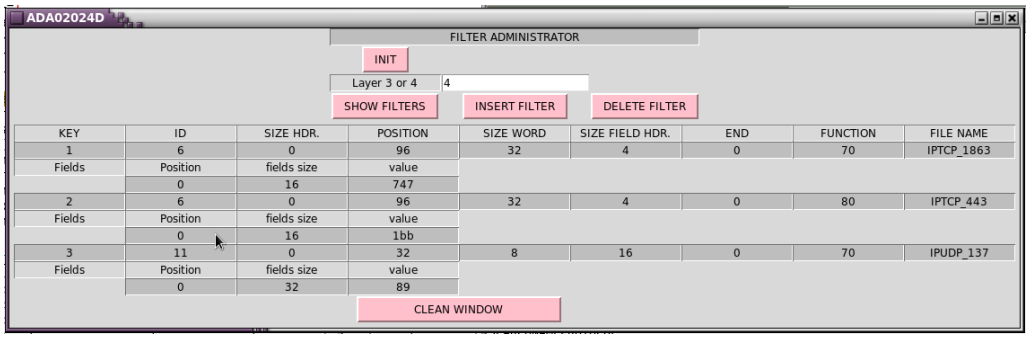


Figura 55 Mostrar filtros.

Primero se muestran los campos generales, en las siguientes filas se muestran los campos especiales que tenga definido el filtro.

La opción borrar filtro, solicita el valor key para buscar el filtro a borrar.

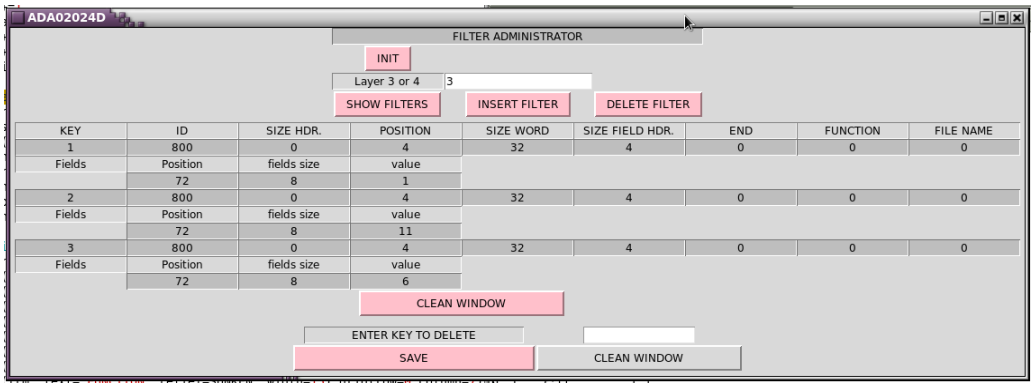


Figura 56 Borrar filtro.

Después de definir los protocolos y posteriormente construir los filtros, se puede llevar a cabo la captura de paquetes.

5.7.3 Captura de paquetes

Captura de paquetes: Para iniciar la captura de paquetes se recopilan los datos de la capa de enlace (ver Figura 5.34). Los datos introducidos se guardan (SAVE DATA). Con el botón “SPECIAL FIELDS” de llama a la pantalla de captura de datos especiales.

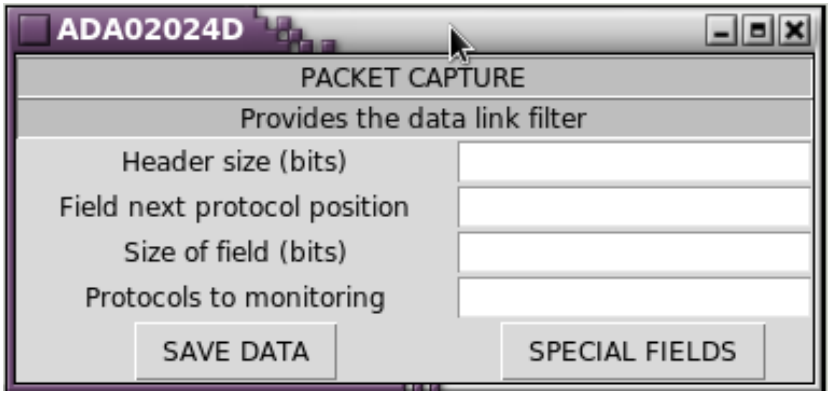


Figura 57 Datos generales para la estructura de la caca de enlace.

La pantalla de captura de campos especiales (ver Figura 5.35) va a mostrar dos campos: Value y Function, por cada campo especial en el encabezado.

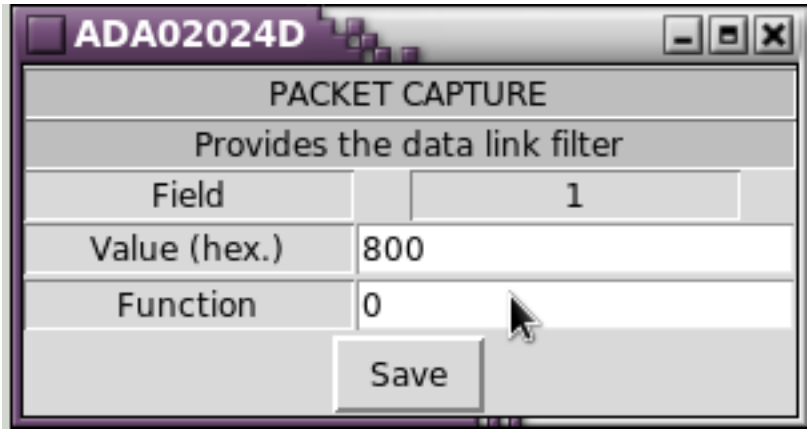


Figura 58 Campos especiales.

Finalmente se piden los datos que necesita Libpcap para llevar a cabo la captura (ver Figura 5.36).

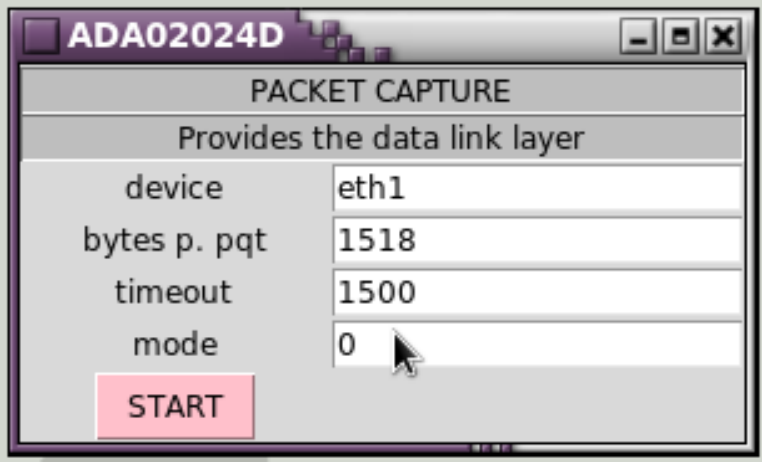


Figura 59 Datos para Libpcap.

En este punto se presiona Start y ADA** comienza a capturar, filtrar y almacenar datos.

Los datos están almacenados en un archivo diferente para cada filtro. Con estos datos se puede dar inicio a las pruebas. Revisando los datos almacenados y comparando con los filtros.

5.8 Resumen

La implementación de ADA** fue una gran experiencia en todos los sentidos ya que se exploró entre diferentes lenguajes de programación. Una parte complicada fue la comunicación entre Python y C. La entrega y recepción de datos para posteriormente darles formato en Python, requirió de un poco de programación adicional. El sincronizar además, Libpcap con C y después con Pytnon fue un punto importante. La compilación se hizo de manera especial para que estos tres factores pudieran comunicarse e intercambiar información entre sí.

En el siguiente capítulo se hacen el reporte de las pruebas que se llevaron a cabo para verificar el funcionamiento de ADA**.

CAPÍTULO 6. PRUEBAS Y RESULTADOS

En este capítulo se comentan las pruebas hechas en ADA** que muestran su funcionalidad y al mismo tiempo se comentan los resultados obtenidos. La primera prueba fue una comparación entre ADA** y Wireshark sin filtros, atrapando todo el tráfico de red. Posteriormente se definieron diferentes filtros (población) y se obtuvo el tráfico filtrado (muestra) para finalmente, obtener la información estadística.

6.1 Captura de paquetes ADA** vs Wireshark

Esta prueba tiene el objetivo de verificar que ADA** recolecta por lo menos la misma cantidad de paquetes que Wireshark. Wireshark es un analizador de protocolos de red popular que utiliza Libpcap como librería para hacer la captura de paquetes de tráfico de red.

La prueba se lleva a cabo sobre un sistema Linux Fedora 14 en donde están instalados: Wireshark y ADA**. La red a la que está conectada la computadora, es una red de difusión casera. La computadora, esta accediendo a YouTube para visualizar videos y ADA** y Wireshark se inicializan al mismo tiempo para capturar paquetes de la red en modo promiscuo. La captura se lleva a cabo de manera continua durante siete horas.

Para ADA**, se define el tamaño de encabezado para el protocolo de enlace Ethernet v. 2 y se pone cero en el número de protocolos a monitorear. Por lo tanto la captura se lleva a cabo sin filtros.

En el caso de Wireshark, tampoco se establecen filtros. Los paquetes capturados son el total de paquetes que pasó por el dispositivo de red de la computadora. Después de siete horas de recolectar paquetes, ADA** terminó con un total de 1099990 y

Wireshark con 1098595 (ver Figura 6.1). En esta prueba ADA** recolecto 1395 paquetes más que Wireshark por lo que la prueba se considera exitosa.

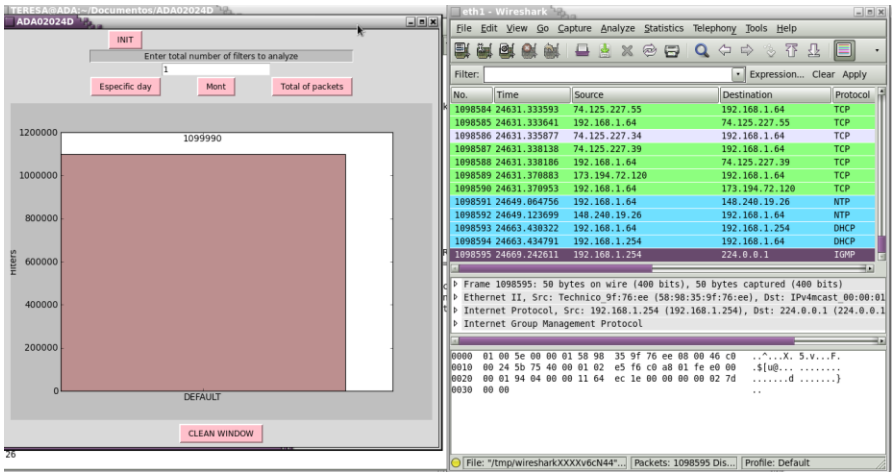


Figura 60 Total de paquetes en ADA** vs Wireshark

Con los paquetes almacenados, se puede llevar a cabo un análisis del número de paquetes almacenados por hora (ver Figura 6.2).

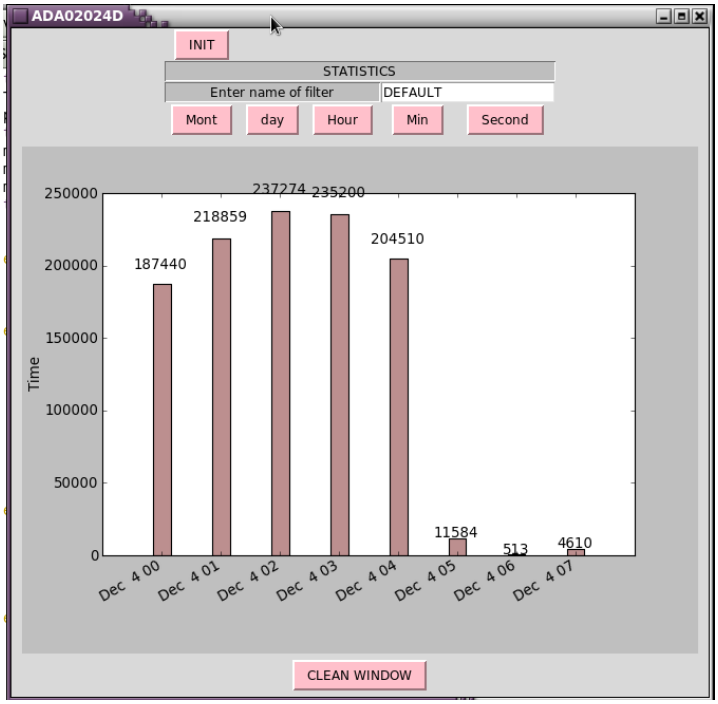


Figura 61 Análisis de paquete por hora

En la gráfica se puede apreciar el flujo de paquetes desde la hora cero, que es la hora en que se inicio la captura. Se puede ver que la mayor cantidad de paquetes capturados fue entre las dos y tres de la mañana. Al final de la prueba hay una disminución drástica de paquetes capturados, la razón: uno de los videos quedo parado.

6.2 Análisis de paquetes

La prueba que se propone es verificar que el sistema ADA** es capaz de recolectar la información que se define en los filtros y mostrar la información en hexadecimal que el usuario puede consultar.

Para esta prueba se definieron los protocolos de la Tabla 6.1.

Tabla 6.1 Protocolos por capa

Capa	Protocolo
3	IP
4	ICMP
4	TCP
4	UDP

Tomando en cuenta los protocolos definidos, se agregan los filtros de la tabla 6.2.

Tabla 6.2 Filtros para la capa tres

Número de filtro	1	2
Id	800	800
sizeHdr	0	0
positionHdr	4	4
sizeWord	32	32
sizeField	4	4
End	0	0
Func	0	0
fileName	0	0
ttlFields	1	1
Fld->position	72	72
Fldd->size	8	8

Fld->value	6	6
Fld->position	48	48
Fldd->size	3	3
Fld->value	2	0

Se puede ver que los dos son filtros para el protocolo IP y que están encadenados con el protocolo con valor hexadecimal 6 (TCP) y ambos están rastreando las banderas del encabezado.

Un protocolo busca las banderas con valor 2 que es la secuencia de bits [0 1 0] que significa: no fragmentar, último paquete.

El otro busca los paquetes con valor cero [0 0 0], que quiere decir: podría fragmentarse, último paquete.

Para la capa cuatro se tienen tres protocolos: ICMP, TCP, UDP. Cabe mencionar en este punto que, el protocolo ICMP pertenece a la capa 3 como un subprotocolo de IP, sin embargo para ADA** se encuentra en la capa cuatro porque es el tercer protocolo en la cadena del filtro. Los filtros definidos se muestran en la Tabla 6.3.

Tabla 6.3 Filtros para la capa cuatro

Núemero de filtro	1	2	3	4	5	6
Id	6	6	6	6	6	6
sizeHdr	0	0	0	0	0	0
positionHdr	96	96	96	96	96	96
sizeWord	32	32	32	32	32	32
sizeFieldHdr	4	4	4	4	4	4
End	0	0	0	0	0	0
Func	95	50	90	90	70	89
fileName	IPTCP5	IPTCP_5	IPTCP_1	IPTCP_2	IPTCP_4	IPTCP_3
ttlFields	1	1	1	1	1	2
Fld->position	0	0	0	0	0	0
Fld->size	16	16	16	16	16	16
Fld->value	50	50	50	50	50	50

Fld->position	109	111	110	110	107	107
Fld->size	1	1	1	1	1	1
Fld->value	1	1	1	0	1	1
Fld->position						110
Fld->size						1
Fld->value						1

Todos los filtros definidos en esta capa son del protocolo 6 (TCP), para este protocolo el tamaño de encabezado puede variar y el valor del tamaño se busca en la posición 96. No es un protocolo final, entonces, se debe buscar el siguiente protocolo que se especificó en la primer tripleta de campos especiales (fld->). Si se revisa la primera tripleta, el filtro va a buscar valor 50 para siguiente protocolo, que es el número de puerto reservado para http.

Los otros campos son similares y al revisar se observa que los datos buscados son las banderas del encabezado TCP: 107 (ACK), 109 (RST), 110 (SYN), 111 (FIN).

Cada que ADA** lee un filtro nuevo con un nombre de archivo especificado, se crea este archivo para grabar los datos capturados. Al iniciar la captura y revisar los resultados, se observa que sólo se crearon tres archivos IPTCP_1, IPTCP_2 y IPTCP_3 (ver Figura 6.3).

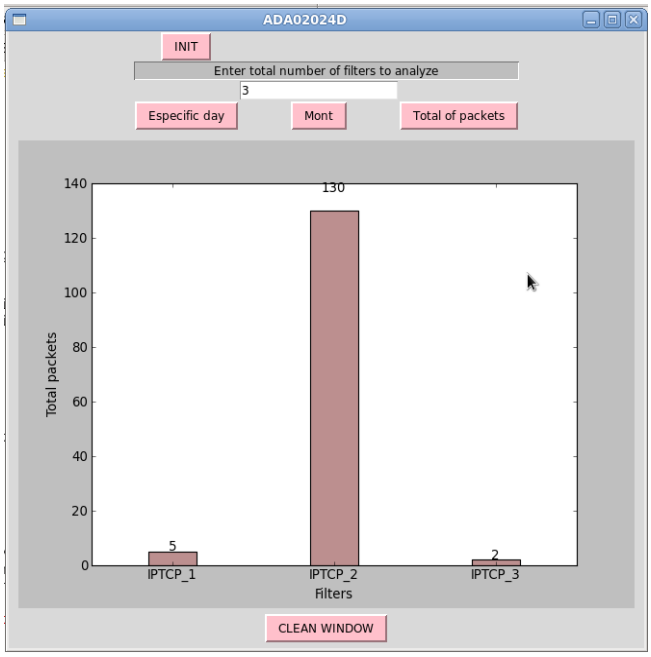


Figura 62 Total de paquetes

Esto quiere decir que los demás filtros no obtuvieron resultados. La cantidad de paquetes por filtro fue para IPTCP_1 5 paquetes, para IPTCP_2 130 paquetes y para IPTCP_3 2 paquetes.

Los paquetes recibidos fueron 5 paquetes con la bandera SYN en uno, 122 paquetes con la bandera SYN en cero, que nos dice que el paquete no es de reconocimiento y dos paquetes SYN/ACK.

El siguiente paso es la revisión de la información almacenada. El filtro a revisar es IPTCP_1 Figura 6.3 que reporta todos los paquetes de tamaño 54. El primer paquete guardado con la bandera SYN en uno

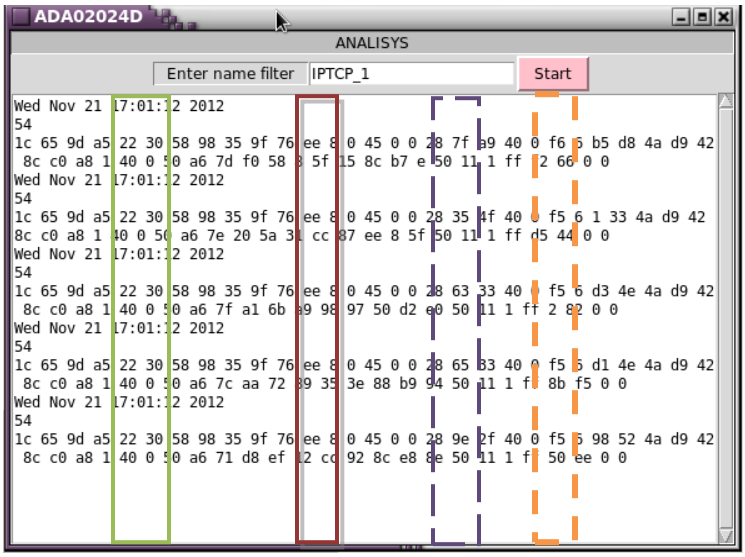


Figura 63 Información en hexadecimal.

La posición 12 de tamaño dos bytes tiene el valor 800 (color rojo) lo que indica que el siguiente encabezado a buscar es un IP. En el byte 10 del siguiente encabezado se tiene un 6 (color naranja) esto indica que el siguiente protocolo esperado es un encabezado TCP, finalmente, en el encabezado TCP, en los primeros dieciséis se encuentra un valor 0 50 (color verde), esto indica que el siguiente protocolo es de tipo http.

Para revisar los campos de interés se revisa el byte 14 del encabezado TCP y encontramos el valor 11 hexadecimal (color morado), si lo traducimos a binario tenemos: 00010001, los seis últimos valores de este octeto representan las banderas URG, ACK,

PSH, RST, SYN, FIN. Concluimos entonces que todos los paquetes capturados con este filtro son paquetes que indican el final de la comunicación ACK/FIN, el servidor ya no enviará más datos.

6.3 Estadísticas

La siguiente prueba se ejecuta en una computadora con Fedora Linux 14 en una red escolar. En esta red no se utilizan los puertos bien conocidos como veremos en el desarrollo de la misma. Los filtros definidos en ADA** son los de la Tabla 6.4, no se definen campos especiales y los puertos para los protocolos https, msnp, netbios-ns en su valor hexadecimal son 1bb, 747, 89 sucesivamente.

Tabla 6.4 Filtros definidos.

Filtro	Capa 1	Capa 2	Capa 3	Capa 4
1	Ethernet v. 2	IP	TCP	HTTPS
2	Ethernet v. 2	IP	TCP	MSNP
3	Ethernet v. 2	IP	UDP	Netbios-ns

ADA** comienza a tomar los paquetes de la red y se lleva a cabo el filtrado de manera continua por dos horas aproximadamente. Se almacenaron datos para el filtro uno y dos, sin embargo, el filtro tres no obtuvo resultados.

Después de la recolección de paquetes se hace un muestreo de la cantidad de paquetes obtenidos por hora (ver Figura 6.5) con el filtro 1 que fue nombrado como IPTCP_443. La línea horizontal representa la media del proceso y a partir de ella se pueden ver las diferencias sobresalientes en los datos. ADA** hace el cálculo de la media, la varianza y la desviación estándar del numero de paquetes capturados a través del tiempo.

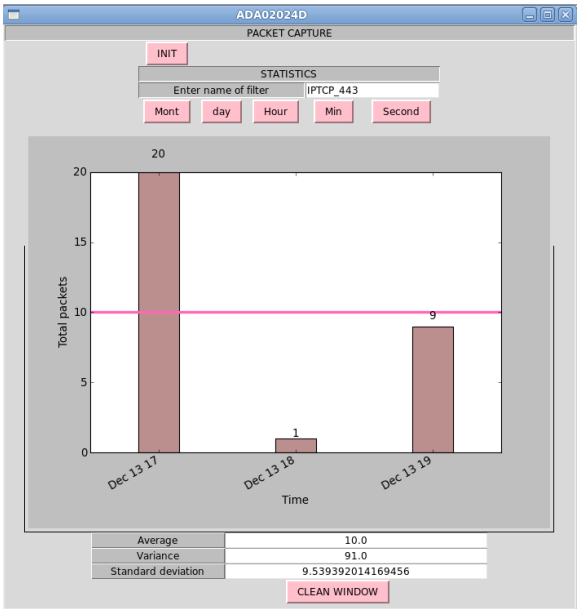


Figura 64 Cantidad de paquetes por hora.

Una gráfica más detalla del tiempo de recolección de paquetes es la representación en segundos (ver Figura 6.6). La media en este caso es de 3, la varianza 11.33 y la desviación estándar 3.36.

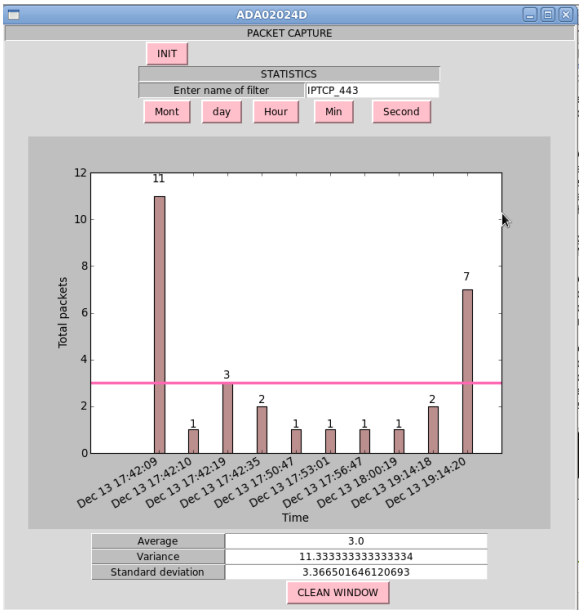


Figura 65 Total de paquetes por segundo.

6.4 Resumen

ADA** Es una herramienta de mucha utilidad para el muestreo de tráfico de red, a diferencia de otras plataformas cerradas, acepta la definición de protocolos no estandarizados. Debido a que el filtrado del tráfico puede ser muy específico, se pueden hacer estudios con fines de investigación. Dependiendo del tipo de red, el tráfico que transita por ella suele tener un comportamiento regular. Llevando a cabo un registro de las estadísticas, los comportamientos del tráfico irregulares requerirán un estudio de los datos almacenados más detallado. La función de probabilidad reduce la cantidad de información almacenada y esto facilita la búsqueda de cadenas en la información hexadecimal.

Por otro lado, la idea de que la recolección de paquetes es distribuida (todas las máquinas son recolectoras), mantiene la información segura porque si se alteran los datos de una computadora, existen otras que contienen un respaldo y la información almacenada se puede verificar.

CAPÍTULO 7. CONCLUSIONES

7.1 Logros Alcanzados

Se llevó a cabo un estudio exhaustivo sobre los principales ataques a las redes de computadoras. Los resultados de esta investigación se encuentran en el capítulo 4 de este documento.

Para determinar qué características deben cumplir los datos recopilados por un sistema, se realizaron las siguientes actividades:

- Se hizo una investigación amplia a través de algunas publicaciones que hablan sobre la problemática forense y otros que abordan alguno de los problemas con soluciones que han sido referenciadas a lo largo del capítulo 2.
- Asistencia al “Congreso de seguridad en cómputo 2011” para los talleres: análisis forense en sistemas Linux y análisis forense de tráfico de red.
- Se hizo una investigación buscando la legislación que existe en México con respecto a este tema y esta referenciado en el anexo B.

Se implemento una herramienta forense ADA** que lleva a cabo la recolección de paquetes de tráfico de red con la ayuda de Libpcap. Los elementos ADA** son:

- Administrador de protocolos. La finalidad de este módulo es brindar una manera de especificar protocolos que están estandarizados o los que no lo están. Esto se logra dando al usuario la facilidad de especificar el tamaño y posición de cada campo.

- Administrador de filtros. De acuerdo a los protocolos existentes en el sistema, se hace la definición de filtros de tal forma que se puede almacenar un paquete por el valor que tiene uno sólo de sus bits.
- El muestreo se lleva a cabo con base en el método Monte Carlo. El muestreo Montecarlo tiene la ventaja de que las muestras preservan las distribuciones originales de la población completa (de todos los flujos de datos).
- El módulo fue probado mostrando que ADA** contiene los datos que se especificaron en los filtros.

Se implemento un módulo en ADA** que crea un archivo para cada filtro definido guardando la información en hexadecimal y la mantiene disponible para su revisión posterior.

Se desarrolló un módulo básico para la presentación de estadísticas y su funcionamiento se explica en el capítulo 5 (Análisis y diseño del sistema de captura aleatoria ADA**).

7.2 Trabajo a futuro

El cómputo forense es una ciencia que está emergiendo. El elevado desarrollo en tecnologías de comunicación por Internet, así como el creciente comercio electrónico con la consecuente necesidad de manejo de información personal a través de Internet y el casi nulo desarrollo de legislación en contra de delitos cometidos por este medio. Hace de la red un lugar favorito para cometer crímenes y los atacantes no necesitan preocuparse mucho por ser atrapados.

Actualmente las empresas cuentan con sofisticadas herramientas de seguridad para proteger su red de posibles ataques pero no invierten muchos recursos en la posible obtención de información cuando el delito ya ocurrió. Para continuar con el desarrollo de herramientas enfocadas a la investigación forense se propone lo siguiente:

- Desarrollar versiones que se ejecuten como parte del núcleo de Linux.

  - Reducir los costos en cuanto al consumo de recursos y reducir la vulnerabilidad del sistema de recopilación y almacenamiento de datos.

- Desarrollar herramientas de análisis de la información recopilada.
  - Mejorar el entendimiento que tenemos sobre los diversos ataques a las redes de computadoras para poder predecirlos, contrarrestarlos o mitigarlos.

BIBLIOGRAFÍA

[1] J. R. Vacca, COMPUTER FORENSICS Computer Crime Scene Investigation, Charles Rever Media, Inc., 2005.

[2] I. C. C. Center, «Internet Crime Complaint Center,» 11 Mayo 2012. [En línea]. Available: http://www.ic3.gov/media/2012.aspx. [Último acceso: 5 Junio 2012].

[3] I. C. C. Center, «2010 Internet Crime Report,» 17 Septiembre 2010. [En línea]. Available: http://www.ic3.gov/media/annualreport/2010_ic3report.pdf. [Último acceso: 2 Diciembre 2011].

[4] mattica, «mattica,» Noviembre 2012. [En línea]. Available: http://www.mattica.com/2012/11/ifai-alerta-de-riesgos-por-delitos-informaticos/. [Último acceso: 16 Noviembre 2012].

[5] E. S. P. a. R. C. J. a. R. Niyogi, «Network forensic frameworks: survey and research challenges,» *Digital Investigation,* vol. 7, pp. 14 - 27, 2010.

[6] L. Z. a. H. Zhang, «An Introduction to data capturing,» 2008.

[7] A. I. C. a. J. L. a. A. Patel, «International Cooperation to Fight Transnational Cybercrime,» *Second International Workshop on Digital Forensics and Incident Analysis,* 2007.

[8] U. CERT, *Congreso de Seguridad en Cómputo 2011,* 2011.

[9] P. G., «A road map for digital forensic research,» *First digital forensic research workshop (DFRWS 2001),* pp. 27 - 30, 2001.

[10] W. V. Dan Farmer, Forensic Discovery, Addison Wesley Professional, 2004.

[11] E. S. P. a. R. C. J. a. R. Niyogi, "Network forensic frameworks: survey and research challenge," 2010.

[12] W. W. a. T. E. Daniels, «Building evidence graphs for network forensics analysis,» *21st Annual Computer Security Applications Conference (ACSAC 2005),* 2005.

[13] C. Y. N. S. Zhin Chen, «Learning Computer Network by Writing Your Own Protocol analyzer,» *2011 Third International Workshop on Education Technology and Computer Science,* 2011.

[14] «Sniffers: Basics and Detection,» [En línea]. Available: http://cns.tstc.edu/cpate/LINUX/Linux_How2/Sniffers.htm. [Último acceso: 27 Julio 2011].

[15] A. S. Tanenbaum, «Redes de computadoras,» Pearson, 2003.

[16] M. A. Q. a. A. I. a. M. Z. a. M. R. Siddiqui, "Network Traffic Analysis and Intrusion Detection using Packet Sniffer," *2010 Second International Conference on Communication Software and Networks,* 2010.

[17] H. Z. Liqiang Zhang, «An Introduction to Data Capturing,» *International Symposium on Electronic Commerce and Security,* 2008.

[18] S. M. a. V. Jacobson, «The BSD Packet Filter: A New Architecture for User-level PAcket Capture,» *The 1993 Winter USENIX Technical Conference,* 1993.

[19] S. R. a. L. Degioanni, «Development of an Architecture for Packet Capture and Network Traffic Analysis,» [En línea]. Available: http://netgroup-serv.polito.it/winpacap.

[20] S. M. a. V. Jacobson, «The BSD PAcket Filter: A new Architecture for User-level Packet Capture,» *Winter USENIX Conference Proceedings,* pp. 1-11, 1992.

[21] A. M. a. A. F. a. F. L. a. J. López, "Ksensor: Multithreaded Kernel-Level Probe for Passive QoS Monitoring," 2007.

[22] J. C. Mogul, «Eliminating Receive Livelock in an Interrupt-Driven Kernel,» *ACM Transactions on Computer Systems,* vol. 15, nº 3, p. 217, 1997.

[23] J. B. a. D. B. T. a. A. P. J. Slay, «Improving the Analysis of Lawfully Intercepted Network Packet Data Captured for Forensic Analysis,» *The Third International Conference on Availability, Reliability and Security,* 2008.

[24] R. G. a. T. C. a. X. Luo, «Application Layer Information Forensics Based on Packet Analysis,» *2010 International Conference of Information Science and Management Engineering,* 2010.

[25] R. Braden, *Requeriments for Internet community.*

[26] D. A. R. P. Agency, *Transmission control protocol,* 1981.

[27] P. J., *User Datagram Protocol,* 1980.

[28] P. J., *Internet Control Message Protocol,* 1981.

[29] I. C. X. C. Digital Equipment Corporation, *The ethernet A local area network, data link layer and physical layer specifications,* 1982.

[30] C. M. P. A. M. M. G. F. P. R. A. Á. R. V. Justo Pérez Agudín, La biblia del hacker Edicion 2006, España: ANAYA multimedia, 2006.

[31] G. D. L. a. G. Schauwers, Network Security Fundamentals, An introduction to the key tools and technologies used to secure network access, Indiana: Cisco Press, 2005.

[32] S. A. Thomas, IP Switching and Routing Essentials, New York: Wiley Computer Publishing, 2002.

[33] J. L. Hernández, Fundamentos de programación, España: Parraninfo España, 1998.

[34] H. B. Christensen, Estadistica paso a paso, Trillas.

[35] Digital, Intel and Xerox, «The ethernet. A Local Area Network Data link layer and physical layer specifications,» 1980.

[36] L. M. Gracia, «Programming with Libpcap Sniffing the Network from our own application,» *Haking, Practical protection hard core it security magazine,* vol. 3, nº 2, p. 40, 2008.

[37] T. f. D. b. farlex, «The free Dictionary by farlex,» 2011. [En línea]. Available: http://www.thefreedictionary.com/cybercrime. [Último acceso: 27 julio 2011].

[38] M. K. a. L. Mitrou, «Internet Forensics: Legal an Technical Issues,» *Proceedings of Second International Annual Workshop on Digital Forensics and Incident Analysis,* pp. 3 - 17, 2007.

[39] J. H. P. a. S. G. a. H. J. a. J. W. a. Guest, «Network Security and Digital Forensics in Next Generation Communications,» *Wireless Communications and Mobile Computing,* vol. 11, pp. 143 - 145, 2011.

[40] J. C. Mogul, «Eliminating Receive Livelock in an Interrupt-Driven Kernel,» *ACM Transactions on Computer Systems,* vol. 15, pp. 217 - 252, 1997.

[41] www.defoenet.com, «CCNA Study Notes - OSI - Data Link Layer,» 3 agosto 2004. [En línea]. Available: http://www.defoenet.com/ccna/osi_l2.html. [Último acceso: 26 septiembre 2012].

[42] A. t. f. poundhead, «llc sublayer details,» [En línea]. Available: http://www.fortypoundhead.com/showcontent.asp?artid=2355. [Último acceso: 26 septiembre 2012].

Anexos

ANEXO A. Glosario

ARP ARP Son las siglas en inglés de Address Resolution Protocol (Protocolo de resolución de direcciones).Es un protocolo de la capa de enlace de datos responsable de encontrar la dirección hardware (Ethernet MAC) que corresponde a una determinada dirección IP

Artefacto Pieza de informática tangible que es crada, modificad y usada por los trabajadores al realilzar actividades; representa una área de responsabilidad, y es candidata a ser tenida en cuenta para el control de la configuración. Un artefacto puede ser un modelo, un elemento de un modelo, o un documento.

BJS Bureau of Justice Statistics (BJS) es un componente de los programas de la oficina de justicia en los E.U. en el Departamento de justicia de los Estados Unidos.

BSD Berkeley Software Distribution (en español, distribución de software berkeley). Así se llamó a las distribuciones de código fuente que se hicieron en la Universidad de Berkeley en California y que en origen eran extensiones del sistema operativo UNIX de AT&T Research.

Buffer Un buffer es una ubicación de la memoria en un Disco o en un instrumento digital reservada para el almacenamiento temporal de información digital, mientras que espera ser procesada.

Checksum Forma de control de redundancia, una medida muy simple para proteger la integridad de los datos, verificando que no hayan sido corruptos.

Core dumps Se refiere a un archivo que contiene una imagen en memoria de un proceso determinado, también se refiere a un listado impreso de todo el contenido de la memoria.

Cracker El término cracker se utiliza para referirse a las personas que rompen algún sistema de seguridad.

Crackers El término cracker se utiliza para referirse a las personas que *rompen* algún sistema de seguridad.

CSMA/CA Protocolo de control de redes de bajo nivel que permite que múltiples estaciones utilicen un mismo medio de transmisión. Cada equipo anuncia opcionalmente su intención de transmitir antes de hacerlo para evitar colisiones entre los paquetes de datos (comúnmente en redes inalámbricas, ya que estas no cuentan con un modo práctico para transmitir y recibir simultáneamente).

CSMA/CD Técnica usada en redes Ethernet para mejorar sus prestaciones. Anteriormente a esta técnica se usaron las de Aloha puro y Aloha ranurado, pero ambas presentaban muy bajas prestaciones.

Datagrama Hay dos formas de encaminar los paquetes en una red conmutación de paquetes. Estas son: datagrama y circuito virtual. En la técnica de datagrama cada paquete se trata de forma independiente, conteniendo cada uno la dirección de destino. La red puede encaminar (mediante un router) cada fragmento hacia el Equipo Terminal de Datos (ETD) receptor por rutas distintas. Esto no garantiza que los paquetes lleguen en el orden adecuado ni que todos lleguen al destino.

Driver Programa informático que permite al sistema operativo interactuar con un periférico, haciendo una abstracción del hardware y proporcionando una interfaz -posiblemente estandarizada- para usarlo.

E-mail Correro eléctronico, es un servicio de red que permite a los usuarios enviar y recibir mensajes y archivos rápidamente.

Ethernet Es un estandar de redes de área local para computadores con acceso al medio por contienda CSMA/CD. CSMA/CD (Acceso Múltiple por detección de portadora con detección de colisiones), es una técnica usada en redes Ethernet para mejorar sus prestaciones.

Exploit Expresa el tamaño en bytes de la unidad de datos más grande que puede enviarse usando un protocolo de comunicaciones.

Firewall Dispositivo que funciona como cortafuegos entre redes, permitiendo o denegando las transmisiones de una red a la otra.

Framework Es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, con base a la cual otro software pude ser más facilmente organizado y desarrollado.

Gateway Una pasarela o puerta de enlace (del inglés gateway) es un dispositivo, con frecuencia una computadora, que permite interconectar redes con

	protocolos y arquitecturas diferentes a todos los niveles de comunicación. Su propósito es traducir la información del protocolo utilizado en una red, al protocolo usado en la red de destino.
Hardware	Partes tangibles de un sistema informático, sus componentes son: eléctricos, electrónicos, electromecánicos y mécanicos.
Honeypot	software o conjunto de computadores cuya intención es atraer a atacantes, 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.
Hub	Dispositivo que tiene la función de interconectar las computadoras de una red local.
ICMP	El Protocolo de Mensajes de Control de Internet o ICMP (por sus siglas en inglés de Internet Control Message Protocol) es el sub protocolo de control y notificación de errores del Protocolo de Internet (IP). Como tal, se usa para enviar mensajes de error, indicando por ejemplo que un servicio determinado no está disponible o que un router o host no puede ser localizado.
IDS	Un sistema de detección de intrusos IDS es un programa usado para detectar accesos no autorizados a un computador o a una red.
IEEE 802.11	Define el uso de los dos niveles inferiores de la arquitectura OSI (capas física y de enlace de datos), especificando sus normas de funcionamiento en una WLAN.
IEEE802.11b	Es una modificación de la Norma IEEE 802.11que amplia la tasa de transferencia hasta los 11 Mbit/s usando la misma banda de 2.4 GHz.
IGMP	Protocolo para intercabiar información acerca del estado de permanencia entre enrutadores IP que admiten la multidifusión y miembros de grupos de multidifusión.
Interfaz	Es la conexión entre dos computadoras o máquinas de cualquier tipo dando una comunicación entre distintos niveles.
IP	Es una etiqueta numérica que identifica, de manera lógica y jerárquica, a un interfaz (elemento de comunicación/conexión) de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP (Internet Protocol), que corresponde al nivel de red del

protocolo TCP/IP.

IRC Es un protocolo de comunicación en tiempo real basado en texto, que permite debates entre dos o más personas. Se diferencia de la mensajería instantánea en que los usuarios no deben acceder a establecer la comunicación de antemano, de tal forma que todos los usuarios que se encuentran en un canal pueden comunicarse entre sí, aunque no hayan tenido ningún contacto anterior.

Kernel Es el principal responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema.

Libpcap libpcap proporciona fuciones para la captura de paquetes a nivel de usuario, utilizada en la monitorización de redes de bajo nivel.

Linux Es un sistema operativo derivado de Unix.

Loopback Es un pseudo-dispositivo que hace que un archivo sea accesible como un dispositivo block. Es una interfaz para un driver.

MAC es un identificador de 48 bits (3 bloques hexadecimales) que corresponde de forma única a una tarjeta o dispositivo de red

Malware También llamado badware, código maligno, software malicioso o software malintencionado, es un tipo de software que tiene como objetivo infiltrarse o dañar una computadora sin el consentimiento de su propietario.

MTU En redes de computadoras expresa el tamaño en bytes de la unidad de datos más grande que puede enviarse usando un protocolo de comunicaciones.

Multicasting Forma de transferencia de datos en donde es posible enviar información de un solo emisor a muchos puntos diferentes (receptores) simultáneamente.

Offset Es un entero que indica la distancia de desplazacimiento desde el inicio del objeto hasta un punto o elemento dado, presumiblemente dentro del mismo objeto.

OSI El modelo de interconexión de sistemas abiertos, también llamado OSI (en inglés open system interconnection) es el modelo de red descriptivo creado por la Organización Internacional para la Estandarización en el año 1984

overflow Es un error de software que se produce cuando un programa no controla adecuadamente la cantidad de datos que se copian sobre un área de memoria reservada a tal efecto (buffer), de forma que si dicha cantidad es superior a la capacidad pre asignada los bytes sobrantes se almacenan en zonas de memoria adyacentes, sobrescribiendo su contenido original.

P2P Una red peer-to-peer, red de pares, red entre iguales, red entre pares o red punto a punto (P2P, por sus siglas en inglés) es una red de computadoras en la que todos o algunos aspectos funcionan sin clientes ni servidores fijos, sino una serie de nodos que se comportan como iguales entre sí.

Plug-in Es un conjunto de componentes de software que agrega habilidades específicas para una aplicación de software más grande.

PPP Point to Point Protocol, es un protocolo de nivel de enlace estandarizado en el documento RFC 1661. Se trata de un protocolo asociado a la pila TCP/IP de uso de Internet.

Protocolo Conjunto de reglas usadas por computadoras para comunicarse unas con otras a través de una red.

Puertos En la informática, un puerto ata ó puerto es una forma genérica de denominar a una interfaz a través de la cual los diferentes tipos de datos se pueden enviar y recibir.

RFC Las Request for Comments ("Petición De Comentarios" en español) son una serie de notas sobre Internet, y sobre sistemas que se conectan a Internet, que comenzaron a publicarse en 1969.1 Se abrevian como RFC.

Router Encaminador, enrutador o direccionador es un dispositivo de hardware usado para la interconexión de redes informáticas que permite asegurar el direccionamiento de paquetes de datos entre ellas o determinar la mejor ruta que deben tomar.

Sniffer Programa utilizado par leer paquetes que viajan a través de la capa de red.

Software Equipamiento lógico o soporte lógico de un sistema informático; comprende el conjunto de los componentes lógico**s** necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos, que son llamados hardware.

Switch El switch es un aparato muy semejante al hub, pero tiene una gran diferencia: los datos provenientes de la computadora de origen solamente son enviados a la computadora de destino. Esto se debe a que los switches crean una especie de canal de comunicación exclusiva entre el origen y el destino.

TCP Transmission Control Protocol, es uno de los protocolos fundamentales en Internet. El protocolo garantiza que los datos serán entregados en su destino sin errores y en el mismo orden en que se transmitieron.

Timer Reloj especializado para medir intervalos de tiempo

Token Un token o también llamado componente léxico es una cadena de caracteres que tiene un significado coherente en cierto lenguaje de programación.

Trama Unidad de envío de datos

UDP User Dtagram Protocol, es un protocolo del nivel de transporte basado en el intercabio de datagramas. Permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera. Tampoco tiene confirmación ni control de flujo, por lo que los paquetes pueden adelantarse unos a otros.

Unix Sistema operativo portable, multitarea y multiusuario

VoIP Es un grupo de recursos que hacen posible que la señal de voz viaje a través de Internet empleando un protocolo IP

Web Sistema de distribución de información basado en hipertexto o hipermedios enlazados y accesibles a través de Internet.

Wireless Es la transferencia de información entre dos o más puntos que no están fisicamente conectados.

Zero-day

Es un ataque contra una computadora, basado en encontrar vulnerabilidades aún desconocidas en las aplicaciones informáticas. Este tipo de exploit circula generalmente entre las filas de los potenciales atacantes hasta que finalmente es publicado en foros públicos. Un exploit de día-cero normalmente es desconocido para la gente y el fabricante del producto. Un *ataque de día cero* se considera uno de los más peligrosos

ANEXO B. Legislación en México

Delitos en materia de vías de comunicación y de correspondencia

Artículo 177.-

A quien intervenga comunicaciones privadas sin mandato de autoridad judicial competente, se le aplicarán sanciones de seis a doce años de prisión y de trescientos a seiscientos días multa.

Revelación de secretos y acceso ilícito a sistemas y equipos de informática

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 de 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 de multa.

Artículo 211 bis 2

Al que sin autorización modifique, destruya o provoque perdida 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 de 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.

A quien sin autorización conozca, obtenga, copie o utilice información contenida en cualquier sistema, equipo o medio de almacenamiento informáticos de seguridad pública, protegido por algún medio de seguridad, se le impondrá pena de cuatro a diez años de

prisión y multa de quinientos a mil días de salario mínimo general vigente en el distrito federal. Si el responsable es o hubiera sido servidor público en una institución de seguridad pública, se impondrá además, destitución e inhabilitación de cuatro a diez años para desempeñarse en otro empleo, puesto, cargo o comisión pública.

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 de 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 de multa.

A quien estando autorizado para acceder a sistemas, equipos o medios de almacenamiento informáticos en materia de seguridad pública, indebidamente obtenga, copie o utilice información que contengan, se le impondrá pena de cuatro a diez años de prisión y multa de quinientos a mil días de salario mínimo general vigente en el distrito federal. Si el responsable es o hubiera sido servidor público en una institución de seguridad pública, se impondrá además, hasta una mitad más de la pena impuesta, destitución e inhabilitación por un plazo igual al de la pena resultante para desempeñarse en otro empleo, puesto, cargo o comisión pública.

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 incrementaran 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 aumentaran hasta en una mitad cuando la información obtenida se utilice en provecho propio o ajeno.

ANEXO C. Símbolos de soporte de información o dispositivos físicos:

La simbología que se utiliza para soporte de información o dispositivos físicos:

Símbolo	**Denominación**	**Tipo de dispositivo**
	Teclado	Entrada
	Soporte Genérico	Entrada
	Pantalla /CRT	Salida
	Impresora	Salida
	Tarjeta perforada	Entrada/Salida
	Cinta de papel	Entrada/Salida
	Disco magnético	Entrada/Salida
	Disco magnético	Entrada/Salida
	Cinta magnética	Entrada /Salida
	Cinta Encapsulada	Entrada/Salida
	Disco Flexible	Entrada/Salida
	Tambor magnético	Entrada/Salida

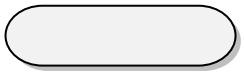
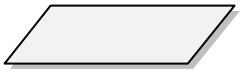
Simbología utilizada en los procesos.

Símbolo	**Función**
	Proceso u operación.
	Clasificación u ordenación de datos en un fichero.
	Fusión o mezcla de dos o más ficheros en uno solo.
	Partición o extracción de datos de un fichero.
	Manipulación de uno o varios ficheros (intercalación).

Significado de las líneas de flujo de datos.

Símbolo	**Función**
	Dirección del proceso o flujo de datos
	Línea conectora. Permite la unión entre unidades o elementos de información.

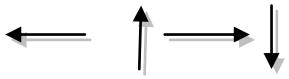
La simbología para los diagramas de flujo es:

Símbolo	**Función**
	Terminal, marca el inicio, final o una parada necesaria realizada en la ejecución del programa.
	Operación de E/S en general. Se utiliza para introducir datos desde un periférico a la memoria de la computadora y la salida de resultados desde la memoria de la computadora hacia un periférico.
	Proceso u operación en general que se utiliza para mostrar cualquier tipo de operación durante el proceso de elaboración de los datos depositados en la memoria.
	Subprograma o subrutina que se utiliza para realizar una llamada a un subprograma o proceso.

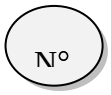
Símbolos de decisión

Símbolo	**Función**
	Decisión de dos salidas. Indica operaciones lógicas o comparativas para seleccionar. La selección se hace en función del resultado entre dos caminos alternativos que se pueden seguir.
	Decisión múltiple con ‘n’ salidas. Indica el camino que se puede seguir entre varias posibilidades según el resultado de la operación lógica o comparación establecida.
	Bucle definido, empleado para modificar una instrucción o bloque de instrucciones que a su vez produce una alteración o modificación en el comportamiento del programa.

Líneas de flujo.

Símbolo	Función
	Flechas indicadoras de la dirección del flujo de datos.
	Línea conectora, también llamada línea de flujo de datos. Se utiliza para conectar los diferentes símbolos del diseño.

Símbolos de conexión.

Símbolo	Función
	Este símbolo es utilizado para el reagrupamiento de las líneas de flujo.
	Conector de las líneas de flujo en la misma página. Es un conector que enlaza dos partes cualesquiera del diseño a través de un conector de salida y un conector de entrada.
	Conector de líneas de flujo en distintas páginas utilizado para enlazar dos partes cualesquiera del diseño a través de un conector de entrada.

ANEXO D. Operación del sistema.

El sistema ADA está diseñado para ser utilizado por un administrador de redes u otra persona que cuente con un nivel intermedio en conocimientos de redes.

Requerimientos:

- Plataforma Linux Fedora 14 o posterior.
- Verificar que el sistema operativo tiene instalado Libpcap y Python.

Para inicializar ADA** es necesario abrir una terminal y ejecutar el programa principal. Esto se hace al teclear:

python principal.py

En seguida aparece el menú principal que tiene las opciones para ingresar a las diferentes áreas del sistema (ver Figura 1).

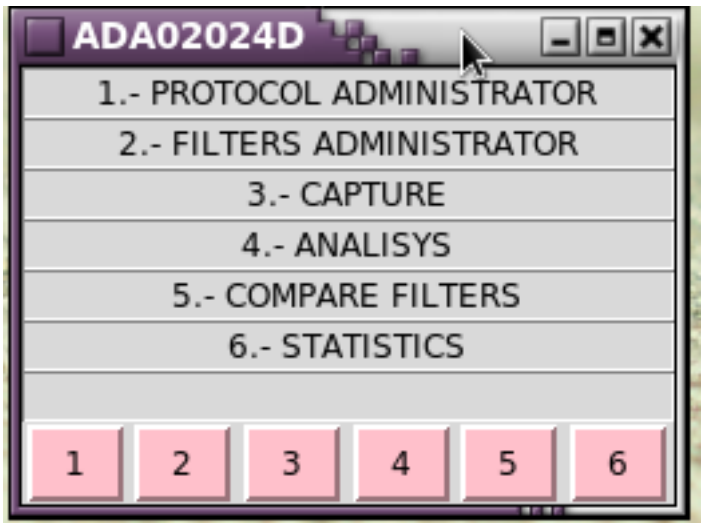


Figura 1 Pantalla principal