INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA

“MODELO PARA LA DETECCIÓN DE ATAQUES POR FLOODING UTILIZANDO GRAFOS”

TESIS

PARA OBTENER EL GRADO DE MAESTRO EN SISTEMAS COMPUTACIONALES

PRESENTA

ISC. YELITZA DIANAHI VIVEROS VARGAS

ASESORES:

DR. LUIS ALBERTO MORALES ROSALES

DR. IGNACIO ALGREDO BADILLO

MISANTLA, VERACRUZ MARZO, 2017

INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA
DIVISIÓN DE ESTUDIOS PROFESIONALES
AUTORIZACIÓN DE IMPRESIÓN DE TRABAJO DE TITULACIÓN MAESTRÍA

FECHA: 27 de Febrero de 2017.

ASUNTO: **AUTORIZACIÓN DE IMPRESIÓN DE TESIS.**

A QUIEN CORRESPONDA:

Por medio de la presente se hace constar que el (la) C:

YELITZA DIANAHI VIVEROS VARGAS

estudiante de la maestría en SISTEMAS COMPUTACIONALES con No. de Control 142T0772 ha cumplido satisfactoriamente con lo estipulado por el **Lineamiento de Posgrado para la obtención del grado de Maestría** mediante **Tesis.**

Por tal motivo se **Autoriza** la impresión del **Tema** titulado:

MODELO PARA LA DETECCIÓN DE ATAQUES POR FLOODING UTILIZANDO GRAFOS

Dándose un plazo no mayor de un mes de la expedición de la presente a la solicitud del examen para la obtención del grado de maestría.

ATENTAMENTE

Dr. Luis Alberto Morales Rosales
Presidente

M.I.A. Roberto Ángel Meléndez Armenta
Secretario

M.S.C. Ignacio Algredo Badillo
Vocal

Archivo.

Agradecimientos.

A Dios ante todo por haberme permitido hallar gracia delante de él, otorgándome fortaleza en los tiempos de debilidad y desanimo; sabiduría para comprender y poner en práctica el conocimiento adquirido y salud para culminar un escalón más en mi vida tanto profesional como espiritual.

A mis padres por el apoyo que me dieron, su comprensión y paciencia en todo momento.

A mi hijo por ser mi motivo para salir adelante aun a pesar de todos los obstáculos que encontré en el camino al éxito.

Al Consejo Nacional de Ciencia y Tecnología (CONACYT) de México, por el apoyo económico que me brindo durante todo el periodo de estudios.

A mi asesor el Dr. Luis Alberto Morales Rosales compañero y amigo, por haberme tenido paciencia, por sus consejos, por sus regaños y por enseñarme el valor de la perseverancia.

A mis revisores de tesis, Dr. Ignacio Algredo Badillo y al MIA Roberto Ángel Meléndez Armenta por su tiempo y dedicación a lo largo de este proyecto.

Y a mis amigos y compañeros que estuvieron conmigo en los desvelos, en las tristezas y en los triunfos. ¡Hoy puedo decirles que lo logramos!

Índice

Índice de figuras.

Índice de tablas.

Capítulo I. Generalidades

1.1 Introducción.

Debido al crecimiento en los volúmenes de información de las empresas y a la evolución de la tecnología, muchas de las organizaciones, tanto públicas como privadas, optaron por utilizar sistemas informáticos, los cuales les permitió manejar, almacenar y procesar la información de la empresa con la ayuda de hardware, software y personal encargado de la administración de las diferentes áreas de la institución.

Con la aparición de los sistemas informáticos y con el surgimiento de la Internet se hicieron presentes los ataques a sistemas informáticos, por un lado, existían los ataques informáticos realizados desde el interior de la red de las empresas las cuales consistían en la alteración de los permisos de acceso con la finalidad de modificar información del sistema; por otra parte se encuentran los ataques realizados de forma externa a la red de la empresa, se producían gracias a la filtración de contraseñas de las computadoras pertenecientes a la red. [1]

Con el paso del tiempo, el nivel de alcance de estos ataques se fueron perfeccionando y desarrollando nuevas formas de vulnerar las redes de las empresas, tanto en su diseño, como en su configuración y estructura de los sistemas informáticos que forman parte de las redes que se conectan en el Internet. Ataques como el Phishing, Denegación de Servicios Distribuidos (DDoS, por sus siglas en inglés) y SQL injection son más comunes al grado que ya encontramos aplicaciones en la Internet que nos indican paso a paso como realizar un ataque informático [2], sin la necesidad de tener un amplio conocimiento sobre las redes o programación. Estos ataques a empresas fueron nombradas por conocedores del área de seguridad informática como “ciberataques” [3], siendo los actos en los cuales se cometen agravios contra la infraestructura informática de instituciones en forma de ataques informáticos.

Ataques como el Phishing y la DDoS han ido en aumento según una encuesta realizada por la OEA y Trend Micro en enero del 2015 [4], donde los Jefes de Seguridad de las principales infraestructuras críticas de América aseguran que han incrementado en un 43% respecto a años anteriores como se muestran en la Gráfica 1. Según la encuesta realizada por Tren Micro en el Reporte de Seguridad Cibernética e Infraestructura Crítica de las Américas [4]

indica que en un 42% de los ataques presentados en sus organizaciones han sido a causa de los DDoS [5] cuyo objetivo es ralentizar un servicio, y pueden llegar hasta el punto de inhabilitar servidores y páginas web completas.

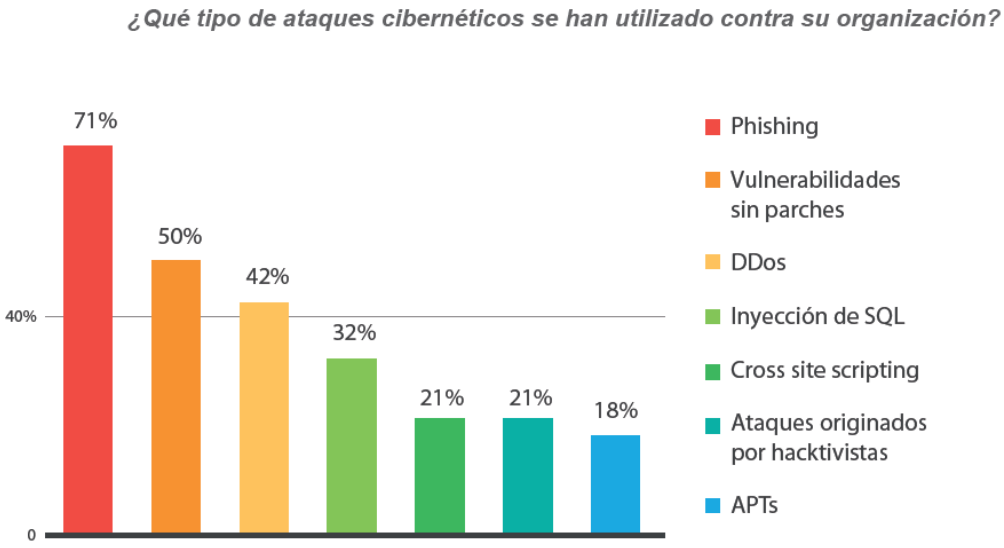


Figura 1. Tipos de Métodos para ataques cibernéticos. Encuesta por TrendMicro (2015).

Dentro de la encuesta realizada por Trend Micro [4], uno de los ataques más difíciles de detectar son los DDoS, ya que éste ataque puede ser provocado tanto por usuarios internos en el sistema de la empresa como por usuarios externos a la institución [1]. Dentro de los usuarios internos de la red, contemplamos a los usuarios con pocos conocimientos informáticos que pueden colapsar el sistema o servicio inconscientemente. Por ejemplo, los usuarios que abusan de los recursos del sistema, ocupando una mayor cantidad de ancho de banda en la búsqueda de archivos de música o de películas, o bien, usuarios malintencionados que aprovechan su acceso al sistema para causar problemas de forma premeditada. Por otro lado los usuarios externos a la institución son aquellos usuarios que han conseguido un acceso al sistema de forma ilegıtima, falseando además la dirección de origen IP con el propósito de evitar la detección del origen real del ataque (mediante ataques de suplantación).

El ataque DDos afecta directamente la disponibilidad de los servicios de empresas e instituciones ya que básicamente consiste en enviar un gran número de peticiones a un servidor de manera distribuida desde múltiples conexiones a Internet, y en consecuencia los

usuarios legítimos del servicio no puedan acceder a esos recursos lo que ocasiona pérdidas económicas ya que por causas del ataque las empresas no pueden ofrecer sus servicios a sus empleados y clientes, propiciando que sus clientes no puedan llevar a cabo sus compras online o la formulación de negocio por vía electrónica.

La forma más clásica de DDos para colapsar el servicio de red de una empresa o de una página web son los ataques por inundación o flooding [6], el cual consiste en él envío masivo de paquetes con la finalidad de saturar a los ciclos de la CPU o la memoria y, a veces al ancho de banda de la red y a la memoria cache de paquetes buffer, lo que imposibilita a la víctima a responder a las peticiones realizadas por los clientes.

Es por ello que este trabajo se enfocará al estudio de los ataques de DDos [5] con la finalidad de detectar posibles intentos de vulnerar al sistema de red con ataques por flooding, dando la posibilidad al administrador de los servicios de red, el poder tomar las precauciones necesarias para evitar la pérdida de conexión de sus servicios y pautar sus comunicaciones durante algún tiempo causando pérdidas económicas.

1.2 Descripción del problema.

Los ataques por inundación (flooding) [6], son ataques diseñados para saturar una red o servicio con grandes cantidades de tráfico utilizando comúnmente una *botnet*, siendo su principal objetivo que la red o servicio se encuentre saturada con peticiones de conexión generadas por el atacante que no pueda procesar las peticiones de conexión reales, teniendo como resultado final un DDoS. La idea de utilizar computadoras que se encuentran distribuidas de manera global y que tienen múltiples conexiones a Internet, es él evitar que el atacante pueda ser localizado fácilmente y debido a la dispersión geográfica de las computadoras que componen la botnet, es mucho más difícil encontrar un patrón con el que se pueda realizar un filtrado de los paquetes de la red y así poder detectar el ataque de manera precisa.

Por lo general, los ataques DDoS suelen comenzar por pequeños problemas de latencia en la conexión a la red que van empeorando gradualmente hasta que llega a un punto donde no se puede acceder a ningún recurso exterior ni servicio. Esta pérdida de conexión causa en muchas de las ocasiones la obstrucción del medio de comunicación entre usuarios de un

servicio y la víctima, de manera que ya no puedan comunicarse adecuadamente. Otra de las consecuencias generadas por la presencia de un ataque DDoS es la interrupción de componentes físicos de la red, la interrupción de sesiones TCP o la alteración de información de configuración, tal como la información de rutas de encaminamiento [1].

El ataque por flooding ataca directamente la disponibilidad de los recursos de red, causándole a la reputación de las empresas con presencia en Internet un gran daño e incluso impidiendo a las compañías el desarrollo normal de sus actividades en caso de que éstas estén basadas en un sistema informático; ejemplo de estas afectaciones tenemos a la compañía Sony que en 2011 recibió un ataque masivo de DDos a todos sus servicios online, teniendo como principal afectado el Playstation Network, el cual trajo a la empresa perdidas económicas bastante altas[7].

Por otro lado, para poder realizar la detección de un ataque DDos, uno de los problemas que se presentan es el poder diferenciar el tráfico de red normal de la empresa con un tráfico potencial para un ataque; esto se debe a que en ocasiones las empresas tienen usuarios que abusan de los recursos del sistema, es decir, ocupando gran cantidad de ancho de banda en la búsqueda de archivos de música o de películas que tiende a ser mal interpretado como un posible ataque de flooding.

Otra de las causas que afecta la mala detección de una ataque son los problemas de suplantación IP [1], es decir que el atacante reemplace la dirección IP de un paquete IP del remitente por la dirección IP de otro equipo, con lo cual le permitirá al atacante enviar paquetes de manera anónima haciéndose pasar por personal de la empresa, lo que ocasiona que los paquetes puedan acceder a la red con mayor facilidad y llegar a ocasionar una saturación de la red.

Otro de los conflictos que se tiene al momento de realizar la identificación de un ataque DDoS, es el uso por parte del atacante de los paquetes encriptados [1], de los cuales la mayoría de los firewall utilizados en las empresas no son capaces de monitorear paquetes con protocolos SSL (Secure Sockets Layer, por sus siglas en inglés) y TLS (Transport Layer Security, por sus siglas en inglés), ya que estas encriptaciones no son desempaquetadas por los firewall para su análisis debido a los problemas de rendimiento que este causaría en la

utilización de los recursos, aparte de las implicaciones legales que conlleva en monitorear un SSL.

Gran parte de las investigaciones que atacan el problema de la detección de DDoS , toman como primer paso el uso de programas para el monitoreo del tráfico de red como SNORT [8], con el cual se apoyan para observar el comportamiento de los paquetes y así poder identificar las fluctuaciones de peticiones de la red para poder diagnosticar en qué momento se está produciendo un ataque por flooding; de lo cual surge una interrogante, ¿cuáles son las variables adecuadas para la identificación del ataque por flooding?, es decir, cómo elegir un subconjunto de variables de un total de características que nos arroja el programa de monitoreo de tráfico de red con las que se pueda llevar a cabo la identificación de un DDos, dando la ventaja de no tener que procesar toda la trama de tráfico completa y así poder disminuir el costo computacional que implicaría procesar toda la información en conjunto.

Después de obtener las variables representativas del conjunto de datos que ayudará a la identificación de un ataque DDoS, surge dos nuevas preguntas, ¿cuál es el grado de correlación que deben tener las variables seleccionadas para poder identificar un patrón que nos permita la identificación de un ataque DDos? y ¿es posible que utilizando sólo un subconjunto de variables correlacionadas se pueda llevar a cabo la identificación del ataque por flooding?; interrogantes que plantean un reto en el área de detección de DDos.

1.3 Hipótesis.

Es posible detectar el ataque DDos por flooding generado en la capa de transporte durante un tráfico denso a partir de la utilización de un algoritmo de agrupamiento y grafos semánticos con la finalidad de disminuir los daños provocados por la falta de disponibilidad de los servicios.

1.4 Objetivos.

1.4.1 Objetivo General.

Desarrollar un modelo de detección del ataque DDos por flooding en la capa de trasporte con la finalidad de contrarrestar los daños provocados en la disponibilidad de los servicios.

1.4.2 Objetivos específicos.

- Capturar los paquetes de tráfico de la red con protocolo TCP, UDP, HTTP e ICMP, utilizando un sniffer de red con la finalidad de obtener las cabeceras de los paquetes y sus trazas para su posterior análisis.
- Seleccionar las variables que aportan los paquetes capturados del tráfico de la red las cuales me proporcionan la información necesaria para identificar si el tráfico que se está generando es un posible ataque por flooding.
- Analizar las características extraídas de los paquetes capturados utilizando el algoritmo k-vecinos para determinar las correlaciones existentes entre ellas.
- Utilizar grafos semánticos en conjunto con el algoritmo MDL con la finalidad de obtener patrones que permitan la identificación de ataque por flooding.

1.5 Justificación.

El ataque DDoS ataca directamente la disponibilidad de los servicios de las empresas, teniendo como consecuencias perdidas económicas que según una encuesta realizada por Kaspersky Lab y B2B International en enero del 2015 [9]. Un ataque DDoS a los recursos en línea de una compañía podría causar pérdidas económicas de entre $52,000 y $444,000 dólares, en función del tamaño de la empresa. [9] Para muchas organizaciones, estos gastos tienen un grave impacto en el balance general, además de perjudicar la reputación de la empresa como consecuencia de la pérdida de acceso a los recursos en línea para los socios y clientes.

Según el estudio realizado por Kaspersky Lab [9], el 61% de las víctimas de ataques DDoS perdieron temporalmente el acceso a la información crítica lo que provoco en un 38% la pauta en sus actividades principales, por lo que en un 33% las empresas perdieron oportunidades de negocios y contratos. Además, el 29% de los incidentes de DDoS, un ataque exitoso tuvo un impacto negativo en la calificación crediticia de la empresa; mientras que el 26% de los casos provocó un aumento en las primas de seguros.

Cuando el ataque se ha realizado a páginas web orientada al público o bien a una aplicación de la empresa las consecuencias de no tener el servicio disponible puede provocar clientes

enojados, ingresos perdidos y daños a la marca, así como también pautar las acciones de las cadenas de suministro de las empresas dedicadas al comercio [10]. Cuando las aplicaciones críticas de negocio no están disponibles, las operaciones, la producción y la productividad de la empresa se detienen lo que representa una pérdida económica para la compañía.

Por lo que se prevé que el modelo para la detección de ataques por flooding beneficie a las compañías en el área de seguridad de red, dándole las estructuras adecuadas para la detección y así evitar que sus servicios sean colapsados y con ello las pérdidas económicas que un DDos representen disminuyan gracias a la detección oportuna del ataque por flooding.

1.6 Propuesta de solución.

Para el desarrollo del modelo para la detección de ataques por flooding se utilizarán las bases de datos de KDD CUP 99, la cual ya ha sido previamente probada por The Third International Knowledge Discovery and Data Mining Tools Competition.

A partir de los datos obtenidos del tráfico de red, se realizará un preprocesamiento de los datos con la finalidad de obtener sólo las variables más representativas que nos proporcionen información acerca del ataque flooding. Para la selección de variables representativas que se utilizarán en esta investigación, se tomó en cuenta la estructura de un ataque por DDos considerada en [16], así como el funcionamiento específico del ataque por Flooding [17], de los cual se ha propuesto lo siguiente:

Debido a que el ataque por flooding consiste en saturar la red con peticiones de conexión generadas por el atacante para que el servicio deje de trabajar de manera normal y ya no procese las peticiones de conexión reales, las variables consideradas para la identificación del ataque por flooding son:

- Count, el cual especifica el número de conexiones que se realizan a la misma máquina, con la finalidad de determinar si existe algún ciclo de peticiones que esté saturando la red.
- Src_bytes, esta ayuda a definir el número bytes por petición, lo cual apoyará a la identificación de alguna saturación en un servicio.
- Protocol_type, este atributo será utilizado para reconocer que tipo de protocolo se está utilizando en cada petición y apoyar en la identificación del ataque por Flooding.

- Wrong_fragment, este representará en número de fragmentos erróneos que se manejan en cada petición, ayudando a la identificación en la saturación de peticiones con fragmentos de la trama de tráfico.
- En el caso de Dst_host_same_srv_rate, Diff_srv_rate, Dst_host_serror_rate y Dst_host_diff_srv_rate serán utilización para identificar los porcentajes de conexiones con algún tipo de error y que se dirijan a un servicio en específico, como puede ser FTP o HTTP.

Ya que se tienen los datos de las variables a utilizar, mediante el uso del algoritmo de Esperanza Múltiple se identificarán los k grupos que ayudarán como punto partida para la implementación del algoritmo de agrupamiento de k vecinos, con el que se determinarán los patrones existentes sobre las variables seleccionadas. A partir de cada clúster se analizarán las trazas de tráfico para formular grafos semánticos con los cuales se pretende descubrir los casos de subestructura que contienen entidades y relaciones anómalas, con la finalidad de realizar la detección del ataque por flooding basada en tales anomalías, esta sección será utilizada en línea en el router principal de la red.

Capítulo II. Marco teórico

Con el comienzo de la interconexión de computadoras para formar redes surgieron las denominadas amenazas o ataques informáticos, donde una de las primeras personas en proponer un mecanismo que automatice la revisión de los eventos de seguridad fue James P. Anderson, quien en 1980 redacto uno de sus trabajo llamado "Monitor de Referencia" el cual sería uno de los primeros trabajos acerca de la detección de intrusiones, exponiendo los conceptos de acción maliciosa en los servidores y los intentos de penetración a la red. [12]

Con el paso de los años el tema de la seguridad en sistemas informáticos fue acrecentándose dentro de las empresas, con propuestas detección de intrusiones; pero a la par de estas propuestas los ataques informáticos fueron creando más fama y se dio paso a la clasificación de los tipos de ataques, como por ejemplo, la clasificación RFC2828 donde se agrupan los ataques como pasivos y activos.

Los ataques pasivos tienen como objetivo obtener la mayor cantidad de información de los mensajes transmitidos y del oponente, siendo un medio de reconocimiento previo a la realización de los ataques activos. Estos tipos de ataques son sutiles, ya que permites al atacante poder acceder al contenido del mensaje y observar la transmisión de los mensajes obteniendo datos como: la frecuencia de emisión de los mensajes y la longitud del mensaje. Esta tipo información es la que usa el atacante posteriormente para inferir la naturaleza de la comunicación de la institución y poder realizar el ataque de manera más precisa. [13]

Por otro lado, los ataques activos involucran y comprender los pilares básicos de las prácticas de seguridad: la confidencialidad, la integridad y la disponibilidad (CIA, por sus siglas en inglés), ya que estos ataques se infiltran en la red con el objetivo de violar las normas de seguridad y dañar a la empresa atacada. Dentro de los ataques activos tenemos [14]:

- Denegación de Servicio DoS (Denegation of Service): El efecto de este ataque es impedir la posibilidad de acceso a toda persona a un determinado servidor.
- Denegación de Servicios Distribuida (DDos): Este ataque al igual que el Dos tiene como objetivo saturar la red con un envío masivo de paquetes, pero a diferencia del

Dos, este envío de paquetes es realizado de manera distribuida desde diferentes computadoras colocadas en distintos lugares geográficos.

- "Masquerade" (Enmascarado): En este caso el atacante se representa él mismo como un legítimo usuario con el objeto de robar, alterar o destruir recursos informáticos.
- "Replay" (Reinterpretar): Este ataque es llevado a cabo mediante una captura pasiva de datos, para que luego sean retransmitidos y con ello producir efectos no autorizados.
- Modificación de contenidos del mensaje: La información original es alterada de tal forma que permita obtener un resultado no autorizado.

Esta investigación tendrá como principal enfoque la detección del ataque DDos, ya que es considerado uno de los ataques más utilizados para inhabilitar los servicios de la empresa y violar uno de los pilares de la CIA, que es la disponibilidad de los servicios.

2.1 Ataque DDos.

El ataque DDos es un intento de hacer que la disponibilidad de los servicios en línea sea abrumada por tráfico de múltiples fuentes con la finalidad de inhabilitar un servidor, un servicio o una infraestructura sobrecargando el ancho de banda del servidor o acaparando sus recursos hasta agotarlos. Durante un ataque DDoS, se envían multitud de peticiones simultáneamente desde múltiples puntos de la Red. La intensidad de este "fuego cruzado" desestabiliza el servicio o, aún peor, lo inhabilita.

Existen tres estrategias que pueden inhabilitar un sitio web, servidor o infraestructura:

- Ancho de banda: Ataque que consiste en saturar la capacidad de la red del servidor, haciendo que sea imposible llegar a él.
- Recursos: Ataque que consiste en agotar los recursos del sistema de la máquina, impidiendo que esta pueda responder a las peticiones legítimas.
- Explotación de fallos de software: Categoría de ataque que explota fallos en el software que inhabilitan el equipo o toman su control.

Según la Computer Emergency Readiness Team (CERT, por sus siglas en inglés) de Estados Unidos [15], los síntomas de los ataques de DDos incluyen las siguientes manifestaciones:

- Lento rendimiento de la red (apertura de los archivos o el acceso a sitios web)
- Falta de disponibilidad de un sitio web en particular.
- Imposibilidad a acceso a cualquier sitio web.

Lin and Tseng en [16], propone una arquitectura general del ataque DDos la cual se divide en dos etapas:

- Control.
- Ataque.

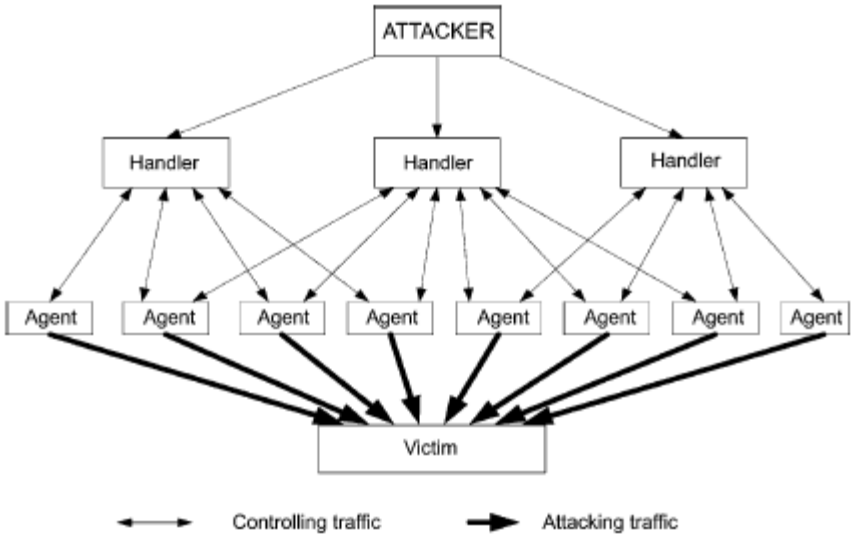


Figura 2. Topología propuesta por Lin y Tseng para realizar ataques DDos.

En la fase de control como se muestra en la Figura 2, se realiza una exploración a gran escala en la red para encontrar una lista de hosts vulnerables, es decir, realiza un ataque pasivo a la red. Generalmente el ataque DDos consiste en tener a los manipuladores y a los agentes, donde los manipuladores son controlados por el atacante y los agentes son controlados por los atacantes a través de los manipuladores. El tráfico de la comunicación en la etapa de control tiene lugar a través de la transmisión de la señal de un atacante a un controlador; sin embargo, la comunicación entre los controladores y agentes es bidireccional.

Dentro de la fase de ataque, los host vulnerables obtenidos en la etapa de control, se utilizan para lanzar un ataque distribuido en el tráfico de red, para sobrecargar los recursos computacionales del sistema de la víctima.

Dentro de los ataques DDos más utilizados se tiene el ataque por inundación o Flooding, el cual se aborda a continuación de manera más detallada.

2.2 Flooding.

El ataque por flooding desactiva o saturan los recursos del sistema. Por ejemplo, un atacante puede consumir toda la memoria o espacio en disco disponible, así como enviar tanto tráfico a la red que nadie más pueda utilizarla. [17]

Dentro del ataque por flooding, el atacante satura el sistema con mensajes que requieren establecer conexión. Sin embargo, en vez de proveer la dirección IP del emisor, el mensaje contiene falsas direcciones IP usando Spoofing y Looping. El sistema responde al mensaje, pero como no recibe respuesta, acumula buffers con información de las conexiones abiertas, no dejando lugar a las conexiones legítimas.

Dentro de los ataques de flooding más utilizados y los cuales serán atacado en esta investigación encontramos los que a continuación se enlistan:

2.2.1 ICMP Flooding

El ataque ICMP es una técnica DDoS que pretende agotar el ancho de banda de la víctima. Consiste en enviar de forma continuada un número elevado de paquetes ICMP Echo request (ping) de tamaño considerable a la víctima, de forma que esta ha de responder con paquetes ICMP Echo Reply (pong) lo que supone una sobrecarga tanto en la red como en el sistema de la víctima como se muestra en la Figura 3. [17]

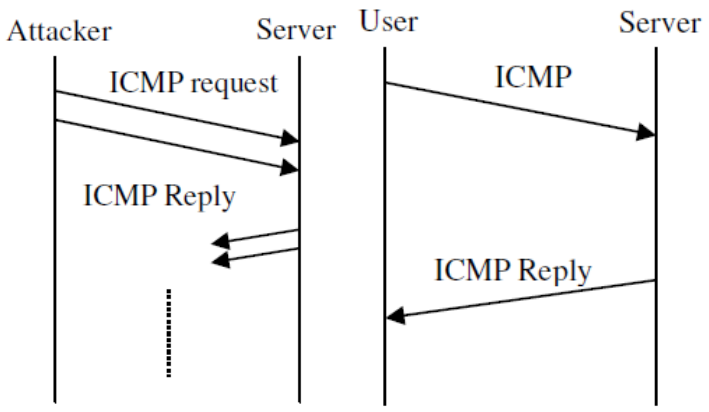


Figura 3. Funcionamiento de comunicación con ICMP y ataque ICMP.

Dependiendo de la relación entre capacidad de procesamiento de la víctima y el atacante, el grado de sobrecarga varía, es decir, si un atacante tiene una capacidad mucho mayor, la víctima no puede manejar el tráfico generado.

2.2.2 Smurf Flooding

Existe una variante a ICMP Flooding denominado Ataque Smurf que amplifica considerablemente los efectos de un ataque ICMP, este ataque realiza una suplantación de las direcciones de origen y destino de una petición ICMP del tipo echo-request como se puede apreciar en la Figura 4. [17]

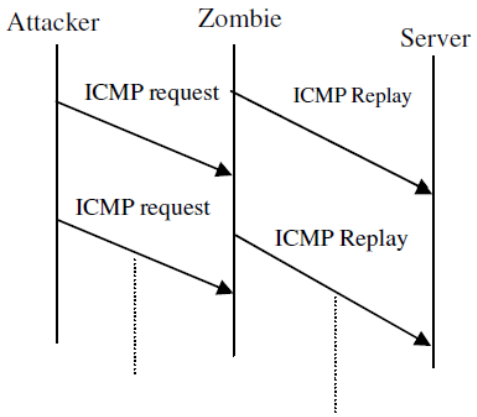


Figura 4. Ataque por Smurf flooding.

En el ataque Smurf se coloca como dirección de origen la dirección IP de la máquina que debe ser atacada. En el campo de la dirección IP de destino se pone la dirección de difusión de la red local o red que se utilizará como trampolín para colapsar a la víctima. Con esta petición fraudulenta, se consigue que todas las máquinas de la red respondan a la vez a una misma máquina, consumiendo todo el ancho de banda disponible y saturando el ordenador atacado.

2.2.3 Fraggle Flooding

Este ataque es una variantes del ataque Smurf flooding el cual consiste en colocar como dirección de origen la dirección IP de la máquina que debe ser atacada y como dirección IP de destino se pone la dirección de difusión de la red local o red que se utilizará como trampolín para colapsar a la víctima, esto con el envío de paquetes UDP (ver Figura 5).

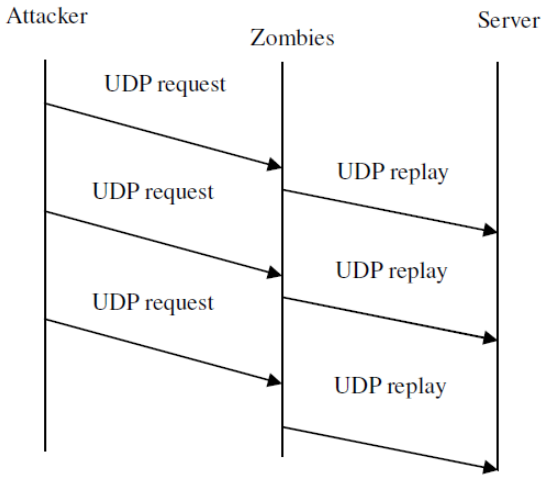


Figura 5. Ataque Fraggle flooding.

2.2.4 Land Flooding

Un ataque LAND se realiza al enviar un paquete TCP/SYN falsificado con la dirección del servidor objetivo como si fuera la dirección origen y la dirección destino a la vez (ver Figura 6). Esto causa que el servidor se responda a sí mismo continuamente y al final falle. [17]

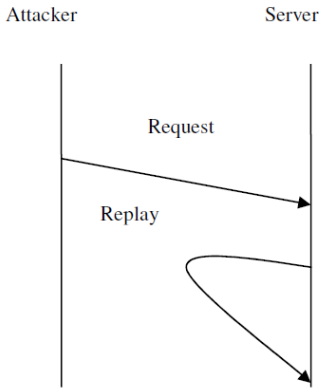


Figura 6. Ataque por Land Flooding.

2.2.5 UDP Flooding

UDP es un protocolo sin conexión y no requiere ningún procedimiento de establecimiento de conexión para transferir los datos. Este ataque es posible cuando un atacante envía un paquete UDP al azar a un puerto en el sistema de la víctima. Cuando el sistema de la víctima recibe un paquete UDP, el sistema determina qué aplicación está esperando en el puerto de destino. Cuando se da cuenta de que no hay ninguna aplicación que está esperando en el puerto, se genera un paquete ICMP de destino inaccesible a la dirección de origen falsificada.

Si los paquetes UDP suficientes se entregan a los puertos de la víctima, el sistema deja de funcionar (ver Figura 7). [17]

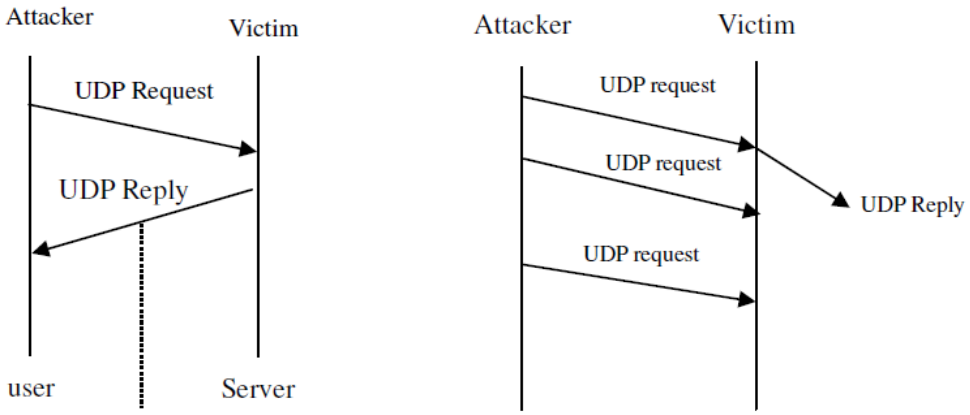


Figura 7. Comunicación de protocolo UDP y ataque UDP.

2.2.6 SYN Flooding.

La inundación SYN envía un flujo de paquetes TCP/SYN, muchas veces con la dirección de origen falsificada. Cada uno de los paquetes recibidos es tratado por el destino como una petición de conexión, causando que el servidor intente establecer una conexión al responder con un paquete TCP/SYN-ACK y esperando el paquete de respuesta TCP/ACK (Parte del proceso de establecimiento de conexión TCP de 3 vías) como se muestra en la Figura 8. Sin embargo, debido a que la dirección de origen es falsa o la dirección IP real no ha solicitado la conexión, nunca llega la respuesta. Estas conexiones a medias consumen recursos en el servidor y limitan el número de conexiones que se pueden hacer, reduciendo la disponibilidad del servidor para responder peticiones legítimas de conexión. [17]

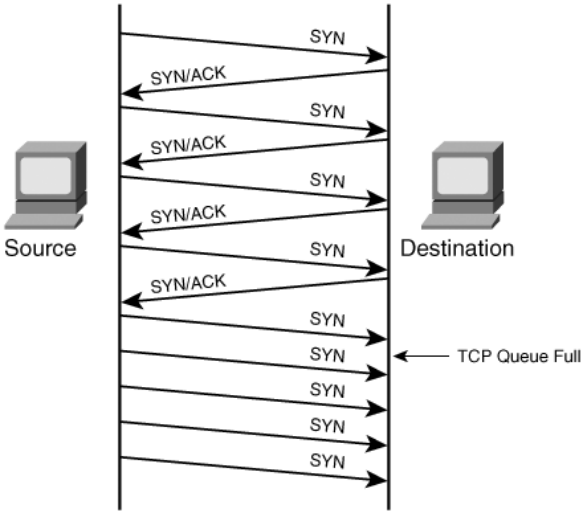


Figura 8. Ataque por SYN Flooding.

2.2.7 HTTP Flooding

La idea principal del ataque HTTP es la de solicitar conexiones con el servidor web (HTTP) y una vez realizado el "handshake" enviar datos de solicitud a una velocidad muy baja, de forma que el servidor se vea obligado a mantener la comunicación abierta esperando que se complete la solicitud (request). Si al mismo tiempo se abren cantidades de conexiones simultáneas con el mismo propósito, se llegará a obstruir las comunicaciones con el servidor ya que este en algún momento podría quedar sin recursos de comunicación disponibles (ver Figura 9). [17]

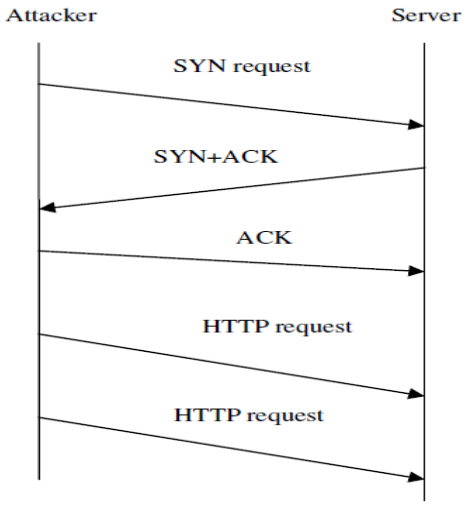


Figura 9. Ataque HTTP Flooding

Capítulo III. Estado del arte

Existen múltiples técnicas con las cuales se ha atacado el problema de la detección de ataques por DDos, sin embargo, sólo algunas de ellas se aplican en un entorno de red real y de trabajo efectivo. El diseño y la implementación de un mecanismo de defensa DDoS perfecto y sensible a cambios es realmente difícil, debido a la complejidad con la cual estos ataques van siendo más robustos y elaborados por los atacantes. Un mecanismo de defensa ante un ataque DDoS ideal según Kumar debe tener las siguientes características: ser eficaz, transparente a la infraestructura existente de Internet, de bajo costo en el gasto de recursos, tolerante a despliegues incrementales y que no tenga ningún impacto en el tráfico legítimo. [44]

El enfoque de un mecanismo de detección ideal ha sido estudiado por muchos investigadores, lo cuales han tratado de darle solución a los retos y problemas abiertos que son enlistados en [44], utilizando técnicas estadísticas, basado en conocimiento, técnicas de computo suave y técnicas basadas en grafos; algunos de las propuestas realizadas por los investigadores para darle solución a la detección de un ataque por DDos se describen a continuación:

3. 1 Técnicas Estadísticas.

Feinstein et al., propone en [45] utilizar métodos estadísticos para la identificación de ataques DDoS calculando entropía y distribuciones ordenadas por frecuencia de los paquetes seleccionados con el método estadístico de chi-square, utilizando trazas de tráfico del núcleo de la red. Entre las desventajas de estos métodos se encontró: Un atacante sofisticado probablemente intente anular el algoritmo de detección mediante la creación de tráfico sigiloso que imite el tráfico legítimo que el detector esperaría. Otro problema con este enfoque es que en algunos entornos de red, una dirección de origen previamente no vista pueden tener un impacto inaceptable en el tráfico legítimo.

No, G., & Ra, I., en [49] propone un enfoque de entropía rápida combinada, con la compresión sin pérdida de entropía del método FOEC, y la calibración de la entropía que

utiliza el número de tipos de paquetes y el número de paquetes basado en la idea de que los ataques DDoS se basan principalmente no solo en los tipos de paquetes como en la entropía convencional, pero si en los tipos de paquetes y el volumen de tráfico (el número de paquetes). La simulación que se utilizó para probar el enfoque de entropía por compresión muestra que el método de entropía rápida no sólo reduce el tiempo de cálculo en más de un 90% en comparación con la entropía convencional, sino que también aumentó la exactitud de detección en comparación con la entropía convencional.

Kim et al. , se centra en [62] en el diseño y evaluación de la caracterización automatizada de ataque, la selección de paquetes descartados y la parte de control de la sobrecarga de la arquitectura propuesta. La idea clave es dar prioridad a los paquetes en base puntuación que estima la legitimidad de un paquete dado los valores de atributo que lleva. Las consideraciones especiales se hacen para garantizar que el sistema sea susceptible a la implementación de hardware de alta velocidad. Una vez que se calcula la puntuación de un paquete, se descartan los paquetes según el que umbral de goteo, el cual se ajusta dinámicamente basado en los resultados de los últimos paquetes entrantes y la corriente del nivel de sobrecarga del sistema. "El esquema propuesto consiste de las 3 fases siguientes:

- Detectar la aparición de un ataque e identificar a la víctima mediante el monitoreo de cuatro estadísticas de tráfico principales de cada objetivo protegido, manteniendo el estado de destino.
- Diferenciar entre paquetes legítimos y paquetes de ataque con destino hacia la víctima basado en el cálculo de la teoría métrica bayesiana de cada paquete. La métrica es la denominada ""probabilidad condicional legítimo"" (CLP).
- Descartar paquetes selectivamente mediante la comparación de la CLP de cada paquete con un umbral dinámico. El umbral se ajusta de acuerdo con la distribución de CLP de todos los paquetes sospechosos y el nivel de congestión de la víctima.

Jin et al., en [22] propone un método de análisis de covarianza basada en estadísticas, pero a diferencia de las obras existentes, que no se basa en presunciones sobre la normalidad distribuciones de paquetes de red. Este método es eficaz porque detecta con precisión los ataques DDoS de diferentes intensidades, y porque el método es simple, puede ser implementado en línea. En este documento, todas los flags en el campo de control de la

cabecera TCP se utilizan como características en el modelo de análisis de covarianza. Los resultados de la simulación mostraron que este método es muy preciso en la detección de ataques de inundación SYN en DDoS. Este método podría también diferenciar eficazmente entre el tráfico normal y ataque.

Lo que es más, se podría detectar ataques de intensidad muy sutil, exponer comportamientos-cerca-a la normalidad. Los resultados de alto exactitud de la detección en los experimentos y en análisis en tiempo real, revelan algunos impactos del análisis multivariante de correlación en la detección de los ataques DDoS.

Sin embargo, el método actual tiene tres limitaciones importantes:

En primer lugar, no hay garantía de que los 6 flags son válidas o que son características suficientes en el modelo de análisis de covarianza para detección DDoS. En segundo lugar, ninguna justificación teórica es proporcionada por la alta tasa de detección como se ha demostrado en los experimentos. En tercer lugar, la forma de seleccionar el intervalo de tiempo adecuado es aún un problema abierto.

Mirkoviac et al., en [23] propone un esquema de defensa DDoS llamado D-WARD [6]. Este sistema está instalado en el router que actúa como fuente entre la red de origen y el Internet. Eso identifica un ataque basado en la observación continua de dos vías el tráfico entre las dos redes. D-WARD compara las estadísticas recogidas con el modelo predefinido de la normalidad tráfico periódicamente. Si el resultado de la comparación muestra que hay es la posibilidad de un ataque DDoS, D-WARD impondrá una límite de velocidad en el flujo saliente sospechoso entre la red de origen y el Internet. Cada registro guardado por D-WARD contiene estadísticas sobre tres tipos de tráfico: TCP, UDP e ICMP. Su viabilidad depende de la voluntaria cooperación de la mayoría de administradores de red de toda la Internet. En teoría, se puede eludir este problema de despliegue mediante la aplicación del enfoque D-WARD a la red troncal. Sin embargo, para realizar un enfoque de este tipo "espina dorsal", se debe abordar los principales problemas de escalabilidad como el gran número de objetivos necesarios para ser protegida y el alto velocidad de funcionamiento dentro de la red troncal.

En conclusión, los trabajos analizados para la detección basada en las estadísticas es que no es posible determinar con certeza la normalidad la distribución de paquetes de red. Más bien, sólo se puede simular como una distribución uniforme.

3. 2 Técnicas Basadas en conocimiento.

En este tipo de método, eventos de ataque se comparan contra patrones predefinidos de ataque. Las características generales de los conocidos ataques están formulados para identificar los sucesos reales de ataques. Los ejemplos de estos enfoques encarnan en sistemas expertos, la organización de los auto- mapas, análisis de firmas, y el estado análisis de transición.

Wang et al., propone en [24] una manera formal y metódica de modelar ataque DDoS por el método de Arbol de Ataque Aumentado (AAT), y presenta un algoritmo de detección de un ataque basado en AAT. Este modelo refleja los patrones de ataque y las transiciones de estado correspondientes en el servidor principal víctima.

Limwiwatkul y Rungsawang, proponen en [25] encontrar firmas de ataques DDoS por el análisis de la cabecera del paquete los protocolos TCP / IP contra las reglas predefinidas y distinguir el tráfico normal y anormal. Este método se centran en los ataques TCP / IP, ICMP, UDP y los ataques por flooding.

Zhang y Parashar proponen en [26] un enfoque distribuido para detectar ataques DDoS. Este enfoque detecta y detiene los ataques DDoS de forma independiente en la red intermedia. Un mecanismo de comunicación se utiliza para intercambiar datos sobre ataques de red entre los nodos de detección independientes para agregar datos sobre los ataques generales de la red. Los nodos de defensa individuales obtienen datos aproximados sobre los ataques de red globales y pueden detener de manera más eficaz y precisa.

Lu et al., en [27] describen sistema de DDoS que analice el tráfico en los enrutadores de borde de una red ISP. Este enfoque puede detectar e identificar los paquetes de ataque sin necesidad de modificar los mecanismos de reenvío de IP existentes en el router de forma precisa.

Dongwon SEO et al. [28] propone la Programación Filtro Probabilístico (PFS), al derrotar de manera eficiente los ataques DDoS y para satisfacer las propiedades necesarias. En la SSP, los routers de filtro identifican caminos de ataque utilizando el marcado de paquetes probabilística, y mantienen filtros utilizando una política de planificación para maximizar la efectividad de defensa. Los resultados muestran que PFS logra eficacia 44% más alto que otros enfoques basados en filtros.

Lin plantea en [29] una taxonomía de las técnicas de detección y mitigación de ataques de denegación de servicio basadas en firmas con sistema como SNORT, PHAD, MADAM, y MULTOPS. Se evaluaron los 4 sistemas de detección propuestos para su análisis, lo que conllevo a identificar que la detección de la vulnerabilidad conocida como ataque DoS en SNORT tiene la mayor tasa de detección ya que usa firmas de ataques actualizados de forma activa, por su parte MADAM también tiene una alta tasa de detección debido al uso de conjuntos de datos de ataques conocidos, en el caso de PHAD tiene una taza de detección moderada ya que no cuenta con conocimiento previo sobre el ataque y MULTOPS no es capaz de detectar ataques Dos. En cuanto ataques no conocidos el sistema de detección PHAD tiene la mayor eficiencia debido a que no requiere conocimientos previos de ataques. Por su parte la tasa de falsas alarmas tanto SNORT y la MADAM tienen baja tasa de falsas alarmas debido a su enfoque en la detección de ataques conocidos. PHAD, como un detector de anomalías en función, tiene relativamente alta tasa de falsas alarmas. La tasa de falsas alarmas de MULTOPS depende de la eficacia de la tasa de paquetes proporcional.

En conclusión, se observa que los métodos basados en conocimiento tienen una desventaja notable, ya que solo comparan el patrón de ataque con los ya conocidos utilizando firmas o conocimiento previo de los ataques, lo que implica que la detección de ataques nuevos no se contempla dentro de esta técnica como una prioridad.

3. 3 Técnicas de cómputo suave.

Las técnicas Softcomputing o cómputo suave es un término general para describir una colección de técnicas de optimización y de procesamiento que son tolerantes de imprecisión e incertidumbre, dentro de esta técnica encontramos diversos investigadores que han tratado de darle solución al problema de la detección de ataque por DDos, los cuales se mencionan a continuación.

Karimazad et al., propone en [30] detectar ataques DDoS con un método de detección de anomalías basado en las diversas características de paquetes de ataque, obtenido del estudio del trafico entrante de la red y el uso de la Función de base radial (RBF) redes neuronales para analizar estas características. La arquitectura del sistema propuesto se divide en cuatro módulos (Modulo de captura de paquetes, módulo de detección DDoS, el módulo de filtrado y módulo de alarma de ataque). En el módulo de captura de paquetes detecta el tráfico de entrada a la red y captura los paquetes con protocolo TCP, UDP e ICMP en ventanas de tiempo fijos. Todos los paquetes capturados son enviados al módulo de detección DDoS. En el módulo de detección DDoS consta de tres partes: Módulo de extracción de características, Módulo de Tráfico agregación y motor de correlación. Este módulo analiza los paquetes para reconocer ataques capturado y clasifica los paquetes en tráfico de ataque y tráfico normal.

El módulo de extracción de características calcula las características seleccionadas para los paquetes capturados, el autor propone utilizar el tamaño promedio del paquete, número de paquetes, varianza de intervalo de tiempo, varianza de tamaño de paquete, número de bytes, calidad y velocidad de paquete como características que se utilizarán en el motor de correlación, donde las siete características obtenidas en el módulo de extracción son utilizadas para activar la red neuronal RBF para clasificar el tráfico de red en normal o ataque. Si el tráfico entrante es reconocido como tráfico de ataque, este será enviado al módulo de alarma.

El módulo de tráfico de agregación, registra el tráfico entrante procedente de diferentes fuentes a un destino específico lo cuales son enviados al motor de correlación para detectar ataque DDoS; el módulo de filtrado por su parte consiste en dos sub-módulos: Tabla IP de Fuente sospechoso y el módulo de actualización. Después la detección de paquetes de ataque, el módulo de detección DDoS envía la dirección IP de origen del ataque al módulo de actualización. El módulo de actualización actualiza la Tabla de Fuentes IP sospechosas y después, el módulo de filtrado filtra las entrada de paquetes a la red basados en la tabla de IP sospechoso. Y por último el módulo de alarma de ataque donde se emite la alarma al administrador para tomar las medidas necesarias.

Se utilizó una red simulada donde se capturaron 41,175 paquetes para el tráfico normal y 36.325 paquetes para el tráfico de los ataques DDoS. Los resultados experimentales muestran que la detección de DDoS es de precisión sobre el 98,2% del conjunto de datos de la UCLA y también sobre el 96,5% de la red simulada.

Rui Zhong and Guangxue Yue propone en [31] un modelo de detección de ataques DDoS con algoritmos de minería como algoritmo de agrupamiento FCM y el algoritmo de asociación a priori. Dentro de las variables seleccionadas para este modelo, se tiene: DST, DST-IP, puertos, flags, src bytes y DST-byte los cuales son utilizados como parámetros de detección del estado del protocolo de paquetes donde DST-IP representa destino IP, DST-puerto representa el puerto de destino, flags representa el estado de extremo de conexión incluye valores SF (extremo de conexión normal), REJ (solicitud de conexión negada), src-byte representa byte fuente, DST-byte representa byte destino. Del análisis de los ataques DDoS en este experimento, se encontró que este sistema tiene un alto grado de eficiencia en la detección, con una tasa de detección del 97% en un entorno controlado LAN.

VeetiL et al., propone en [32] un sistema de detección de intrusos que emplea un algoritmo bayesiano que se ejecuta de manera distribuida. El clasificador utiliza las API de Apache Hadoop y Streaming para detectar intrusiones en tiempo real. Los datos de tráfico de red generado por los rastreadores de paquetes no comerciales sirven como entrada a nuestro clasificador. El objetivo es desarrollar un fallo en el sistema distribuido, escalable, tolerante y confiable. El sistema está diseñado de tal manera que se puede ejecutar sin esfuerzo en un clúster compuesto de hardware obsoleto. Los resultados indican que el rendimiento del sistema propuesto es aceptable en escenarios reales. El resultado muestra que la formación en el clúster homogéneo es 37% más rápido que el algoritmo bayesiano independiente y los algoritmos mejorados discutidos en [32]. Usando el modelo de conocimiento, nos encontramos con un experimento para determinar el rendimiento del clasificador basado en Hadoop en términos del número de registros que se clasifica para una unidad de tiempo dada. Snort captura 527 paquetes en 49 segundos, lo que significa que Snort captura al menos 10 paquetes por segundo. Nuestro clasificador de secuencias se puede clasificar a un promedio de 434 paquetes por minuto. La tasa de detección de streaming es del 90%. Por lo tanto, llegamos a la conclusión de que el sistema tiene éxito en demostrar la técnica de prueba de

concepto utilizando el clasificador bayesiano basado en Hadoop para la detección de intrusiones.

Gavrilis y Dermatas en [33] presentan un detector de DDoS basados en características estadísticas estimadas en tiempo corto de análisis de paquetes de datos entrantes en redes públicas. Los descriptores estadísticos se utilizan para describir el comportamiento de los ataques DDoS. Una clasificación exacta es lograda mediante redes neuronales RBF.

Wu et al., en [34] propone detectar ataques DDoS usando análisis relacional grey y árboles de decisión. Ellos usan quince atributos que controlan la tasa de entrada/ salida de paquetes, bytes y compila paquetes del protocolo TCP, SYN, y las tasas de flags ACK, para describir el patrón de tráfico. La técnica de árbol de decisión desarrolla un clasificador para detectar el tráfico anormal, también utilizan un patrón de coincidencia de patrones de tráfico para detectar el flujo de ataque y respaldar el origen de un ataque basado en esta similitud.

Chen et al. [35] presentan un marco global para la detección de ataques DDoS conocido como DDoS de Container. Funciona en el modo en línea para examinar y manipular el tráfico en tiempo real. DDoS Container va a hacer la inspección de los flujos de datos y correlaciona eventos de ataque en diferentes sesiones. Finaliza una sesión cuando se detecta un ataque.

Muchas de las estrategias para la detección DDoS se han reportado en la literatura, sin embargo, los principales retos que todo sistema de defensa debe DDoS superar para ser idealmente utilizable se dan a continuación. [44]

1. Se debe de dar más énfasis a la velocidad en la que se realiza ya que por lo general consume una mayor potencia de procesamiento que también puede afectar a la precisión de detección.

2. Detección en tiempo real de los ataques por flooding, el cual se realice con precisión aminorando falsas alarmas, ya que este tipo de ataques sigue una distribución normal del tráfico cotidiano.

3. Sistemas de detección escalables en redes de alta velocidad.

4. El desarrollo de un enfoque combinado basado en tanto enfoques supervisados y no supervisados con la capacidad de detectar ataques tanto conocidos como desconocidos en tiempo real o casi en tiempo real es de suma necesidad.

5. Detección de las variantes de los ataques por DDos, deben ser efectivas con mínimas tasas de error.

6. Transparencia a la infraestructura de Internet existente es increíblemente importante en términos de despliegue. Por lo tanto, un esquema de defensa ante un DDos debe poder desplegarse en redes reales.

7. Actualizaciones en tiempo real de las estadísticas de la red y rápida identificación de direcciones IP simuladas.

8. Un mecanismo de defensa DDoS con el objetivo de dar una cerca solución en tiempo real puede tener que basarse en una copia de algún algoritmo de agrupamiento incremental para segregar el ataque de tráfico normal. Esto requiere una medida de proximidad apropiada que trabaja con sensatez, rápida y fiable.

10. El método de detección debería depender de un número mínimo de parámetros de entrada si no es independiente de parámetros y también deben estar basados en un mínimo parámetros de tráfico o características.

3. 1 Técnicas basadas en grafos.

En el ámbito de la minería de datos, otra de las técnicas utilizadas para la identificación de anomalías en el tráfico de red, se encuentra la implementación de la teoría de grafos. Dentro de esta área de investigación podemos encontrar un enfoque muy particular de aplicación en los sistemas de detección de intrusiones; a continuación se mencionan algunos de las retos a los que se han enfrentado los investigadores para darle solución al problema de la detección de intrusiones basada en grafos.

Jeena,R et al. [39] Propone un modelo basado en grafo de ataque para la detección de ataques DDoS en la nube, donde los IDS se coloca en el conmutador de red para controlar todas las actividades de los nodos en el sistema de nube; después con el VMprofiling se exploran todas las máquinas virtuales en la nube, con la finalidad de capturar los datos de las VM

(Virtual Machine) incluyen el nombre de máquina virtual, la dirección IP, CPU, memoria y sistema operativo.

En el segundo proceso se detectan los nodos infectados, cada vez que el usuario accede al servicio, se registran sus datos, si la tasa de envío de datos es mayor que el umbral propuesto se genera una alerta, y el modelo de grafo de ataque se construye. El tercer proceso consta de la construcción del grafo de ataque, el cual se divide en SAG (Escenario de grafo de ataque) y ACG (Grafo de correlación de ataque).

El SAG es un subgrafo que representa una alerta producida por el sistema la cual contiene dirección de IP de origen y destino, tipo de alerta y marca de tiempo de alerta. Para ACG las alertas producidas por el sistema en SAG son asignadas a los nodos del grafo, las aristas dirigidas representarán la correlación entre las alertas, lo que da como resultado un conjunto de alertas relacionadas en orden cronológico para la detección de un ataque.

Noble, C et al. [40] Propone dos métodos para la detección de anomalías en la red utilizando grafos, el primero método es la detección de subestructuras anómalas, la cual busca subestructuras específicas, inusual dentro del grafo. En el segundo método es la detección de subgrafos anómalos, el grafo se divide en distintos conjuntos de vértices (subgrafos), cada uno de los cuales se comprueba frente a los otros por los patrones inusuales. Además, se introduce una medida de regularidad del grafo llamada entropía condicional, que define el número de bits necesarios para describir los alrededores de una subestructura arbitraria.

Swapnali G et al. [41] Propone un enfoque que utiliza la detección de vulnerabilidades de fases múltiples a nivel distribuido utilizando un mecanismo llamado NICE, que está basado en los modelos basados en grafos de ataque y utiliza contramedidas basadas en redes virtuales reconfigurables.

Las funciones clave del sistema NICE se llevan a cabo por el analizador de ataques, incluyendo procedimientos como construcción de grafo de ataque y actualización, la correlación de alertas, y la selección de contramedidas. El proceso de construcción y utilizando el SAG consta de tres fases:

I. La recopilación de información,
II. Construcción de grafo de ataque, y

III. Explotar el análisis potencial de la ruta.

Mediante el uso de esta información, los caminos de ataque se puede demostrar utilizando SAG. Cada nodo en el grafo de ataque representa un exploit por un atacante, cada camino sencillo de un nodo inicial a una nodo objetivo denota un ataque con éxito. El analizador de ataques se ocupa de la correlación y el análisis de operaciones de alerta. Esto con la finalidad de tener dos funciones principales: 1. Las construcciones ACG, 2. Proporciona información sobre amenazas y realizar las contramedidas apropiadas en la red.

Eberle, W et al. [42] Propone un método de análisis de grafos usando un enfoque de partición paralelo que puede descubrir subestructuras anómalos, llamado PLADS, lo que hace frente al problema de la escalabilidad asociado con la realización de la detección de anomalías basados en grafos, debido a la gran cantidad de datos a procesar y a las estructura que debe detectar. Este enfoque utiliza como entrada N número de particiones en el grafo, lo que le permite al algoritmo dividir un grafo grande en segmentos de los datos que se procesa de forma individual, acelerando así la búsqueda de subestructuras inusuales con GBAD en esas pequeñas porciones de forma paralela con la búsqueda en las otras particiones, lo que permite tener un resultado más rápido.

Le, D et al. [43] Propone la detección de tráfico de dispersión basado en grafos de anomalías, utilizan un nuevo método para detectar el tráfico de red basado en conceptos de teoría de grafos anómalos, tales como distribución de grado, grado máximo y distancia DK-2. En este enfoque, se ha usado grafos de dispersión de tráfico (TDG) para modelar el tráfico de red en series de tiempo. Se analizan las diferencias de grafos TDG para detectar anomalías e introducir un método para identificar patrones de ataque en el tráfico anómalo. Donde el TDG es un grafo en el que cada nodo representa una dirección IP y cada borde representa una conexión entre dos direcciones IP; lo primero que se detecta son los TDG anormales y luego se identifica la patrones de ataque dentro de ellos. De esta manera, se reduce la complejidad computacional del método del grafo utilizado.

El enfoque de detección de dispersión basado en grafos de anomalías propone varios pasos en el proceso de detección de anomalías:

I. Muestreo de tráfico de red y generación de flujos de red.

II. Creación de TDG (formato de punto) donde el flujo de red fluye en intervalos de muestreo de tiempo.

III. Cálculo de matrices de adyacencia de la TDG y el cálculo de las métricas del gráfico de la TDG.

IV. Comparación de los valores de las métricas del gráfico de la TDG con su valor umbral. Si un valor de una de las métricas del gráfico excede su valor de umbral, entonces el TDG es un TDG anormal, de lo contrario el TDG es uno normal. Para determinar los valores de umbral, se estudió el tráfico de red a largo los registros de la salida a Internet POSTECH. En este trabajo, se utilizar valores umbrales estáticos, que se determinan a partir de la comparación de tráficos normales y anómalos. Los valores umbral tienen que determinarse en cada caso de acuerdo con los entornos de red particulares.

Como se puede observar en los trabajos mencionados anteriormente, el problema de detección de ataques DDos sigue siendo un caso abierto a la investigación, en esta propuesta lo que se pretende es atacar el problema de detección de ataques de DDoS, en específico los ataques por inundación (Flooding), reduciendo las falsas alarmas y proponiendo un modelo que pueda ser implementado en redes con tráfico denso. Dentro de los trabajos analizados se tiene la siguiente Tabla 1.

Tabla 1. Comparativa de trabajos relacionados que atacan el problema de detección de DDos y la propuesta de investigación.

Tipos de Ataques por Flooding	Reyhaneh et al. [29]	Rui Zhong et al. [30]	Jin et al. [62]	Propuesta
ICMP Flooding	X			X
Smurf Flooding				X
Land Flooding				X
TCP Flooding	X	X		X
UDP Flooding	X			X
Fraggle Flooding				X
HTTP Flooding				X
SYN Flooding		X	X	X

Capítulo IV. Modelo para la detección de ataques por flooding utilizando grafos.

Recientemente, los daños causados por los ataques DDoS aumenta, la rápida detección y los mecanismos de respuesta adecuados son urgentes. Sin embargo, con la variación en los tipos de ataques DDos, la detección se ha convertido en un caso de estudio ya que los mecanismos de defensa utilizados actualmente ante estos ataques no proporcionan un método eficaz en la detección. Es por ello que dentro de esta investigación se propone la detección de ataques por flooding analizando primeramente los procedimientos a realizar para efectuar los ataques y posteriormente seleccionando las variables basadas en estas características. Mediante el uso del algoritmo de Esperanza Múltiple se identificarán los k grupos que ayudarán como punto partida para la implementación del algoritmo de agrupamiento de k vecinos, con el que se determinarán los patrones existentes sobre las variables seleccionadas. A partir de cada clúster se analizarán las trazas de tráfico utilizando grafos híbridos para descubrir los casos de subestructura que contienen entidades y relaciones anómalas, con la finalidad de realizar la detección del ataque por flooding basada en tales anomalías.

4. 1 Descripción del problema

En este capítulo se presenta la metodología a seguir para la elaboración del modelo de detección de ataques por Flooding utilizando grafos. La metodología será detallada en dos secciones como se muestra en la figura 10, la primera consta de obtención de los patrones del tráfico de la red estudiada, la cual será llevará a cabo fuera de línea; y la segunda sección analizará el tráfico de red con grafos híbridos con la finalidad de buscar anomalías en el tráfico de red generado en línea.

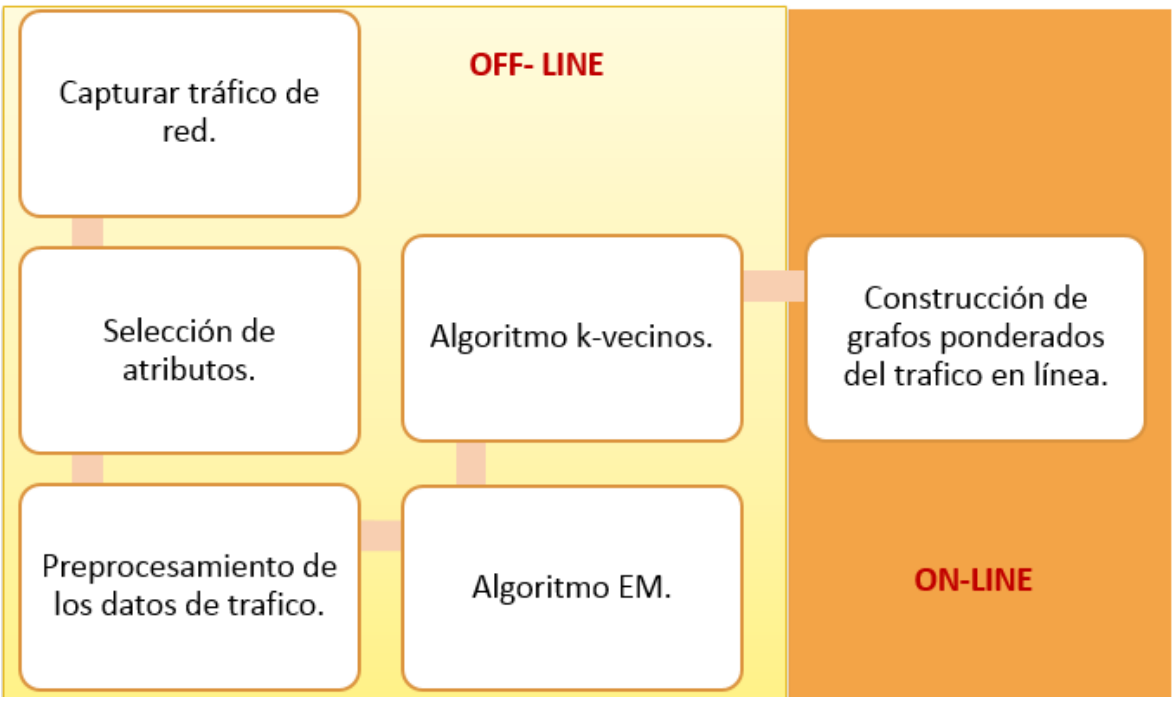


Figura 10. Diagrama del modelo de detección de ataques por flooding utilizando grafos.

4. 1.1 Adquisición de datos

En este módulo se capturará el tráfico de entrada a la red, lo que dará la facilidad de conocer la duración de los paquetes, el protocolo bajo el cual se ha realizado la petición, el servicio y algunas otras características de los paquetes, lo cual será de ayuda para la selección de las variables que se necesitarán para la detección del ataque por flooding.

Para la adquisición de datos se utilizará una base de datos llamada KDD CUP 99, la cual incluye las características básicas de una conexión TCP individual, tales como: su duración, tipo de protocolo, número de bytes transferidos y la bandera que indica el estado normal o de errores de la conexión. Estas características intrínsecas " proporcionan información para uso general con fines de análisis de redes de tráfico. Dado que la mayoría de los ataques DDos y Probe implican un envío de una gran cantidad de conexiones a la misma máquina (s) al mismo tiempo, se puede tener patrones secuenciales frecuentes, que son diferentes para el tráfico normal.

Para estos modelos, una característica "same host" examina todas las demás conexiones en los 2 segundos anteriores, que tenían el mismo destino que la conexión actual. Del mismo modo, una característica "same services" examina todas las demás conexiones en los 2

segundos anteriores, que tenían el mismo servicio que la conexión actual. Estas características temporales y estadísticas se refieren como el "time based" de las características de tráfico.

En total, hay 41 variables, incluyendo el tipo de ataque en cada registro de conexión, con la mayor parte de ellas teniendo en valores continuos, como se muestra a continuación en la tabla 2.

Tabla 2. Características de la base de datos KDDCUP99.

Características básicas de conexiones TCP individuales		
Variable	Descripción	Tipo de dato
Duration	Longitud (número de segundos de la conexión)	Continuo
Protocol_type	Tipo de protocolo (UDP/TCP)	Discreto
Service	Servicio de red en el destino (http/telnet)	Discreto
src_bytes	Número de bytes de datos del origen al destino	Continuo
dst_bytes	Número de bytes de datos del destino a la fuente	Continuo
Flag	Estado normal o error en conexión	Discreto
Land	1 si la conexión es desde/hasta el mismo host/puerto; y 0 en otro caso	Discreto
Wrong_fragment	Número de fragmentos erróneos	Continuo
Urgent	Número de paquetes urgentes	Continuo

Características de contenido dentro una conexión sugerido por conocimiento del dominio		
Variable	Descripción	Tipo de dato
Hot	Número de indicadores hot	Continuo
Num_failed_logins	Número de intentos de acceso fallido	Continuo
Logged_in	1 si se inicia sesión correctamente; 0 para otro caso	Discreto
Num_compromised	Número de condiciones comprometidas	Continuo

Root_shell	1 si se obtiene root shell; 0 en otro caso	Discreto
Su_attempted	1 si el comando su root se intenta; 0 en otro caso	Discreto
Num_root	Número de accesos root	Continuo
Num_file_creations	Número de operaciones de creación de archivos	Continuo
Num_shell	Número de shell prompts	Continuo
Num_acces_files	Número de operaciones en los archivos de control de acceso	Continuo
Num_outbound_cmds	Número de comandos de salida en una sesión FTP	Continuo
Is_hot_login	1 si la entrada pertenece a la lista de hot;0 de otra manera	Discreto
Is_guest_login	1 si la entrada es una entrada guest; 0 en otro caso	Discreto

Calculo de características de tráfico usando una ventana de dos segundos de tiempo

Variable	Descripción	Tipo de Dato
Count	Número de conexiones a la misma máquina que la conexión actual en los últimos 2 segundos	Continuo

Nota: Las siguientes características se refieren a estas conexiones entre personas del mismo huésped

Variable	Descripción	Tipo de Dato
serror_rate	% de conexión que tienen errores SYN	Continuo
rerror_rate	% de conexiones que tiene errores REJ	Continuo
Same_srv_rate	% de conexiones al mismo servicio	Continuo

Diff_srv_rate	% de conexiones a diferentes servicios	Continuo
Srv_count	Número de conexiones a los mismos servicios que la conexión actual en los últimos 2 segundos	Continuo

Nota: Las siguientes características se refieren a estas conexiones entre personas del mismo servicio		
Variable	Descripción	Tipo de dato
Srv_serror_rate	% de conexiones que tienen errores SYN	Continuo
Srv_rerror_rate	% de conexiones que tienen errores REJ	Continuo
Srv_diff_host_rate	% de conexiones a diferentes servidores	Continuo

Características de tráfico calculadas usando las cien- ventana de conexión Las funciones de tráfico utilizando un centenar- ventana de conexión En el mismo host cxn		
Variable	Descripción	Tipo de dato
dst_host_count	Número de conexiones al mismo host que la conexión actual en los últimos 2 segundos	Continuo
dst_host_serror_rate	% de conexiones que tienen errores SYN	Continuo
dst_host_rerror_rate	% de conexiones que tienen errores REJ	Continuo
dst_host_same_srv_rate	% de conexiones del mismo servicio	Continuo
dst_host_diff_srv_rate	% de conexiones de diferentes servicios	Continuo

En el mismo servicio cnx		
Variable	Descripción	Tipo de dato
dst_host_srv_serror_rate	% de conexiones que tienen errores SYN	Continuo
dst_host_srv_count	Número de conexiones para el mismo servicio que la conexión actual en lo últimos 2 segundos	Continuo
dst_host_ srv_ rerror_rate	% de conexiones que tienen errores REJ	Continuo
dst_host_srv_diff_host_rate	% de conexiones de diferentes host	Continuo

dst_host_same_src_port_ rate	% de conexiones a la máquina de corriente que tiene el mismo puerto src	Continuo

4. 1.2 Selección de atributos

La selección de atributos permitirá determinar los atributos relevantes y desechar aquellos atributos que no aporten la información suficiente para la detección del ataque por Flooding, con el objetivo de obtener un subconjunto de atributos que describa lo más acertado posible el funcionamiento del ataque.

La selección de atributos tiene ventajas como [36]:

- Mejora el rendimiento de los algoritmos de aprendizaje automático.
- Reducción de dimensionalidad.
- Posibilita usar modelos simples ganando así velocidad de procesamiento.

Existen diversas investigaciones donde cada uno de los autores han determinado cuales son las variables más representativas para la detección del ataque por DDoS, a continuación se hará un estudio de las diferentes variables seleccionadas por cada uno de estos investigadores y el porqué de su decisión por tomar como base esos atributos.

Reyhaneh Karimazad propone en [30] el uso de 7 variables obtenidas de la base de datos UCLA CSD las cuales son propuestas con base en la observación de las características del ataque por DDos, dentro de las que menciona en sus articulo está el promedio de tamaño de paquete, número de paquetes, varianza de intervalo de tiempo, varianza de tamaño de paquete, número de bytes, calidad y velocidad de paquete.

Prof Bill Buchanan propone en [37] siete métricas de como tasa de pérdida de paquetes, tiempo para responder, ancho de banda disponible, estado latente, confiabilidad, carga de la CPU y el uso de memoria para la detección de ataques por DDos.

Mihui KiM propone en [38] algunos atributos candidatos como son cantidad de paquetes por flujo (P / F), el octeto TCP recuento por flujo (A / F), el recuento de paquetes TCP por flujo (TP / F), octeto UDP por flujo (UO / F), cantidad de paquetes por flujo UDP (UP / F), la varianza de puerto de origen para el tráfico TCP (srcTport), la varianza de puerto de origen

para el tráfico UDP (srcUport), destino varianza de puerto para el tráfico TCP (dstTport), varianza puerto de destino para UDP tráfico (dstUport), origen de la dirección IP de la varianza (srcVar), la relación de tráfico TCP (Tratio), y la relación de tráfico UDP (Uratio); las cuales fueron obtenidas por NetFlow.

Yoohwan Kim propone en [62] considerar los siguientes atributos enlistados para ser utilizados para perfilar el tráfico: IP protocol-type values, packet size, *server2* port numbers, source/ destination IP prefixes 3, Time-to-Live (TTL) values, IP/TCP header length 4, TCP flag patterns.
También estamos interesados en el fracción de paquetes que utilizan la fragmentación de IP y llevan incorrectas / TCP / UDP de comprobación IP. Vale la pena considerar la distribución conjunta de la fracción de paquetes que tienen varias combinaciones de valor TTL y la fuente prefijo IP, packet-size y protocol-type, server port number y ProtocolType, así como prefijo IP fuente y el valor TTL.

Otro de los autores quien exploro tres métodos para la selección de atributos fue Mukherjee and Sharma en [39] quien exploro con métodos de selección de características la base de datos KDD con la finalidad de obtener las variables más representativas:

- Correlation-based Feature Selection.
- Information Gain and Gain Ratio.
- Feature Vitality Based Reduction Method.

De lo cual se obtuvo las siguientes características, como se muestra en la tabla 3:

Tabla 3. Atributos de KDDCUP99 seleccionados por cada método utilizado por Mukherjee and Sharma.

Algoritmo de selección	Cantidad de atributos seccionados	Atributos seleccionados
CFS+ BestFirst	10	Service, src_bytes, dst_bytes, Flag, Logged_in, Same_srv_rate, Srv_serror_rate, Srv_rerror_rate, dst_host_srv_serror_rate, dst_host_srv_count

GR+Ranker	14	Service, src_bytes, dst_bytes, Flag, Num_failed_logins, Logged_in, Is_guest_login, rerror_rate, Same_srv_rate, Srv_serror_rate, Srv_rerror_rate, dst_host_srv_serror_rate, dst_host_srv_count, dst_host_ srv_ rerror_rate
InfoGain+Ranker	20	Service, src_bytes, dst_bytes, Flag, Logged_in, Count, serror_rate, rerror_rate, Same_srv_rate, Srv_serror_rate, Srv_rerror_rate, Srv_diff_host_rate, dst_host_count, dst_host_serror_rate, dst_host_rerror_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_srv_serror_rate, dst_host_srv_count, dst_host_ srv_ rerror_rate
FVBRM	24	Duration, Service, src_bytes, dst_bytes, Flag, Land, Wrong_fragment, Urgent, Hot, Num_failed_logins, Logged_in, Num_compromised, Root_shell, Su_attempted, Num_root, Num_file_creations, Num_shell, Num_acces_files, Count, serror_rate, dst_host_count, dst_host_serror_rate, dst_host_diff_srv_rate, dst_host_srv_count, dst_host_srv_diff_host_rate

De lo anterior, se puede observar que cada uno de los investigadores utilizó diferentes técnicas para la selección de atributos, por lo que se ha retomado para este trabajo algunas de las consideraciones contempladas en los artículos mencionados en el Capítulo III y IV, proponiendo mediante la observación de las características de los paquetes de ataque DDoS los siguientes atributos:

- Protocol_type, el cual especifica el tipo de protocolo utilizado
- Src_bytes, es el número de bytes de datos del origen al destino
- Dst_host_same_srv_rate, siendo el porcentaje de conexiones al mismo servicio
- Count, es el número de conexiones a la misma máquina que la conexión actual en los últimos 2 segundos
- Diff_srv_rate, es el porcentaje de conexiones a diferentes servicios entre personas del mismo huésped
- Dst_host_serror_rate, porcentaje de conexiones que tiene error SYN
- Dst_host_diff_srv_rate, es el porcentaje de conexiones a diferentes servicios del mismo host
- Wrong_fragment, es el número de fragmentos erróneos de las conexiones TCP individuales

4. 1.3 Preprocesamiento de datos

Este módulo consta de dos secciones, en la primera se realizará la extracción de los atributos y en la segunda se realizará la normalización de los tipos de datos de cada atributo.
Sección I. Extracción de atributos.

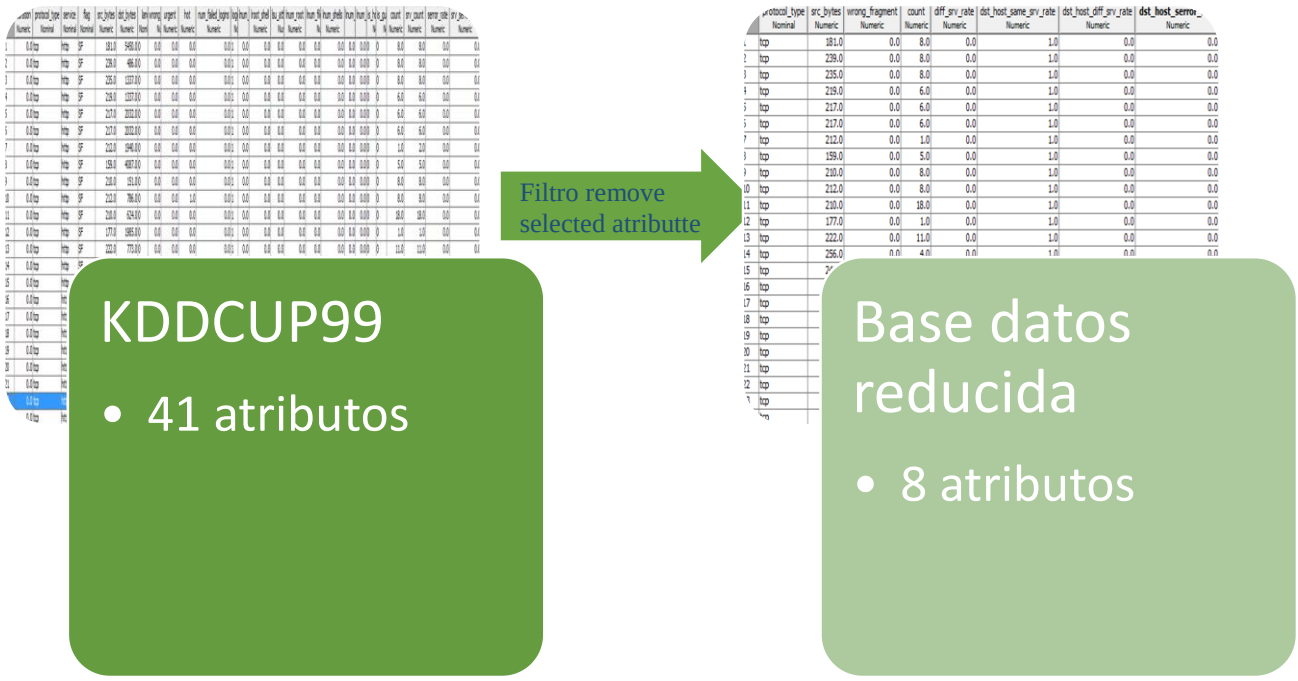


Figura 11. Esquema representativo de la extracción de atributos usando filtro removed selected atributte.

De un total de 41 atributos incluidos en la base de datos KDDCUP99, se extraerán los siguientes atributos Protocol_type, Src_bytes, Dst_host_same_srv_rate, Count, Diff_srv_rate, Dst_host_serror_rate, Dst_host_diff_srv_rate y Wrong_fragment utilizando el filtro Remove selected atributte; cada uno de ellos con diferente tipo de dato como se muestra en la tabla 4.

Tabla 4. Descripción de los atributos seleccionados en el preprocesamiento.

Nombre de atributos	Tipo de dato	Características
Protocol_type	Nominal	Etiquetas: TCP, UDP y ICMP
Src_bytes	Numérico	Mínimo: 0, Maximo:693375640, Media: 3036.207 y DevEst: 990084.697
Dst_host_same_srv_rate	Numérico	Mínimo: 0, Maximo:1, Media: 0.753y DevEst: 0.411

Count	Numérico	Mínimo: 0, Maximo:511, Media: 333.349 y DevEst: 212.773
Diff_srv_rate	Numérico	Mínimo: 0, Maximo:1, Media: 0.021 y DevEst: 0.082
Dst_host_serror_rate	Numérico	Mínimo: 0, Maximo:1, Media: 0.177 y DevEst: 0.381
Dst_host_diff_srv_rate	Numérico	Mínimo: 0, Maximo:1, Media: 0.031 y DevEst: 0.109
Wrong_fragment	Numérico	Mínimo: 0, Maximo:3, Media: 0.006 y DevEst: 0.135

Sección II. Normalización de atributos.

En esta sección se aplican algunas técnicas de normalización estadística a cada uno de los atributos, con la finalidad de que los datos puedan ser utilizados para la estructuración de los grafos semánticos. Se comenzará con el análisis de cada uno de los atributos, lo cual nos permitirá que cada atributo se convierta en nominal para facilitar la estructuración del grafo.

Protocol_type.

Es un atributo de tipo nominal, donde se contemplan los protocolos UDP, TCP e ICMP, teniendo un total de 494020 instancias de las cuales 190064 son del protocolo TCP, 20354 son del protocolo UDP y 283602 son del protocolo ICMP.

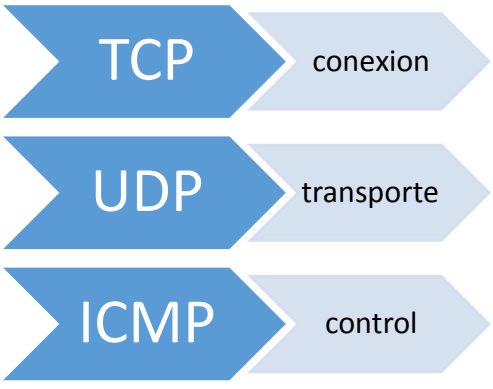


Figura 12. Esquema de normalización para atributo Protocol_type

Para este caso la normalización aplicada está relacionada con la función de cada protocolo en la red (Figura 12), es decir, el protocolo TCP es un protocolo orientado a la conexión

entre maquinas; el protocolo UDP es un protocolo de transporte sin conexión y el ICMP es un protocolo de control de mensajes de Internet.

Src_bytes.

Es un atributo de tipo numérico, el cual se pre- procesará con la finalidad de convertirlo en un atributo nominal; para su normalización se aplicaron algunas técnicas las cuales se describen a continuación:

Como primer paso se graficaron los datos para conocer las frecuencias del número de bytes de datos, como se muestra en la Figura 13.

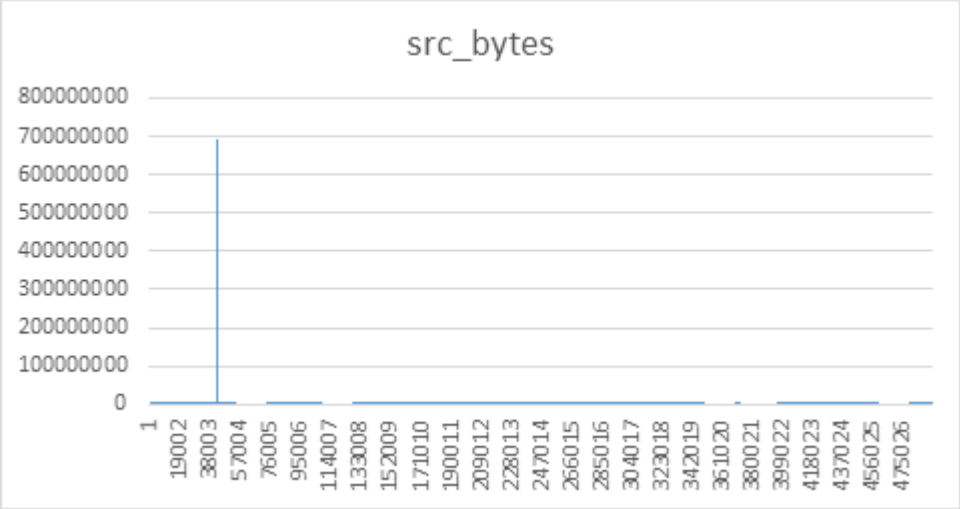


Figura 13. Gráfica de datos del tráfico del número de bytes de origen al destino

Para normalizar se aplicó estadística descriptiva a los datos, utilizando las siguientes formulas:

Para el cálculo de la media aritmética:

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{N}$$ Obteniendo como resultado 3036.207.

Para el cálculo de la Desviación Estándar.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (X_i - \bar{X})^2}{N}}$$

Obteniendo como resultado 990084.697, lo cual nos indica que los datos se encuentran dispersos y difíciles de agrupar en clases como se haría aplicando estadística descriptiva, ya

que se puede observar en la gráfica que existe un punto con un valor de 69337560 bytes, que propicia a que la desviación estándar sea muy grande.

Para eliminar este problema el área de minería de datos maneja algunos consejos a ser utilizados al momento del pre- procesamiento de los datos, entre los cuales se tiene la recuperación de información incompleta, la eliminación de registros por anomalía y la eliminación de outliers que afecten en el preprocesamiento; por lo cual se eliminará el valor atípico observado la Figura 14.

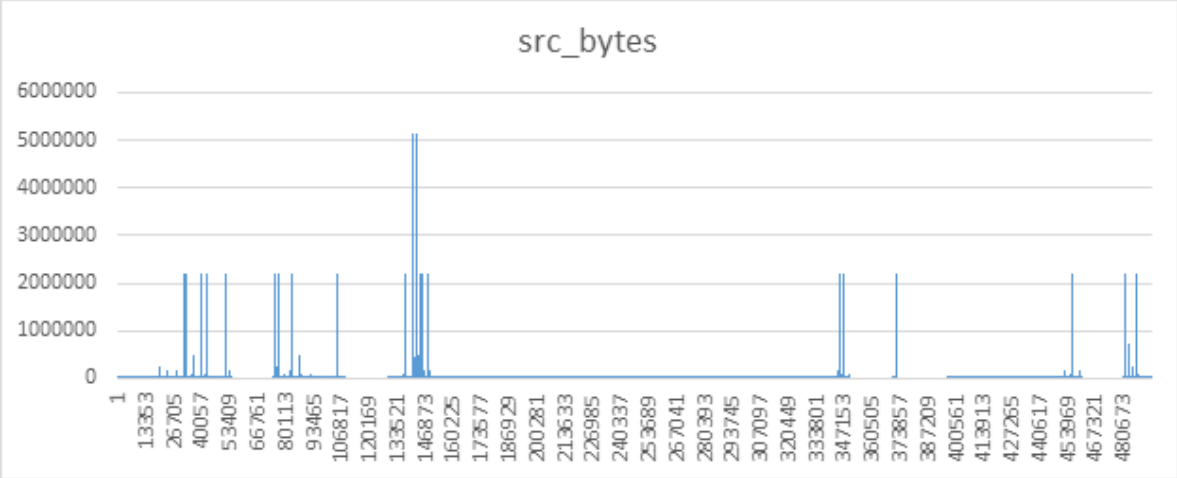


Figura 14. Gráfica de datos del tráfico del número de bytes de origen al destino con eliminación de outlier.

Después de eliminar el outlier, se recalculo la media con un valor de 1622.07 y la desviación estándar con un valor de 58333.27, por lo que se puede observar que los datos se encuentran aún muy dispersos. En conclusión se puede aseverar que no hay un modelo que determine el comportamiento del flujo del tráfico por Internet, que se ajuste a un modelo estadístico y estocástico, ya que los datos no se ajustan a una distribución conocida, es por ello que se propone normalizar los datos a partir de conjuntos difusos [46] para manejar la incertidumbre de la información que se presenta.

A continuación se muestra los conjuntos difusos para la normalización del atributo src_bytes (Fig 15).

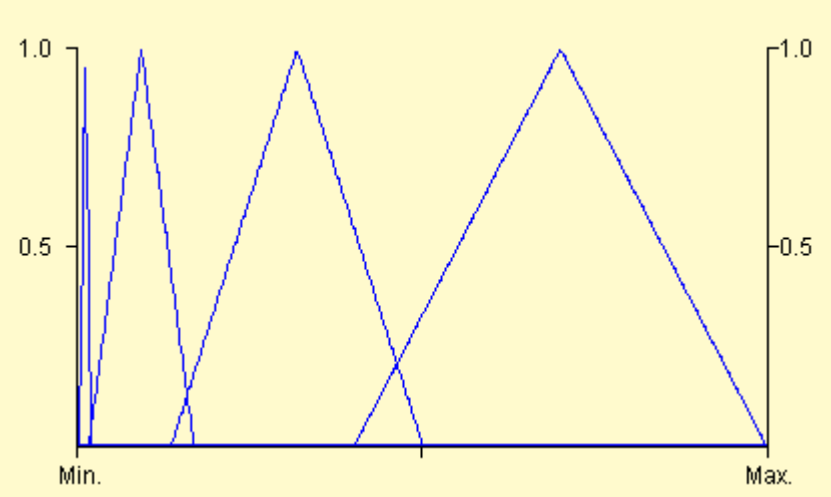


Figura 15. Conjuntos difusos para la normalización de src_bytes.

Dentro de la figura 15, el universo del discurso consta de un rango de valores de U= [0, 6000000] bytes, 4 conjuntos difusos con un valor lingüístico de bajo [0,100000], medioBajo[80000,1000000], medioAlto [800000, 3000000] y Alto [2400000,6000000], obtenidos debido a la observación del tráfico de la red mostrado en la Figura 14.

Count.

Es un atributo de tipo numérico, el cual se pre- procesará con la finalidad de poder convertido en un atributo nominal, para su normalización se utilizaron conjuntos difusos debido a que los datos se encuentran dispersos como se muestra en la Fig. 16.

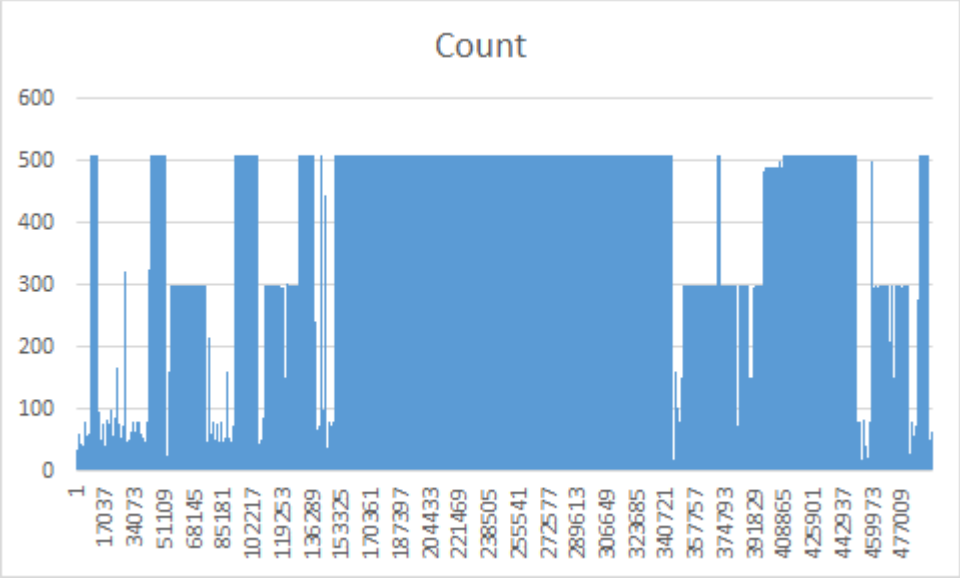


Figura 16. Gráfica de datos del tráfico del número de conexiones a la misma máquina que la conexión actual en los últimos 2 segundos.

Dentro de la figura 16, el universo del discurso consta de un rango de valores de U= [0, 600] bytes, 4 conjuntos difusos con un valor lingüístico de bajo [0,150], medioBajo[120,250], medioAlto [200, 450] y Alto [360,600], obtenidos debido a la observación del tráfico de la red mostrado en la Figura 16.

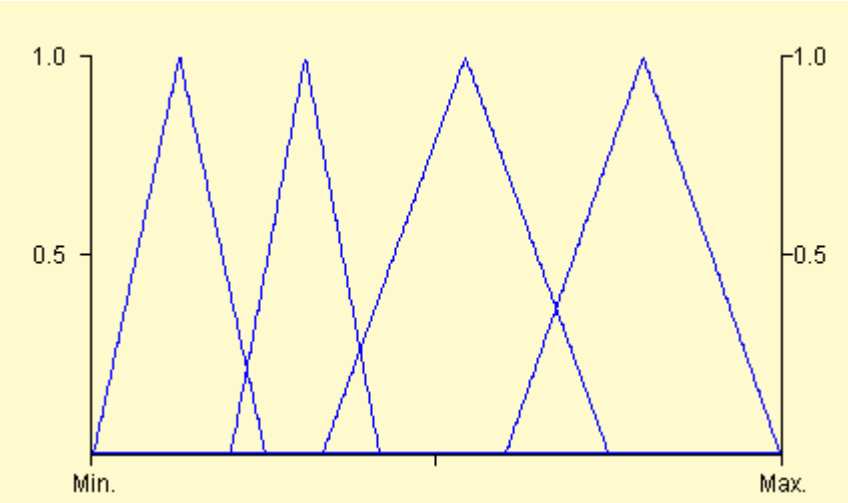


Figura 17. Conjunto difuso para normalización del atributo Count.

Wrong_fragment.

Es un atributo de tipo numérico, en el cual se especifica el número de fragmentos erróneos en una conexión TCP; para normalizar se aplicó estadística descriptiva a los datos, utilizando las siguientes formulas:

Para el cálculo de la media aritmética:

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{N}$$ Obteniendo como resultado 0.006.

Para el cálculo de la Desviación Estándar.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (X_i - \bar{X})^2}{N}}$$ Obteniendo un resultado de 0.135

Lo que indica que la dispersión de los datos es mínima (Ver Fig. 17) en comparación con la dispersión de los atributos count y src_bytes; por tal motivo no se aplicará normalización con conjuntos difusos, ya que para efectos de la estructuración del grafo semántico solo se requiere saber si dentro del tráfico de red, el paquete enviado genera un error en uno de los fragmentos enviados o fue enviado con éxito. Por lo que la normalización constará de dos conjuntos [ConError, SinError].

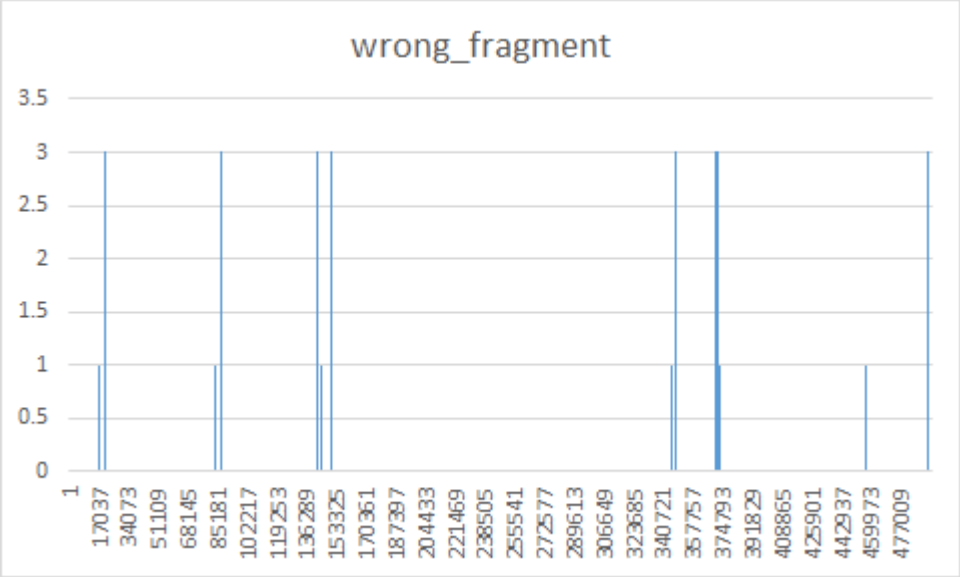
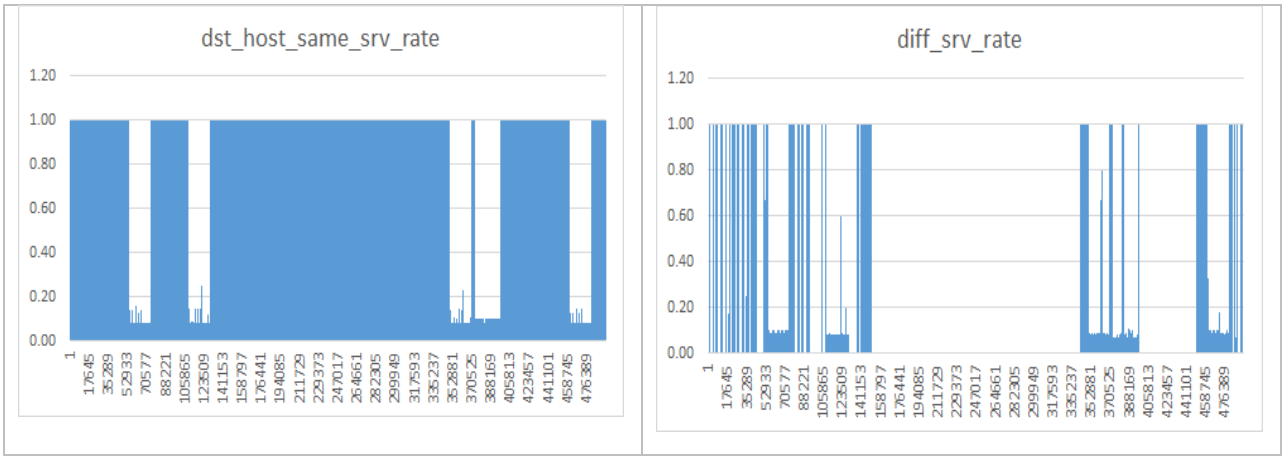


Figura 18. Gráfica de dispersión de los datos en el atributo wrong_fragment.

Dst_host_same_srv_rate, Diff_srv_rate, Dst_host_serror_rate, Dst_host_diff_srv_rate.

En el caso de las variables que manejan porcentajes de conexiones como los son Dst_host_same_srv_rate, Diff_srv_rate, Dst_host_serror_rate, Dst_host_diff_srv_rate; la normalización se realizó con conjuntos difusos debido a la dispersión que tienen los datos, cuyos valores lingüísticos utilizados son [SinConexiones, PocasConexiones, MuchasConexiones], determinadas por los rangos observados en la Fig. 17.



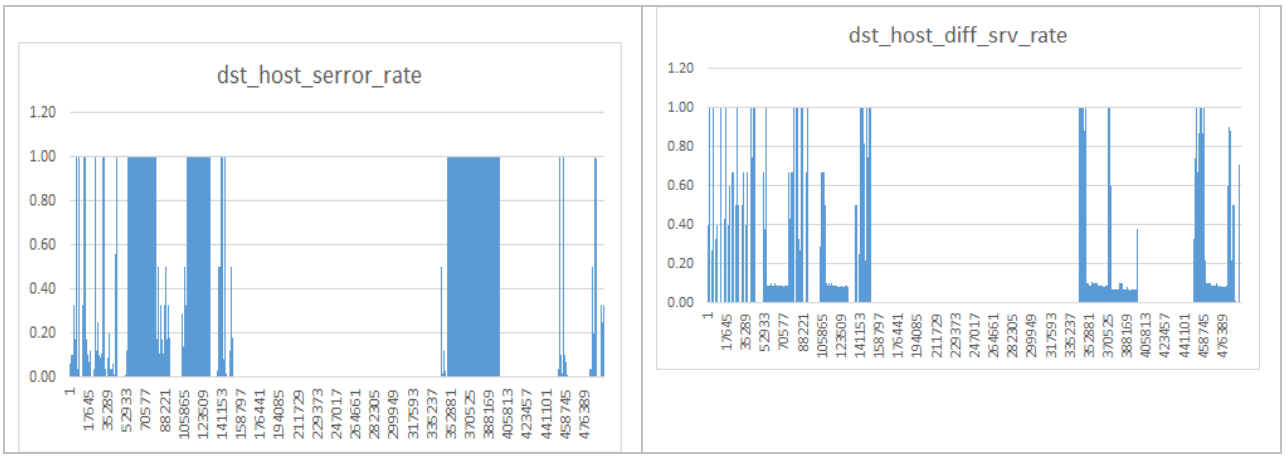


Figura 19. Dispersión de los datos en el atributo dst_host_same_srv_rate, diff_srv_rate, dst_host_serror_rate, dst_host_diff_srv_rate.

Los conjuntos difusos propuestos fueron modelados por una función singleton ya que existen valores nulos y 3 funciones triangulares cuyos valores lingüísticos son nulo [0], poco [0.01, 0.20], medio [0.16, 0.60] y alto [0.48, 1] como se aprecia en la Fig.18.

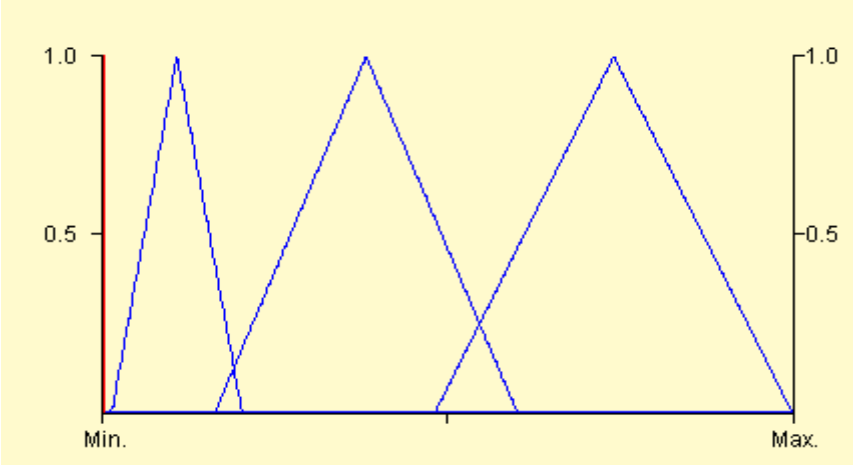


Figura 20. Conjuntos difusos propuestos para la normalización de los atributos dst_host_same_srv_rate, diff_srv_rate, dst_host_serror_rate, dst_host_diff_srv_rate.

Al terminar estas dos secciones los datos normalizados quedarán de la siguiente manera:

Tabla 5. Base de datos KDD CUP99 normalizada.

protocol_type	src_bytes	wrong_fragment	count	diff_srv_rate	dst_host_same_srv_rate	dst_host_diff_srv_rate	dst_host_serror_rate
conexión	BAJO	SINERROR	bajo	bajo	alto	bajo	bajo
conexión	BAJO	SINERROR	bajo	bajo	alto	bajo	bajo
conexión	BAJO	SINERROR	bajo	bajo	alto	bajo	bajo
conexión	BAJO	SINERROR	bajo	bajo	alto	bajo	bajo
conexión	BAJO	SINERROR	bajo	bajo	alto	bajo	bajo

4. 1.3 Algoritmo de Esperanza múltiple.

Se utilizará el algoritmo EM (Esperanza Múltiple) para la identificación de los k-grupos de datos que tengan alguna relación entre sí, lo cual ayudará a que el algoritmo KNN tenga como entrada la K para los agrupamientos y la obtención de los patrones de la red a estudiar, partiendo del hecho que el problema que se tiene para el estudio de los datos del tráfico de la red es que se desconoce el comportamiento de cada uno de los datos y en que distribución se encuentra, por lo que se optó por el uso del algoritmo EM [44], el cual realiza iteraciones para calcular las probabilidades de distribución de cada uno de los datos con el fin de determinar a qué clúster va a pertenecer. La información que genera busca la relación y similitudes entre cada uno de los datos. Las estimaciones de los parámetros se siguen iterando usando las probabilidades para re-estimar los parámetros hasta converger dando como resultado que cada uno de los elementos pertenezca a una clase donde estos tengan alguna característica en común.

El cálculo de las probabilidades de las clases o los valores esperados de las clases es la parte de expectation. El paso de calcular los valores de los parámetros de las distribuciones, es maximization, maximizar la verosimilitud de las distribuciones dados los datos.

Para estimar los parámetros, tenemos que considerar que tenemos únicamente las probabilidades de pertenecer a cada cluster y no los clusters en sí. Estas probabilidades actúan como pesos:

$$\mu_A = \frac{w_1 x_1 + w_2 x_2 + \ldots w_n x_n}{w_1 + w_2 + \ldots w_n}$$

$$\sigma_A^2 = \frac{w_1 (x_1 - \mu)^2 + w_2 (x_2 - \mu)^2 + \ldots w_n (x_n - \mu)^2}{w_1 + w_2 + \ldots w_n}$$

Donde $\frac{w_i}{}$ es la probabilidad de que el objeto i pertenezca al cluster A y se suma sobre todos los objetos (no solo los de A).

El algoritmo tiende a converger pero nunca llega a un punto fijo.

Podemos ver que tanto se acerca calculando la versorimilitud general de los datos con esos parámetros, multiplicando las probabilidades de los objetos individuales (i):

$$\prod_i (P_A P(x_i|A) + P_B P(x_i|B))$$

Esta medida crece en cada iteración, y se itera hasta que el crecimiento es despreciable.

4. 1.4 Algoritmo KNN

Se ha determinado utilizar el algoritmo KNN (K nearest neighbors o K vecinos más cercanos) debido a que tiene la ventaja de ser de fácil interpretación, tolerante a ruido en los datos y un bajo coste en de cálculo. El KNN [45] requiere el valor inicial de k, donde la elección de k dependerá fundamentalmente del comportamiento de los datos; generalmente mientras más grande sea k reduce el efecto del ruido de la clasificación, pero crean límites entre las clases que son parecidas. Dado que no se puede determinar la distribución total de la información del tráfico de red se optó por obtener el número de k que será utilizado por KNN, a partir de los cálculos de EM; una vez encontrado el número de elementos asociados y de haber realizado la discriminación de clúster formado a partir del algoritmo EM, el resultado del número de clúster será utilizado como la k de entrada del algoritmo KNN, el cual funciona de la siguiente manera.

Los vecinos más cercanos a una instancia se obtienen, en caso de atributos continuos, utilizando la distancia Euclideana sobre los n posibles atributos.

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2}$$

El resultado de la clasificación de k-NN puede ser discreto o continuo.

En el caso discreto, el resultado de la clasificación es la clase más común de los k-vecinos, como se muestra a continuación.

Entrenamiento:

almacena todos los ejemplos de entrenamiento $(x, f(x))$

Clasificación:

Dada una instancia x_q:

Sean $x_1, \ldots, x_k$ los k vecinos más cercanos a x_q.

Entonces:

$$f(x_q) = argmax_{v \in V} \sum_{i=1}^{k} \delta(v, f(x_i))$$

donde: $\delta(a, b) = 1$ si $a = b$ y 0 en caso contrario.

La forma que se genera con $k = 1$ es un diagrama de Voronoi alrededor de las instancias almacenadas. A una nueva instancia se le asigna la clasificación del vecino más cercano.

Para clasificaciones continuas, se puede tomar la media de las clasificaciones.

$$f(x_q) = \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

Un extensión obvia al algoritmo es pesar las clasificaciones de los vecinos de acuerdo a su distancia con el objeto a clasificar (la clasificación de vecinos más cercanos tienen más peso).

Para clases discretas:

$$f(x_q) = argmax_{v \in V} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

Dónde: $w_i = \frac{1}{d(x_q, x_i)^2}$ (si la distancia es 0 entonces $w = 0$).

Para clase continuas:

$$f(x_q) = \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

Una suposición es que los vecinos más cercanos nos dan la mejor clasificación y esto se hace utilizando todos los atributos.

4. 1.4 Estructuración del grafo.

El manejo actual de grandes cantidades de datos, que aun las bases de datos convencionales no pueden manejar, ha dado paso a las representaciones basadas en grafos de datos. El uso de vértices para representar entidades tales como personas, lugares y cosas, y bordes para representar las relaciones entre esas entidades, como amigo, vive en y es propietaria de, permite que la expresión de estas asociaciones sea más rica y fácil de comprender, que una representación tabular de la información.

Las aplicaciones de grafos de datos han ido desde conjuntos de datos como registros de llamadas de telecomunicaciones, información financiera, hasta la formación de grafos de las redes sociales con la que se han descubierto las propiedades estructurales de los datos que no son evidentes utilizando métodos tradicionales de minería de datos. Entre los problemas presentes que se atacan con el uso de grafos es el problema de la escalabilidad relacionada con el manejo de grandes cantidades de datos, cuando se trata de aprender patrones e identificar anomalías en los datos.

Por ejemplo, en el caso de tráfico de red, una representación gráfica del tráfico podría consistir en vértices que representan las computadoras, y las aristas representan las comunicaciones entre los equipos correspondientes. Además, otras fuentes potenciales de datos para ayudar en el análisis del tráfico de red podrían incluir detalles sobre los usuarios, como su ubicación y el registro de sus consultas diarias. La incorporación de estos conjuntos de datos heterogéneos para el tráfico de red, representada como un grafo, podría servir de base para el descubrimiento de patrones estructurales interesantes y anomalías, que pueden alertar a un analista de seguridad de la amenaza potencial en un intento de irrumpir en la red.

Es por ello que surge la necesidad de modelos y sistemas de detección de ataques informáticos robustos, que puedan ser utilizados en redes de tráfico denso. Aunque existe investigación realizada en el área [29-49], poco de ello se ha centrado en el análisis de datos del tráfico de red para la detección de ataques basados en grafos [50-54].

Sin embargo, la gran mayoría de los enfoques basado en grafos han abordado el tema utilizando concepto como la identificación de agrupaciones inusuales en subgrafos, mediante el análisis estadístico de la existencia de asociaciones entre los datos que determine una anomalía en el grafo. En este caso el uso de la teoría de grafos será utilizado para la estructuración de subestructuras que representen un ataque potencial por Flooding. Tales subestructuras serán generadas a partir de las agrupaciones obtenidas por el algoritmo KNN, con las cuales se pretende se logren identificar la forma de trabajo de los ataques por inundación [17].

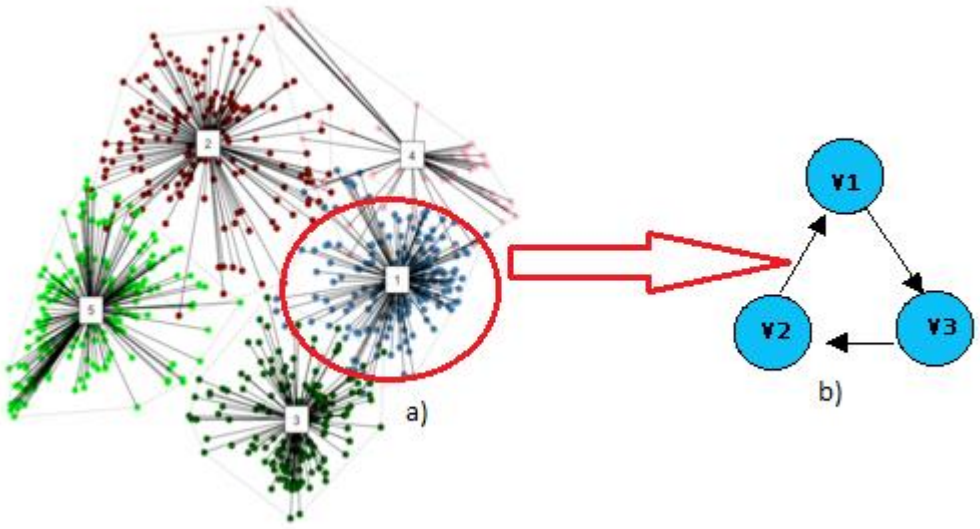


Figura 21. A) Agrupamientos generados por algoritmo KNN. B) Generación de grafo a partir del comportamiento de los datos en casa cluster.

El diseño de los grafos se estructurará tomando en cuenta el comportamiento del ataque por flooding [27,28] y analizando la conducta de los datos que forman parte de cada uno de los cluster que se han generado por KNN.

Para la estructuración de las subestructuras de grafos semánticos, se utilizará la base de datos normalizada, para poder asignar el peso de los arcos y los vértices. En donde los vértices

serán los atributos Protocol_type, Src_bytes, Dst_host_same_srv_rate, Count, Diff_srv_rate, Dst_host_serror_rate, Dst_host_diff_srv_rate y Wrong_fragment; y sus relaciones serán obtenidas de la normalización de sus registros (Ver Fig. 22), generando el grafo en un formato aceptable para Subdue [60- 62].

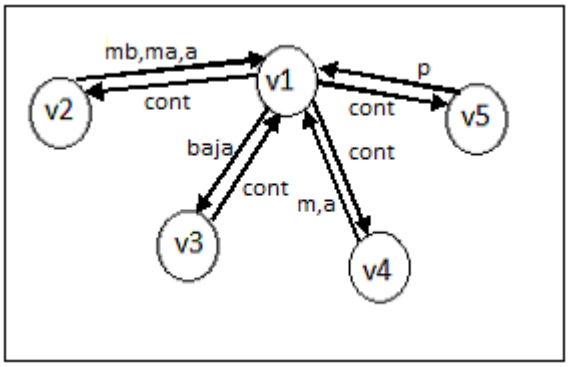


Figura 22. Ejemplo de grafo de ataque Smurf.

Se ha elegido utilizar la herramienta Subdue, ya que es un sistema de aprendizaje relacional utilizado para encontrar subestructuras (subgrafos) que aparecen repetidamente en la representación basada en grafos de bases de datos. Una vez que la base de datos está representada con grafos, Subdue busca la subestructura que mejor comprime al grafo utilizando el principio MDL. Después de encontrar esta subestructura, Subdue comprime el grafo y puede iterar repitiendo este proceso. Subdue tiene la capacidad de realizar un macheo inexacto que permite descubrir subestructuras con pequeñas variaciones.

Otra característica importante que posee Subdue es que permite utilizar conocimiento previo representado como subestructuras predefinidas, concepto del cual se aprovechará en este apartado, ya que los grafos generados como se muestra en la Fig. 22 serán los que se busquen dentro de la base de datos con la finalidad de corroborar si esas estructuras representan un ataque por flooding.

Capítulo V. Evaluación y resultados

En este capítulo se presentan las evaluaciones y resultados de los algoritmos utilizados en la metodología propuesta para la detección del ataque por Flooding utilizando grafos. El primero de ellos fue el algoritmo EM, donde se utiliza validación cruzada de k iteraciones. El segundo algoritmo que se presenta es el algoritmo KNN, donde se realizó una selección de instancias reduciéndolas a la mitad de forma aleatoria debido a que el coste computacional se elevaba si se procesaban las 494020 instancias de la base de datos. Finalmente con el resultado del algoritmo KNN, se construyeron los grafos semánticos y se probaron las subestructuras utilizando Subdue.

5. 1 Algoritmo de Esperanza Múltiple (EM).

Para determinar el número de clúster de mejor ajuste para K en el algoritmo KNN, se utilizó EM con validación cruzada de k iteraciones (k- folds cross- validation), donde uno de los subconjuntos se utilizó como datos de prueba y el resto k-1 como datos de entrenamiento. El proceso de validación cruzada es repetido durante k iteraciones, con cada uno de los posibles subconjuntos de datos de prueba. Finalmente se realiza la media aritmética de los resultados de cada iteración para obtener un único resultado. Este método es preciso puesto que evaluamos a partir de K combinaciones de datos de entrenamiento y de prueba, aunque tiene una desventaja, es lento desde el punto de vista computacional. En la práctica, la elección del número de iteraciones depende de la medida del conjunto de datos. Lo más común es utilizar la validación cruzada de 10 iteraciones (10-fold cross-validation) según [47].

En la tabla 6, se muestra los resultados obtenido por el algoritmo EM, utilizando como máximo de iteraciones 100, numFolds 10 y numKmeanRuns 10.

Tabla 6. Resultados del algoritmo EM usando 10-folds

Instances: 494020

Number of clusters selected by cross validation: 9
Number of iterations performed: 1

Attribute	Cluster 0 (0.01)	1 (0.02)	2 (0.03)	3 (0.03)	4 (0.57)	5 (0.18)	6 (0.12)	7 (0.05)	8 (0.01)
protocol_type									
tcp	485.1004	478.4001	13361.001	61.4457	1	86548.5232	60537.2209	22881.4634	5718.8455
udp	238.367	7567.7154	51.3112	12473.877	14.8215	2.0692	5.6035	8.2353	1
icmp	3189.5957	6.9297	5.2441	58.8588	280299.4597	36.9058	4.0437	8.9624	1
[total]	3913.0631	8053.0452	13417.5563	12594.1815	280315.2812	86587.4982	60546.8681	22898.6611	5720.8455
src_bytes									
mean	288.7788	109.9507	24249.492	80.3692	935.6483	8879.0376	2229.3783	4.8345	42.9732
std. dev.	462.9472	57.8514	321291.4927	32.6394	200.1213	2357073.9216	10068	91.5415	853.3322
wrong_fragment									
mean	0.0662	0.3626	0	0	0	0	0	0	0
std. dev.	0.2486	0.9768	0.0042	0	0.1348	0.1348	0.1348	0	0.1348
count									
mean	34.6411	10.0307	2.9066	14.3946	507.469	190.034	9.0657	194.7267	1.507
std. dev.	94.7031	26.0273	4.6732	39.6271	13.11	69.7284	8.662	106.7844	2.4278
diff_srv_rate									
mean	0.1492	0.1267	0.04	0.004	0	0.0612	0	0.1257	0
std. dev.	0.3179	0.2709	0.1643	0.0368	0.0822	0.0148	0.0004	0.2335	0.0822
dst_host_same_srv_rate									
mean	0.7246	0.0591	0.4801	0.9287	1	0.0434	0.9965	0.0369	0.9963
std. dev.	0.3946	0.1245	0.2437	0.0874	0.0006	0.0251	0.0179	0.0259	0.026

dst_host_diff_srv_rate

mean	0.0208	0.5775	0.0762	0.0107	0	0.0652	0.0013	0.1597	0.0006
std. dev.	0.0791	0.3162	0.108	0.0102	0.0001	0.0111	0.0069	0.2686	0.0047

dst_host_rerror_rate

mean	0.0009	0.0463	0.0082	0.001	0	0	0.0031	0.99	0.9369
std. dev.	0.0131	0.1555	0.0513	0.0097	0.0001	0.0004	0.0157	0.0458	0.1685

=== Model and evaluation on training set ===

Clustered Instances

0	3729 (1%)
1	8005 (2%)
2	16227 (3%)
3	15002 (3%)
4	280267 (57%)
5	84960 (17%)
6	57124 (12%)
7	22861 (5%)
8	5845 (1%)

Log likelihood: 6.86106

5. 2 Algoritmo de KNN

Para obtener los cluster que se van a estudiar en la estructuración de los grafos se utiliza el algoritmo KNN, considerando como entrada de K el resultado obtenido en EM que es k=9. La métrica utilizada para medir la proximidad, será la distancia euclideana por razones computacionales; ya que el cálculo de la distancia Euclídea tiene un coste lineal respecto a d (coste global: *O(Nd))* mientras que el coste de una distancia de Mahalanobis es cuadrático respecto a d (coste global: *O(Nd2)).*

Para reducir el coste computacional de KNN, existen diferentes estrategias que permiten la aplicación de la regla del vecino más cercano para conjuntos de referencia númerosos el cual es el caso de la base de datos KDDCUP99, ya que contamos con casi 500,000. Estas estrategias pueden agruparse en dos clases según [48], la primera es reducir el conjunto de referencia y la segunda es mejorar la eficiencia computacional en la búsqueda del vecino

más cercano; de la cual se recurre a la reducción del conjunto de referencia que consiste en la selección de un subconjunto del conjunto de referencia que tenga las mismas propiedades que el conjunto original para, una vez reducido, aplicar la regla del vecino más cercano en su formulación original sobre el nuevo conjunto.

Con lo que si M es el número de prototipos del conjunto reducido, el orden de complejidad sigue siendo lineal respecto a M, con la salvedad de que ahora M < N. No obstante, cabe mencionar que la selección de un conjunto de referencia reducido que refleje todas las características del conjunto general no está del todo garantizado.

Para la reducción de la población se utilizó una muestra a un nivel de confianza del 99%, con un error del 5%, utilizando la siguiente formula:

$$n = \frac{k^2 * p * q * N}{(e^2 * (N-1)) + k^2 * p * q}$$

Obteniendo una muestra de 665 elegidos de manera aleatoria de un total de 494020 lo que representa un 13% del total de la población y un error del 5%.

En la tabla 7, se puede apreciar los resultados de la ejecución del algoritmo KNN, con k=9, nearestNeighbourSearchAlgorithm= LinearNNSearch, y crossValidate=10 folds; procesando el 13% de los datos originales que fueron seleccionados de manera aleatoria.

Tabla 7. Resultados de ejecución del algoritmo KNN con una muestra de 13% de los datos, con un nivel de confianza del 99%.

=== Stratified cross-validation ===			
Correctly Classified Instances	649	97.741	%
Incorrectly Classified Instances	15	2.259	%
Kappa statistic	0.955		
K&B Relative Info Score	61838.5486 %		
K&B Information Score	718.3646 bits	1.0819 bits/instance	
Class complexity | order 0	768.7737 bits	1.1578 bits/instance	
Class complexity | scheme	99.1772 bits	0.1494 bits/instance	
Complexity improvement (Sf)	669.5965 bits	1.0084 bits/instance	
Mean absolute error	0.0192		
Root mean squared error	0.1107		
Relative absolute error	5.6806 %		
Root relative squared error	26.9824 %		

```
Total Number of Instances              665
=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0.998    0.004    0.998      0.998    0.998      0.994    0.999     0.999     icmp
                 0.983    0.025    0.954      0.983    0.968      0.951    0.993     0.979     tcp
                 0.643    0.005    0.857      0.643    0.735      0.733    0.966     0.785     udp
Weighted Avg.    0.977    0.011    0.976      0.977    0.976      0.968    0.996     0.983
=== Confusion Matrix ===

   a   b   c   <-- classified as
 404   1   0 |   a = icmp
   1 227   3 |   b = tcp
   0  10  18 |   c = udp
=== Predictions on test data ===

inst#,actual,predicted,error,prediction
1,1:icmp,1:icmp,,1
2,1:icmp,1:icmp,,1
3,1:icmp,1:icmp,,1
4,1:icmp,1:icmp,,1
5,1:icmp,1:icmp,,1
6,1:icmp,1:icmp,,1
7,1:icmp,1:icmp,,1
8,1:icmp,1:icmp,,1
9,1:icmp,1:icmp,,1
10,1:icmp,1:icmp,,0.999
11,1:icmp,1:icmp,,1
12,1:icmp,1:icmp,,1
13,1:icmp,1:icmp,,1
14,1:icmp,1:icmp,,1
15,1:icmp,1:icmp,,1
16,1:icmp,1:icmp,,1
17,1:icmp,1:icmp,,1
18,1:icmp,1:icmp,,1
19,1:icmp,1:icmp,,1
20,1:icmp,1:icmp,,1…..
```

Con lo cual se analizaron las predicciones de los datos y su comportamiento dentro de cada cluster, para observar las relaciones que existían entre ellos y poder formular los grafos semánticos, utilizando como base los cluster formados.

Como podemos observar en la Fig. 23, la relación que tienen el atributo count con {dst_host_diff_srv, dst_host_serror, dst_host_same_s, diff_serror_rate} se encuentra más remarcada por lo que para la formulación del grafo semántico se partirá del estas relaciones.

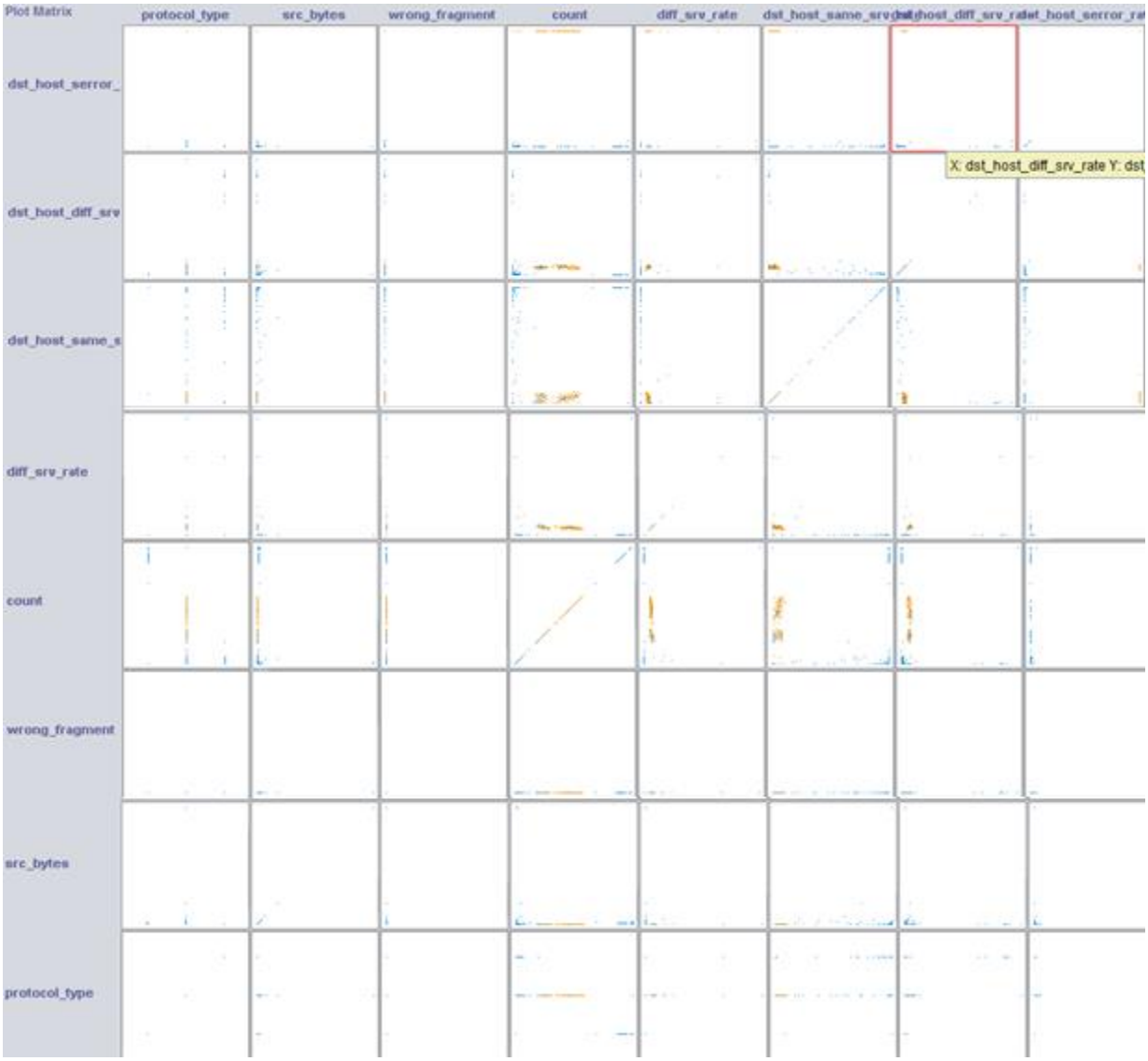
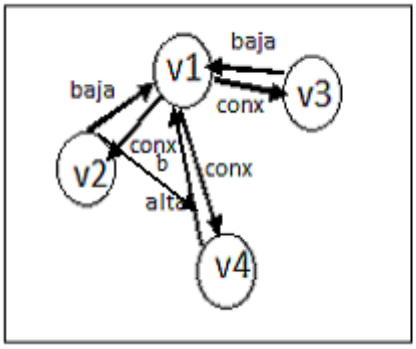


Figura 23. Visualización de resultados de la aplicación del algoritmo KNN con K=9

5. 3 Estructuración de Grafo semántico.

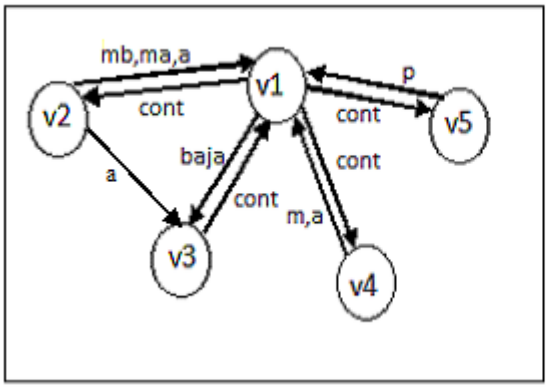
Una vez obtenidos los cluster, se analizó la pertenencia de los datos a cada uno de los cluster para conocer el comportamiento de cada uno de ellos y así comenzar la estructuración de los grafos semánticos, para ello se utilizaron las relación y las similaridades que se observaron entre los datos de cada atributo, y tomando en cuenta el comportamiento del ataque por flooding.

Los grafos obtenidos se muestran a continuación:



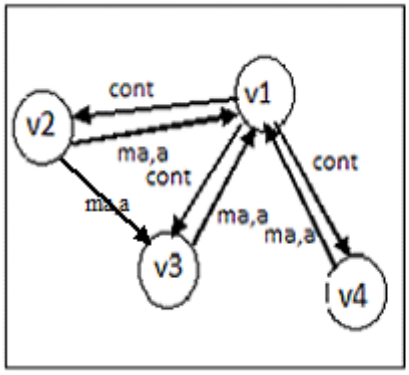
Donde v1: protocol, v2: src_bytes, v3: count y v4: dst_host_same_srv_rate

Figura 24. Grafo 1, Ataque Multihop



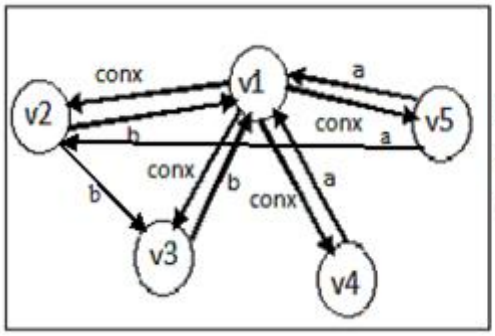
Donde v1: protocol, v2: count, v3: src_bytes, v4: dst_host_same_srv_rate y v5: dst_host_diff_srv_rate.

Figura 25. Grafo 2, Ataque Smurf



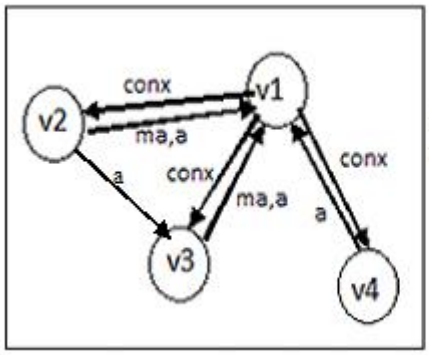
Donde v1: protocol, v2: count, v3: src_bytes y v4: dst_host_same_srv_rate.

Figura 26. Grafo 3, Ataque ICMP Flooding



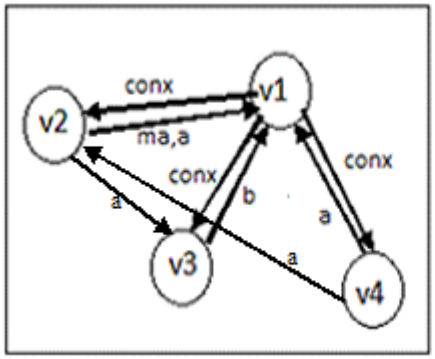
Donde v1: protocol, v2: count, v3: src_bytes, v4: dst_host_same_srv_rate y v5: diff_srv_rate

Figura 27. Grafo 4, Ataque LoadModule



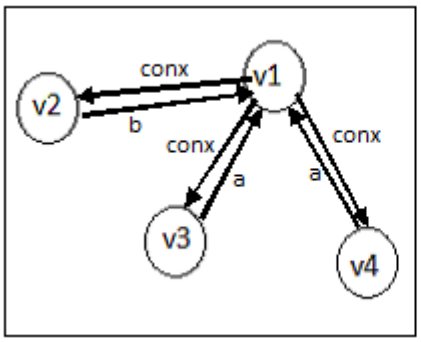
Donde v1: protocol, v2: count, v3: src_bytes y v4: dst_host_serror_rate

Figura 28. Grafo 5, Ataque SYN flooding.



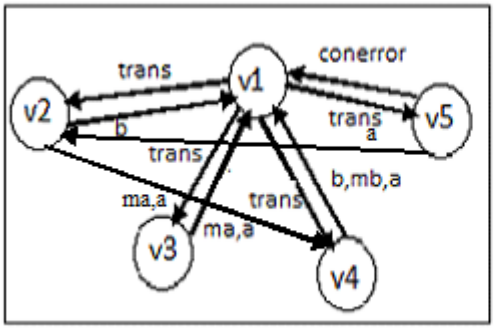
Donde v1: protocol, v2: count, v3: src_bytes y v4: dst_host_serror_rate

Figura 29. Grafo 6, Ataque HTTP Flooding.



Donde v1: protocol, v2: count, v3: dst_host_same_srv_rate y v4: dst_host_serror_rate

Figura 30. Grafo 7, Ataque Land Flooding



Donde v1: protocol, v2: count, v3: src_bytes, v4: dst_host_same_srv_rate y v5: wrong_fragment

Figura 31. Grafo 8, Ataque UDP Flooding

Una vez obtenidos cada uno de los grafos, se realizaron pruebas de búsqueda utilizando la herramienta Subdue, en la cual se utilizó la estructura del grafo como una estructura predeterminada. Los resultados de la búsqueda del grafo se muestran en la tabla 8, donde cabe destacar que para poder realizar la búsqueda, la base de datos de KDDCUP99, tuvo que ser transformada a una base de datos orientada a grafos (BDBG).

Debido a que la capacidad de procesamiento a la cual está restringida Subdue, se realizaron grupos de prueba equivalente al 10% de los datos, lo cuales fueron elegidos entre el tráfico de red considerado como normal y los datos del ataque según corresponde al grafo, obteniendo como resultado los mostrado en la tabla 8.

Tabla 8. Resultados en porcentaje de error de la detección de la subestructura predefinida en la base de datos KDDCUP99

No. Grafo	% de error en la detección de la subestructura
1	12%
2	6%

3	7%
4	9%
5	6%
6	10%
7	8%
8	6%

Capítulo VI. Conclusiones y trabajo futuro

6. 1 Conclusiones.

En este trabajo se presentó un modelo para la detección de ataques de Flooding utilizando grafos, partiendo del conocimiento a priori del comportamiento del flujo de los datos en un ataque por inundación y a su vez del uso de técnicas de minería de datos como son el algoritmo EM y KNN para apoyar a la estructuración correcta de los grafos partiendo del hecho que los datos comparten características similares entre sí, lo cual hace que establecer las relaciones entre los nodos de los grafos sea una tarea más sencilla.

Entre las principales razones que nos llevó a utilizar grafos es la importancia que existe para el manejos de los datos utilizando las relaciones entre entidades para descubrir conocimiento obteniendo patrones y anormalidades, ya que actualmente esta tendencia está siendo aplicada a redes sociales, ayudando a encontrar relaciones y dar sentido al puzzle del manejo de los datos y apoyar a sectores como e-commerce entre otros.

Una aportación que se obtuvo fue la estructuración de grafos representativos de ataques por Flooding, demostrando que el comportamiento del ataque puede ser manejado de manera más sencilla utilizando relaciones entre los datos. Cabe mencionar, que parte de los resultados tiene mucho que ver con la parte del preprocesamiento de los datos y el uso de técnicas de minería para el proceso de análisis inteligente de los datos, ya que si este proceso no se realiza de manera adecuada, los resultados de detección del ataque pueden variar.

Otra de las aportaciones que se vale la pena mencionar, es que comprobó que utilizando 8 variables de un total de 41 que maneja la base de datos de KDDCUP99, se pueden obtener estructuras que nos ayuden a predecir un ataque por Inundación.

Entre las recomendaciones que se realizan si se quiere enfrentar un problema similar, es que la normalización de los datos utilizando estadística descriptiva, es que el tráfico de datos no se ajusta a alguna distribución conocida, ya que se manejan muchos datos cuya fluctuación puede ser o muy grande o muy pequeña para la utilización de técnicas como el uso de media y desviación estándar, por lo que la recomendación es utilizar conjuntos difusos o en su caso

de no necesitar normalizaciones tan especificas se puede llegar a utilizar los filtros pasa alta y pasa baja.

Otra de las recomendaciones es que si se desea utilizar alguna de las técnicas de minería lazy learning, se debe tener presente el costo computacional que este conlleva, ya que para manejar extraer comportamiento o patrones a partir de estos algoritmos utilizando grandes cantidades de datos, se debe recurrir a la reducción del conjunto de referencia buscando que no se pierdan las características principales de los datos o en su defecto optar por alguna otra técnica que reduzca la carga computacional del proceso como por ejemplo utilizar computo paralelo.

6. 2 Trabajo Futuro.

Como trabajo futuro existen aún diversas problemáticas en el área de seguridad informática por explorarse y por resolver en lo que concierne a la detección de ataques informáticos; en nuestra línea futura pretendemos realizarlo en tres lapsos de tiempo, divididos en corto, mediano y largo plazo.

A corto plazo se pretende utilizar las subestructuras generadas en la detección de ataques por Flooding en alguna empresa o institución para corroborar que el método de detección propuesto puede ser utilizado en otros entornos, siguiendo los pasos mencionados en el Capítulo 4.

A mediano plazo se utilizará cómputo paralelo en la implementación de los algoritmos EM y KNN, para mejorar la velocidad de procesamiento de los datos y así obtener resultados en menor tiempo.

A largo plazo se pretende explorar la posibilidad de implementar grafos en la detección de otros ataques informáticos y aumentar el modelo de detección.

Referencias.

1 Siles R. (2002), Análisis de seguridad de la familia de protocolos TCP/IP y sus servicios asociados, Recuperado 28 de octubre del 2015, de http://elpuig.xeill.net/departaments/informatica/fitxers/xarxes/analisis-de-seguridad-de-la-familia-de-protocolos-tcp-ip-y-sus-servicios-asociados/view

2 Mieres J (2009), Ataques informáticos. Debilidades de seguridad comúnmente explotada, Recuperado 28 de octubre del 2015, https://www.evilfingers.com/publications/white_AR/01_Ataques_informaticos.pdf

3 Urueña Centeno F. J. (2015), Ciberataques la mayor amenaza actual, Recuperado 30 de octubre del 2015, http://www.ieee.es/Galerias/fichero/docs_opinion/2015/DIEEEO09-2015_AmenazaCiberataques_Fco.Uruena.pdf

4 Klopfenstein N (2015), Report on Cybersecurity and Critical Infrastructure in the Americas, Recuperado 23 noviembre del 2015, de http://www.oas.org/cyber/documents/OASTrendMicroLAC_ENG.pdf

5 Arun Raj Kumar, P. & Selvakumar, S. (2009). Distributed denial of service (DDoS) threat in collaborative environment a survey on DDoS attack tools and traceback mechanisms. 2009 IEEE International Advance Computing Conference (IACC 2009). India March 2009.

6 R. Meher, S. Ladhe “Review Paper on Flooding Attack in MANET” in Int. Journal of Engineering Research and Applications, ISSN : 2248-9622, Vol. 4, Issue 1(Version 2), January 2014, pp. 39-46

7 Peter Bright (2014, Dec). FBI claimed to be investigating Xbox Live, PlayStation Network DDoS perps, Recuperado el 23 noviembre del 2015, de http://arstechnica.com/security/2014/12/fbi-claimed-to-be-investigating-xbox-live-playstation-network-ddos-perps/

8 C. G. Martin Roesch. (2007) Snort homepage on.Recuperado el 19 noviembre del 2015, de http://www.snort.org/docs/snort_htmanuals/htmanual_280/node1.html.

9 Global IT Security Risks Survey. (2014) Distributed denial of service (DDoS) attacks, Recuperado 23 de noviembre del 2015, de https://media.kaspersky.com/en/B2B-International-2014-Survey-DDoS-Summary-Report.pdf.

10 The Security Division of Netscout Arbor Network (2015) Soluciones contra ataques Ddos, Recuperado 25 de noviembre del 2015, de http://es.arbornetworks.com/ddos/.

11 Salvatore J. Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip K. Chan, (1999), Intrusion detector learning, Recuperador el 24 de noviembre del 2015, de https://kdd.ics.uci.edu/databases/kddcup99/task.html.

12 J. P. Anderson, "Computer security technology planing study," ESD-TR-73-51 Electronics System Division(AFSC), vol. 1, pp. 1–32, october 1972.

13 Britos J.D., (2010), Detección de Intrusiones en redes de datos con captura distribuida y procesamiento estadístico, Recuperado en Octubre del 2015, de

14 A. S. Quist. (2007) Security Classification of Information, Recuperado Octubre del 2015, de http://www.fas.org/sgp/library/quist/index.html.

15 CERT. (2007) CERT Coordination Center Denial of Service Attacks, Recuperado en Noviembre del 2015, de http: //www.cert.org/tech_tips/denial_of_service.html.

16 S. C. Lin, and S. S. Tseng, "Constructing detection knowledge for DDoS intrusion tolerance", Expert Systems with Applications, 2004, Vol. 27, pp. 379–390.

17 Garcia Alfaro J., Ataque contra redes TCP/IP , Recuperado en 25 de Noviembre del 2015, de http://docplayer.es/644900-Ataques-contra-redes-tcp-ip.html.

18 Kumar et al.. (2013). A Survey on Defense Mechanisms countering DDoS Attacks in the Network. International Journal of Advanced Research in Computer and Communication Engineering, 2(7), 2599-2606. In-text citation: (Kumar et al, 2013)

19 Feinstein, L., Schnackenberg, D., Balupari, R., & Kindred, D. (2003, April). Statistical approaches to DDoS attack detection and response. In DARPA Information Survivability Conference and Exposition, 2003. Proceedings (Vol. 1, pp. 303-314). IEEE.

20 No, G., & Ra, I. (2009, September). An efficient and reliable DDoS attack detection using a fast entropy computation method. In Proceedings of the 9th International Symposium on Communications and Information Technology (ISCIT'09) (pp. 1223-1228).

21 Kim, Y., Lau, W. C., Chuah, M. C., & Chao, H. J. (2004, March). PacketScore: Statistics-based overload control against distributed denial-of-service attacks. In INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies (Vol. 4, pp. 2594-2604). IEEE.

22 Jin, S., & Yeung, D. S. (2004, June). A covariance analysis model for DDoS attack detection. In Communications, 2004 IEEE International Conference on(Vol. 4, pp. 1882-1886). IEEE.

23 Mirkoviac, J., Prier, G., and Reiher, P. (2002) Attacking DDoS at the source. Proceedings of the 10th IEEE International Conference on Network Protocols, Paris, France, 12-15 November, pp. 1092–1648. IEEE CS.

24 Wang, J., Phan, R. C. W., Whitley, J. N., and Parish, D. J. (2010) Augmented attack tree modeling of distributed denial of services and tree based attack detection method. Proceedings of the 10th IEEE International Conference on Computer and Information Technology, Bradford, UK, 29 June-1 July, pp. 1009–1014. IEEE CS.

25 Limwiwatkul, L. and Rungsawang, A. (2004) Distributed denial of service detection using TCP/IP header and traffic measurement analysis. Proceedings of the IEEE International Symposium Communications and Information Technology, Sapporo, Japan, 26-29 October, pp. 605–610. IEEE CS.

26 Zhang, G. and Parashar, M. (2006) Cooperative defence against DDoS attacks. Journal of Research and Practice in Information Technology, 38, 1–14. [31] Lu, K., Wu, D., Fan, J., Todorovic, S., and Nucci, A. (2007) Robust and efficient detection of DDoS attacks for large-scale Internet. Computer Networks, 51, 5036– 5056.

27 Lu, K., Wu, D., Fan, J., Todorovic, S., and Nucci, A (2007) Robust and efficient detection of DDoS attacks for large-scale Internet. Computer Networks, 51, 5036– 5056.

28 Dongwon Seo, Heejo Lee and Adrian Perrig, “PFS: Probabilistic Filter Scheduling Against Distributed Denial-of-Service Attacks”, 36th Annual IEEE Conference on Local Computer Networks(LCN 2011), pages 9-17.

29 Lin, D. (2013). Network Intrusion Detection and Mitigation against Denial of Service Attack.

30 Karimazad, R & Faraahi, A. (2011). An Anomaly-Based Method for DDoS Attacks Detection using RBF Neural Networks. International Conference on Network and Electronics Engineering, 11, 44-48. In-text citation: (Karimazad & Faraahi, 2011) & Ahmad , 2011)

31 Zhong, R., & Yue, G. (2010, April). DDoS detection system based on data mining. In Proceedings of the 2nd International Symposium on Networking and Network Security, Jinggangshan, China (pp. 2-4).

32 VeetiL, S. A. N. J. A. I., & Gao, Q. I. G. A. N. G. (2013). A real-time intrusion detection system by integrating hadoop and naive bayes classification. InDalhousie Computer Science In-house Conference (DCSI). Dalhousie University, Halifax, Canada. OpenURL.

33 Gavrilis, D. and Dermatas, E. (2005) Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features. Computer Networks and ISDN Systems, 48, 235–245.

34 Wu, Y. C., Tseng, H. R., Yang, W., and Jan, R. H. (2011) DDoS detection and traceback with decision tree and grey relational analysis. International Journal of Ad Hoc and Ubiquitous Computing, 7, 121–136.

35 Chen, Z., Chen, Z., and Delis, A. (2007) An inline detection and prevention framework for distributed denial of service attacks. Comp. J., 50, 7–40.

36 (Guyon and Elisseeff, 2003)

37 Buchanan, W. J., Flandrin, F., Macfarlane, R., & Graves, J. (2011). A methodology to evaluate rate-based intrusion prevention system against distributed denial-of-service (DDoS).

38 Kim, M., Na, H., Chae, K., Bang, H., & Na, J. (2004). A combined data mining approach for DDoS attack detection. In Information Networking. Networking Technologies for Broadband and Mobile Networks (pp. 943-950). Springer Berlin Heidelberg.

39 Jeena,R., & Rajeswari,M., (2015, February). Attack Graph Model: A New Approach for DDOS Attack Detection in Cloud, International Journal of Innovative Research in Computer and Communication Engineering on(Vol. 3,Issue 2, pp. 1276-1280).

40 Noble, C. C., & Cook, D. J. (2003, August). Graph-based anomaly detection. InProceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 631-636). ACM.

41 Game, S. G., & Natikar, S. B. (2014, August). Graph-based Attack Detection in Cloud using KDD CUP 99 Dataset, International Journal of Science and Research, (Vol. 3, Issue 8, pp.511-516).

42 Eberle, W., & Holder, L. (2013, December). Incremental Anomaly Detection in Graphs. In *2013 IEEE 13th International Conference on Data Mining Workshops* (pp. 521-528). IEEE.

43 Le, D. Q., Jeong, T., Roman, H. E., & Hong, J. W. K. (2011, October). Traffic dispersion graph based anomaly detection. In *Proceedings of the Second Symposium on Information and Communication Technology* (pp. 36-41). ACM.

44 Morales F.E. (2012), Algoritmo EM, Recuperado 22 de Noviembre del 2015, de https://ccc.inaoep.mx/~emorales/Cursos/NvoAprend/node81.html

45 Morales F.E. (2012), Vecinos más cercanos, Recuperado 24 de Noviembre del 2015, de https://ccc.inaoep.mx/~emorales/Cursos/NvoAprend/node70.html

46 Departamento de lenguaje y ciencias de la computacion. (2009), Teoría de conjuntos difusos y lógica difusa, Recuperado 22 de Febrero del 2016, de http://www.lcc.uma.es/~eva/aic/apuntes/fuzzy.pdf

47 Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. In Encyclopedia of database systems (pp. 532-538). Springer US.

48 Cambronero, C. G., & Moreno, I. G. (2006). Algoritmos de aprendizaje: knn & kmeans. Intelgencia en Redes de Comunicación, Universidad Carlos III de Madrid.

49 Gonzalez Bernal J.A. (2011), Descubrimiento de conocimiento basado en grafos, Recuperado 22 de Enero del 2016, de https://ccc.inaoep.mx/~jagonzalez/ML/principal/node50.html

50 Vázquez, A. C., & Concepción, L. P. (2011). Descubrimiento de Conocimiento Basado en Grafos. Revista de investigación de Sistemas e Informática, 8(2), 17-25.

62 The Subdue Proyect (2011), Subdue Manual, Recuperado 22 de Enero del 2016, de http://ailab.wsu.edu/subdue/