PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

IMPLEMENTACIÓN DE UN SISTEMA DE SEGURIDAD DE REDES CENTRALIZADO MEDIANTE UN FIREWALL CON AUTOCONFIGURACIÓN BASADO EN DETECCIÓN DE TRÁFICO ANÓMALO UTILIZANDO ALGORITMOS BIOINSPIRADOS

Tesis para obtener el título profesional de Ingeniera Informática

AUTORA:

Katherine Laura Borja Torres

ASESORES:

Mg. Moisés Antonio Villena Aguilar

Mg. Rony Cueva Moscoso

Lima, Diciembre, 2025

Informe de Similitud

Yo, Moisés Antonio Villena Aguilar, docente de la Facultad de Ciencias e Ingeniería de la Pontificia

Universidad Católica del Perú, asesor(a) de la tesis/el trabajo de investigación titulado:

Implementación de un Sistema de Seguridad de Redes centralizado mediante un firewall con autoconfiguración basado en detección de tráfico anómalo utilizando algoritmos bioinspirados

de la autora Katherine Laura Borja Torres

dejo constancia de lo siguiente:

- El mencionado documento tiene un índice de puntuación de similitud de 18%. Así lo consigna el reporte de similitud emitido por el software *Turnitin* el 01/12/2025.
- He revisado con detalle dicho reporte y la Tesis o Trabajo de Suficiencia Profesional, y nose advierte indicios de plagio.
- Las citas a otros autores y sus respectivas referencias cumplen con las pautas académicas.

Lugar y fecha: San Miguel, Lima, lunes 01 de diciembre del 2025

Apellidos y nombres del asesor: Villena Aguilar, Moisés Antonio	
DNI: 10064527	Firma
ORCID: 0000-0002-1106-9551	 Firmado digitalmente por: VILLENA AGUILAR Moises Antonio FAU 20254165035 soft Motivo: Soy el autor del documento Fecha: 27/01/2026 19:19:45-0500

Dedicatoria

A mi padre, por ser mi inspiración en la investigación y en el amor por la ciencia, a mi madre por ser ejemplo de perseverancia y de resiliencia y por eneseñarme a nunca rendirme. A mi profesor Miguel Guanira, Ana Roncal y Layla Hirsh por su apoyo en momentos dificiles de mi carrera, me demostraron que además de ser ingeniero, se es humano. A los profesores que me enseñaron y de quienes llevo un parte de ellos en la profesional que soy hoy.

Agradecimientos

Agradezco a mis asesores Moisés Villena y Rony Cueva por su apoyo, orientación y conocimientos en esta etapa, larga y cansina, pero también muy productiva.

Agradezco a Dios por permitirme dar este gran paso que sé que me llevará mas cerca de mis objetivos. Esta tesis es solo el comienzo de cosas más grandes.

Resumen

Con el uso generalizado de las redes de computadoras, aplicadas a todas las funciones de las instituciones y empresas en general, poder garantizar la seguridad de las redes es un factor de primordial importancia que debe ser implementado con sistema avanzados para una reacción inmediata.

En este trabajo de tesis se propone un sistema centralizado de seguridad de redes de computadoras con detección de intrusiones basado en algoritmos bioinspirados como la redes neuronales artificiales (RNA) y bloqueo automático de tráfico de intrusiones mediante reglas de firewall sobre la plataforma Linux.

El sistema centralizado de seguridad propuesto captura logs de tráfico de red mediante *iptables*, procesa los logs para obtener características de conexiones de tráfico anómalo, analiza las conexiones de tráfico mediante redes neuronales artificiales para reconocer intrusiones, y si son reconocidas como tráfico anómalo de intrusiones se bloquean mediante reglas de *iptables* del firewall de Linux.

Finalmente se realizaron pruebas de cada módulo por separado y todos los módulos integrados operando de forma automática realizando todos los procesos cada minuto logrando generar reglas de bloqueo de conexiones de intrusiones y garantizando la reacción inmediata a las intrusiones de red, para lo cual la RNA pasó por un entrenamiento utilizando muestras de tráfico anómalo de intrusiones.

Índice

Índice de Tablas

Índice de Figuras

Capítulo 1. Generalidades

1.1. Problemática

La definición del problema a resolver en la presente tesis se realizó elaborando el árbol de problemas, la descripción de la problemática y el problema seleccionado.

1.1.1. Árbol de problemas

Para elaborar el árbol de problemas se identificó el problema central, seguidamente los problemas efecto y finalmente los problemas causa, presentado en la Tabla 1.

Tabla 1. Árbol de problemas

Problemas efectos	Los criterios conocidos son incompletos dejando en riesgo la seguridad de los sistemas computacionales de la organización.	No se atiende la seguridad global de la organización.	Reacción tardía a un ataque informático.
Problema central	La seguridad de la información en una organización es vulnerable debido a que no es posible detectar patrones de ataques y tomar una medida inmediata para gestionar el riesgo.		
Problemas causas	La configuración de los firewalls se realiza en base a criterios de seguridad conocidos, como bloquear números ips conocidos, bloquear puertos, etc.	La seguridad es aplicada de manera aislada a cada sistema de cómputo dentro de la organización	La configuración del firewall se realiza manualmente, después de detectar alguna anomalía analizando los logs.

1.1.2. Descripción

La amplia difusión de uso de las redes de datos en los procesos empresariales y personales ha llevado a un escenario donde todas las actividades diarias se realizan usando servicios en internet (S. Helal, 2002) lo que requiere una conexión total a internet de todos los equipos de cómputo, equipos móviles y maquinas con IoT en general.

Esta necesidad de conexión a la red ha hecho que los fabricantes de computadoras, celulares y maquinas incluyan en sus sistemas operativos los mecanismos de conexión, múltiples aplicaciones de acceso a servicios, aplicaciones que tienen vulnerabilidades (S. Jin, 2009) que son usadas para acceder irregularmente a la información que contienen.

Por otro lado, se ha venido desarrollando un grupo de personas a nivel mundial, que trabajan ya sea con buena o mal intención en explotar las vulnerabilidades de los sistemas (Gundecha, 2011) y acceder a información privada sin autorización, creando situaciones riesgosas para la información empresarial u organizacional ya sea por accesos indebidos usuarios internos o externos.

Como respuesta a esta situación de los equipos conectados a internet se han desarrollados múltiples estrategias de protección de la información a nivel aislada para equipos de cómputo y a nivel de la red privada de la empresa u organización, con una serie de aplicaciones como: antivirus, firewalls (Krupa C. Patel, 2017), sistemas de gestión de eventos de la seguridad de la información (SIEM) (Gustavo González-Granadillo, 2021), sistemas de detección de intrusiones (IDS) (Jakub Svoboda, 2015), sistema de prevención de intrusiones (IPS) (D. Stiawan, 2010), entre otros.

Como alternativa de seguridad global de una red privada empresarial u organizacional se utiliza firewalls (Krupa C. Patel, 2017) que es un sistema que se instala entre la red interna e internet creando una separación física y el tráfico de red circula por el mismo, en donde se puede implementar reglas de control acorde a las políticas de seguridad que tiene la empresa. Esta solución, por un lado, tiene la desventaja de que la definición de las políticas de seguridad depende del criterio y experiencia de especialistas, y, por otro lado, las reglas de seguridad son implementadas manualmente, lo que hace sistemas de seguridad con una reacción lenta a nuevos riesgos.

El enfoque de seguridad basada en asegurar aisladamente cada equipo de cómputo (Bukac, 2012) tiene el mismo principio que se implementa por experiencias pasadas y además la sobrecarga en el personal de servicio que debe atender cada equipo de cómputo de forma aislada.

Para agilizar el proceso de reacción ante nuevos riesgos de seguridad en la red se implementan sistemas de monitoreo y detección como los SIEM (Gustavo González-Granadillo, 2021), que mediante análisis de registros de los sistemas operativos y de red dan información de eventos que usan los especialistas para tomar acciones, los IDS son sistemas que detectan intrusiones e informan sobre esta situación para la toma de acciones de seguridad (Jakub Svoboda, 2015) y finalmente los IPS (D. Stiawan, 2010) son sistemas que al detectar una intrusión toman acciones de seguridad como reportar, bloquear, eliminar convirtiéndose en los sistemas más avanzados en seguridad de redes también conocidos como firewalls de la siguiente generación (NGFW) (Kim, 2022). Sin embargo, la detección y prevención de intrusiones se basa en patrones conocidos, lo que no permitiría la detección de nuevos tipos de intrusión y los más avanzados utilizan la detección de anomalías de tráfico en la red que dependiendo de los métodos de detección podrían generar falsos positivos.

La detección de trafico anómalo es una labor de alta complejidad que no se resuelve con algoritmos comunes computacionales, por lo que recientemente se vienen desarrollando nuevos métodos para la detección de anomalías en el tráfico de red mediante algoritmos bio-inspirados como algoritmos genéticos (Reza Rafeh, 2022), redes neuronales artificiales (Soodeh Hosseini, 2020), algoritmos de colonia de abejas (Ali Hussein Shamman Al-Safi, 2021), método de optimización por enjambre de partículas (Almomani, 2021), entre otros que están teniendo resultados alentadores en los últimos tiempos.

En este contexto, el problema es que la seguridad de la información en una organización es vulnerable debido a que no es posible detectar nuevos patrones de intrusiones y tomar una acción inmediata para eliminar el riesgo (Glesias, 2015), (Reza Rafeh, 2022).

1.1.3. Problema seleccionado

Por lo presentado, en esta tesis se realizará una propuesta que permita mejorar la seguridad de la información de la organización usando un sistema centralizado con capacidad de tomar acciones inmediatas reportando, bloqueando y eliminando la intrusión, basándose en la detección de tráfico anómalo utilizando nuevas herramientas con algoritmos bio-inspirados.

1.2. Objetivos

Con la meta de resolver el problema seleccionado se plantean el objetivo general y los objetivos específicos considerando el árbol de problemas.

1.2.1. Objetivo general

Implementar un sistema de seguridad de redes centralizado mediante un firewall con autoconfiguración basado en detección de tráfico anómalo utilizando algoritmos bioinspirados.

Objetivos específicos

O 1. Implementar en el firewall un módulo de identificación de criterios de seguridad. (C1)

O 2. Implementar el sistema de seguridad centralizado en el firewall de la red que se desea asegurar. (C2)

O 3. Implementar el módulo de autoconfiguración del firewall. (C3)

Resultados esperados

O 1. Implementar en el firewall un módulo de identificación de criterios de seguridad.

R 1. Módulo de aprendizaje de criterios de seguridad.

R 2. Módulo de detección de intrusiones.

O 2. Implementar el sistema de seguridad centralizado en el firewall de la red que se desea asegurar.

R 3. Servidor firewall instalado con la configuración física de tal forma que todo el tráfico de la red pase a través de ese servidor.

R 4. Servidor firewall con el software instalado y configurado.

O 3. Implementar el módulo de autoconfiguración del firewall

R 5. Módulo de autoconfiguración del firewall utilizando información entregada por el módulo de detección de intrusiones.

1.2.2. Mapeo de objetivos, resultados y verificación

Tabla 2. Extensión del objetivo 1

Objetivo 1: Implementar en el firewall un módulo de identificación de criterios de seguridad.		
Resultado	**Medio de verificación**	**Indicador objetivamente verificable**
Módulo de aprendizaje de criterios de seguridad	1. Informe técnico de pruebas del módulo	2. Porcentaje de pruebas de aprendizaje exitosas al 100%
Módulo de detección de intrusiones	3. Informe técnico de pruebas del módulo	4. Porcentaje de pruebas de detección exitosas al 100%

Tabla 3. Extensión del objetivo 2

Objetivo 2: Implementar el sistema de seguridad centralizado en el firewall de la red que se desea asegurar.		
Resultado	**Medio de verificación**	**Indicador objetivamente verificable**
Servidor firewall instalado con la configuración física de tal forma que todo el tráfico de la red pase a través de ese servidor.	5. Informe técnico de la instalación.	6. Verificación física del servidor instalado
Servidor firewall con el software instalado y configurado.	7. Informe técnico de la instalación y configuración del software.	8. Porcentaje de pruebas de conexión de los usuarios y los servicios al 100%.

Tabla 4. Extensión del objetivo 3

Objetivo 3: Implementar el módulo de autoconfiguración del firewall.		
Resultado	**Medio de verificación**	**Indicador objetivamente verificable**
Módulo de autoconfiguración del firewall utilizando información entregada por el módulo de detección de intrusiones.	9. Informe técnico de implementación y pruebas del software	10. Porcentaje de pruebas de autoconfiguración exitosas al 100%

1.3. Métodos y Procedimientos

Los métodos y procedimientos para desarrollar e implementar el aprendizaje de criterios de seguridad, detección de intrusiones, instalación y configuración de servidor firewall basado en iptables y configuración automática de firewall se presentan en la tabla 6.

Tabla 5. Métodos y procedimientos.

RESULTADOS ESPERADOS	MÉTODOS Y PROCEDIMIENTOS
1.1. Módulo de aprendizaje de criterios de seguridad.	Se aplica redes neuronales artificiales que se implementa utilizando el lenguaje PHP y la base de datos PostgreSQL sobre Linux, y se prueba experimentalmente con el prototipo de software desarrollado.
1.2. Módulo de detección de intrusiones.	Se aplica redes neuronales artificiales que se implementa utilizando lenguaje PHP y la base de datos PostgreSQL sobre Linux, y se prueba experimentalmente con el prototipo de software desarrollado.
2.1. Servidor firewall instalado con la configuración física de tal forma que el tráfico de la red pase a través de ese servidor.	Se utiliza la especificación de cableado estructurado para realizar la instalación del servidor firewall con una configuración física de tal forma que todo el tráfico de red para por el firewall y se prueba experimentalmente con pruebas de conexión.
2.2. Servidor firewall con el software instalado y configurado.	Se utiliza la guías técnicas de instalación y configuración de Linux y del software complementario requerido y se prueba experimentalmente con pruebas de operación del firewall .
3.1. Módulo de auto configuración del firewall utilizando información entregada por el módulo de detección de intrusiones.	Se aplica procesamiento de cadenas que se implementará en lenguaje de programación PHP y base de datos PostgreSQL sobre Linux para generar las reglas del firewall y se prueba experimentalmente con el prototipo de software desarrollado.

El desarrollo del software de todos los módulos se realiza con lenguaje de programación PHP y para almacenar datos se realizó utilizando bases de datos en PostgreSQL.

La detección de intrusiones es realizado con Redes Neuronales Artificiales (RNA) con diversos métodos de aprendizaje (Soodeh Hosseini, 2020) y (Zouhair Chiba, 2022), en la presente tesis se implementa una RNA utilizando el lenguaje de programación PHP utilizando una base de datos en PostgreSQL, para esto primero se implementa el módulo de aprendizaje de criterios de seguridad basados en los patrones, con este conocimiento se implementara el módulo de detección de intrusiones con una RNA, reconoce los criterios de seguridad que representan intrusiones basados en el tráfico de la red. Para desarrollar los dos módulos se realizará utilizando logs de trafico de paquetes IP de bases de datos y luego se probará con logs capturados por el firewall implementado en el proyecto.

El firewall se instala de acuerdo a la disposición física de tal forma que todo el tráfico de la red pase por el firewall (Sharma, 2020), y se realiza la instalación de sistema operativo Linux Fedora con soporte de PHP, servidor de base de datos PostgreSQL, servidor web Apache e iptables. Además, se realiza la configuración para la captura de logs de trafico de paquetes IP, y se realizan pruebas de operación básica del firewall.

La configuración automática del firewall (D. Bringhenti, 2020) de acuerdo a los resultados de detección de intrusiones se realiza aplicando procesamiento de cadenas y se aplica modificando directamente las reglas del firewall mediante comandos en el Shell desde el software de configuración automática.

1.4. Viabilidad

1.4.1. Viabilidad Técnica

Científicamente es viable ya que se tiene acceso a artículos científicos publicados que proponen y explican diversas formas de aplicación de RNA aplicadas a soluciones de seguridad. Por otro lado, se cuenta con preparación en el uso de herramientas de programación, software libre, instalación y configuración de firewall basada en Linux, durante el estudio de la carrera y finalmente se cuenta con la asesoría dos docentes especialistas en las áreas requeridas

1.4.2. Viabilidad Económica

Los recursos necesarios para la implementación de prototipo de desarrollo es una computadora core I7 para usar como servidor firewall, sistema operativo Linux y otros softwares libres que no tienen costos de licencia y el acceso a una red de aproximadamente diez usuarios con servidores web, correo electrónico y otros servicios es de libre disponibilidad para el proyecto, por lo que la infraestructura experimental para el desarrollo es adecuada. En este contexto el proyecto de investigación es viable por contar con todos los recursos necesarios para el desarrollo de la tesis.

1.5.Alcance, limitaciones y riesgos

1.5.1. Alcance

El alcance es implementar un prototipo de un sistema de seguridad de redes centralizado mediante un firewall con autoconfiguración basado en detección de tráfico anómalo utilizando algoritmos bioinspirados (Redes neuronales artificiales) para una red de diez usuarios con servicios en de web, correo electrónico, ftp, etc utilizando una computadora core I7 como servidor con sistema operativo Linux y software de desarrollo libre como *iptables* para implementar el firewall, php para desarrollo de aplicaciones, apache como servidor web y Postgres como servidor de base de datos.

1.5.2. Limitaciones

Las principales limitaciones de la tesis fue la decisión de cambiar de base de datos de MariaDB a Postgres debido al mejor desempeño en velocidad de la segunda y la indisponibilidad para realizar ataques reales para verrificar el sistema.

1.5.3. Riesgos

Los riesgos del proyecto están relacionados con el tiempo para el desarrollo del proyecto y la disponibilidad de ataque reales a la red, para lo cual se determina los síntomas, las probabilidades, el impacto, la severidad como mitigar el riesgo y la contingencia en caso se presente que se observan en la tabla 3. Los valores de las probabilidades están comprendidos entre 0 y 1, donde 0 es nada probable y 1 es totalmente probable. Los posibles valores del impacto y severidad son bajo, medio y alto.

Tabla 6: Riesgos del proyecto

Nº	Descripción	Síntomas	Probabilidad	Impacto	Severidad	Mitigación	Contingencia
1	El plazo de 14 semanas de ejecución del proyecto	El avance no de acuerdo al cronograma	0,5	Alto	Media	Planificar una dedicación de medio tiempo del tesista	Incrementar la dedicación.
2	Intrusiones a la red para pruebas del sistema	Ausencia de intrusiones	0,5	Alto	Media	Activar servicios expuestos	Simular ataques

2. Marco Conceptual

2.1. Modelo OSI y Modelo TCP/IP

OSI y TCP/IP son los modelos de referencia (Tanenbaum, 2005). Cada modelo ejemplifica una arquitectura distinta de red. Los protocolos del modelo OSI no son muy usados, pero como modelo son ampliamente referenciado. Por otra parte, el modelo TCP/IP no es tan referenciado pero sus protocolos son usados ampliamente.

2.1.1. Modelo OSI

El modelo OSI se fundamenta en la propuesta presentada por la Organización Internacional de Normas (ISO), el cual consta de 7 capas como se aprecia en la Figura 1.

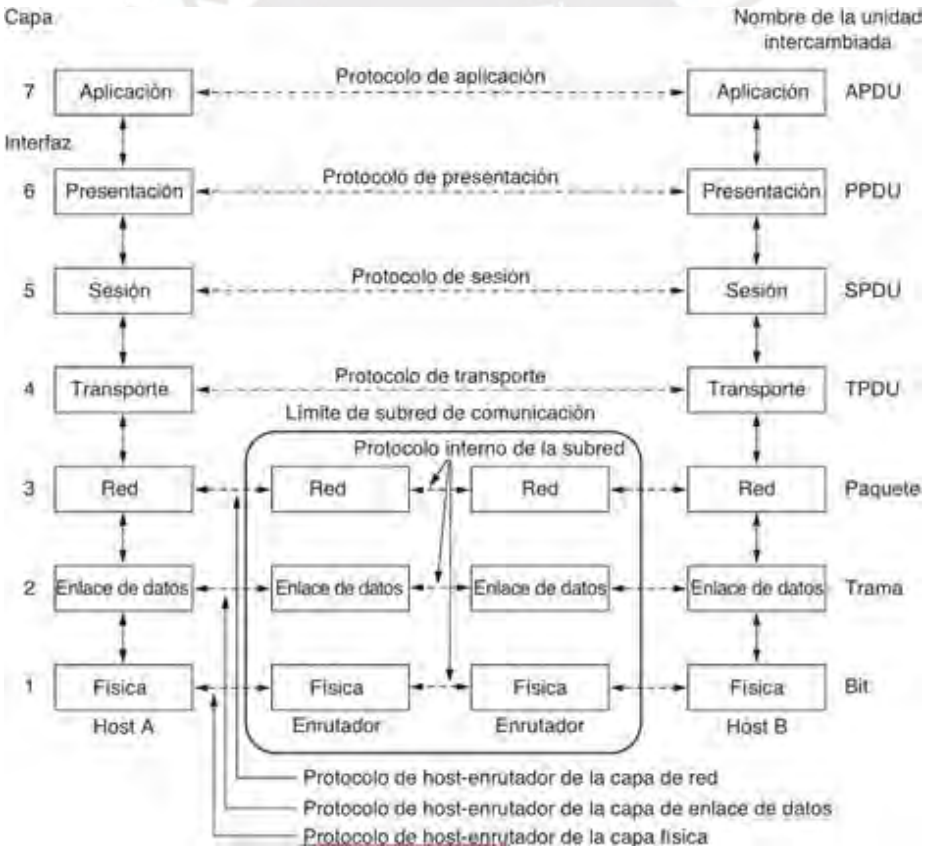


Figura 1. Modelo de Referencia OSI (Tanenbaum, 2005)

Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

2.1.2. Capa física

La primera capa del modelo OSI, trasmite los bits por un medio físico de comunicación. Siendo su principal función convertir los bits en señales electrónicas, electromagnéticas u ópticas, para que puedan ser transmitidos por cables, fibra óptica, radio u otros medios. En esencia, la capa física maneja la conexión física entre dispositivos, estableciendo y manteniendo la comunicación a través de señales físicas.

Las tecnologías de transmisión de esta capa son: cable coaxial; par trenzado; fibra óptica y líneas eléctricas.

Par trenzado

Es un cable de red que consiste en pares de hilos conductores aislados que se trenzan entre sí como se presenta en la Figura 2. Esta disposición de los hilos ayuda a reducir la interferencia electromagnética (EMI) y la diafonía, mejorando la calidad de la señal.

Figura 2. Cable par trenzado
Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

Cable Coaxial

Este cable tiene un conductor central cubierto por aislamiento y un conductor exterior en forma de malla como se ve en la Figura 3.

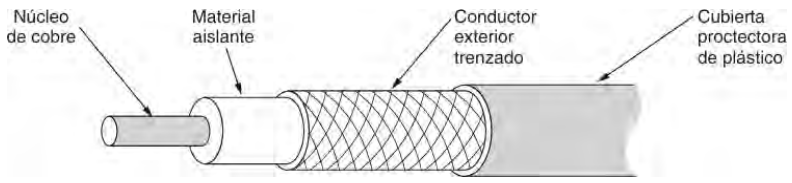


Figura 3. Cable coaxial
Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

Fibra Óptica

Es un cable que transmite información a grandes distancias mediante pulsos de luz. Se fabrica con hilos de vidrio o plástico de alta pureza, en la Figura 4 se ve el cable de fibra óptica

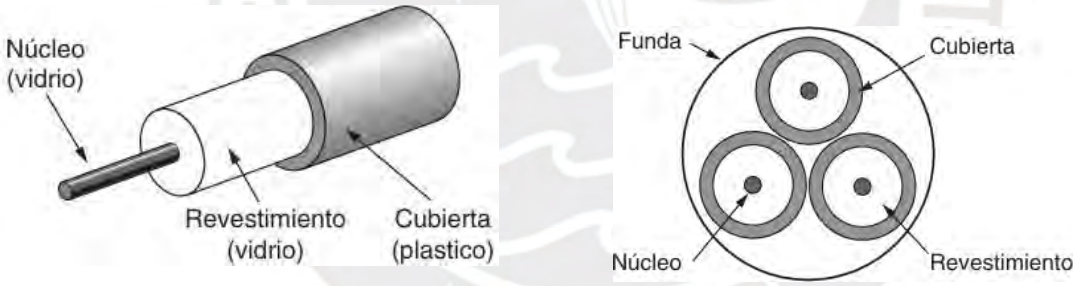


Figura 4. Vista lateral y de extremo de una sola fibra óptica
Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

2.1.3. Capa de enlace

Es la segunda capa que transmite de manera fiable de datos a través de un enlace físico entre dos nodos adyacentes en una red. Su función principal es facilitar la transmisión de datos entre equipos de la una red que utiliza el mismo medio de trasmisión, garantizando una transmisión segura y sin errores.

La función principal de la capa de enlace es la transmisión de datos libre de errores para la capa de red. Esto implica la creación de tramas, la gestión de direccionamiento y el control de errores para garantizar una transferencia de datos confiable a través del medio físico. Las funciones específicas de la capa de enlace son las siguientes:

1. Proporcionar una interfaz de servicio definida a la capa de red.
2. Gestionar los errores que se producen en la transmisión.
3. Controlar el flujo de datos de tal forma que los recpetores lentos no sean saturados por los emisores rápidos.

La capa de enlace cumple con estas funciones, tomando los paquetes de la tercera capa y los divide en marcos para transmitirlos. Los marcos tienen un encabezado, un espacio para la carga útil de datos del paquete y una bandera de fin de marco, mostrado en la Figura 5.

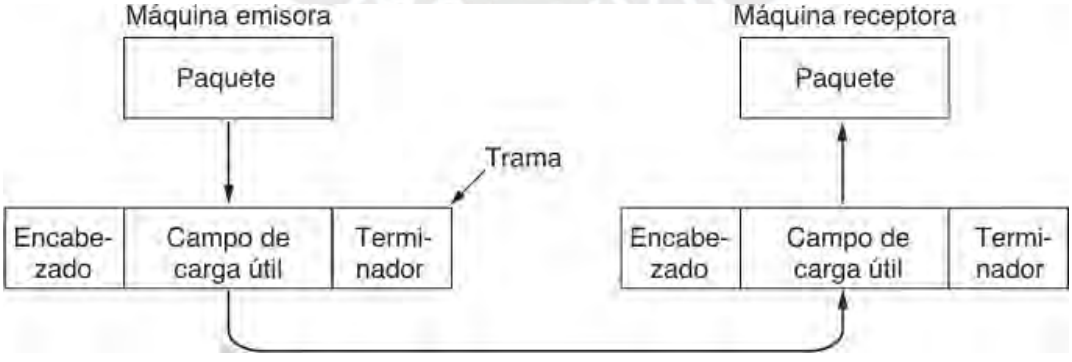


Figura 5. Relación entre paquete y trama.
Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

2.1.4. Capa de red

Esta capa transmite datos entre diferentes redes o nodos de una red. Esta capa define la ruta de los paquetes de datos desde el host origen hasta el host destino, utilizando el protocolo IP para direccionar y enrutar los datos. Un esquema de cómo funciona la capa de red se muestra en la Figura 6.

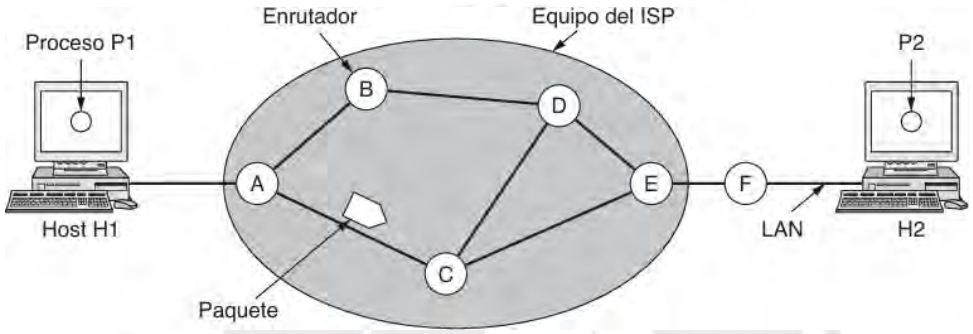


Figura 6. La capa de red: Entorno de protocolos
Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

Dentro de las principales funciones de la capa de red tenemos:

- Direccionamiento. – La direcciones IP (Internet Protocol) se utilizan para identificar y localizar los dispositivos en la red.
- Enrutamiento. –Esta capa decide la ruta optima de los paquetes de datos para que sean entregados al destino, considerando factores como la congestión del tráfico y la distancia entre los nodos.
- Encapsulamiento. -La capa de red agrega una cabecera IP a los datagramas recibidos de la capa superior o capa de transporte, formado los paquetes IP.
- Desencapsulamiento. -La capa de red del dispositivo receptor elimina la cabecera IP del paquete para entregarlos datos a la capa superior.

En conclusión, cuando envías un correo electrónico, la capa de red, enruta los paquetes de datos a por la red de Internet hasta el servidor de correo del destinatario, utilizando la dirección IP del servidor.

El protocolo IP versión 4

El datagrama IPv4 esta formado por dos secciones: la cabecera IP y el cuerpo que contiene la carga útil. La cabecera IP contiene una sección de tamaño fijo de veinte bytes y una longitud variable de datos opcionales. El formato de la cabecera IP esta en la Figura 7. Los bits se trasmiten en orden big endian, es decir de izquierda a derecha y de arriba hacia abajo iniciando por el bit de mayor orden.

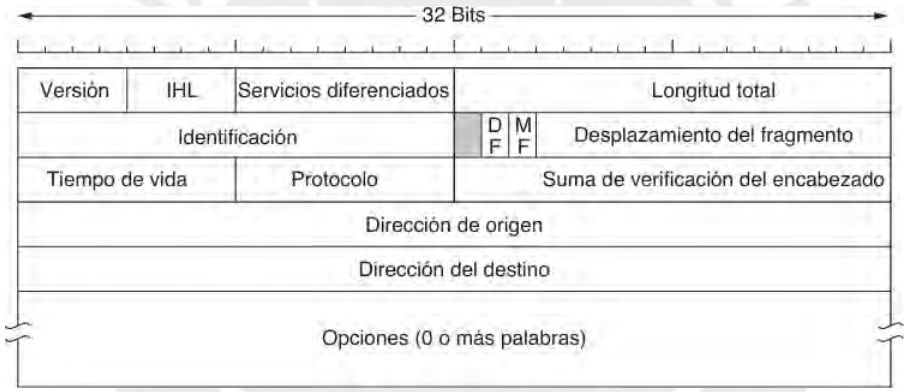


Figura 7. El encabezado de IP (Protocolo de Internet).

Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

La versión del protocolo del datagrama esta en el campo versión, lo que permite la transición de datagramas de diferentes versiones manteniendo la compatibilidad.

El campo IHL inidica la longitud de la cabecera IP en palabras de 32 bits, siendo 5 el valor mínimo, para los datos indispensables del protocolo cuado no tiene datos opcionales. Este campo es de 4 bits por lo que puede terner un valor máximo de 15 que hacen un total de 60 bytes.

El campo Servicios Diferenciados (DS) en el encabezado de IP es de 8 bits, utilizado para para la gestión del tráfico de red, específicamente indica la clase de servicio o prioridad de un paquete IP

En la cabezera IP, el tipo de servicio (ToS) se especifica con 8 bits, el cual inida el tratamiento que se da al paquete IP, aunque en la actualidad, no es generalizado el uso. Este campo contiene subcampos que especifiquen la prioridad, retardo, rendimiento y confiabilidad deseados para el paquete.

El campo Longitud Total de 16 bits se utiliza para especificar la longitud del datagrama IP en bytes, considerando la cabecera y los datos, lo que hace un máximo de 65 535 bytes.

El host destino identifica a que paquete pertenece un fragmento mediante el contenido del campo identificación, ya que el host origen asigna el mismo numero de indentificación a todos los fragmentos de un paquete.

El campo DF (del ingles: Dnon’t Fragment) especifica que el paquete no puede ser fragmentado por lo enrutadores.

En la cabecera IP, los campos MF (Más Fragmentos) y DF (No Fragmentar) controlan la fragmentación de paquetes. MF indica si el paquete es parte de un conjunto de fragmentos (1 para fragmentos intermedios, 0 para el último fragmento). DF indica si el paquete debe fragmentarse (0 para fragmentación, 1 para no fragmentar.

El campo Desplazamiento de Fragmento de 13 bits en el encabezado de un paquete IP especifica la posición inicial de los datos del fragmento en el paquete IP original. Este campo se utiliza para armar el paquete original con los fragmentos recibidos. El desplazamiento se mide en múltiplos de 8 bytes.

El tiempo de vida es implementado con campo TTL, que es un contador que disminuye cada vez que pasa el paquete por un enrutador, y si llega a cero el paquete es desechado, evitando asi que los paquetes se queden en la red por tiempo indefinido.

La capa de red reensambla el paquete con los fragmentos recibidos y lo entrega al a la capa de transporte indicado por el campo protocolo.

El campo suma de comprobación de 16 bits es utilizado para verificar los errores en la trasmisión de la cabecera IP.

En el protocolo de internet, la dirección del host que envía es contenida en el campo dirección origen y la dirección del host destino en el campo dirección destino, datos que utilizadan los enrutadores para entregar los paquetes al destino y este pueda ser respondido.

El campo Opciones en la cabecera IP es un campo opcional que permite incluir información adicional en un paquete IP, como información de seguridad, registro de ruta o marcación de tiempo. Este campo no es obligatorio y su longitud es variable, con un máximo determinado por la longitud total de la cabecera.

En la Tabla 1, se muestra las posibles opciones que se pueden dar en este campo.

Tabla 7. Algunas de las opciones del protocolo IP

Opción	Descripción
Seguridad.	Especifica qué tan secreto es el datagrama.
Enrutamiento estricto desde el origen.	Proporciona la ruta completa a seguir.
Enrutamiento libre desde el origen.	Proporciona una lista de enrutadores que no se deben omitir.
Registrar ruta.	Hace que cada enrutador adjunte su dirección IP.
Estampa de tiempo.	Hace que cada enrutador adjunte su dirección y su etiqueta de tiempo.

Nota *Tomado de "redes de computadoras"*, A. S. Tanenbaum *(Madrid), 2005*

Direcciones IP

La dirección IP identifica de manera única a un host en la red global de internet, por lo que a todos los dispostivos conectados a internet se debe asignar una dirección ip única para que sea posible la trasmisión de datos.

Existen las versiones IPv4 con direcciones de 32 bits y IPv6 con direcciones de 128 bits. IPv6 es la versión más reciente y está diseñada para reemplazar a IPv4, ya que ofrece una mayor cantidad de direcciones disponibles. Las direcciones del protocolo internet pueden ser estáticas o dinámicas, y privadas o públicas.

- Las direcciones estáticas. - No cambian y son utilizadas por servidores o dispositivos que necesitan una dirección fija.
- Las direcciones dinámicas. - Cambian cada vez que los hosts se conectan a la red y se asignan automáticamente mediante el servicio de red DHCP (Del ingles: Dinamic Host Configuration Protocol).
- Las direcciones privadas. – Es un rango de IPs que se usan para hosts en las redes locales y no son accesibles desde Internet.
- Las direcciones públicas. – Es un rango de IPs, que se utilizan para los hosts que deben ser accesibles desde internet.

Formatos de direcciones IP

Se definieron cinco clases de direcciones IP de acuerdo la cantidad de bits asignados para especifcar el numero de red y numero de host, como de puede ver en la figura 8.

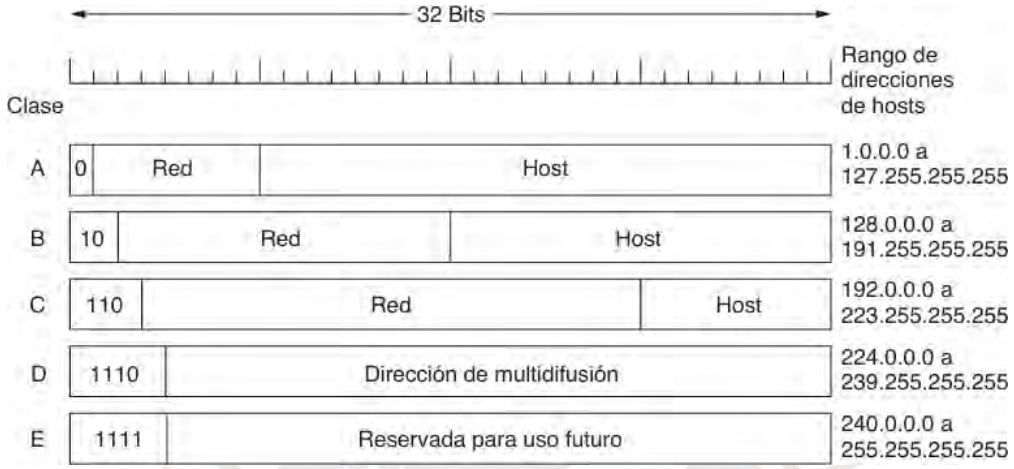


Figura 8. *Formatos de direcciones IP*
Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

Las direcciones IP se organizan en clases para facilitar su gestión y el enrutamiento en la red. Estas clases se basan en el valor del primer octeto de la dirección y determinan el numero de red y la cantidad de dispositivos de cada red.

En la clase A, puede haber máximo, 128 redes y cada red puede terner hasta 16 millones de hosts, la clase B puede tener hasta 16384 redes, cada una con un máximo de 65534 hosts y de clase C pueden existir has 16777214 de redes y cada red puede tener hasta con hasta 254 hosts, en donde la primer a IP de rango es numero de red y la ultima IP es numero de difusión. Para enviar datagramas a multiples hosts se puede usar la clase D.

Las diercciones de internet es clasifcados de la siguiente forma:

Clase A: El primer octeto va de 0 a 127, son las que identifican redes grandes.

Clase B: El primer octeto va de 128 a 191, son las que identifican redes medianas.

Clase C: El primer octeto va de 192 a 223, son las que identifican redes pequeñas.

Clase D: El primer octeto va de 224 a 239, son direcciones multicast utilizadas para enviar datos a múltiples dispositivos al mismo tiempo.

Clase E: El primer octeto va de 240 a 255, son direcciones reservadas para investigación y desarrollo.

2.1.5. La capa de transporte

La comunicación entre el host origen y destino es gestionada por la capa esta, garantizando la entrega segura y ordenada de datos entre aplicaciones. Las funciones son las siguientes:

- Segmentación y ensamblado de datos: Los datos recibidos de la capa de aplicación son divididos en partes de tamaño máximo de 65536 bytes, que son posibles trasmitir en un datagrama, también conocidos como segmentos y se encarga de reensamblar los segmentos en el extremo receptor.
- Control de flujo y de congestión: Ayuda a gestionar el flujo de datos entre aplicaciones para evitar la sobrecarga de la red y asegurar una transmisión eficiente.
- Control de errores: Verifica la integridad de los datos y retransmite los segmentos perdidos o corruptos, asegurando una entrega confiable.
- Multiplexado y desmultiplexado: Permite que múltiples aplicaciones compartan la misma conexión de red al etiquetar los datos con información de la aplicación de origen y destino, utilizando puertos.
- Gestion a la conexión: Gestiona la conexión entre el host origen y el host destino en el caso del protocolo TCP, o una transmisión sin conexión en el caso de protocolo UDP.

La capa de trasporte usa primitivas de comunicación para interctua con la capa superior de aplicación y la capa inferior de red. Estas primitivas permiten establecer, mantener y terminar conexiones, enviando y recibiendo datos de forma segura y confiable. Las primitivas pueden apreciarse en la tabla 2.

Tabla 8. Las primitivas para un servicio simple de transporte.

Primitiva	Paquete enviado	Significado
LISTEN	(ninguno)	Se bloquea hasta que algún proceso intenta conectarse.
CONNECT	CONNECTION REQ.	Intenta activamente establecer una conexión.
SEND	DATA	Envía información.
RECEIVE	(ninguno)	Se bloquea hasta que llegue un paquete DATA.
DISCONNECT	DISCONNECTION REQ.	Solicita que se libere la conexión

Nota *Tomado de "redes de computadoras"*, A. S. Tanenbaum *(Madrid), 2005*

2.1.6. Capa de aplicación

La capa superior es la de aplicación que interactua con las aplicaciones del usuario. Proporciona servicios que soportan aplicaciones como los navegadores web, la transferencia de archivos, el correo electrónico y la resolución de nombres de dominio. Esta capa define los protocolos que utilizan las aplicaciones para intercambiar datos. Las funciones clave de la capa son las siguientes:

Proporciona la interfaz para las aplicaciones en la red:

- Es el punto de contacto de las aplicaciones del usuario y la red de comunicación de datos entre host.
- Define los protocolos de comunicación. -Establece las reglas y los formatos que las aplicaciones siguen para comunicarse.

- Soporta aplicaciones de usuario. - Permite que las aplicaciones accedan a servicios de FTP, HTTP, SMTP, DNS entre otros.

Aplica las funcionalidades de las capas superiores del modelo OSI:

- En esencia, la capa de aplicación de TCP/IP maneja las funciones de las capas de sesión, presentación y aplicación del modelo OSI.

Existen varios protocolos en la capa de aplicación, entre los cuales tenemos:

- Protocolo de Transferencia de Hipertexto (HTTP): Del servcio web.
- Protocolo de Transferencia de Archivos (FTP): Del servcio de transferencia de archivos.
- Protocolo Simple de Transferencia de Correo (SMTP): Del servcio de correo electrónico.
- Telnet (Red de Telecomunicaciones): Para acceder a una computadora remota.
- Sistema de Nombres de Dominio (DNS) : Para obtener las direcciones IP de los nombres de dominio.
- Protocolo Simple de Administración de Red (SNMP): Permite gestionar la red.
- Protocolo de Configuración Dinámica de Host (DHCP): Gestiona la asignación dinámica de direcciones IP.
- POP (Protocolo de Oficinas de Correos): Gestiona el acceso a correos electrónicos del servicio, permitiendo bajar a una aplicación local.
- Protocolo de Acceso a Mensajes de Internet (IMAP): Gestiona el acceso a correos electrónicos remotamente mediante una aplicación web.

Sistema de Nombres de Dominio (DNS)

Los nombres de dominio son gestionados por el Sistema de Nombres de Dominio, permitiendo traducir los nombres de dominio a direcciones IP que los navegadores necesitan para acceder un sitio web.

El DNS funciona de la siguiente manera:

1. El usuario escribe la dirección de un sitio web en el navegador.
2. El navegador envía una solicitud para obtener la ip al domicio al servidor DNS local.
3. El servidor DNS consulta su caché y si tiene la dirección IP del dominio, la envia inmediatamente.
4. Si el servidor DNS no tiene la ip en la caché, consulta los servidores raíz del DNS, de nivel superior y autoritativos para encontrar la dirección IP.
5. Una vez que el servidor DNS local encuentra la dirección IP, la devuelve a su navegador.
6. El navegador utiliza la dirección IP para acceder al servidor web.

El nivel superior de la jerarquía de nombres de dominio es administrado por la organización ICANN (Del inglés: Internet Corporation for Assigned Names and Numbers), creada en 1998 para organizar el Internet. Los nombres de dominio esta orgnizados en más de 250 dominios de nivel superior, los cuales tienen muchos hosts. Cada una de los dominios superiores se dividen en subdominios de menor gerarquía, los que se dividen en nuevas gerarquias sucesivamente. La gerarquia de los dominios representan como un árbol, como se puede ve en la Figura 9.

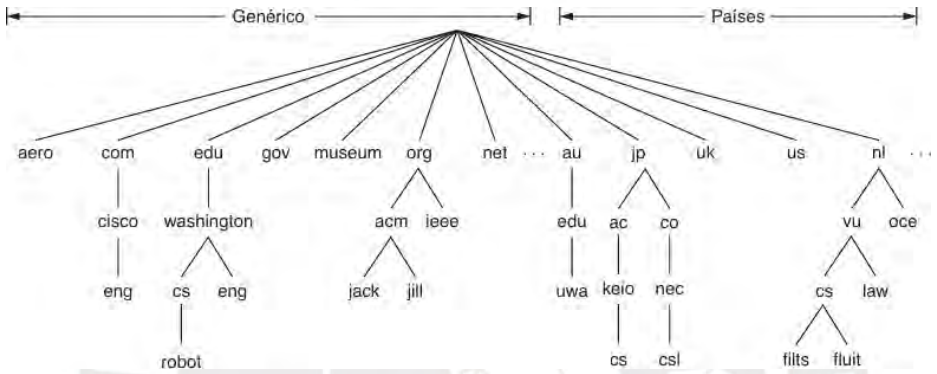


Figura 9. Una porción del espacio de nombres de dominio de Internet. Tomado de "redes de computadoras", A. S. Tanenbaum *(Madrid), 2005*

1.1. Seguridad redes de computadoras

La seguridad en redes informáticas tiene como función asegurar la integridad de los datos, la confidencialidad de los datos y la disponibilidad de los datos en una red. Esto implica el despliege de tecnologías, políticas y procedimientos para prevenir y detectar ciberataques, accesos no autorizados y pérdidas de datos.

La seguridad de las redes incluye:

- Protección de la red: Incluye la protección de dispositivos, servidores, usuarios y datos dentro de la red.
- Control de acceso: Reglas y mecanismos para determinar quién puede acceder a qué recursos de la red.

- Detección de intrusiones: Sistemas y procesos para identificar actividades sospechosas o ataques en curso.
- Prevención de la pérdida de datos: Herramientas y políticas para evitar que datos sensibles sean robados o accedidos de forma no autorizada.
- Gestión de riesgos: Identificar, evaluar y mitigar los riesgos de seguridad en la red

Para establecer medidas de seguridad en las redes se utilizan:

- Firewalls los cuales bloquean el tráfico no autorizado que entra o sale de la red.
- Anti-malware es un software que detecta y elimina software malicioso (virus, ransomware, etc.).
- VPN se encarga de crear conexiones seguras y cifradas entre dispositivos y la red.
- Detección de intrusiones (IDS) se encarga de monitorizar la red en busca de actividad sospechosa.
- Prevención de intrusiones (IPS) se encarga de bloquear el tráfico malicioso detectado por el IDS.
- Cifrado se asegura de proteger la confidencialidad de los datos mientras son transmitidos o almacenados.
- Formación de usuarios, el cual permite capacitar a los usuarios sobre seguridad en la red y cómo reconocer y evitar amenazas.
- Políticas de seguridad son documentos que establecen las reglas y procedimientos de seguridad para la red.
- Auditorías de seguridad consisten en revisiones periódicas de la red para encontrar las vulnerabilidades y correguirlas.

Es importante la seguridad en redes por lo siguiente:

- Protección de datos confidenciales: Evita el robo o la pérdida de información sensible.
- Mantenimiento de la disponibilidad de los servicios: Previene interrupciones en los servicios de la red.
- Cumplimiento normativo: Asegura que se cumplen las reglas y leyes de la seguridad de la información.
- Protección de la reputación: Evita daños ocasionados por un ciberataque a la reputación de una organización.
- Reducción de riesgos financieros: Minimiza las pérdidas financieras causadas por incidentes de seguridad

2.1. Firewall

Un firewall de red es un sistema de seguridad como el que se muestra en la Figura 10, que actúa como una barrera entre redes, controlando el tráfico entrante y saliente de la red. El objetivo es proteger la red de intrusiones externas e internas, aplicando reglas de seguridad basadas en los paramentros de los paquetes para bloquear o dejar pasar el tráfico de red.

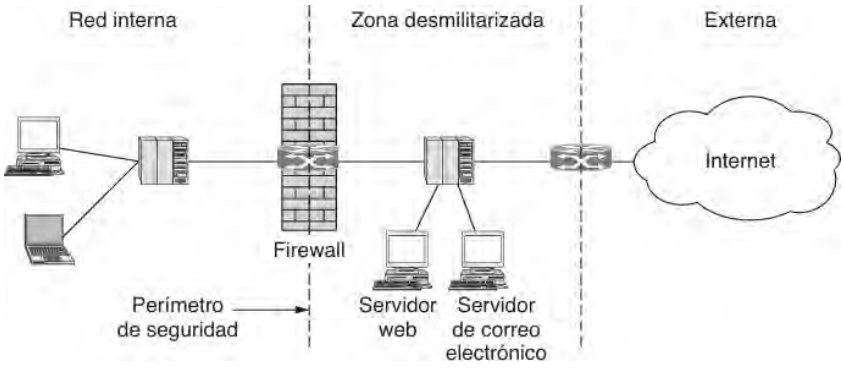


Figura 10. Un firewall que protege a una red interna.

El firewall de red monitorea, filtra y controla el tráfico de red que entra y sale, según las políticas de seguridad establecidas y se implementa en el borde de la red, donde se conecta a otras redes, como Internet. Existen diferentes tipos de firewalls, como firewalls de hardware y software, y firewalls de nueva generación

Analiza el tráfico de red, verifica las características (direcciones IP, protocolos, etc.) y bloquea o permite el tráfico según la política configurada. Reduce el riesgo de ataques cibernéticos, protege la información sensible y ayuda a cumplir con las normativas de seguridad.

2.2. Redes de computadoras

Son multiples computadoras interconectadas entre sí que comparten recursos de hardware y de software para realizar todo tipo de procesos en la vida diaria (Jakub Svoboda, 2015).

2.3. Seguridad de redes

Visto desde la perspectiva en (Marin, 2005) la seguridad de redes se encuentra incluida en la seguridad computacional. Los aspectos de la seguridad en redes incluyen detección de intrusiones en la computadora, análisis de tráfico y monitoreo de la red.

2.4. SIEM

SIEM (Del ingles: Security Information and Event Management). Es una aplicación de software que ha evolucionado para convertirse en un sistema compresivo que permite identificar riesgos en la red y se enfocar proactivamente en mitigar los riesgos para reducir costos de incidencia (Gustavo González-Granadillo, 2021).

2.5. IDS

Sistema que detecta la intrusiones (Del ingles: Intrusion Detection System) es usado para detectar actividades sospechosas y prevenir ataques a la red o sistema (Movva Sai Chaithanya, 2021).

2.6. IPS

Sistema de prevención de intrusiones (Del ingles: Intrusion Prevention System) utiliza firmas para identificar la actividad en el tráfico de la red y realizar la detección de paquetes entrantes y salientes y bloquear esa actividad antes de que dañe y acceda a los recursos de la red (D. Stiawan, 2010).

2.7. Firewall

Un firewall es una tecnología de seguridad que es ampliamente desplegada en redes de computadoras. Esta tecnología es usada para restringir tráfico de red entrante y saliente de acuerdo a las necesidades de seguridad (Watkins, 2020).

2.8. Iptables

Iptables es un aplicación para los administradores de seguridad para configurar las tablas del firewall implmentado en el núcleo Linux, permietiendo agregar reglas de seguridad en las cadenas (Watkins, 2020).

2.9. Linux

Linux es un sistema operativo basado en Unix pero de software libre y de código abierto, que incluye software del proyecto GNU iniciado por en 1983 por Richard Stallman y el nucleo de Linux de Linus Torval quien inicio su desarrollo en 1991 (Watkins, 2020).

2.10. Tráfico de red

El tráfico de red se compone de flujo y formato. El flujo refiere a los protocolos y mensajes de red que viajan a través de los dispositivos. El formato refiere a la estructura de la celda, marcos, paquetes, datagramas y segmentos que comprende el flujo (Marin, 2005).

2.11. Algoritmos bioinspirados

Algoritmos cuya implementación se basa en la forma como funcionan ciertos sistemas naturales, replicando la forma como ésta se optimiza. Imitando la evolución natural de las especies para subsistir en el medio o la respuesta natural de los sistemas sociales a la presencia de nuevos desafíos. Entre ellos esta, la familia de los algoritmos evolutivos, como los algoritmo genéticos, que simulan la evolución de las especies a través de los cromosomas(Movva Sai Chaithanya, 2021).

2.12. PSO

PSO (Del ingles: particle swarm optimization) u optimización por enjambre de partículas, que simula el comportamiento de las manadas de ciertas especies para optimizar una actividad común (Glesias, 2015).

2.13. Redes neuronales artificiales

Es un modelo computacional con una arquitectura fuertemente paralelizada que emula las redes neuronales biológicas de cerebro (Glesias, 2015).

2.13.1. Colonia de abejas

La solución de problemas de automatización se puede realizar mediante el comportamiento inteligente de los enjambres de abejas en la búsqueda de miel, conocido como algoritmo de colonia de abejas inteligente (CAA), desarrollado por Karaboga en 2005 (Movva Sai Chaithanya, 2021).

2.13.2. Logs

Los registros o también conocidos como logs por la palabra en ingles, en informática se refiere a archivos de texto en los que se registra cronológicamente los cambios, actualizaciones y modificaciones que se generan en un sistema informático, servidor, aplicación o programa (H. Shiravi, 2011).

3. Estado del Arte

3.1. Introducción

En este capítulo da cuenta de los resultados obtenidos mediante una revisión sistemática, la cual fue guiada por la metodología propuesta por Kitchenham y Charters (Kitchenham B & Charter S, 2007). La aplicación de este protocolo implicó el desarrollo secuencial de sus tres etapas fundamentales: la planificación de la revisión, su ejecución práctica, y el subsiguiente reporte y análisis de los estudios pertinentes, con el objetivo último de resolver los interrogantes de la investigación. (Kitchenham, 2007)

3.2. Metodología de revisión

La revisión sistemática se realizó utilizando formulando una pregunta, utilizando métodos sistemáticos de identificación, selección y evaluación crítica de la información relevante, que son posibles de reproducir. Recopilando y analizando datos de los estudios obtenidos durante la revisión.

3.3. Objetivos de revisión

- Identificar los trabajos que permitan entender las técnicas de detección de tráfico anómalo usando algoritmos bioinspirados para configurar el firewall, que sirvan como base para la elaboración del Estado del Arte.
- Conocer los conceptos respecto a la detección de tráfico anómalo, específicamente al firewall aplicado en este marco de trabajo.
- Conocer cómo se adapta el uso de algoritmos bioinspirados a la detección de tráfico anómalo.

3.4. Preguntas de revisión

Durante la revisión, para identificar las técnicas de detección de tráfico anómalo usando algoritmos bioinspirados para configurar el firewall, se formularon las preguntas:

P1. ¿Cuáles son las causas de que la información en las organizaciones sea vulnerable?

P2. ¿Qué soluciones se están dando al problema de seguridad de la información en las organizaciones?

P3. ¿Qué características y conceptos se han investigado acerca de la forma de asegurar la información en las organizaciones?

P4. ¿Qué algoritmos bioinspirados se están utilizando para la solución del problema de seguridad de la información en las organizaciones y cómo se han evaluado, comparado y verificado?

3.5. Estrategia de búsqueda

3.5.1. Motores de búsqueda

En la revisión bibliográfica se utilizó literatura especializada sobre ciberseguridad de las bases de datos proporcionadas por la PUCP. Asimismo, estas fueron recomendadas por los asesores de tesis. Los motores de búsqueda utilizados fueron:

- Google Académico
- SCOPUS
- IEEE Xplore

3.5.2. Cadenas de búsqueda

Considerando que la literatura especializada esta en ingles, las cadenas de búsqueda se formularon mediante palabras claves en ingles, las cuales se muestran en la tabla 4.

Tabla 9. Palabras clave

Concepto	**Términos relacionados**
C1: Firewall	packet filter / circuit-level gateway / intrusion prevention system

C2: Firewall logs	packet filter logs
C3: Abnormal network traffic	Anomalous network traffic / Unusual network traffic / network intrusion detection
C4: bio-inspired algorithms	immunological algorithms / swarm intelligence / evolutionary algorithms / neural networks / genetic algoritms
C5: Anomaly clasification	anomalous traffic detection

La cadena de búsqueda general incluye los conceptos de la tabla 4, de la siguiente forma:

C1 **AND** C2 **AND** C3 **AND** C4 **AND** C5

Particularmente para cada base de datos seleccionada se formula cadena de búsqueda de acuerdo las recomendaciones.

- **GOOGLE ACADÉMICO**

(firewall OR packet filter OR circuit-level gateway OR intrusion prevention system) AND (firewall logs OR packet filter logs) AND (Abnormal network traffic OR Anomalous network traffic OR Unusual network traffic) AND (bio-inspired algorithms OR immunological algorithms OR swarm intelligence OR evolutionary algorithms OR neural networks) AND (detection of abnormal traffic OR anomalous traffic detection)

- **SCOPUS**

TITLE-ABS-KEY((firewall OR packet filter OR circuit-level gateway OR intrusion prevention system) AND (firewall logs OR packet filter logs) AND (Abnormal network traffic OR Anomalous network traffic OR Unusual network traffic) AND (bio-inspired algorithms OR immunological algorithms OR swarm intelligence OR evolutionary algorithms OR neural networks) AND (detection of abnormal traffic OR anomalous traffic detection))

- **IEEExplore**

((firewall OR packet filter OR circuit-level gateway OR intrusion prevention system) AND (firewall logs OR packet filter logs) AND (Abnormal network traffic OR Anomalous network traffic OR Unusual network traffic) AND (bio-inspired algorithms OR immunological algorithms OR swarm intelligence OR evolutionary algorithms OR neural networks) AND (detection of abnormal traffic OR anomalous traffic detection))

3.5.3. Articulos científicos encontrados

Se obtuvieron 43 resultados como resultado de la búsqueda en las bases de datos selecionadas, en la Tabla 5 se meustran la cantidad de artículos por buscador y en la tabla 6 se presenta los artículos.

Tabla 10: Resultados del Proceso de Búsqueda Sistemática

Motor de búsqueda	**Resultados de la Búsqueda**
GOOGLE ACADÉMICO	25
SCOPUS	10
IEEE Xplore	8
Total	**43**

3.5.4. Criterios de inclusion y de exclusión

- **Criterios de Inclusión**
  - El trabajo trata un tema relacionado a la seguridad de información en las organizaciones (CI1).
  - La investigacion presenta el tema del problema de seguridad de la información con las herramientas tradicionales de ciberseguridad (CI2).
  - La investigación presenta comparaciones metodológicas de la eficiencia de las herramientas usadas para el problema de seguridad de la información (CI3).
- **Criterios de Exclusión**

- La investigación presenta un análisis correlacional de la seguridad de información en las organizaciones, más no una propuesta con algoritmos bioinspirados para detectar problemas de seguridad de la información (CE1).
- El documento no esta redactado en idioma ingles o español, que son los idiomas seleccionados para el presente trabajo por el desconocimiento de otros idiomas. (CE2).
- La investigación fue publicada hace mas de 10 años, ya que se requiere los últimos avances desarrollados sobre el tema (CE3).
- El trabajo no se relaciona no aborda aspectos de la informática y/o ciencias de la computación (CE4).

La investigación no se refiere a temas relacionados con lo que se desea desarrollar. (CE5).

Tabla 11. Artículos relevantes para la revisión sistemática

ID	Título	Autor	Año de publicación
S1	Anomaly Classification Using Genetic Algorithm-Based Random Forest Model for Network Attack Detection	Adel Assiri	2020
S2	Bio-Inspired, Host-based Firewall	Lanier Watkins, James Ballard, Kevin Hamilton, Jay Chow, Aviel Rubin, William H. Robinson, Cleon Davis	2020
S3	Feature selection using cloud-based parallel genetic algorithm for intrusion detection data classification	Dzelila Mehanovic, Dino Keco, Jasmin Kevric, Samed Jukic, Adnan Miljkovic, Zerina Masetic	2020

S4	New hybrid method for attack detection using combination of evolutionary algorithms, SVM, and ANN	Soodeh Hosseini, Behnam Mohammad Hasani Zade	2020
S5	A Hybrid Model Using Bio-Inspired Metaheuristic Algorithms for Network Intrusion Detection System	Omar Almomani	2021
S6	Intelligent IDS: Venus Fly-trap Optimization with Honeypot Approach for Intrusion Detection and Prevention	Movva Sai Chaithanya, Suresh Nikudiya, Varsha S Basanaik, Damodar Reddy, Hanumanthu Bhukya	2021
S7	Using A Hybrid Algorithm and Feature Selection for Network Anomaly Intrusion Detection	Ali Hussein Shamman Al-Safi, Zaid Ibrahim Rasool Hani, Musaddak M. Abdul Zahra	2021
S8	A Hybrid Evolutionary Algorithm for Anomaly Detection in Computer Networks	Reza Rafeh, Mehdi Naja	2022
S9	Automatic Building of a Powerful IDS for The Cloud Based on Deep Neural Network by Using a Novel Combination of Simulated Annealing Algorithm and Improved Self-Adaptive Genetic Algorithm	Zouhair Chiba, Moulay Seddiq El Kasmi Alaoui, Noreddine Abghour, Khalid Moussaid	2022
S10	Defense against distributed DoS attack detection by using intelligent evolutionary algorithm	Shubhra Dwivedi, Manu Vardhan, Sarsij Tripathi	2022

3.6. Extracción de datos

Para respoder a las preguntas de la revisión se debe realizar la extracción de datos incluyendo toda la información necesaria, para lo cual se plantea un formulario con la estructura de la Tabla 7.

Tabla 12: Formulario de extracción de datos

Información general	**Valor**
Identificador del trabajo	Identificador único asigando a los trabajos seleccionados por su relevancia.
Título	Aquí se incluye el título del paper
Autor(es)	Aquí se incluyen los autores del paper
Año de publicación	Aquí se especifica el año de publicación del paper
Idioma	Aquí se especifica el idioma utilizado en el paper (inglés/español)
Motor de búsqueda	Google Académico / Scopus / IEEE Xplore
Link de consulta	Aquí se incluye el link de acceso al paper
Abstract	Aquí se incluye el abstract completo del paper
Información a extraer para contestar las preguntas	
Información a extraer	Preguntas que responde
Causas de que la información en las organizaciones sea vulnerable.	P1
Soluciones propuestas al problema de seguridad de la información en las organizaciones.	P2
Nuevas teorías desarrolladas para asegurar la información en las organizaciones.	P3
Algoritmos bioinspirados que se están aplicando para resolver los problemas de seguridad de la información en las organizaciones y estrategias para evaluar, comparar y verificar los resultados.	P4

3.7. Resultados de la revision bibliográfica

3.7.1. Respuesta de la pregunta 1

Las respuestas obtenidas de la literatura para la pregunta ¿Cuáles son las causas de que la información en las organizaciones sea vulnerable?, se muestra en la tabla 8.

Tabla 13: Causas de que la información en las organizaciones sea vulnerable.

ID	**Causas de que la información en las organizaciones sea vulnerable**
S2	Los firewalls tradicionales requieren nuevos métodos para detectar y proteger las redes.

S3	La gran cantidad de características de datos de tráfico de intrusiones dificulta que los algoritmo de aprendizaje de maquina tengan un alto desempeño.
S6	Los sistemas tradicionales detectan mediante firmas de ataques previos o mediante la detección de trafico anómalo.
S10	Ataques de denegación del servicio distribuido causado para miles de computadoras.

3.7.2. Respuesta de la pregunta 2

En la tabla 9 se presenta las respuestas a la pregunta ¿Qué soluciones se están dando al problema de seguridad de la información en las organizaciones?

Tabla 14: Soluciones que se están dando al problema de seguridad de la información en las organizaciones.

ID	Soluciones propuestas al problema de seguridad de la información en las organizaciones
S1	Un clasificador de anomalías basado en el tráfico de la red para monitorear y detectar los ataques de intrusión a la red utilizando algoritmos genéticos para optimizar los para metros del método de optimización Randon Forest.
S2	Utilización de Limux IP Tables firewall modificado.
S3	Selección de las características de los datos del tráfico para mejorar los datos de detección de intrusiones.
S4	Nuevo método híbrido en dos fases, una fase de selección de características con space vector machine y la fase de detección de intrusiones con redes neuronales artificiales.
S5	Nuevo método híbrido en dos fases, una fase de selección de características con particle swarm optimization y la fase de detección de intrusiones con space vector machine.
S9	Detección de intrusiones reduciendo las falas alarmas.

3.7.3. Respuesta de la pregunta 3

A la pregunta ¿Qué características y conceptos se han investigado acerca de la forma de asegurar la información en las organizaciones?, se obtuvieron las respuestas presentadas en la tabla 10.

Tabla 14: Características y conceptos que se han investigado acerca de la forma de asegurar la información en las organizaciones.

ID	Características y conceptos que se han investigado acerca de la forma de asegurar la información en las organizaciones
S2	Uso de aprendizaje de máquina para filtrar paquetes IP.
S3	Selección de características aplicando algoritmos genéticos para la clasificación de datos de detección de intrusión

S4	Métodos con dos fases, una fase de selección de características de tráfico de intrusión y otra fase de detección de la intrusión.
S5	Métodos con dos fases, una fase de selección de características de tráfico de intrusión y otra fase de detección de la intrusión.
S9	Detección de intrusiones reduciendo las falsas alarmas mediante aprendizaje profundo de redes neuronales artificiales y algoritmos genético auto- adaptativo.

3.7.4. Respuesta de la pregunta 4

La pregunta ¿Qué algoritmos bioinspirados se están utilizando para la solución del problema de seguridad de la información en las organizaciones y cómo se han evaluado, comparado y verificado?, tiene las respuestas mostradas en la Figura 11.

Tabla 15. Algoritmos bioinspirados que se están utilizando para la solución del problema de seguridad de la información en las organizaciones y cómo se han evaluado, comparado y verificado.

ID	**Algoritmos bioinspirados que se están utilizando para la solución del problema de seguridad de la información en las organizaciones y cómo se han evaluado, comparado y verificado**
S1	Algoritmos genéticos
S3	Algoritmos genéticos
S4	Space vector machine y redes neuronales artificiales.
S5	Particle swarm optimzation y Space vector machine.
S7	Algoritmo de colonia de abejas y space vector machine.
S8	Algoritmo competitivo imperialista y algoritmo genético.
S9	Redes neuronales artificiales con aprendizaje profundo y algoritmos genético auto-adaptativo

3.8. Conclusiones

Los sistemas de seguridad basados en firewalls no protegen de nuevos ataques por lo que requieren métodos para detectar las intrusiones.

Para la detección de intrusiones al ser un problema de reconocimiento de patrones de características en el tráfico se utiliza algoritmos de aprendizaje de máquina como RNA.

4. Implementación del sistema de seguridad centralizado en el firewall de la red que se desea asegurar

4.1. Introducción

La seguridad debe centralizarse en un sistema que pueda mantener el control del tráfico de red que ingresa y que sale de la red local, para lo cual se debe instalar el servidor firewall entre la red interna y externa mediante dos tarjetas de red que separan físicamente las dos redes.

El propósito es establecer un control completo sobre el tráfico de la red mediante la implementación de un sistema de seguridad centralizado en el firewall, gestionando así todo el trafico entrante y saliente de la red.

El sistema de seguridad centralizado se implementó ensamblando un servidor Core I7, con 32 GB RAM, un dispositivo de almacenamiento DSS de 1 TB y dos tarjetas de red que permitieron separar físicamente la red interna de internet para tener control total sobre el tráfico, con sistema operativo Linux Fedora con soporte de bases de datos PostgreSQL, servidor web Apache con php e iptables para la manipulación de los paquetes IP entrantes y salientes.

El objetivo se alcanzó en su totalidad lográndose implementar el servidor físico conectado de tal forma que separa físicamente la red interna de internet y tiene control total de tráfico de la red, con las herramientas de software necesarias para la captura de registros ip, manipulación de tráfico entrante y saliente, procesamiento de registros de tráfico.

4.2. Resultados alcanzados

4.2.1. Servidor firewall instalado con la configuración física de tal forma que el tráfico de la red pase a través de ese servidor

4.2.1.1. Ensamblado del servidor

Las pruebas del proyecto se realizarán en un prototipo implementado en un servidor con sistema operativo Linux para capturar, almacenar y procesar registros de paquetes ip, correr el programa de algoritmos de aprendizaje de nuevos patrones de ataque, ejecutar el programa de detección de intrusiones y el programa de configuración automática del firewall. Para dar soporte a los procesos del sistema se ensambla un servidor con procesador Core I7, 32 GB de RAM y un dispositivo de almacenmaiento DSS de 1 TB.

En la Figura 11 se presenta la instalación de la placa y el procesador, en la Figura 12 se presenta la instalación de dos bancos de 16 MB de memoria RAM y el ventilador del procesador.

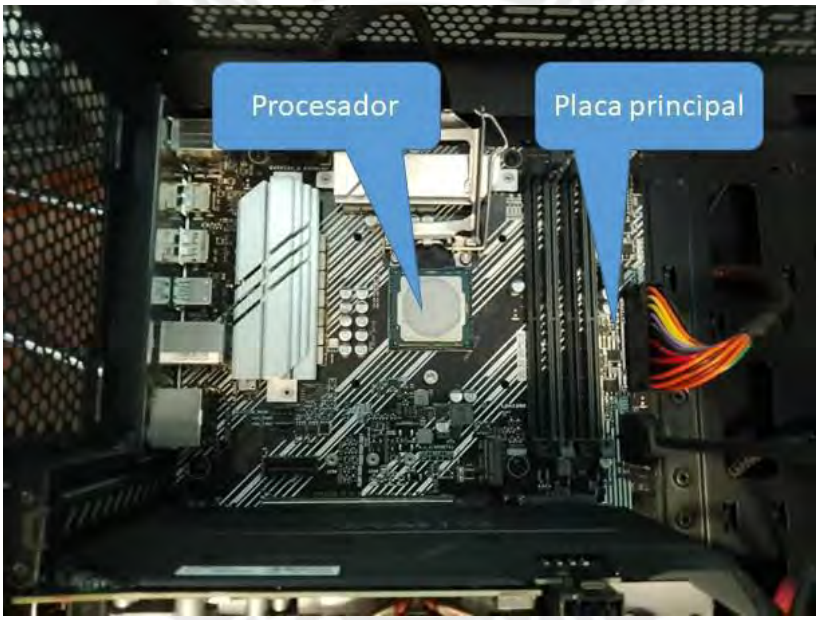


Figura 11. Instalación de placa principal y procesador.

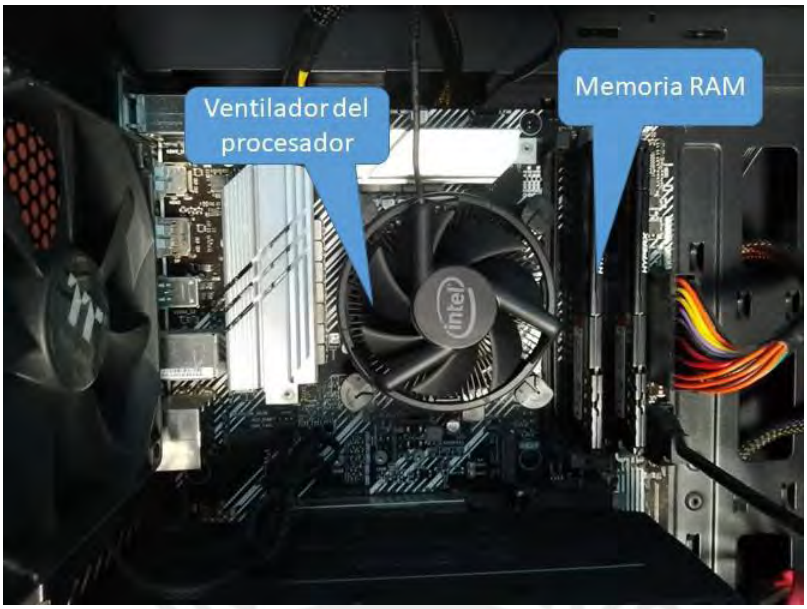


Figura 12. Instalación de memoria RAM.

La instalación de del disco de estado sólido SATA de 1 TB se presenta en la Figura 13, para separar físicamente la red interna con la red externa en este caso internet se instala una tarjeta de red adicional que se presenta en la Figura 14, además de la que viene en la placa principal.

Figura 13. Instalación de disco de estado sólido.

Figura 14. Instalación de tarjeta de red adicional.

La Figura 15 muestra la vista posterior donde se puede ver las dos tarjetas de red necesarias en el firewall para separar la red externa o pública de la red privada o local. La Figura 16 presenta la vista frontal del servidor ensamblado.

Figura 15. Vista posterior de servidor con dos interfaces de red.

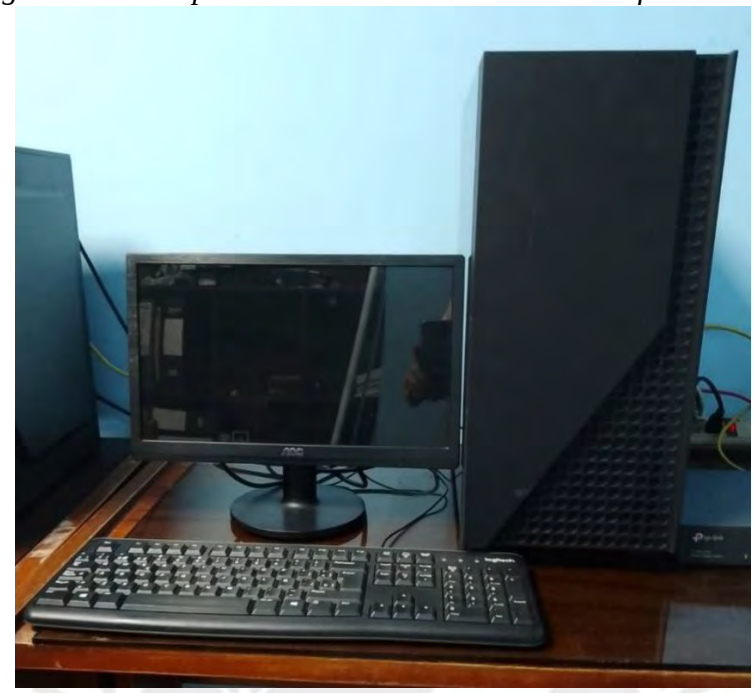


Figura 16. Vista frontal de servidor.

4.2.1.2. Instalación física del servidor con las conexiones a red

La conexión del servidor en la red se realiza como esta presentado en la Figura 17, donde la tarjeta de red interna se usa para conectar la red local protegida y con la tarjeta de red externa se usa para conectar la red externa (internet). En la red interna se conectará cuatro usuarios de con computadoras, un servidor web, servidor de correo electrónico, cinco usuarios de celulares, cuatro usuarios de televisores inteligentes haciendo un total de 13 usuarios y un servidor con servicios de internet.

Utilizando la conexión mostrada en la Figura 17 todo el tráfico de la red externa e interna pasa por el servidor firewall, dando la posibilidad de capturar los registros de los paquetes IP de tráfico, para analizarlos y también permite controlar el tráfico mediante la configuración del firewall.

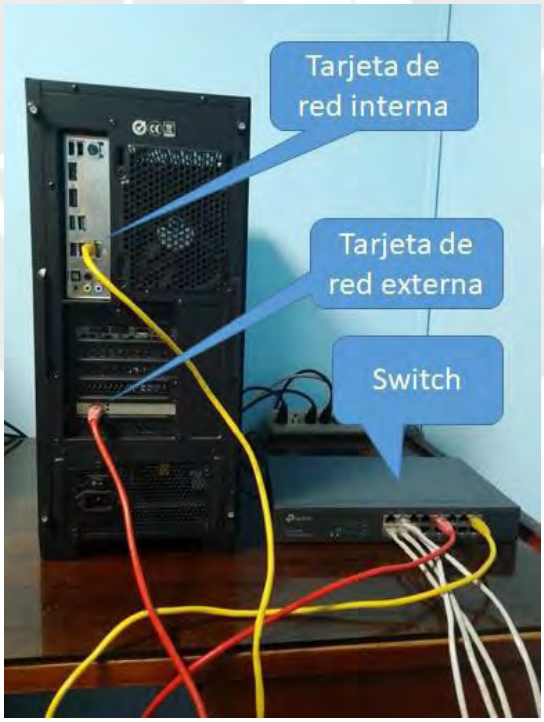


Figura 17. Conexión a red del servidor.

4.2.2. Servidor firewall con el software instalado y configurado

4.2.2.1. Instalación de sistema operativo Linux Fedora

El sistema operativo Linux Fedora es instalado mediante una USB de arranque que se ve en la Figura 18.

Figura 18. Instalación de USB de arranque de Linux.

Después del proceso de inicialización, la computadora presenta la pantalla que se observa en la Figura 19, donde se debe seleccionar “Instalación de Linux” e inicia el cargado de los programas de instalación que se muestran e la Figura 20.

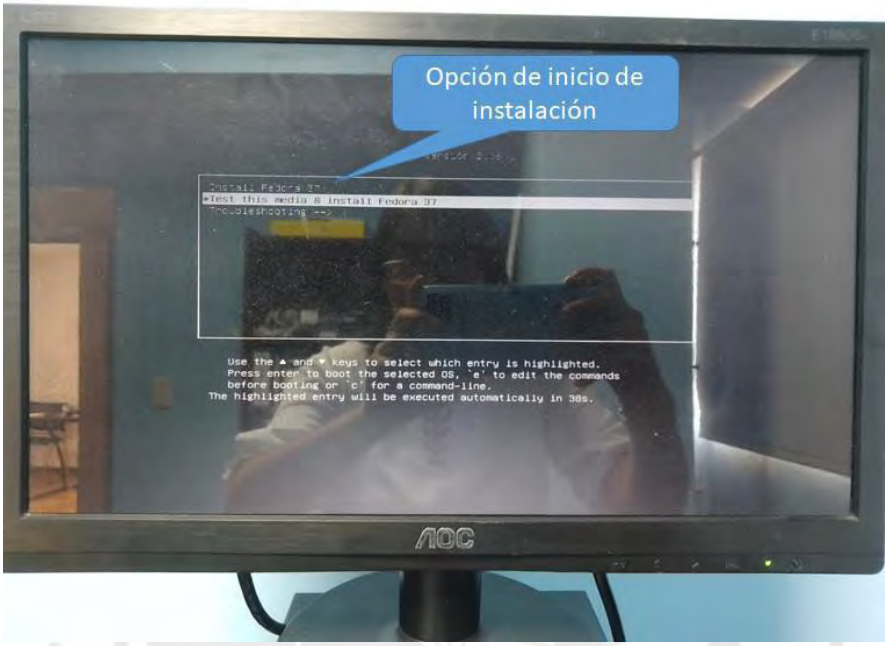


Figura 19. Inicio de instalación de Linux.

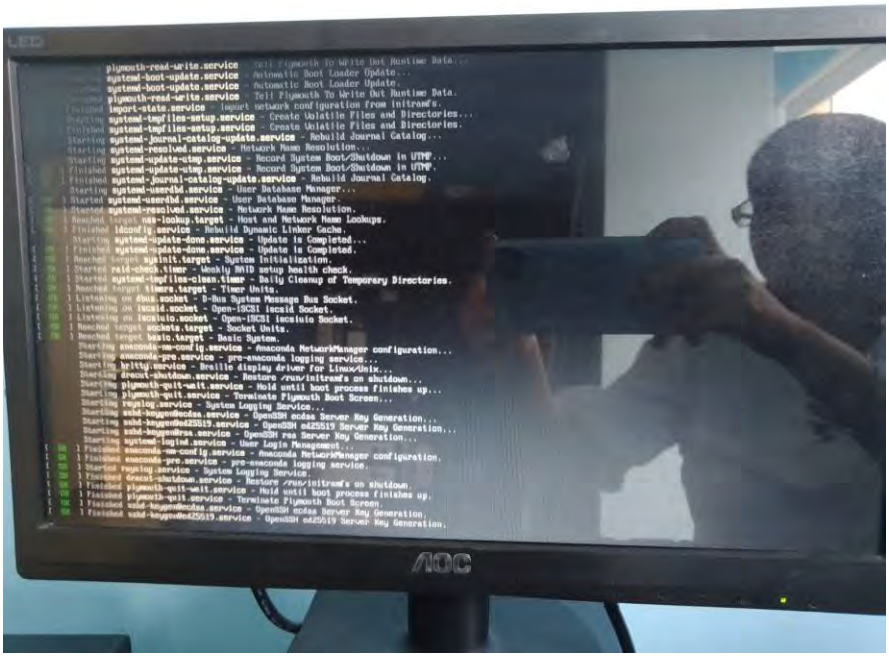


Figura 20. Cargado de programas de instalación de Linux.

Cuando finaliza el cargado de los programas de instalación de Linux se presenta una pantalla para configurar la instalación mostrada en la Figura 21, en donde se selecciona el dispositivo de almacenamiento que presenta la Figura 22, se ingresa un número IP de la red interna manualmente como se muestra en la Figura 23.

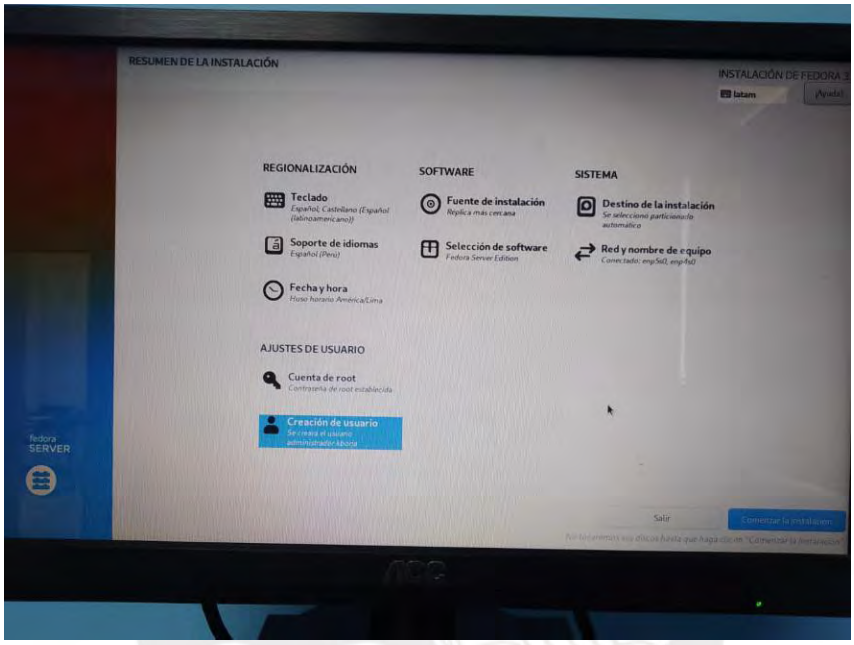


Figura 21. Pantalla de configuración de instalación de Linux.

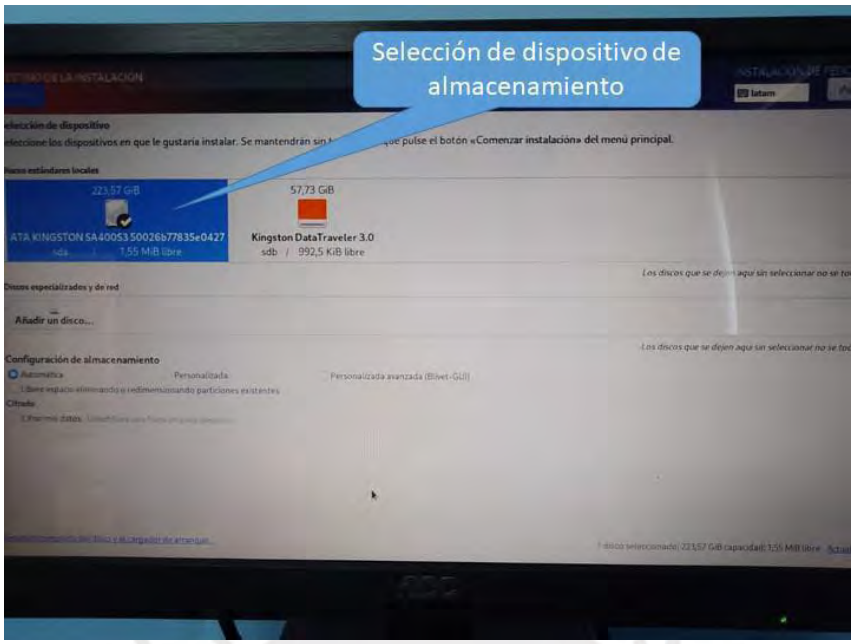


Figura 22. Selección de dispositivo de almacenamiento.

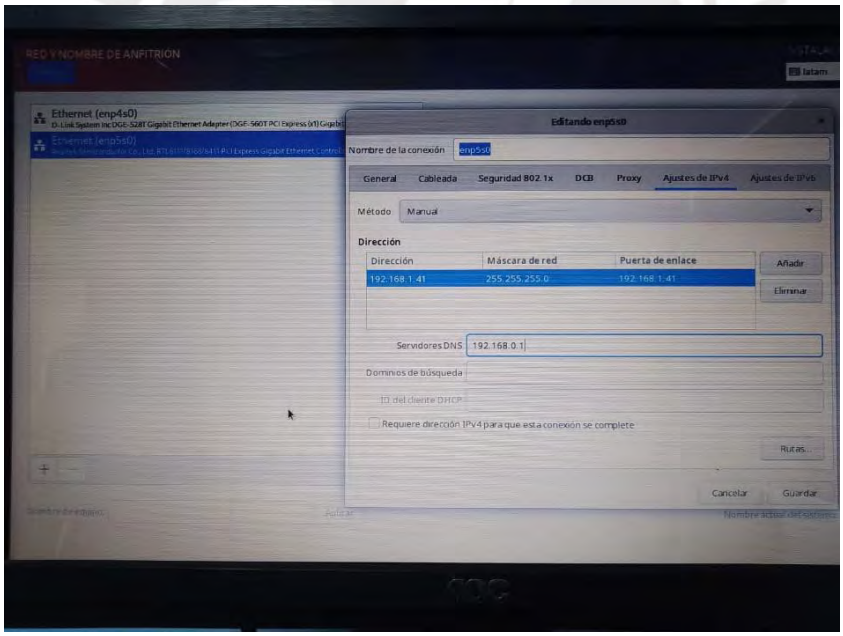


Figura 23. Configuración de IP privada.

La Figura 24 presenta la IP pública asignada, que por esquema de configuración del proveedor internet, se realizar mediante DHCP.

Con el fin de facilitar la operación posterior del sistema operativo se activa el acceso remoto con el super usuario root y se asigna una clave como se presenta en la Figura 25, además se agrega un usuario adicional para operaciones como usuario común en la pantalla presentada en la Figura 26, también se selecciona el software a instalar en la interface presentada en la Figura 27, en este caso se elige la edición de servidor que incluye los paquetes de servidor necesarios para el sistema y se minimiza el uso de recursos con programas y servicios que no se aplican a este caso.

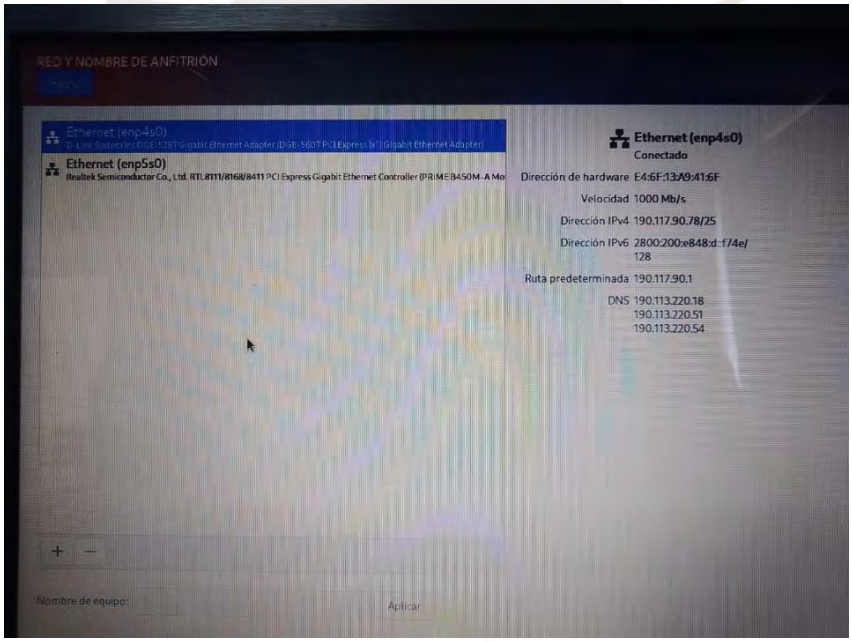


Figura 24. Configuración de IP pública.

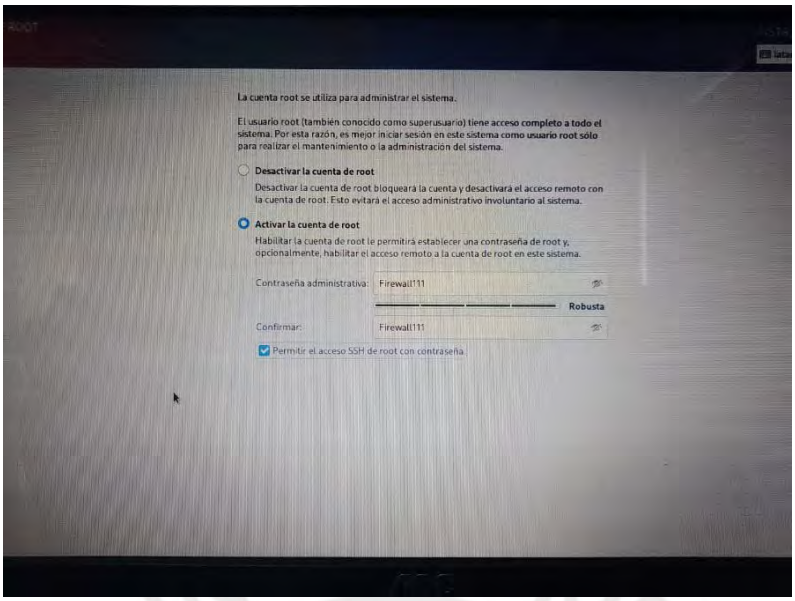


Figura 25. Activar cuenta de root.

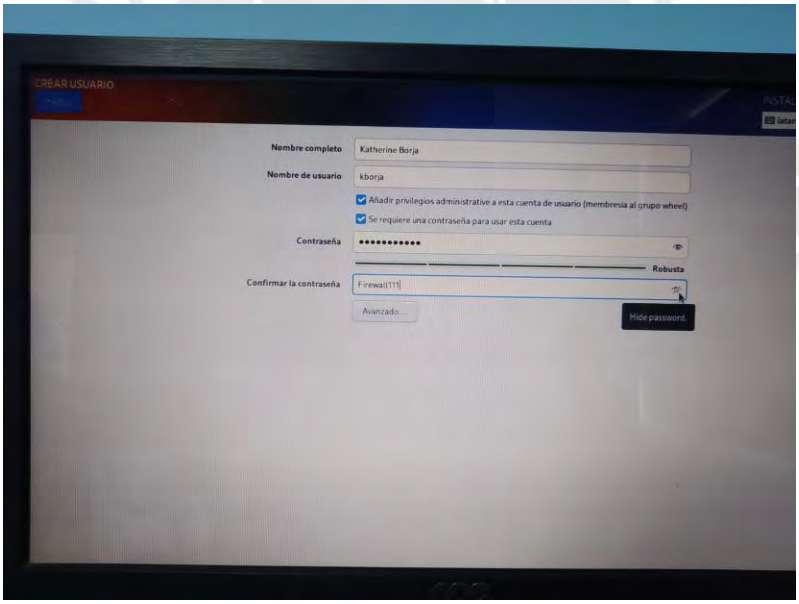


Figura 26. Creación de una cuenta de usuario adicional.

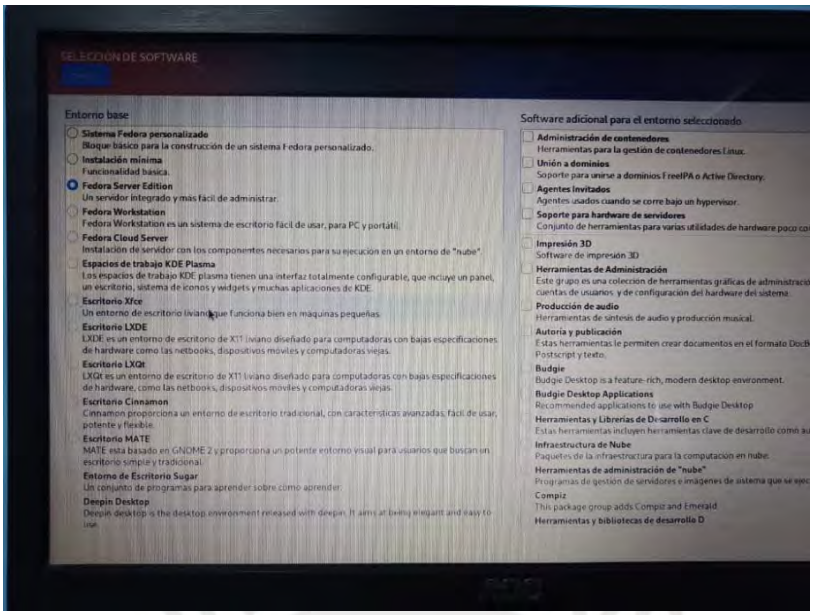


Figura 27. Selección de software para la instalación.

Concluyendo la configuración de la instalación se procede con el proceso de instalación que se presenta en la Figura 28, con una duración de 5 minutos, finalizando en la pantalla presentda en la Figura 29, con una opción para reiniciar el sistema y finalmente presentando la pantalla para ingresar al sistema mostrado en la Figura 30.

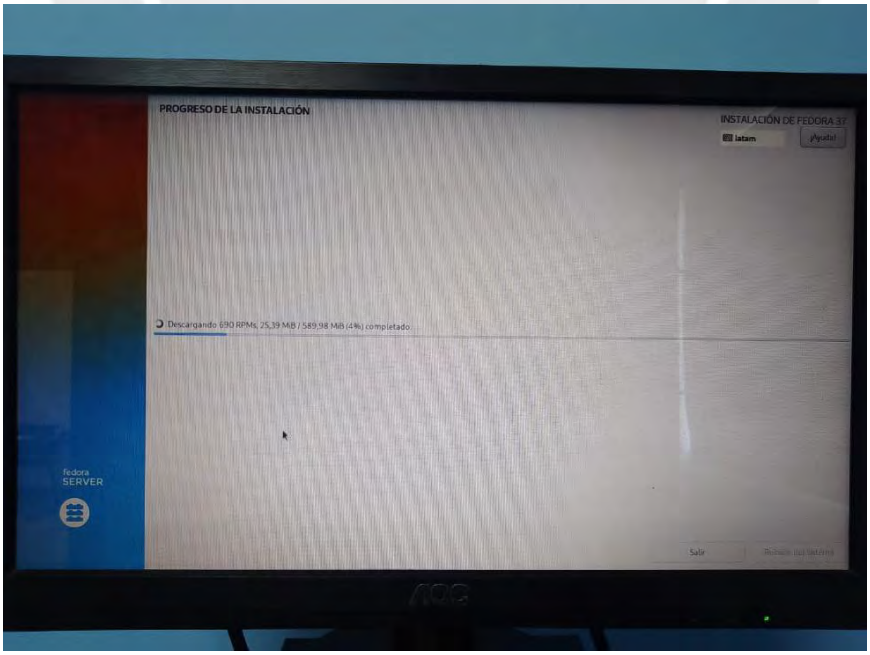


Figura 28. Proceso de instalación de Linux.

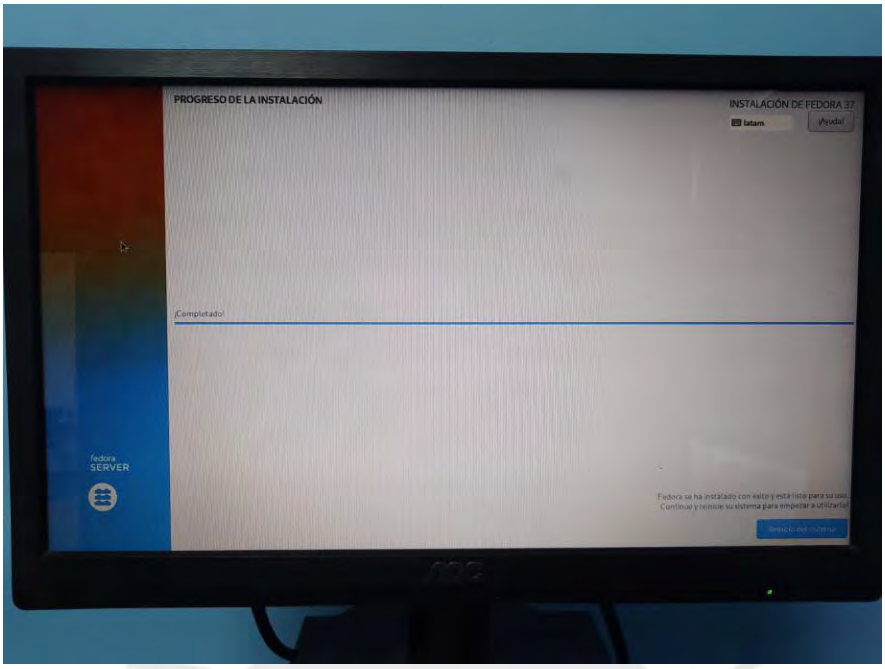


Figura 29. Fin instalación de Linux.

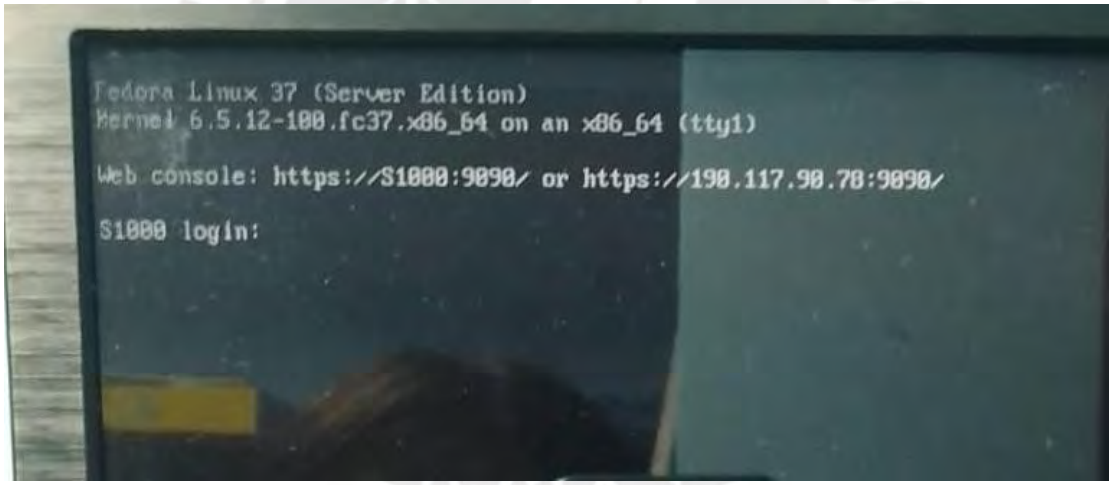


Figura 30. Pantalla de ingreso al sistema.

Ingresando al sistema, se puede operar el sistema en modo texto utilizando el teclado y un monitor conectada a la computadora en la pantalla presentado en la Figura 31. O activando el acceso remoto SSH, se puede acceder con la aplicación para Windows putty como se presenta en la Figura 32, con una IP externa y con el usuario root accediendo a una pantalla como la presentada en la Figura 33, se accede al sistema para operar remotamente en calidad de super usuario en una pantalla como se presenta en la Figura 34.

Figura 31. Pantalla de operación del sistema en modo texto.

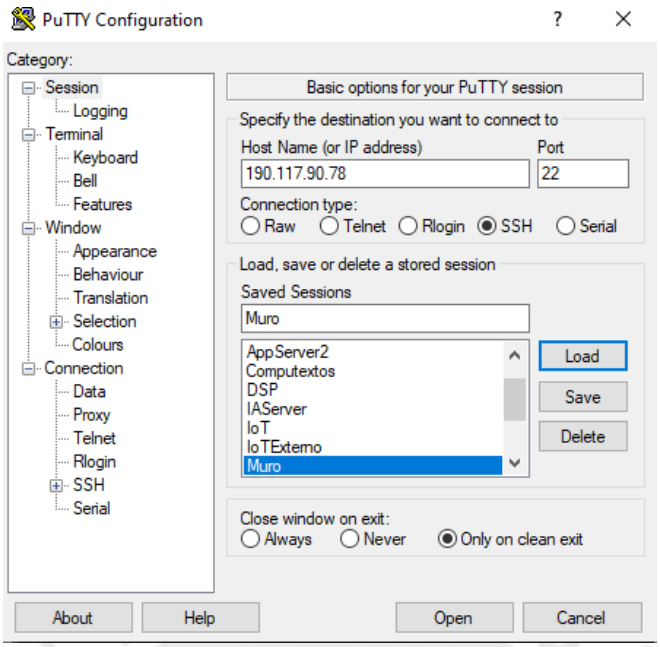


Figura 32. Acceso remoto al servidor Linux.

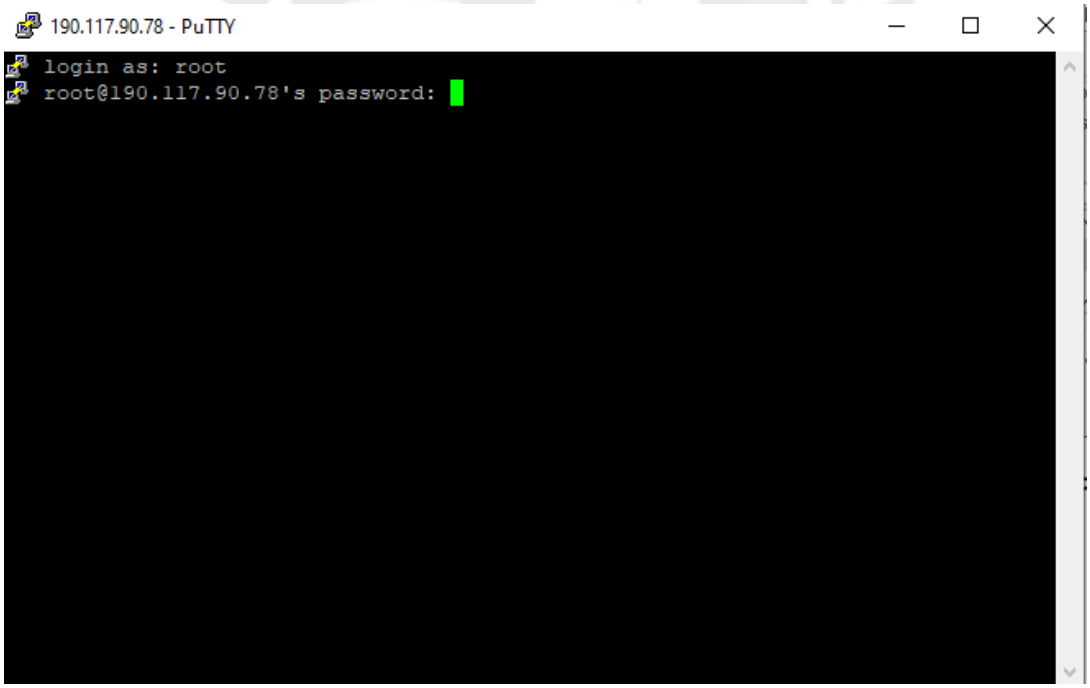


Figura 33. Ingreso de usuario y clave.

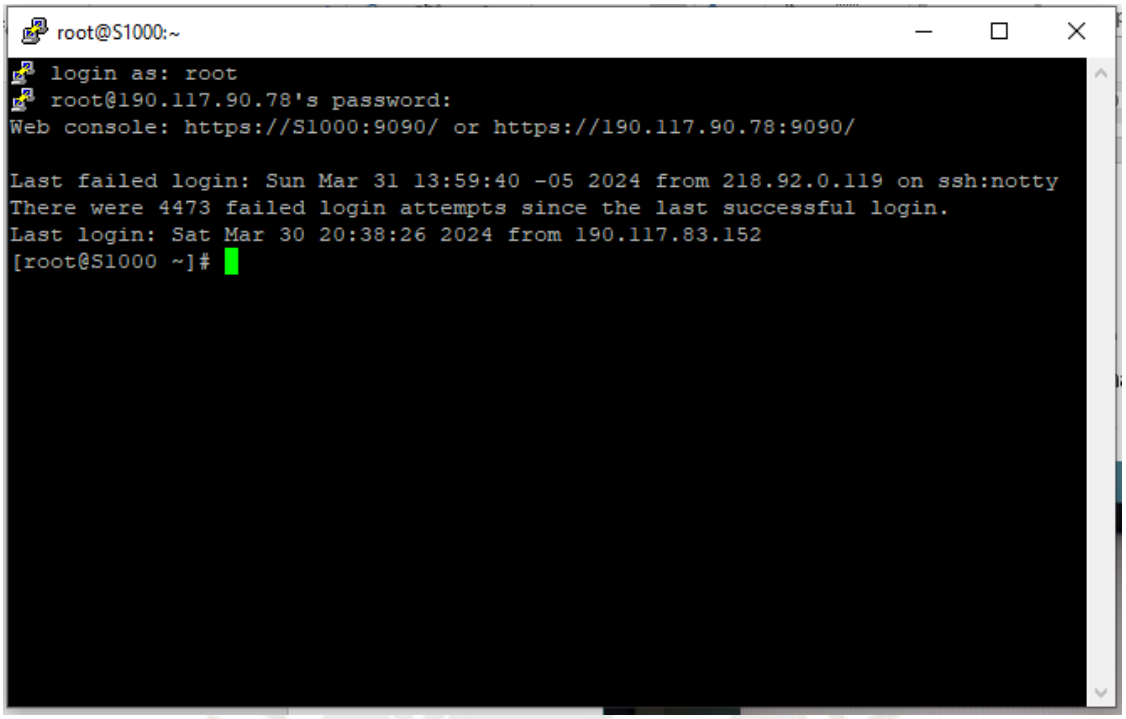


Figura 34. Pantalla de operación remota de Linux.

4.2.2.2. Configuración de base de datos PostgreSQL

La instalación y configuración de PostgreSQL se realiza con el manual de (Server, PostgreSQL 14 : Install, 2022) y en la Figura 35 se presenta los comando de instalación, inicialización de la base de datos y activación de PostgreSQL.

```
[root@www ~]# dnf module -y install postgresql:14/server

[root@www ~]# postgresql-setup --initdb
 * Initializing database in '/var/lib/pgsql/data'
 * Initialized, logs are in /var/lib/pgsql/initdb_postgresql.log

[root@www ~]# systemctl enable --now postgresql
```

Figura 35. Instalación de PostgreSQL.

La dirección y el puerto de acceso al servidor PostgreSQL se realiza en el archivo */var/lib/pgsql/data/pg_hba.conf* como se muestra en la Figura 36, donde la dirección privado de escucha es el 192.168.1.41 y la IP pública es el 190.117.90.78.

La configuración de acceso remoto de usuarios se realiza en el archivo */var/lib/pgsql/data/PostgreSQL.conf* como se muestra en la Figura 37, donde se puede agregar las redes que accederán al servidor o los ips de cada computadora.

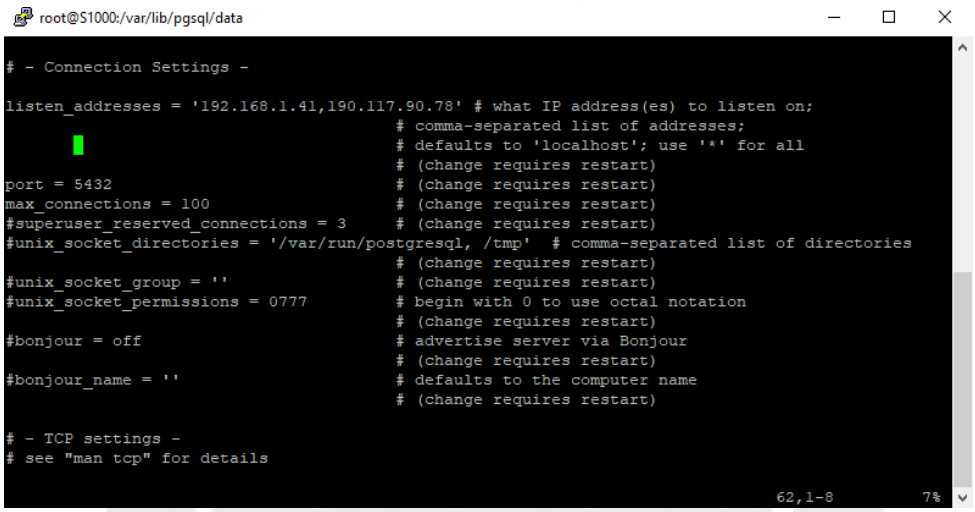


Figura 36. *Configuración de puerto e ip.*

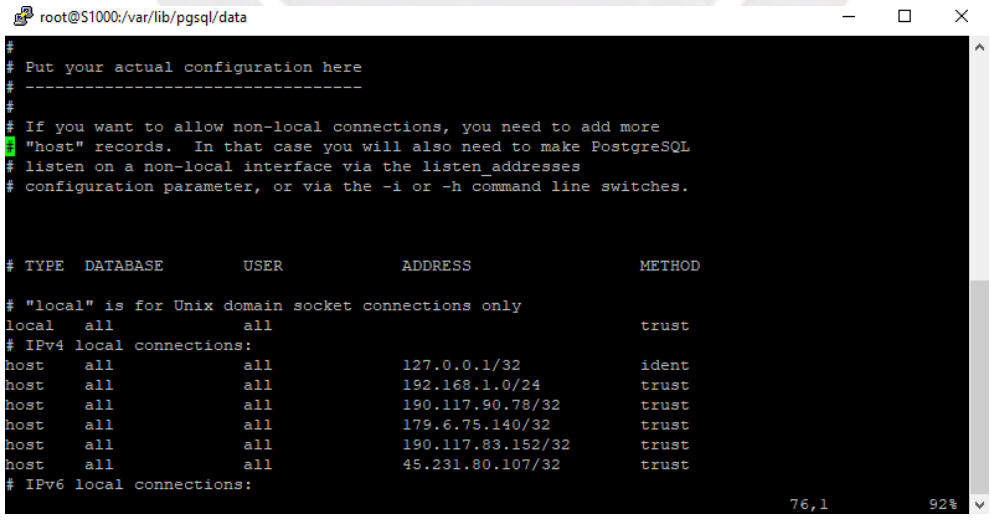


Figura 37. Configuracion de acceso remoto de usuarios.

La operación de PostgreSQL se realizó utilizando el cliente de Windows DBeaver, que se configura el acceso de acuerdo a la Figura 38 y la Figura 39 presenta la prueba de conexión. En la Figura 40 se ve la base de datos S1000 de sistema y se lista datos de logs.

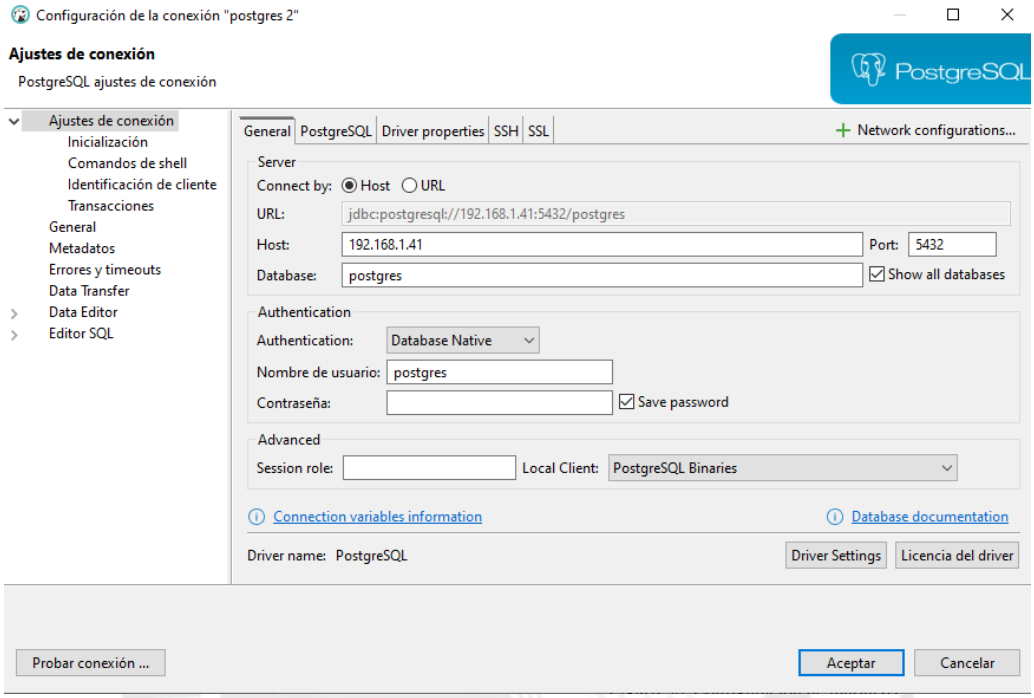


Figura 38. Configuración de acceso desde el cliente de Windows DBeaver.

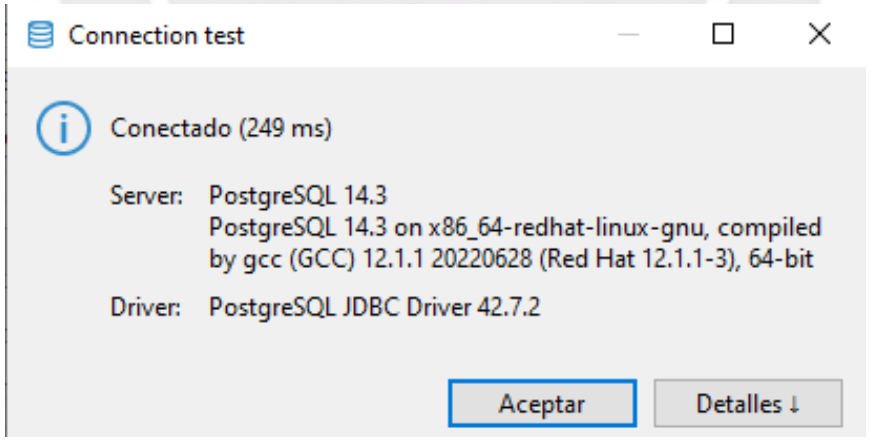


Figura 39. Prueba de acceso al servidor PostgreSQL.

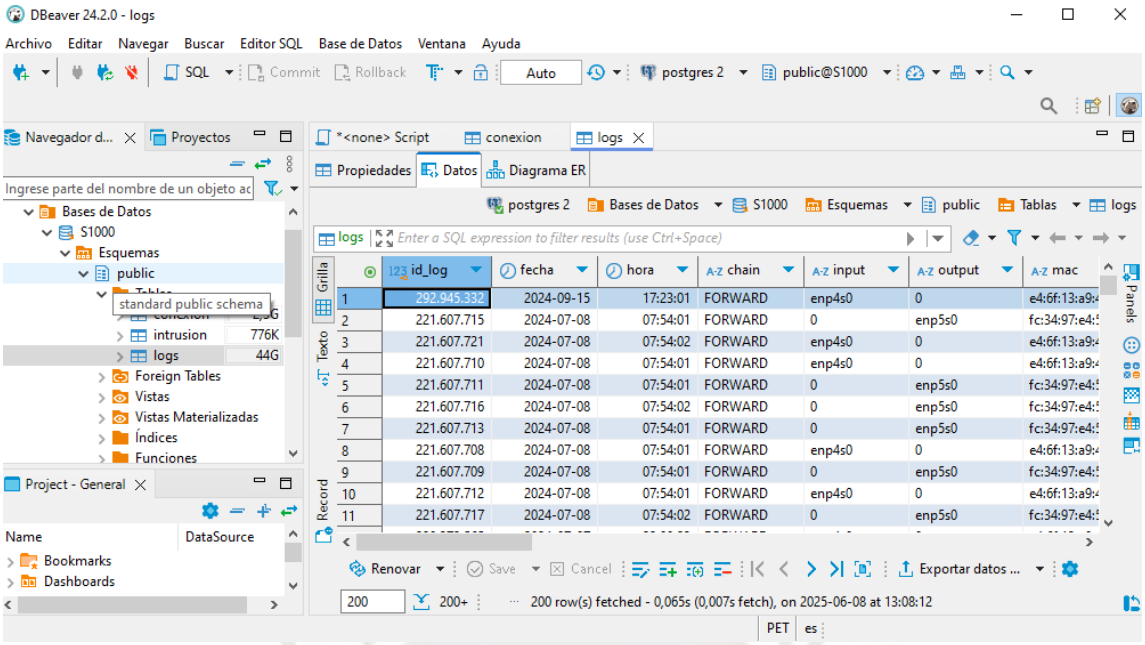


Figura 40. Prueba de acceso al servidor mariaDB desde windows.

4.2.2.3. Configuración de servidor web Apache

La instalación y configuración de servidor web apache se realiza con el manual de (Server, Install Apache httpd, 2022) con el comando dnf -y install httpd se realiza la instalación y el resultado se presenta en la Figura 41 y la Figura 42 muestra el acceso desde un navegador para Windows.

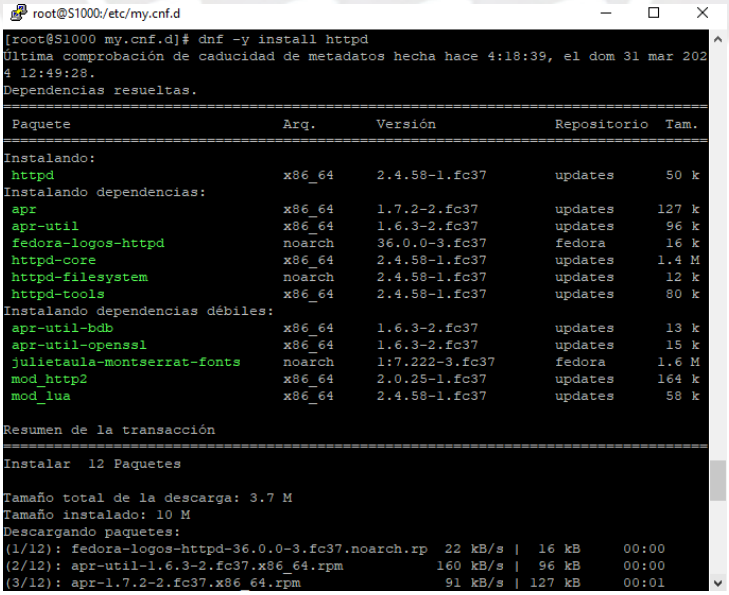


Figura 41. Instalación de servidor web apache.

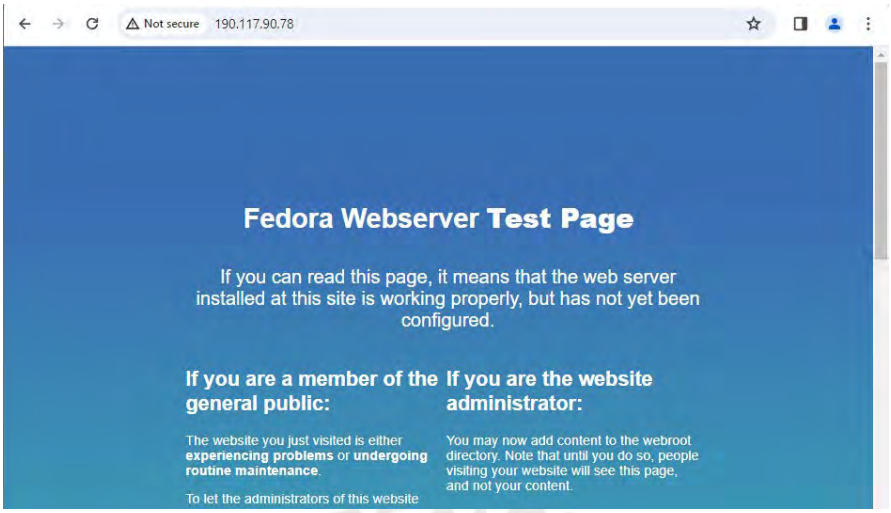


Figura 42. Acceso al servidor web desde un navegador de Windows.

4.2.2.4. Instalación de php y configuración de php 8

La instalación de php se realiza con el manual de (Server, Install PHP, 2022), utilizando el comando dnf -y install php php-mbstring php-pear se realiza la instalación de los paquetes que se presenta en la Figura 43 y la Figura 44 presenta la página de información de php en el servidor.

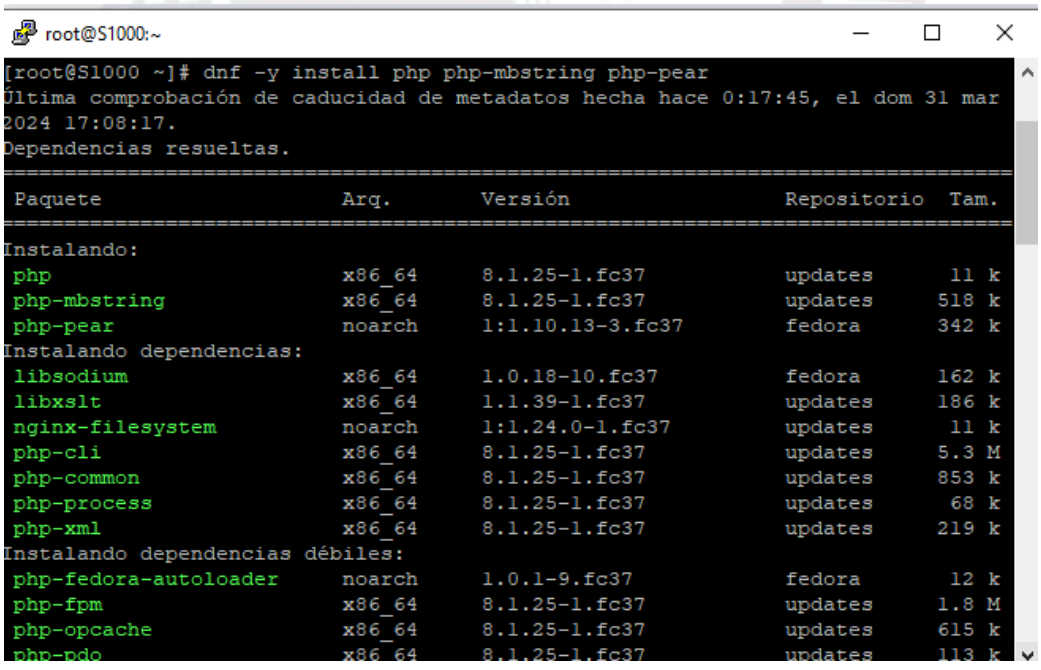
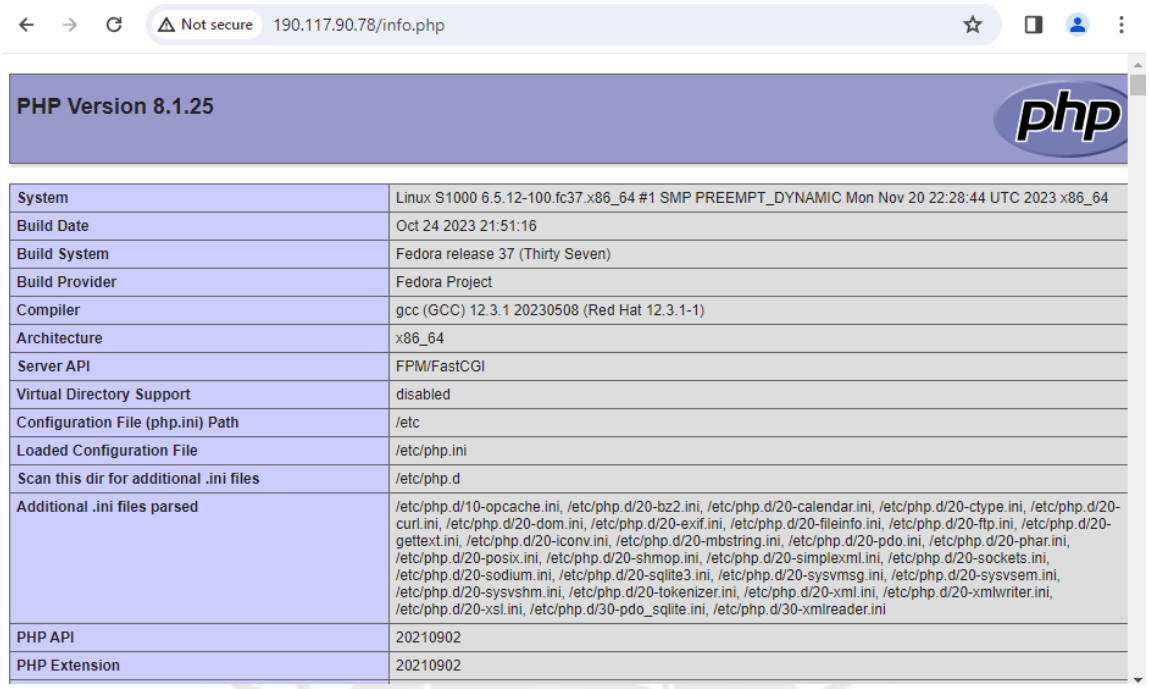


Figura 43. Instalación de php.



System	Linux S1000 6.5.12-100.fc37.x86_64 #1 SMP PREEMPT_DYNAMIC Mon Nov 20 22:28:44 UTC 2023 x86_64
Build Date	Oct 24 2023 21:51:16
Build System	Fedora release 37 (Thirty Seven)
Build Provider	Fedora Project
Compiler	gcc (GCC) 12.3.1 20230508 (Red Hat 12.3.1-1)
Architecture	x86_64
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/10-opcache.ini, /etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-dom.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-mbstring.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-posix.ini, /etc/php.d/20-shmop.ini, /etc/php.d/20-simplexml.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sodium.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-sysvmsg.ini, /etc/php.d/20-sysvsem.ini, /etc/php.d/20-sysvshm.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/20-xml.ini, /etc/php.d/20-xmlwriter.ini, /etc/php.d/20-xsl.ini, /etc/php.d/30-pdo_sqlite.ini, /etc/php.d/30-xmlreader.ini
PHP API	20210902
PHP Extension	20210902

Figura 44. Página de información de php.

4.2.2.5. Configuración de firewall con iptables

El núcleo de los sistemas operativos Linux y Unix tiene una pila TCP/IP muy robusta que permite implementar ruteo, contrafuegos, conmutación entre otras funcionalidades e inclusive es usado ampliamente por compañías como CISCO y otras para desarrollar productos propietarios (Paulikas, 2021), y es posible configurar el filtrado y ruteo de paquetes utilizando el comando iptables. Asi para ver las cadenas de paso de los paquetes podemos utilizar el comando iptables –L y el comando iptables –t nat –L, lo resultados de ejecución de estos comandos se presentan en la Figura 45, donde se aprecia las cadenas de paso de paquetes INPUT, OUTPUT, FORWARD, PREROUTONG y POSTROUTING.

Se activa el reenvió de paquetes asignando 1 al bit de IP forward con el comando echo 1 > /proc/sys/net/ipv4/ip_forward, para limpiar las reglas de las cadenas se utiliza iptables –F y iptables -t nat –F y para agregar una regla para realizar SNAT para reenviar los paquetes hacia Internet permitiendo acceder a internet a usuarios internos se puede utilizar el comando iptables -t nat -A POSTROUTING -s 192.168.1.0/255.255.255.0 -j SNAT --to 190.117.90.78, cambiando la dirección fuente del paquete por la dirección pública del firewall 190.117.90.78, que se observa en la Figura 46.

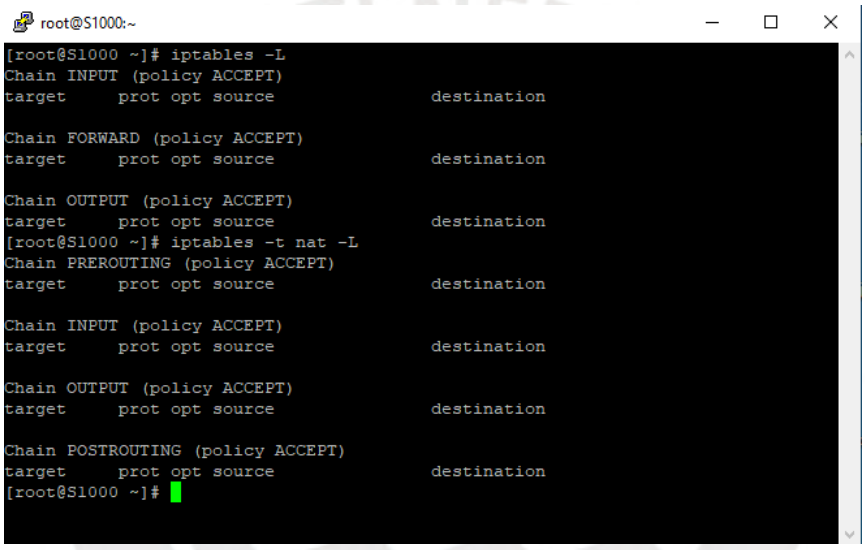


Figura 45. Cadenas de paso de paquetes del núcleo de Linux.

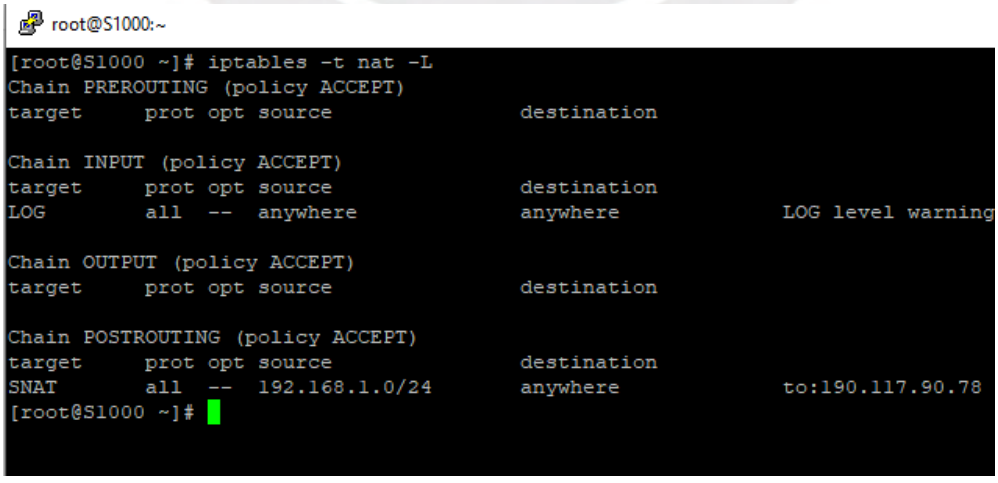


Figura 46. Regla de SNAT en la cadena POSTROUTING.

4.2.2.6. Configuración de captura de logs del firewall

La captura de registros de los paquetes IP en las cadenas INPUT, OUTPUT y FORWARD se realizar agregando las reglas iptables -A INPUT -j LOG --log-prefix "INPUT: " --log-level 4; iptables -A OUTPUT -j LOG --log-prefix "OUTPUT: " --log-level 4 y iptables -A FORWARD -j LOG --log-prefix "FORWARD: " --log-level 4, previamente configurando en rsyslog un formato de tiempo en horas, minutos segundos y diez milesimas de segundo; y el archivo destino para la captura como se observa en la Figura 47, donde se configura que los registros se almacenaran en el archivo /var/log/kernely.

La Figura 48 muestra las reglas para capturar los registros de paquetes IP para las cadenas INPUT, OUTPUT y FORWARD.

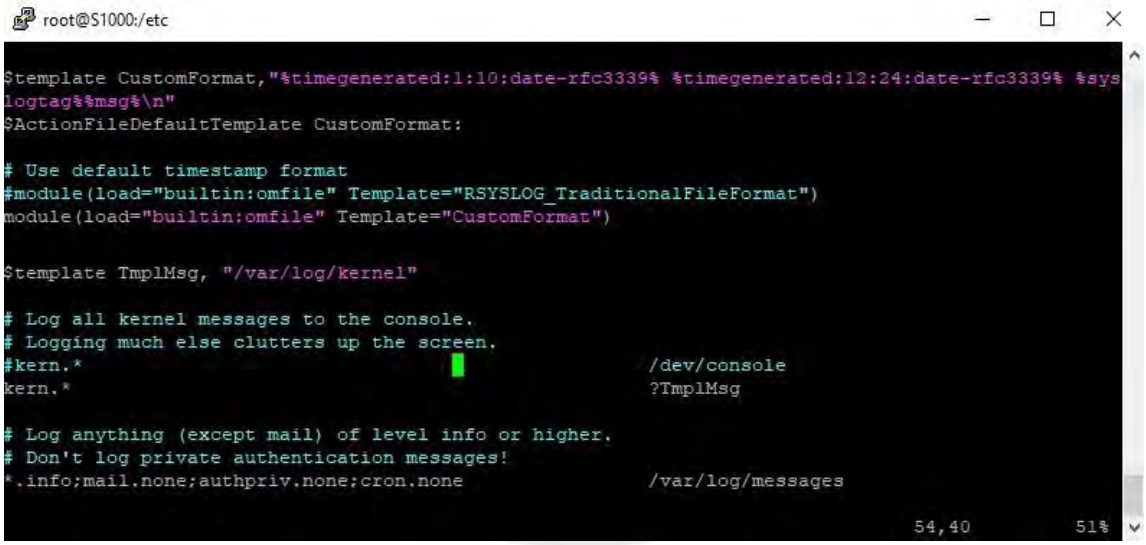


Figura 47. Configuración de archivo destino de registros.

Figura 48. Reglas de captura de registros de paquetes ip.

Finalmente, la Figura 49 presenta los registros de los paquetes IP capturados en el archivo /var/log/kernel, donde se ve el tiempo captura con una presión una de diez milecima de segundo muy importante para ordenar el tiempo de paso de los paquetes.

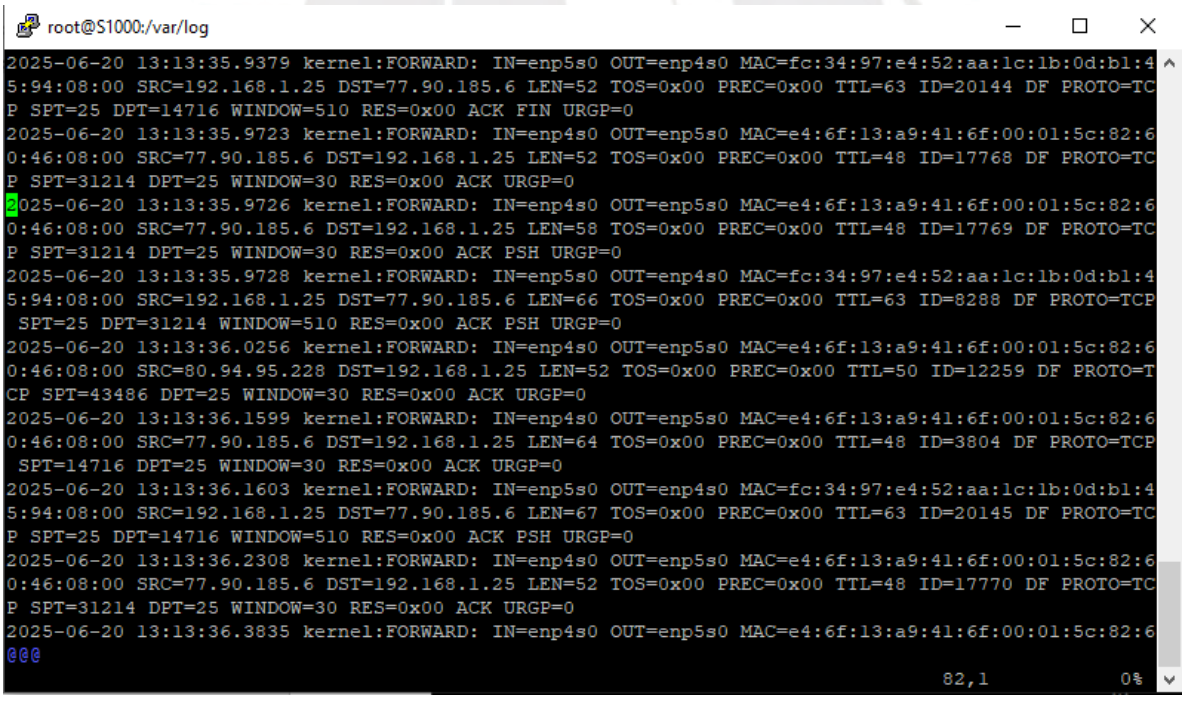


Figura 49. Registros de paquetes ip.

4.3. Discusión

La implementación del sistema de seguridad centralizado requiere un servidor físico instalado para lo cual se ensambla un servidor con un procesador Core I7 con 32 MB de RAM, dispositivo de almacenamiento DSS de 1 TB y con dos tarjetas de red, el proceso de ensamblaje se presenta en las Figuras 11 – 16, donde se muestra la instalación del procesador, la instalación del ventilador del procesador, la instalación de dos bancos de memoria RAM de 16 GB, la instalación del disco de estado sólido SATA de 1 TB y la instalación de la tarjeta de red adicional. Además, se realizó la conexión del servidor a la red como se presenta en la Figura 17.

La implementación del sistema de seguridad utiliza el sistema operativo Linux de la distribución Fedora con soporte de base de datos PostgreSQL, servidor web Apache, php e iptables, lo cual se instaló como se muestra en las Figuras 19 – 44, además se probó la configuración de reglas de captura de registros de paquetes ip, reglas de manipulación de trafico como SNAT, se configuro el archivo destino de los registros de paquetes IP como se muestra en las Figuras 46 – 48 y se presenta la captura de registros de paquetes IP en la Figura 49.

El ensamblaje del servidor se realizó siguiendo el procedimiento de estándar para ensamblar computadoras indicado por los fabricantes de partes de equipos de cómputo y es posible reproducir el proceso siguiendo los manuales que adjuntan lo fabricantes.

La instalación y configuración de software se realizó utilizando los manuales proporcionados por el sitio web Server World que incluyen comando de instalación y configuración.

Como medio de verificación se presentan fotografías y capturas de pantalla del proceso de ensamblaje de los diversos componentes, se presenta una fotografía de la conexión del servidor ensamblado a la red, capturas de pantallas de instalación configuración y operación de servidor de bases de datos PostgreSQL, servidor web Apache, iptables y captura de registros ip.

Con los resultados obtenidos el objetivo específico de implementar el sistema de seguridad centralizado en el firewall de la red que se desea asegurar es cumplido.

5. Implementación del módulo de identificación de criterios de seguridad

5.1. Introducción

El tráfico de red presenta muchas características o patrones que pueden ser usadas para la identificación de trafico anómalo que constituyen intrusiones a la red protegida, mediante el análisis con redes neuronales artificiales, sin embargo, para identificar el tráfico anómalo no siempre es necesario utilizar todas las características del tráfico, por lo que se requiere seleccionar las características apropiadas para la identificación del trafico anomalo de los ataques.

Las características del tráfico anómalo de red permiten definir criterios de seguridad los cuales pueden se expresados como criterios para aceptar el tráfico de red o criterios para rechazar el tráfico de red. El tráfico de paquetes en la red debe ser analizado en tiempo real y de acuerdo a los criterios de seguridad se debe generar un mensaje con los datos del tráfico de red que son intrusiones para que el módulo de auto configuración del firewall tome acciones.

El análisis se realiza utilizando RNA, las cuales después del entrenamiento identifican las intrusiones a la red, analizando el tráfico de paquetes en la red en tiempo real y genera un mensaje sobre la intrusión y los datos de la fuente que realiza la intrusión.

5.2. Resultados alcanzados

5.2.1. Diseño la RNA para el reconocimiento de criterios de seguridad

Para analizar el tráfico de red es importante realizar un proceso de extracción de características (A. Shahraki, 2022), en general existen cuatro tipos de características de tráfico de red: series temporales, cabeceras de protocolos, carga útil y características estadísticas.

Entre las características de series temporales están, tiempo máximo de llegada entre paquetes, tamaño máximo de los paquetes en bytes y tiempo de llegada entre paquetes.

Las cabeceras contienen información de las diferentes capas de red, como: números de puerto, protocolo y bits especiales de estado.

Las capas de red también ofrecen información sobre la carga útil como por ejemplo en el tráfico BitTorrent.

Las características estadísticas de tráfico longitud media de paquete, mediana de la longitud de paquetes, entre otros. Una particularidad de este tipo de características es que no pueden ser usadas para análisis en línea ya que primero se deben realizar los cálculos de los de los parámetros estadísticos. La base de datos (Yang, 2022) tiene las siguientes características del tráfico de red usadas son estadísticas y temporales como:

- Cantidad de bytes enviados en el flujo.
- Tasa de bytes del flujo enviado
- Cantidad de bytes del flujo recibido
- Tasa de bytes del flujo recibido
- Tamaño medio del paquete
- Mediana del tamaño del paquete
- Moda del tamaño del paquete
- Varianza del tamaño del paquete
- Desviación estándar del tamaño del paquete
- Coeficiente de variación del tamaño del paquete
- Desviación de la mediana del tamaño del paquete
- Desviación de la moda del tamaño del paquete
- Tiempo medio del paquete

- Mediana del tiempo del paquete
- Moda del tiempo de paquete
- Varianza del tiempo de paquete
- Desviación estándar del tiempo de paquete
- Coeficiente de variación del tiempo de paquete
- Desviación de la mediana del tiempo de paquete
- Desviación de la moda del tiempo de paquete
- Media de la diferencia de tiempo de solicitud/respuesta
- Mediana de la diferencia de tiempo de solicitud/respuesta
- Moda de la diferencia de tiempo de solicitud/respuesta
- Varianza de la diferencia de tiempo de solicitud/respuesta
- Deviación estándar de la diferencia de tiempo de solicitud/respuesta
- Coeficiente de variación de la diferencia de tiempo de solicitud/respuesta
- Desviación de la mediana de la diferencia de tiempo de solicitud/respuesta
- Desviación de la moda de la diferencia de tiempo de solicitud/respuesta.

Los logs de trafico considerado anómalo se obtiene mediante la captura de trafico de red de ataques simulados, realizando un procesamiento de los logs se extrae las características del tráfico anómalo que luego será bloqueado por el sistema.

5.2.2. Diseño de la RNA para el reconocimiento de criterios de seguridad

Los criterios de seguridad que se utilizan para detectar las intrusiones están especificados mediante las características de tráfico anomalo, por lo que hay criterios de seguridad especificados con un conjunto de características de tráfico que representan intrusiones.

Dependiendo de tamaño de la red local de la empresa u organización y la cantidad de usuarios externos que tengan los servicios ofrecidos, puede generar un tráfico de cientos hasta miles de paquetes por segundo que debe ser analizado en tiempo real para detectar intrusiones, por lo que se requiere un algoritmo computacional que reciba las características de tráfico y en tiempo real tome la decisión si el tráfico es normal o es una intrusión.

En la presente tesis se propone una solución basada en redes neuronales artificiales (RNA) que funcionan bajo el principio de las redes neuronales biológicas de las personas donde primero se realiza un proceso de aprendizaje sobre lo que se quiere reconocer y luego en base a este conocimiento se puede realizar el reconocimiento de forma instantánea.

La estructura de la RNA es presentada en la Figura 50, en donde hasta M entradas son características del tráfico, capas ocultas con varias neuronas por capa y tiene una capa de salida, que si reconoce que es tráfico que representa un ataque se activa. El tipo de neurona artificial utilizado es el perceptrón formando una red multicapa de perceptrones, con función de activación sigmoidea.

El número de capas y de neuronas en cada RNA, se define experimentalmente en las pruebas de entrenamiento.

El proceso de aprendizaje de la RNA es supervisado, que consiste en entrenar la RNA poniendo muestras de características de tráfico que son intrusiones y se repite el proceso hasta que se active la neurona de la salida, lo que demuestra que la RNA ya aprendió. La Figura 51 presennta el algoritmo de aprendizaje de la RNA, en donde se inicializa la RNA con valor de pesos y umbrales de activación aleatorios, se inicia con la primera repetición, se pone la primera muestra de características de tráfico que representa intrusión en la entrada y se ajusta los pesos hasta que se active la salida de la RNA, luego se coloca en la entrada la siguiente muestra y se repite el proceso de ajuste de pesos hasta que se active la RNA, y así sucesivamente se realiza el proceso de aprendizaje con todo el conjunto de muestras y se de repetir el proceso con el conjunto de muestras hasta que se active la RNA con todas las muestras.

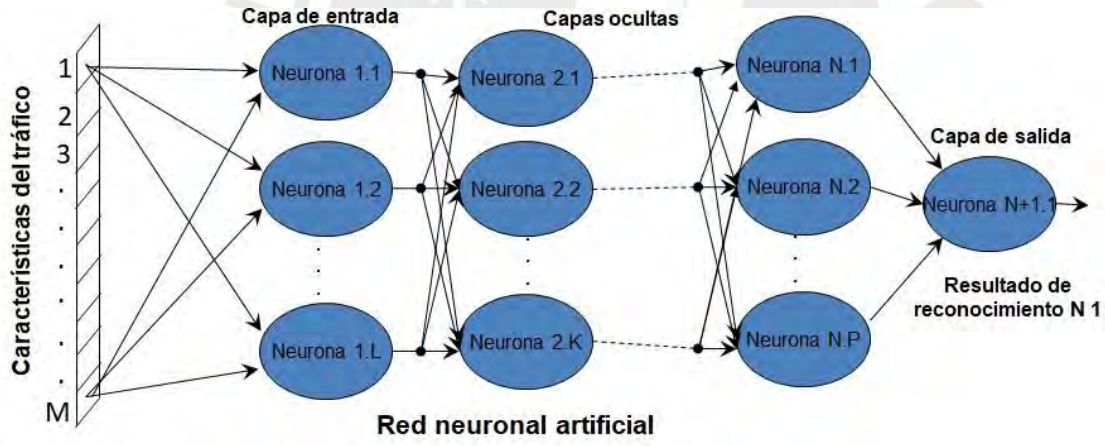


Figura 50. Estructura de la red neuronal artificial.

EL proceso de detección de intrusiones se realiza cargando los pesos obtenidos en el proceso de aprendizaje en la RNA, se carga las características de tráfico que se desea analizar en en el vector de entrada, se realiza el calculo de la RNA y si se activa la salida de la RNA, entonces el tráfico corresponde a una intrusión en caso contrario no, como se presenta en el flujograma del algoritmo de operación de la RNA en la Figura 52.

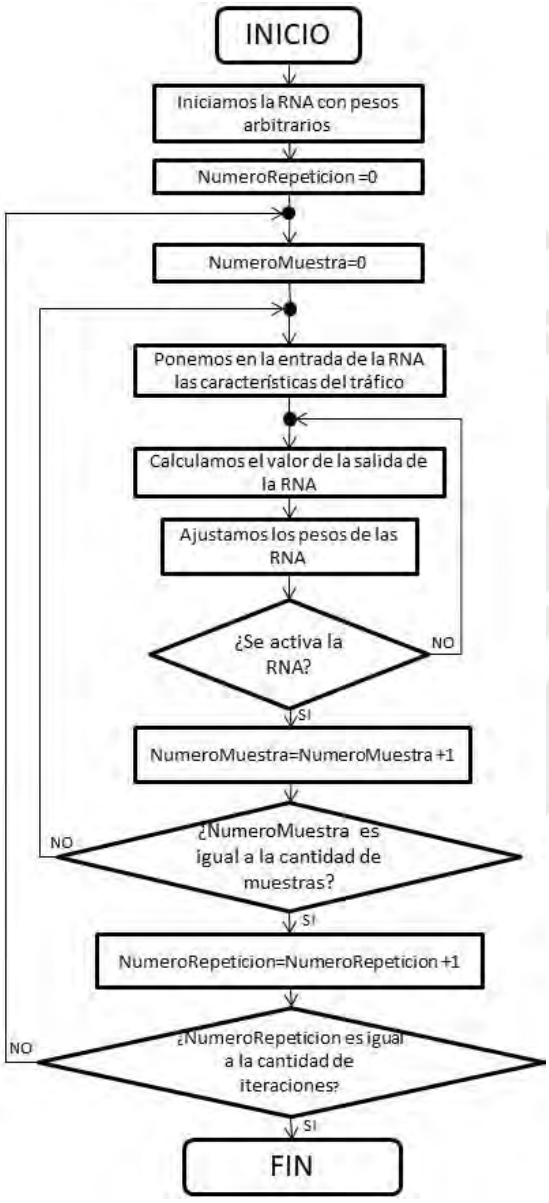


Figura 51. Flujograma del algoritmo de aprendizaje de la RNA.

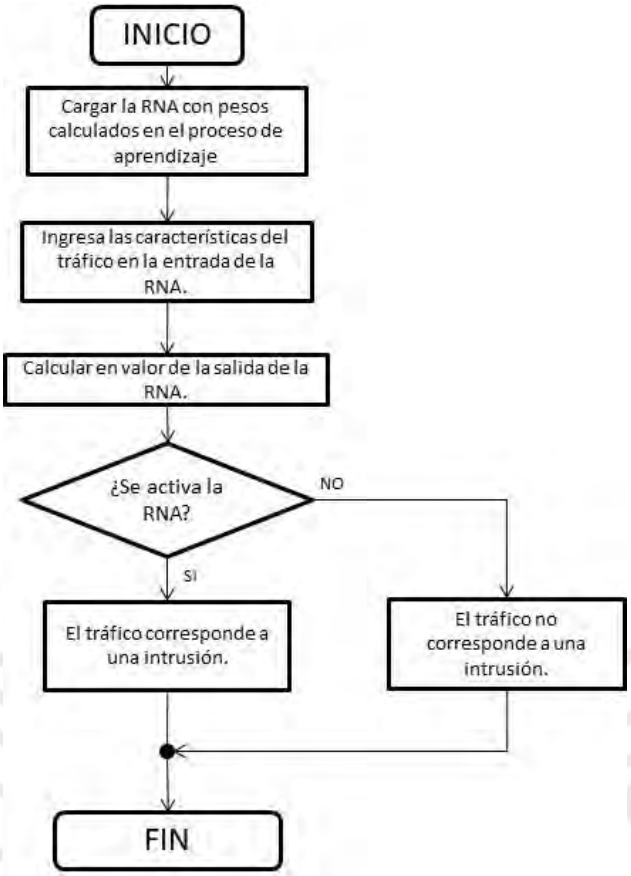


Figura 52. Flujograma del algoritmo de operación de la RNA.

5.2.3. Implementación del algoritmo de aprendizaje de la RNA del reconocimiento de criterios de seguridad

La implementación de la RNA para el reconocimiento de los criterios de seguridad se realiza utilizando una base de datos de se presenta en Figura 53, que tiene las tablas ESTRUCTURA_NEURONA, RNA, NEURONA y PESO. Para generar las RNAs se define las estructuras de cada RNA en la tabla de la base de datos ESTRUCTURA_NEURONA, en la cual en el primer registro es el número de capas en el campo dato_estructura_neurona, seguido una cantidad de registros igual al número de capas y en el campo dato_estructura_neurona se especifica la cantidad de neuronas en la capa empezando por la primera capa hasta la última capa.

Con la información de las estructuras de las RNAs se genera las RNAs y se guardan en las tablas RNA, NEURONA y PESO utilizando el programa en PHP del APÉNDICE B. La generación de las RNAS consiste en generar el número de entradas *m*, el umbral aleatorio *u*, la capa *capa* de cada neurona, registrar la RNA en la tabla *RNA* y generar el valor de los pesos aleatorio *valor_peso* en la tabla peso.

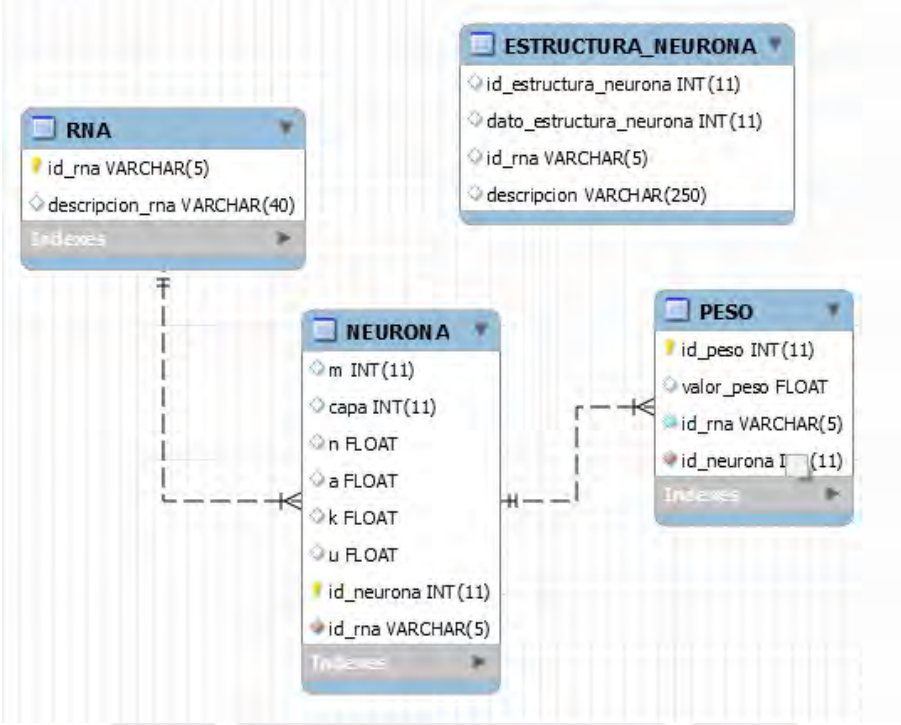


Figura 53. Modelo de base de datos.

El resultado de la ejecución del programa de generación de la RNA se presenta en la Figura 54, en donde se presenta la generación de los pesos aleatorios de los pesos de la neurona 1 de la RNA 00001 de la primera capa.

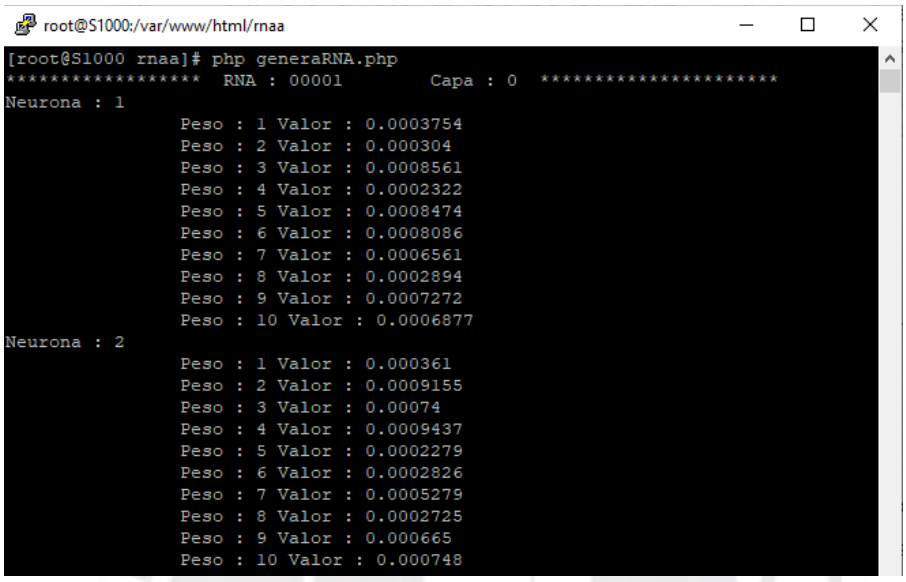


Figura 54. Ejecución del programa de generación de las RNAs.

La implementación del algoritmo de aprendizaje se realiza leyendo la informacion de la RNA generada de la base de datos, se ejecuta el proceso de entrenamiento y se actualizan los nuevos pesos, como se muestra en el flujograma en la Figura 55.

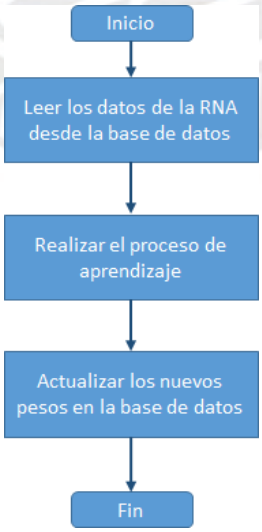


Figura 55. Flujograma implementación del proceso de aprendizaje.

La implementación del proceso de aprendizaje se realizó en base al flujograma de la Figura 55 con el programa en lenguaje php del APENDICE C, los resultados de ejecución se presenta en la Figura 56, donde se presenta el valor inicial de la RNA es de 1.8, el umbral de la salida RNA es 2154 y en la última columna se muestra como evoluciona la salida durante el proceso de entrenamiento. El proceso completo es mostrado en la Figura 57, donde se realiza el proceso con dos muestras en 5 iteraciones separadas por la línea vertical verde, en la primera iteración con la primera muestra actualiza aproximadamente 6.6 millones de pesos para llegar al umbral, con la segunda muestra el valor es superior al valor del umbral y se reduce hasta llegar al umbral finalmente en la siguientes iteraciones realiza ajuste a los pesos casi imperceptibles en el gráfico.

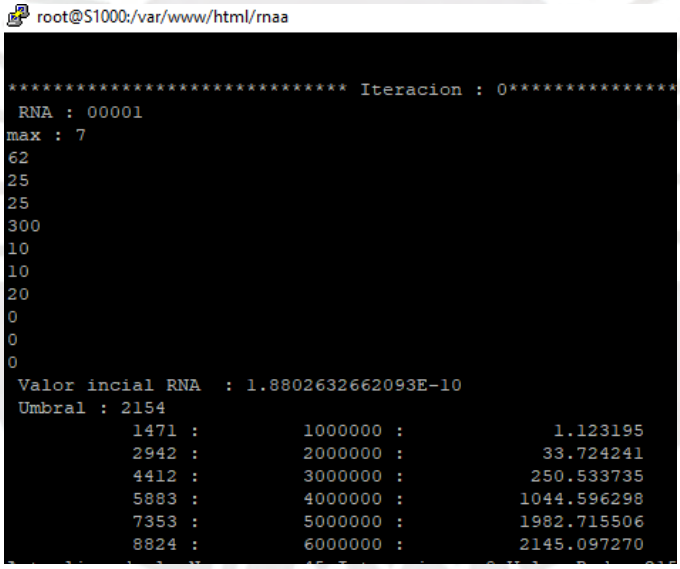


Figura 56. Resultado de ejecución del proceso de aprendizaje.

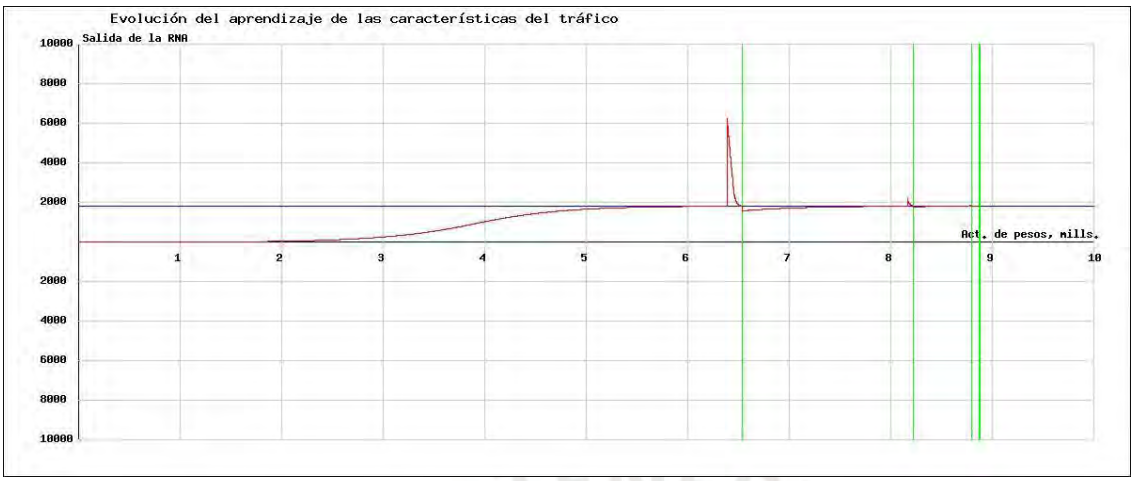


Figura 57. Evolución del aprendizaje de la RNA.

5.2.4. Implementar el algoritmo de operación de la RNA del reconocimiento de criterios de seguridad

La operación de la RNA del reconocimiento de criterios de seguridad se realizar utilizando los pesos obtenidos del proceso de aprendizaje que se conservaron en la base de datos en la tabla PESOS y los datos de las neuronas de la RNA almacenadas en la tabla NEURONA que se muestran en la Figura 53.

La implementación del algoritmo de operación se realizó con un programa en php que se presenta en el APÉNDICE D, utilizando el flujograma del algoritmo de operación de la RNA en la Figura 52.

Se probaron preliminarmente realizando el proceso de aprendizaje en una RNA con una catidad de neuronas por capa de 20; 30; 20; 10; y 1 neuronas, con 30 muestras de características de tráfico anomalo y se probó con 20 muestras que no pertenecían al grupo de muestras utilizadas durante el entrenamiento. El resultado del entrenamiento se realizó en 5000 iteraciones muestrado en la Figura 58, y se puede apreciar que el entrenamiento converge, en donde se muestra la curva de aprendizaje con color rojo y la separación entre iteraciones con líneas verticales de color verde, en aproximadamente 80 millones de actualizaciones de pesos.

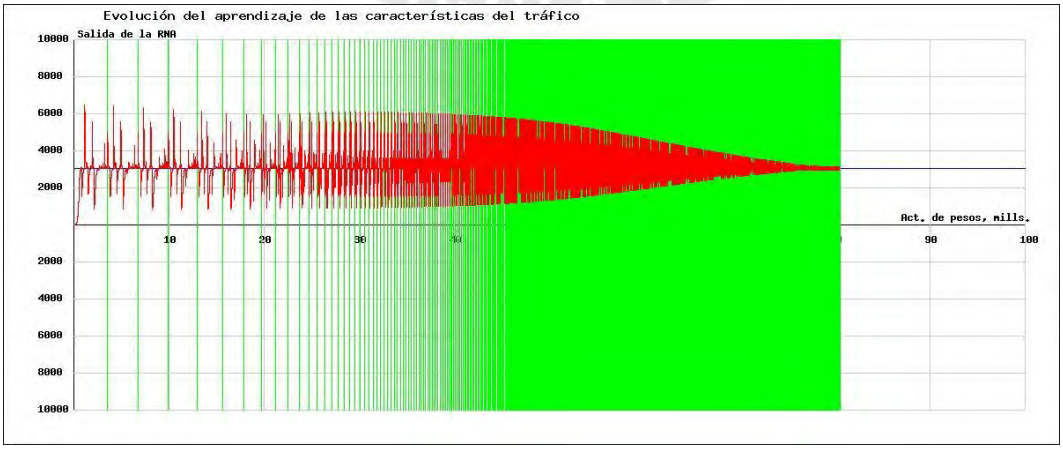


Figura 58. Curva de aprendizaje de la RNA.

La RNA reconoció de 100% de las muestras utilizadas para el proceso de aprendizaje y el 100% de las muestras de prueba, resultados muestrados en la Figura 59.

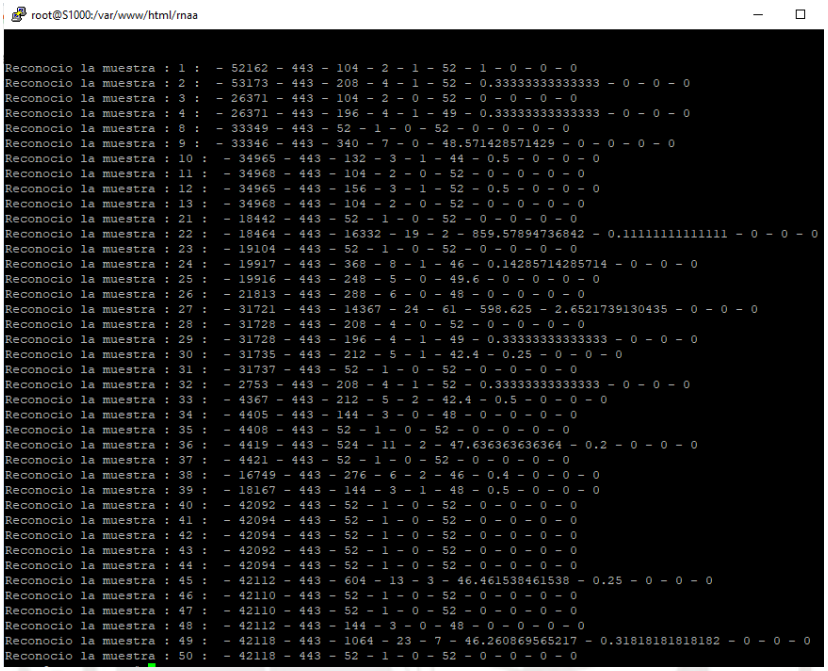


Figura 59. Resultado de operación de la RNA.

5.3. Discusión

Para la detección de intrusiones en la red protegida por el firewall, se diseñó la RNA multicapa de perceptrones con aprendizaje supervisado, se desarrollo el flujograma de aprendizaje, el algoritmo de funcionamiento y se realizo el desarrollo del software de entrenamiento de la red neuronal artificial, se probó con treinta muestras demostrando la funcionalidad del programa, se implementó la operación de la RNA y se probó con veinte muestras adicionales.

6. Implementación del módulo de autoconfiguración del firewall

6.1. Introducción

La reacción tardía ante la detección de una intrusión en la red en los sistemas con configuración manual de firewall es uno de los principales incovenientes ya que pude causar detención de servcios de red por un periodo de tiempo, pérdida de información valiosa para la institución o reducción de la eficiencia de la red.

Para resolver esto se propone en el presente trabajo un programa que configura automáticamente las reglas de bloqueo a los números IP de las intrusiones utilizando la información del flujo de tráfico de la intrusión.

Se realizaron pruebas para lo cual el programa se ejecuta continuamente y verifica si hay registros de intrusión creados por el modulo de detección de intrusiones inmediatamente se genera la regla de bloqueo de la IP del intruso y se aplica al firewall.

6.2. Resultados alcanzados

6.2.1. Implementación del software de generación de reglas del firewall

La autoconfiguración del firewall permite que el sistema de seguridad de redes propuesto bloquee inmediatamente si es detectado una intrusión a la red por le modulo de detecciones de intrusiones, para esto el modulo de detección de intrusiones genera un registro en la tabla intrusiones con los datos del flujo de tráfico de la intrusión. El registro de intrusiones tiene información para bloquear el IP desde el cual realizan el ataque y de esta manera bloquear el ataque mediante una regla en el firewall.

La generación de reglas con iptables se realiza con el programa en php del APENDICE E, en donde lee de la tabla de intrusiones los flujos y genera la regla correspondiente con el comando iptables.

6.2.2. Implementación y pruebas de la configuración automática del firewall

El modulo de detección de intrusiones con redes neuronales, al detectar una intrusión graba en la tabla “intrusion” el flujo de trafico de la intrusión muestrado en la Figura 60. El programa en php que genera las reglas de configuración analiza si hay intrusiones pendientes que procesar y genera la regla y ejecuta en el comando el el Shell de linux para agregar la regla en el kernel del firewall, como se puede ver en la Figura 61 la cadena FORWARD del firewall no tiene una regla del bloqueo a la IP 190.231.111.14 y después de la ejecución del programa, aparece una regla de bloqueo a la ip como se ve en la Figura 61.

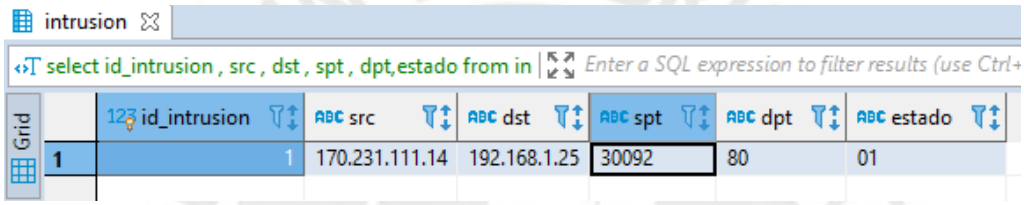


Figura 60. Registro de intrusión generado por la RNA.

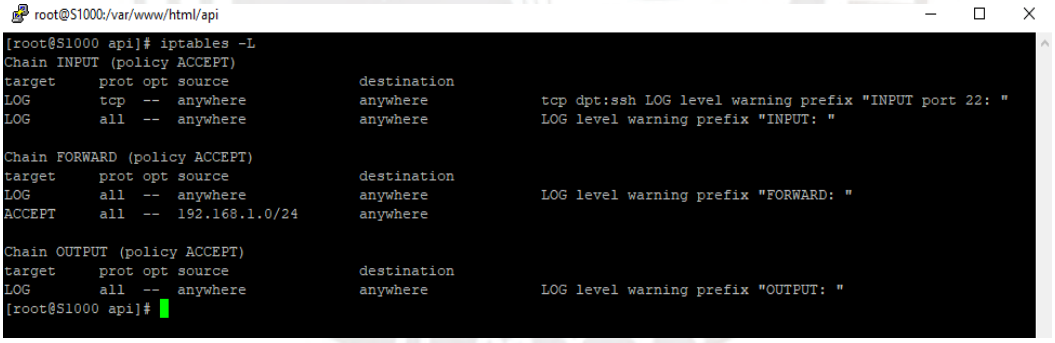


Figura 61. *Cadena FORWARD sin regla de bloqueo.*

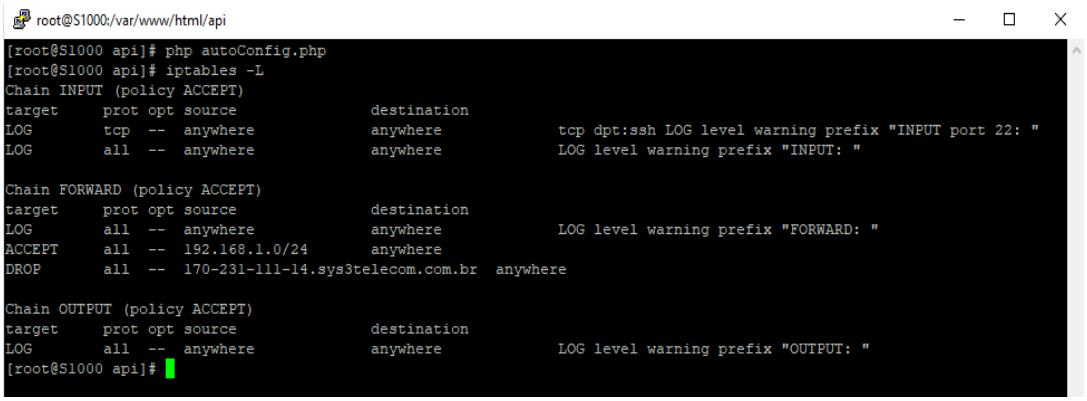


Figura 62. Cadena FORWARD con regla de bloqueo.

Por otro lado, para conservar las reglas generadas automáticamente en caso que se deba reiniciar el sistema operativo del firewall la reglas son conservados en el archiv0 firewall, con se puede ver en la Figura 63 el archivo firewall sin regla de bloqueo al IP 190.231.111.14 y luego de la ejecución de programa se muestra en la Figura 64 la regla autogenerada.

Ademas, se cambia de estado a 02 el registro en la tabla intrusión para que no vuelva a utilizar el programa de generación de reglas como se meustra en la Figura 65.

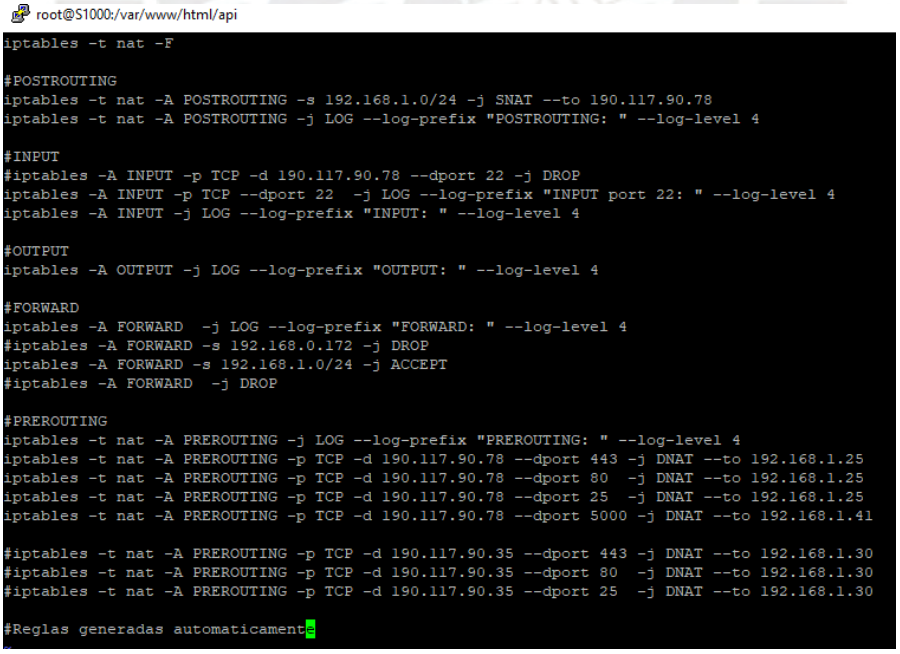


Figura 63. Archivo firewall sin regla de bloqueo al final.

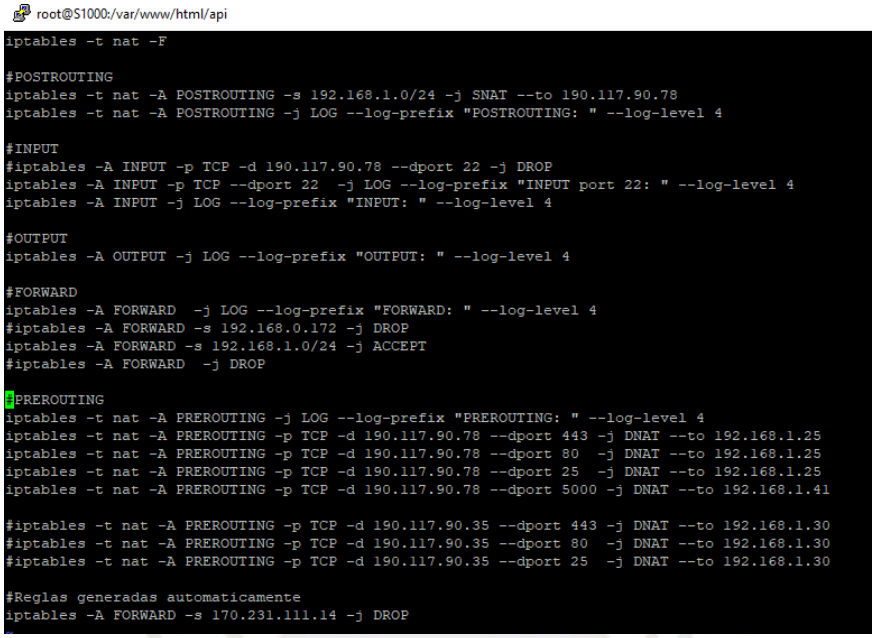


Figura 64. Archivo firewall con regla de bloqueo al final.

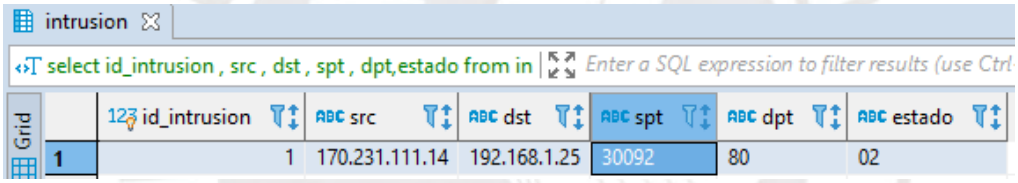


Figura 65. Registro de flujo de intrusión en estado 02.

6.3. Discusión

La configuración automática del firewall es importante para el bloqueo imediato de las intrusiones detectadas por el modulo de detección de intrusiones y eso es posible utilizando el comando iptables en Linux, para realizar esta configuración automáticamente se desarrollo un programa en php que genera la regla de bloqueo al IP que realiza la intrusión, ejecuta el comando el el Shell de Linux para cargar la regla en le firewall como se presenta en las Figuras 61 y 62, y conserva la regla en un archivo pare recargar las reglas en caso sea necesario como se puede ver en la Figuras 63 y 64.

7. Resultados

7.1. Introducción

Los resultados se obtuvieron haciendo pruebas utlizando datos de trafico anomalo de ataques para los servicios de acceso remoto con SSH, servicio web, servicio de correo electrónico y uso común de usuarios internos.

Se realizo una selección de muestras de tráfico para los servicios mensionados para realizar el proceso de entrenamiento, reconocimiento y de bloqueo automatico si se reconoce conexiones de tráfico de intrusiones.

Se obtuvo resultados de operación de sistema para los modulos por separados y finalmente integrado.

7.2. Resultados de aprendizaje de nuevos criterios de seguridad generando trafico de ataques simulados y trafico normal

Los criterios de seguridad están definidos por: el conjunto de datos de trafico anomalo de intrusiones que deben ser detectadas y bloqueadas mediante reglas de firewall y datos de trafico normal que debe ser confirmado para que el sistema no tome acciones.

Las pruebas se realizan con logs de trafico anómalo capturados por el servidor firewall implementado durante el proyecto denominado S1000, en el archivo kernel, en fomato de texto como las que se muestran en la Figura 66, estos logs son procesados por el programa subelogpgs.php y almancena la información en la tabla logs que se presenta en la Figura 67, haciendo un total de seis millones quinientos mil seicientos treinta y nueve (6500639) de logs en los utimos 5 dias como se presenta en la Figura 68, con datos como los mostrados en la Figura 69.

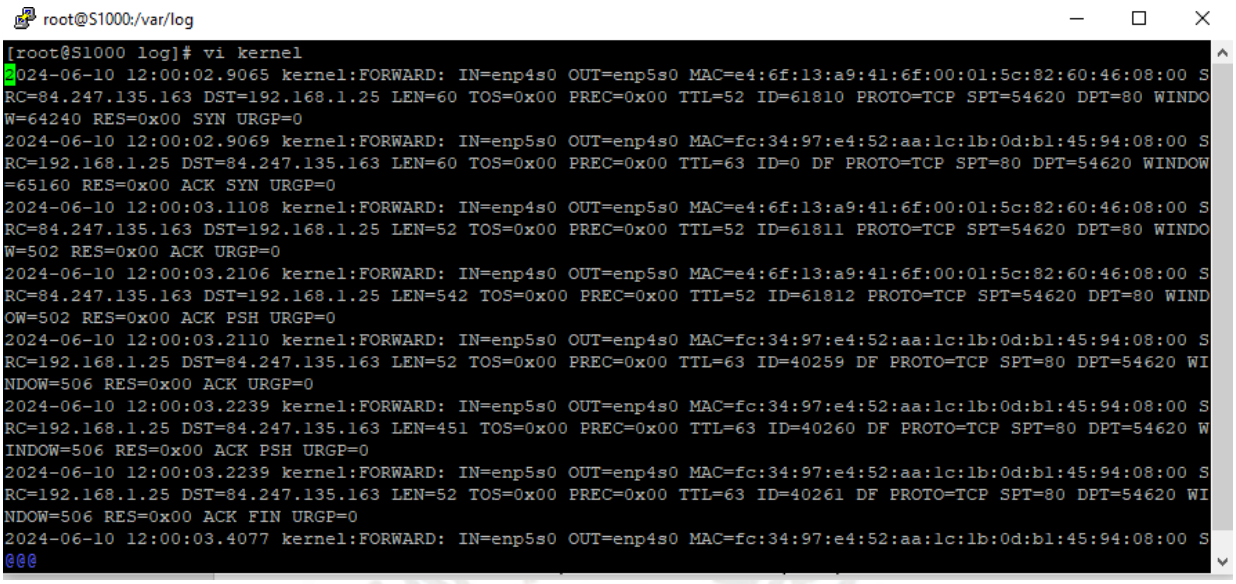


Figura 66. Archivo de logs kernel.

Los logs capturados pertenecen a ciento dieciocho mil novecientos dos (118902) IP de host que generaron tráfico en la red, de los cuales aproximadamente dos millones seiscientos mil son del IP 192.168.1.25 que pertence al servidor con servicios web, correo electrónico y ftp, y mas de un millón doscientos mil logs son del servicio web seguro (HTTPS) en el puerto 443, cuatrocientos mil aproximadamente del servicio web común (HTTP) en el puerto 80, doscientos sesenta mil aproximadamente del servicio acceso remoto (SSH) en el puerto 22 y doscientos cuarenta mil aproximadamente del servicio de correo electrónico en el puerto 25 como se presentan en la Figuras 70 y 71.

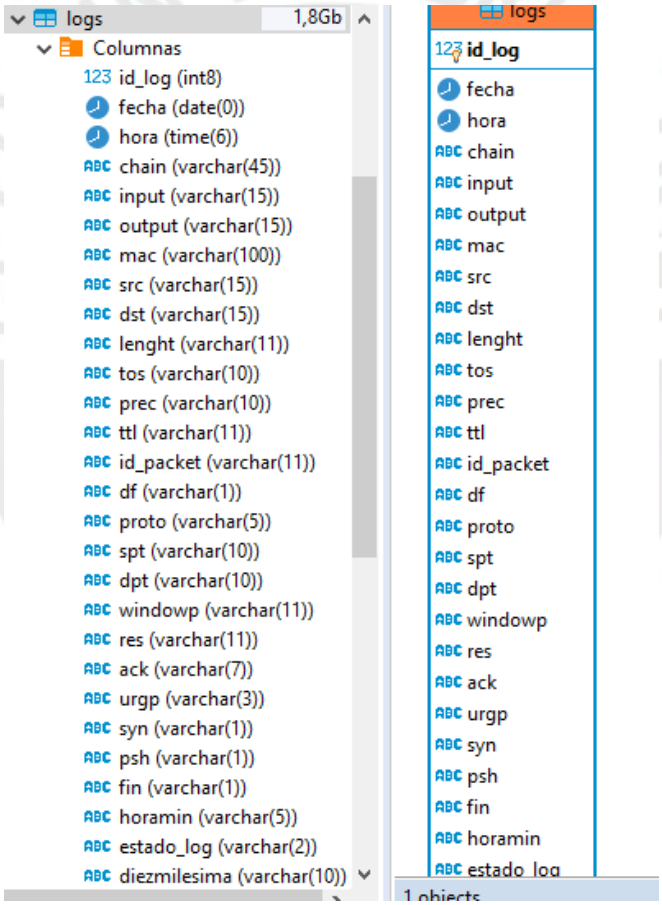


Figura 67. Tabla logs.

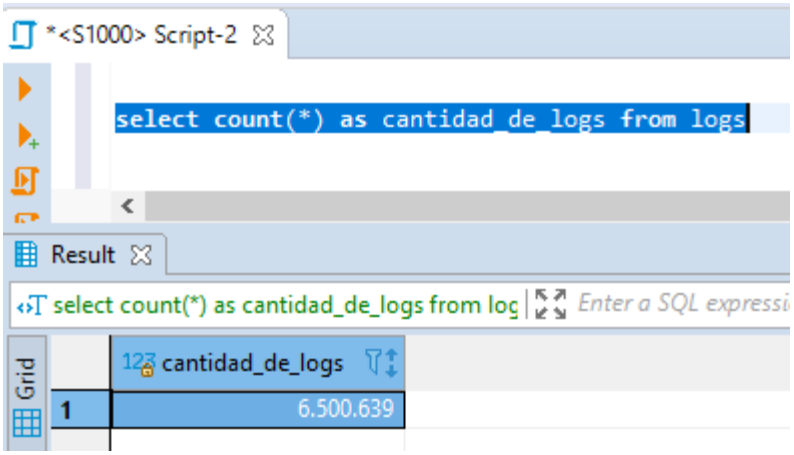


Figura 68. Cantidad de log capturados en los ultmos cinco días.

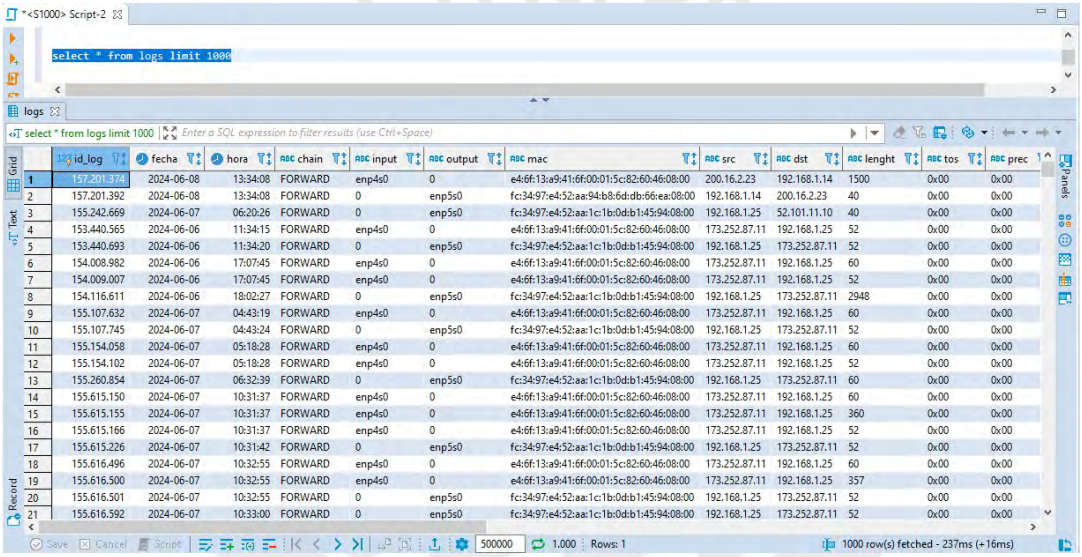


Figura 69. Datos de logs capturados y almacenados a la tabla logs.

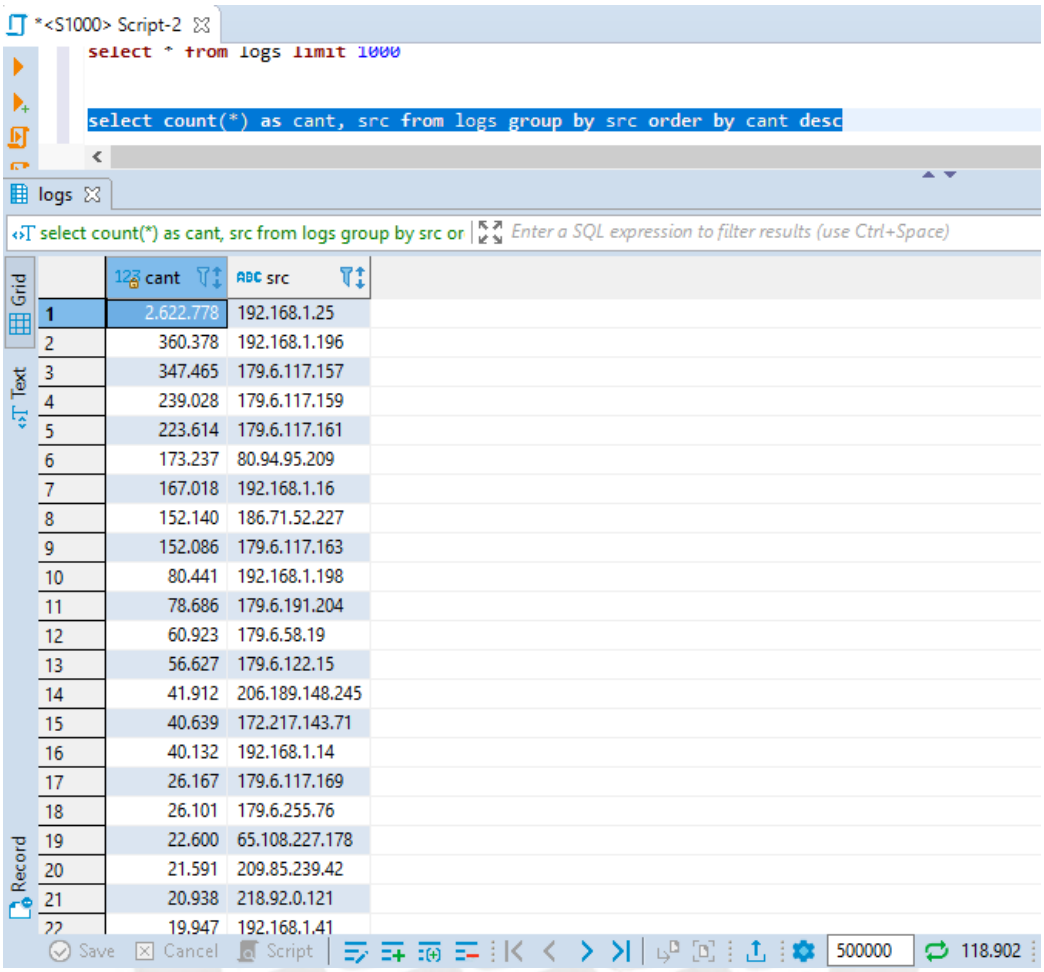


Figura 70. Cantidad de logs por IP fuente.

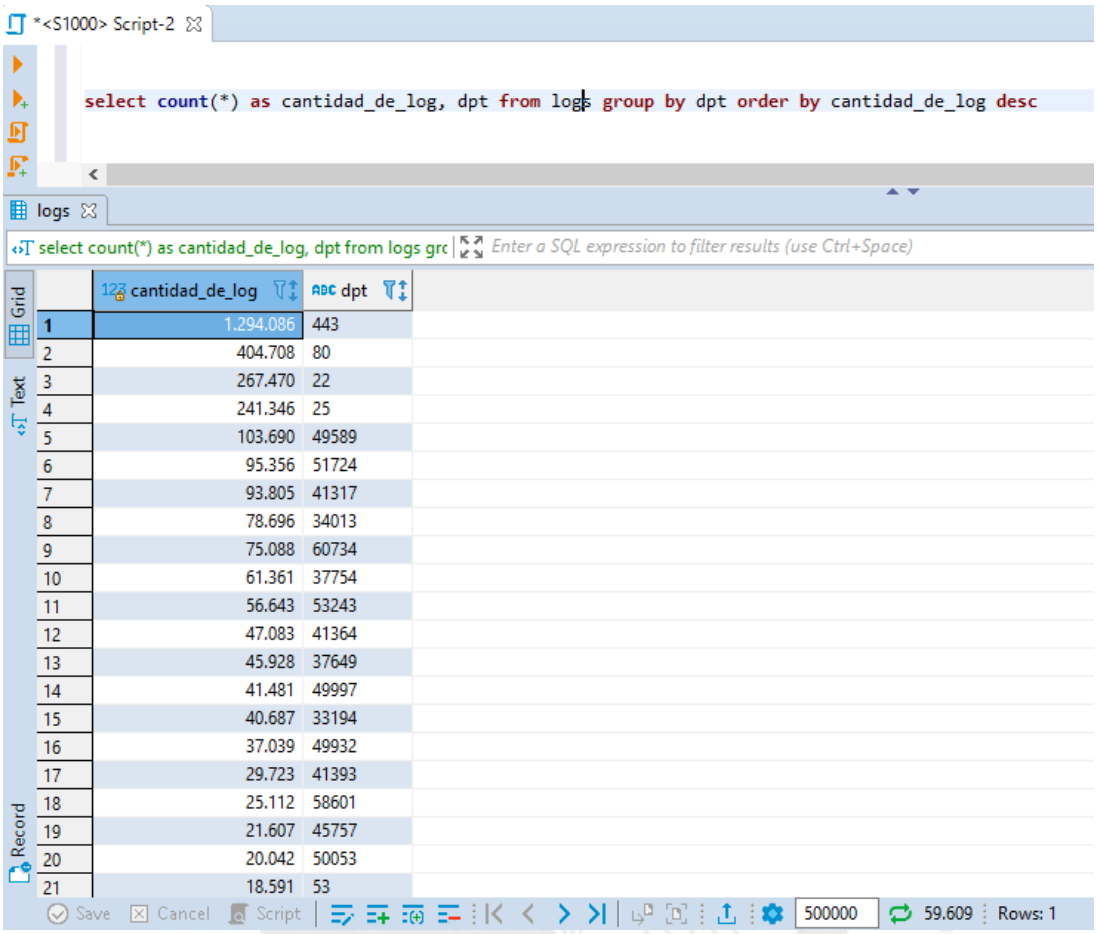


Figura 71. Cantidad de logs por servicio.

Para evaluar el trafico se requiere parámetros del trafico anómalo en términos de conexiones del protocolo TCP, que consiste en que un host se conecta a otro para intercambiar datos y cierra la conexión. A nivel operátivo el host inicia la conexión enviando un datagrama con el bit SYN de la cabecera TCP activo y finaliza con el bit FIN activo o simplemente no hay mas intercambio de paquetes entre los hosts y todos los datagramas internedios pertenecen a la conexión.

Procesando los más de seis millones de logs con el programa extrae_caracter.php del APENDICE A, se obtiene caraterísticas de las conexiones que se guardan en la tabla conexión mostrada en la Figura 72, haciendo un total del trescientas veinte un mil trecientas cincueta y siete conexiones como se presenta en la Figura 73, con datos mostrados en la Figura 74, los cuales estan dsitribuidos por ips como se presenta en la Figura 75 y por servicios como se muestra en la Figura 76.

El conjunto de datos de conexiones de trafico anómalo de intrusiones o ataques se obtuvieron de ataques al servidor publicado al puerto 22 del servicio de acceso remoto SSH como se puede ver las conexiones intermitentes desde la IP 103.108.94.89 en la Figura 77.

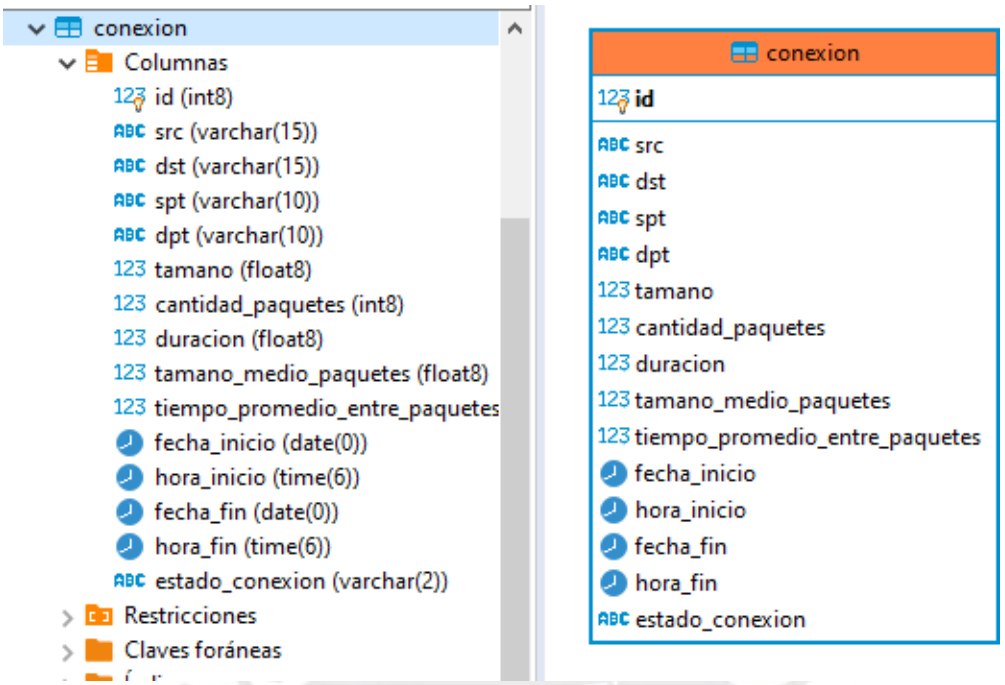


Figura 72. Tabla conexion.

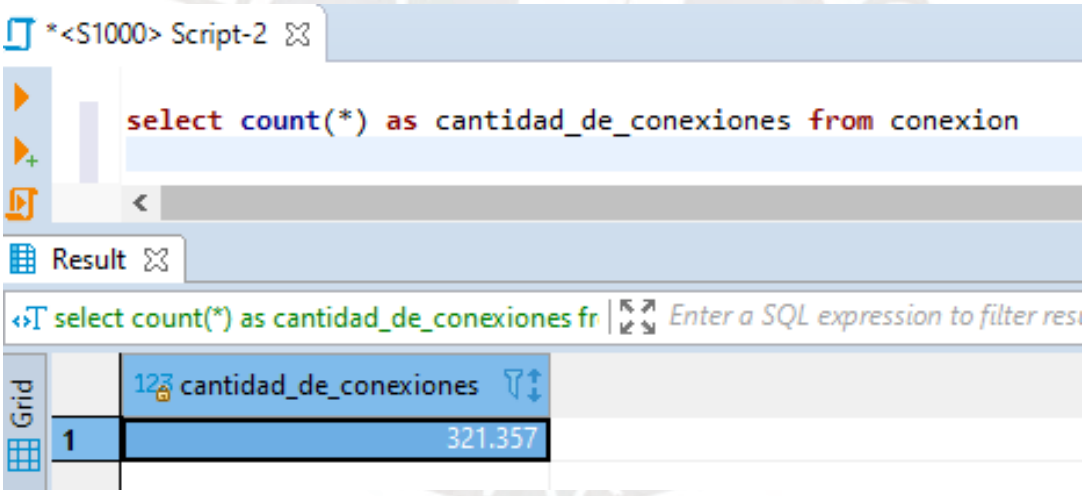
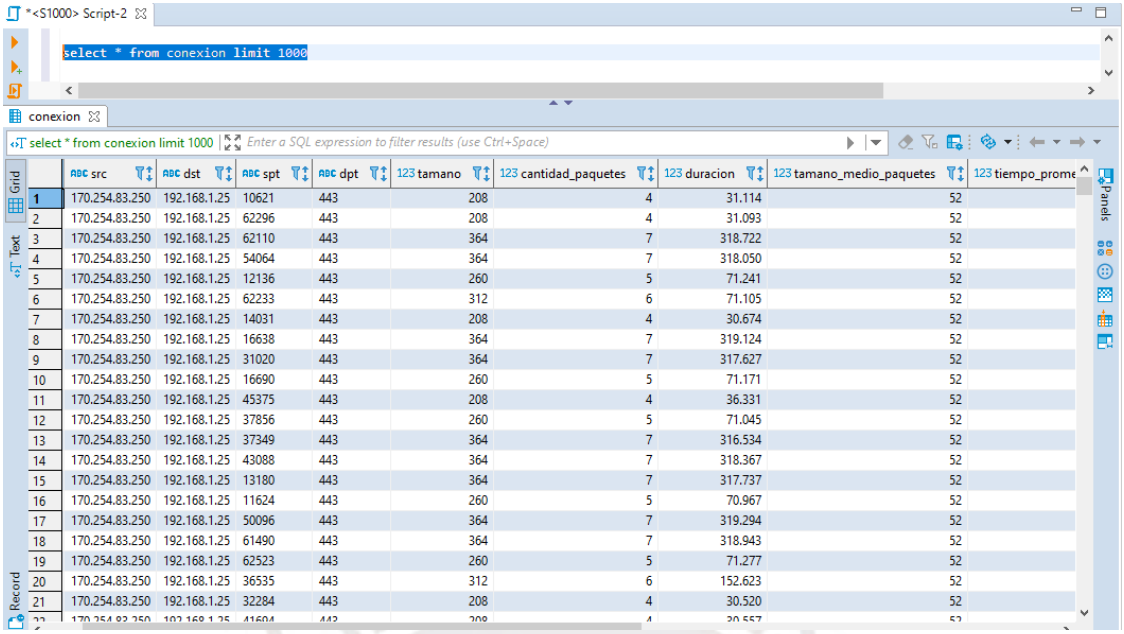


Figura 73. Cantidad de conexiones.



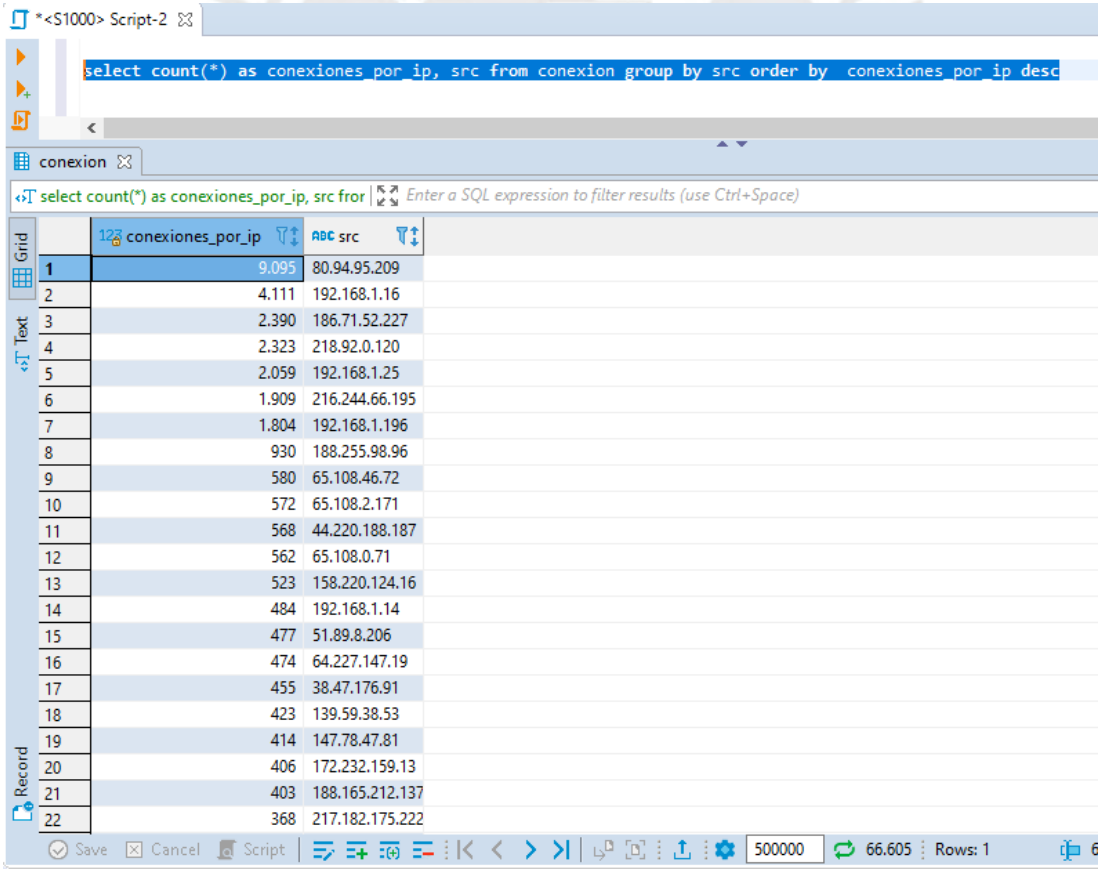
```
select * from conexion limit 1000
```

conexion

select * from conexion limit 1000 | Enter a SQL expression to filter results (use Ctrl+Space)

	src	dst	spt	dpt	tamano	cantidad_paquetes	duracion	tamano_medio_paquetes	tiempo_prome
1	170.254.83.250	192.168.1.25	10621	443	208	4	31.114	52	
2	170.254.83.250	192.168.1.25	62296	443	208	4	31.093	52	
3	170.254.83.250	192.168.1.25	62110	443	364	7	318.722	52	
4	170.254.83.250	192.168.1.25	54064	443	364	7	318.050	52	
5	170.254.83.250	192.168.1.25	12136	443	260	5	71.241	52	
6	170.254.83.250	192.168.1.25	62233	443	312	6	71.105	52	
7	170.254.83.250	192.168.1.25	14031	443	208	4	30.674	52	
8	170.254.83.250	192.168.1.25	16638	443	364	7	319.124	52	
9	170.254.83.250	192.168.1.25	31020	443	364	7	317.627	52	
10	170.254.83.250	192.168.1.25	16690	443	260	5	71.171	52	
11	170.254.83.250	192.168.1.25	45375	443	208	4	36.331	52	
12	170.254.83.250	192.168.1.25	37856	443	260	5	71.045	52	
13	170.254.83.250	192.168.1.25	37349	443	364	7	316.534	52	
14	170.254.83.250	192.168.1.25	43088	443	364	7	318.367	52	
15	170.254.83.250	192.168.1.25	13180	443	364	7	317.737	52	
16	170.254.83.250	192.168.1.25	11624	443	260	5	70.967	52	
17	170.254.83.250	192.168.1.25	50096	443	364	7	319.294	52	
18	170.254.83.250	192.168.1.25	61490	443	364	7	318.943	52	
19	170.254.83.250	192.168.1.25	62523	443	260	5	71.277	52	
20	170.254.83.250	192.168.1.25	36535	443	312	6	152.623	52	
21	170.254.83.250	192.168.1.25	32284	443	208	4	30.520	52	

Figura 74. Datos de conexiones.



```
select count(*) as conexiones_por_ip, src from conexion group by src order by  conexiones_por_ip desc
```

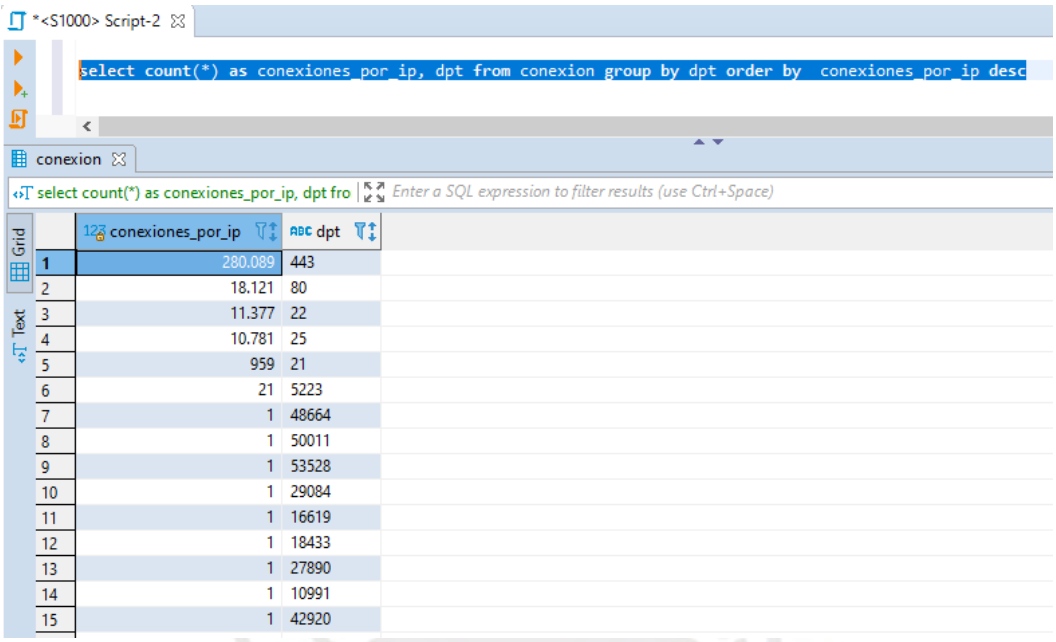
conexion

select count(*) as conexiones_por_ip, src fror | Enter a SQL expression to filter results (use Ctrl+Space)

	conexiones_por_ip	src
1	9.095	80.94.95.209
2	4.111	192.168.1.16
3	2.390	186.71.52.227
4	2.323	218.92.0.120
5	2.059	192.168.1.25
6	1.909	216.244.66.195
7	1.804	192.168.1.196
8	930	188.255.98.96
9	580	65.108.46.72
10	572	65.108.2.171
11	568	44.220.188.187
12	562	65.108.0.71
13	523	158.220.124.16
14	484	192.168.1.14
15	477	51.89.8.206
16	474	64.227.147.19
17	455	38.47.176.91
18	423	139.59.38.53
19	414	147.78.47.81
20	406	172.232.159.13
21	403	188.165.212.137
22	368	217.182.175.222

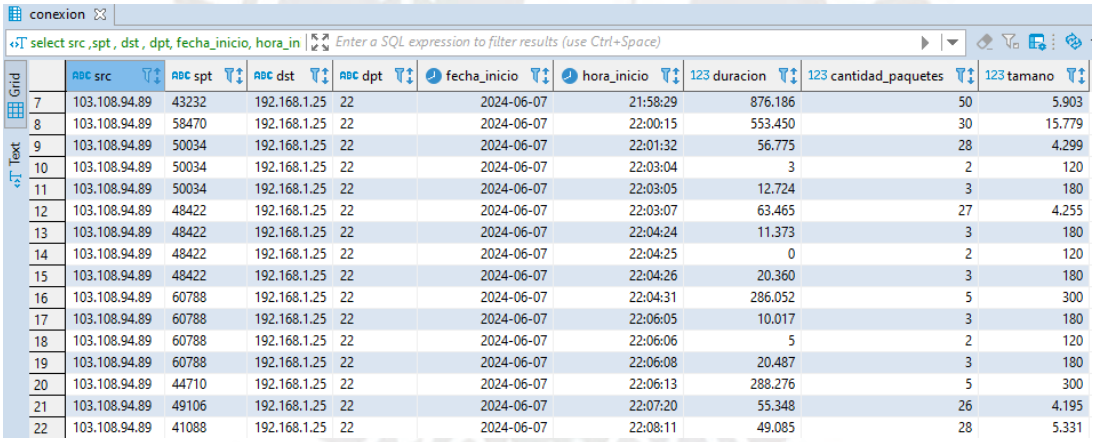
Save | Cancel | Script | 500000 | 66.605 | Rows: 1

Figura 75. Cantidad de conexiones por ip.



	conexiones_por_ip	dpt
1	280.089	443
2	18.121	80
3	11.377	22
4	10.781	25
5	959	21
6	21	5223
7	1	48664
8	1	50011
9	1	53528
10	1	29084
11	1	16619
12	1	18433
13	1	27890
14	1	10991
15	1	42920

Figura 76. Cantidad de conexiones por servicio.



	src	spt	dst	dpt	fecha_inicio	hora_inicio	duracion	cantidad_paquetes	tamano
7	103.108.94.89	43232	192.168.1.25	22	2024-06-07	21:58:29	876.186	50	5.903
8	103.108.94.89	58470	192.168.1.25	22	2024-06-07	22:00:15	553.450	30	15.779
9	103.108.94.89	50034	192.168.1.25	22	2024-06-07	22:01:32	56.775	28	4.299
10	103.108.94.89	50034	192.168.1.25	22	2024-06-07	22:03:04	3	2	120
11	103.108.94.89	50034	192.168.1.25	22	2024-06-07	22:03:05	12.724	3	180
12	103.108.94.89	48422	192.168.1.25	22	2024-06-07	22:03:07	63.465	27	4.255
13	103.108.94.89	48422	192.168.1.25	22	2024-06-07	22:04:24	11.373	3	180
14	103.108.94.89	48422	192.168.1.25	22	2024-06-07	22:04:25	0	2	120
15	103.108.94.89	48422	192.168.1.25	22	2024-06-07	22:04:26	20.360	3	180
16	103.108.94.89	60788	192.168.1.25	22	2024-06-07	22:04:31	286.052	5	300
17	103.108.94.89	60788	192.168.1.25	22	2024-06-07	22:06:05	10.017	3	180
18	103.108.94.89	60788	192.168.1.25	22	2024-06-07	22:06:06	5	2	120
19	103.108.94.89	60788	192.168.1.25	22	2024-06-07	22:06:08	20.487	3	180
20	103.108.94.89	44710	192.168.1.25	22	2024-06-07	22:06:13	288.276	5	300
21	103.108.94.89	49106	192.168.1.25	22	2024-06-07	22:07:20	55.348	26	4.195
22	103.108.94.89	41088	192.168.1.25	22	2024-06-07	22:08:11	49.085	28	5.331

Figura 77. Ataque de fuerza bruta al servicio de acceso remoto SSH.

El ataque simulado de DoS al servidor web publicado con IP publica 190.117.90.78 e IP interna 192.168.1.25, desde la IP externa 186.71.52.227 mediante el programa saludo_web.php del APENDICE G, como se nuestra en la Figura 78. Se capturaron mas de ocho cientos mil logs como se presenta en la Figura 79, con los datos presentados en la Figura 80 y los datos de las conexiones se presentan en la Figura 81.

Como datos de intrusiones al servicio de correo electrónico se utiliza el trafico generado por la IP 80.94.95.209 que durante todo el tiempo se conecta al servicio de correo electrónico publicado en la IP 192.168.1.25 como se puede ver en la Figura 82, habiendo generado mas de seis cientos mil logs en los últimos doce días y mas de treinta y ocho mil conexiones como se presenta en la Figura 83.

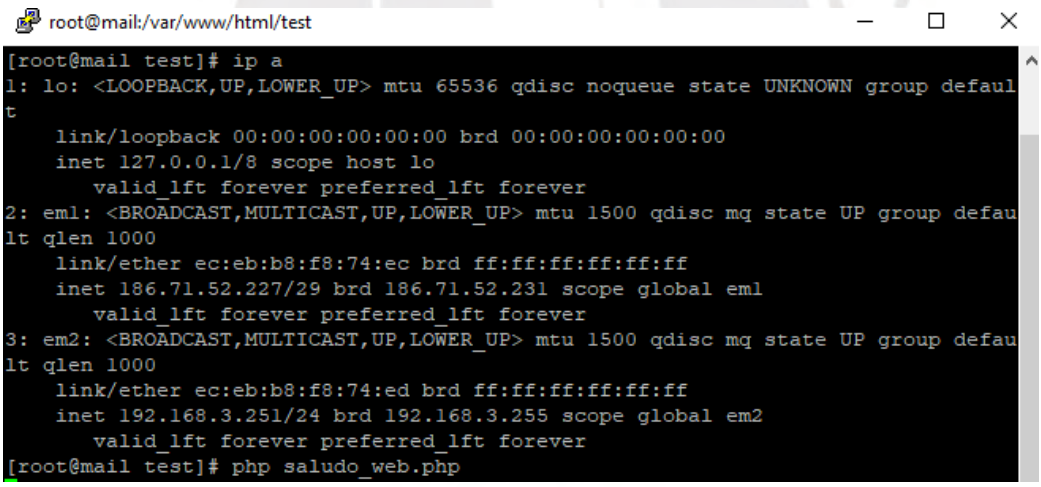


Figura 78. Ataque DoS al servidor web.

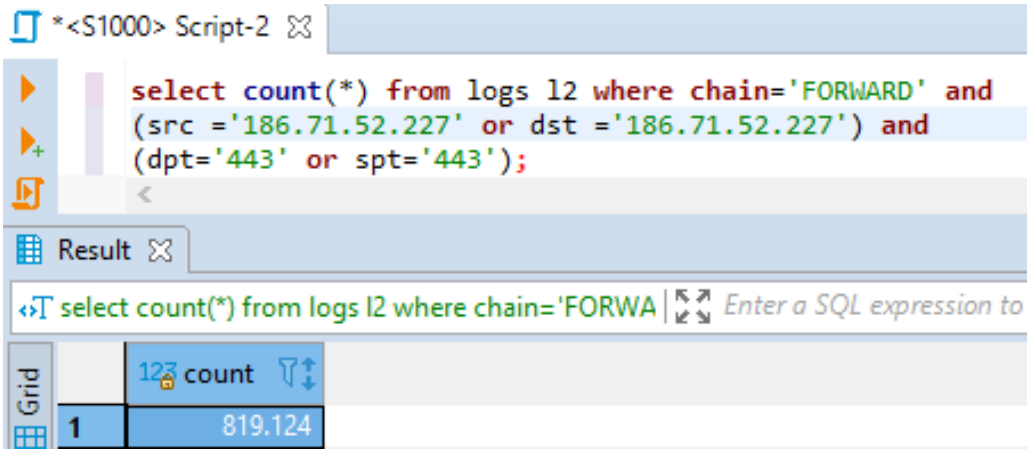


Figura 79. Cantidad de logs de ataque DoS al servidor web.

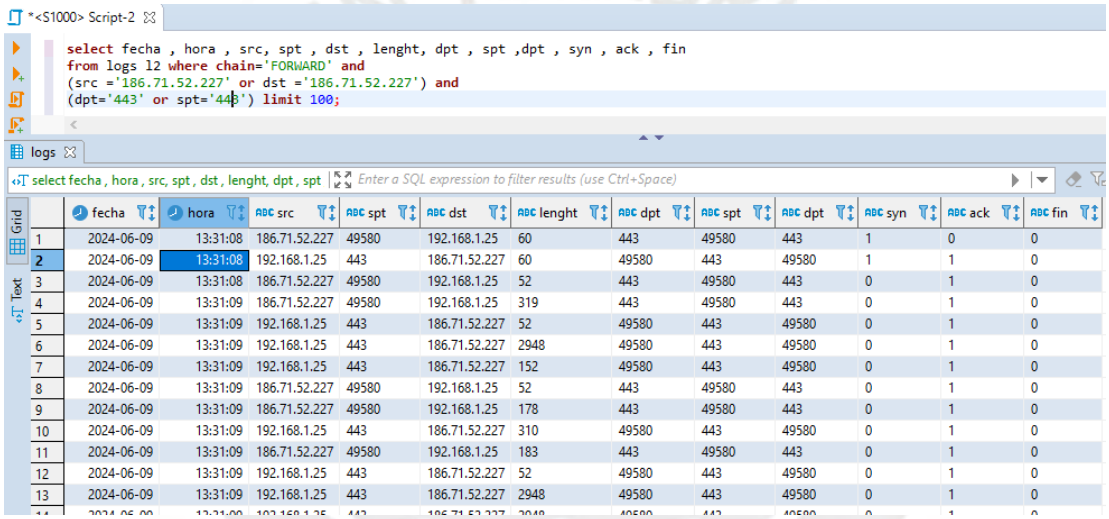


Figura 80. Datos de logs de ataque DoS al servidor web.

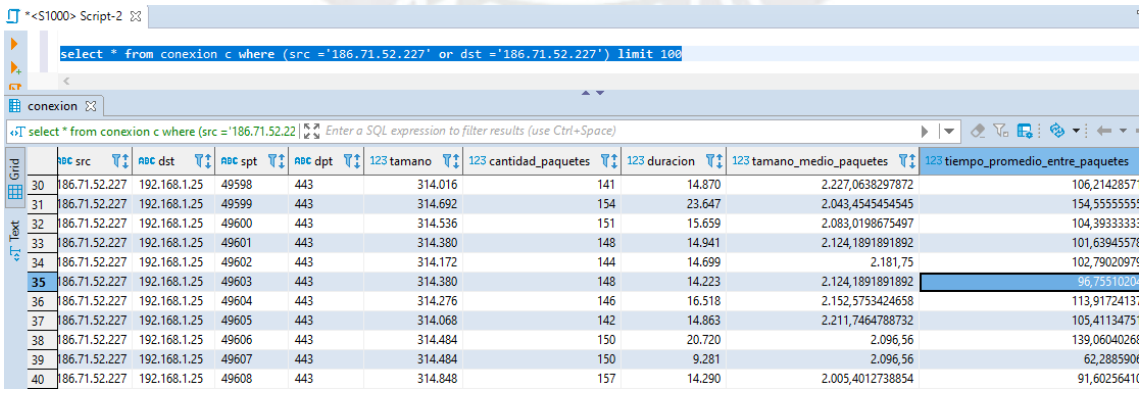


Figura 81. Datos de las conexiones de ataque DoS al servidor web.

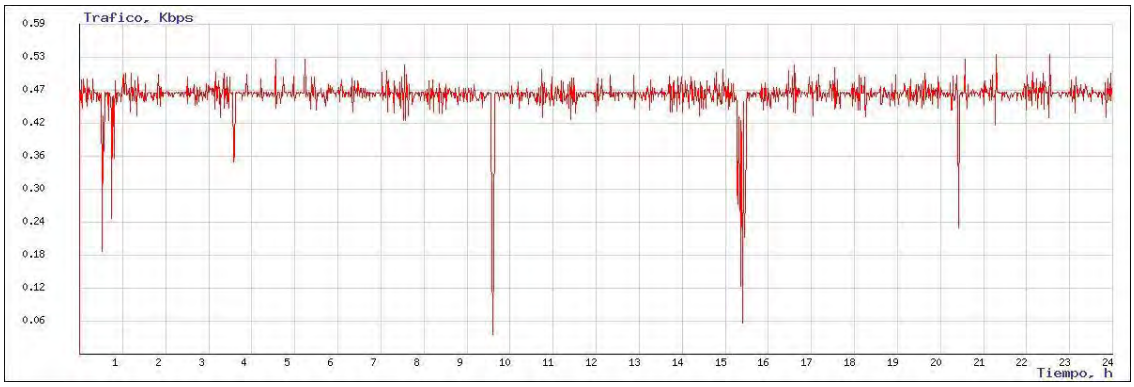


Figura 82. Trafico del IP 80.94.95.209 al servicio de correo electrónico en el puerto 25.

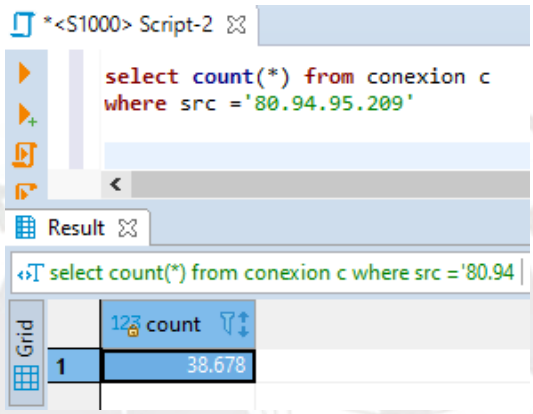


Figura 83. Conexiones del IP 80.94.95.209 al servicio de correo electrónico en el puerto 25.

El análisis de tráfico se realiza mediante redes neuronales artificiales, y para garantizar el reconocimiento del trafico adecuadamente se utiliza una RNA para reconocer las conexiones de trafico anómalo de intrusiones.

Cuando el sistema reconoce una conexión de trafico anómalo de intrusión de red. la conexión es marcada como una intrusión para que el modulo de auto configuración lo procese bloqueando ese trafico mediante una regla de iptables.

La RNA funciona utilizando el conocimiento almacenado en los pesos de cada neuronal artificial de la RNA, para esto la RNA debe pasa por un proceso de entrenamiento que ajusta los pesos utilizando muestras de conexiones tráfico de intrusiones, y luego del proceso de aprendizaje la RNA operan reconociendo el tráfico para el cual fueron entrenadas.

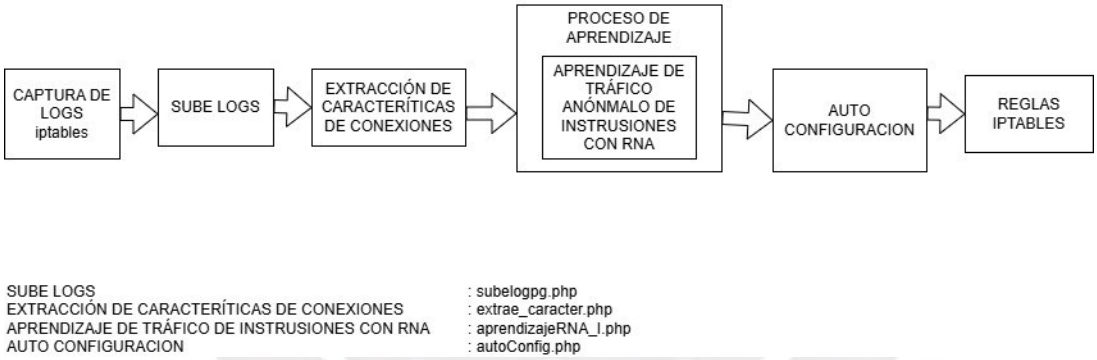


Figura 84. Diagrama del sistema.

En la Figura 85 se presenta la catidad de logs capturados por día, que luego de procesarlos se convierten conexiones por día mostradas en la Figura 86

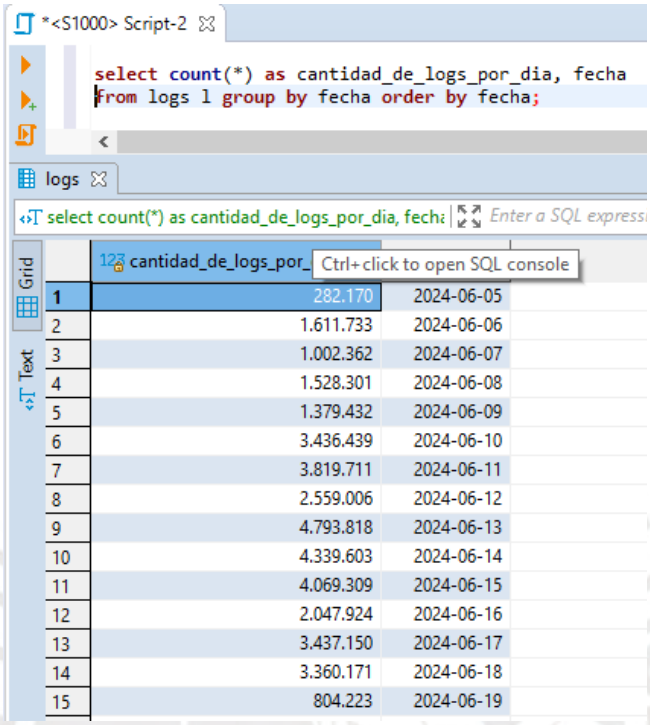


Figura 85. Cantidad de logs por dia.

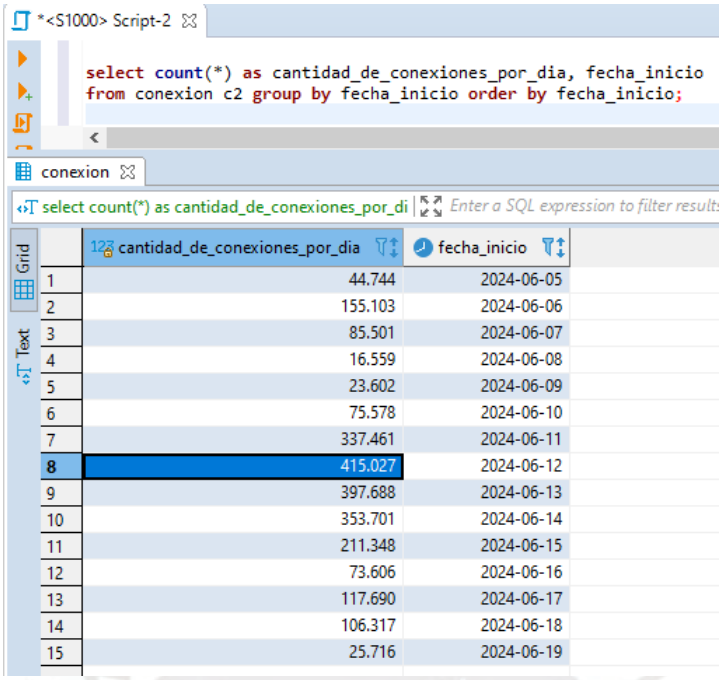


Figura 86. Cantidad de conexiones por dia.

Las RNA utilizada en el sistema tienen una estructura con cinco capas de veinte, treinta, veinte, diez y una, neuronas por capa, haciendo un total 81 neuronas tipo perceptron.

Tabla 16: Conjunto de datos de aprendizaje de la RNA para el reconocimiento de conexiones de trafico de instrusiones.

Nº	Descripción	Total de muestras	Muestras de aprendizaje	Muestras de prueba
1	Conexiones de trafico de intrusiones de fuerza bruta al servcio SSH	114.641	103.176	11.465
2	Conexiones de trafico de intrusiones al servicio web	5.890	5.301	589
3	Conexiones de trafico de intrusiones al servicio de correo	38.678	36.000	2.678
	Total :	159.209	143.288	15921

Se realizo el proceso de aprendizaje con los diferentes conjuntos de datos de las tablas yel proceso de aprendizaje con cincuenta muestras en cien iteraciones con tráfico de intrusiones al servicio SSH se puede ver en la Figura 87, en donde se aprecia que converge en aproximadamente cuarenta millones de actualizaciones de los pesos. En las Figuras 88, 89 y 90 se presenta el proceso de entrenamiento con docientas, quinientas y mil muestras en docientas iteraciones, donde se puede apreciar que converge en todos los casos.

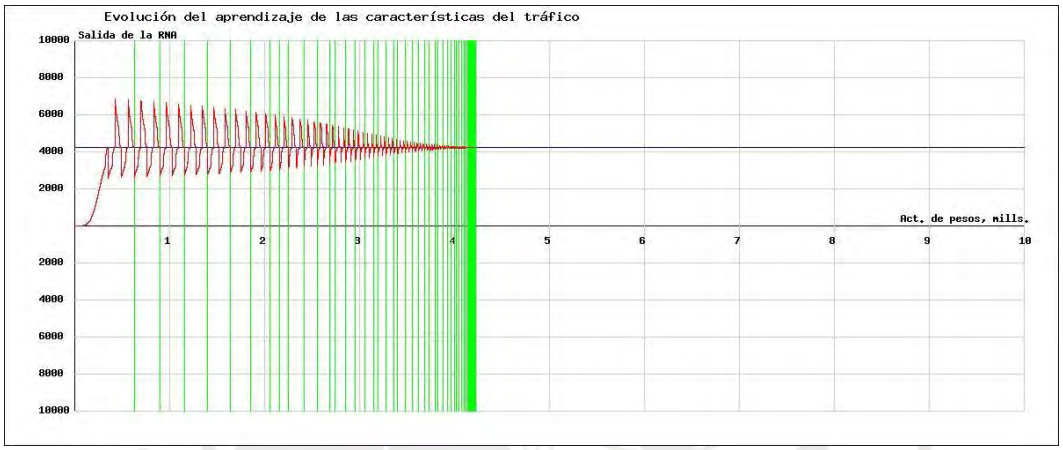


Figura 87. Proceso de aprendizaje con 50 muestras en 100 iteraciones para trafico de intrusión al servicio SSH.

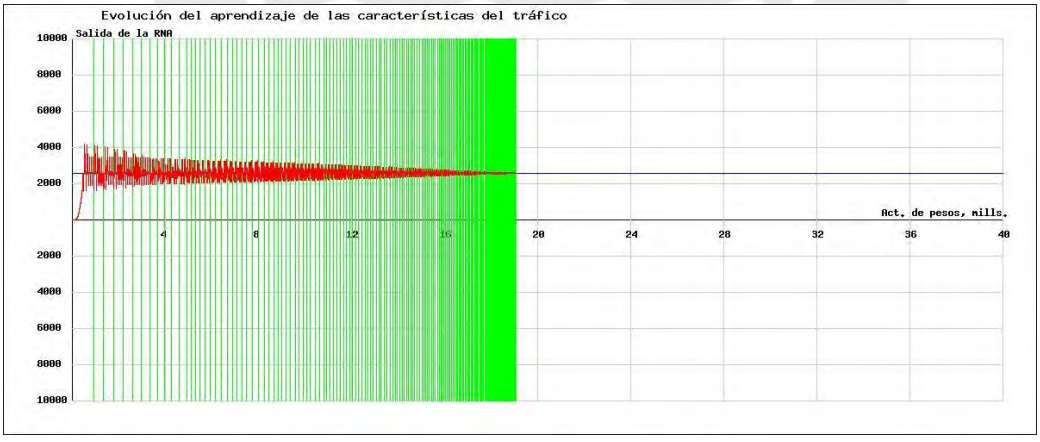


Figura 88. Proceso de aprendizaje con 200 muestras en 200 iteraciones para trafico de intrusión al servicio SSH.

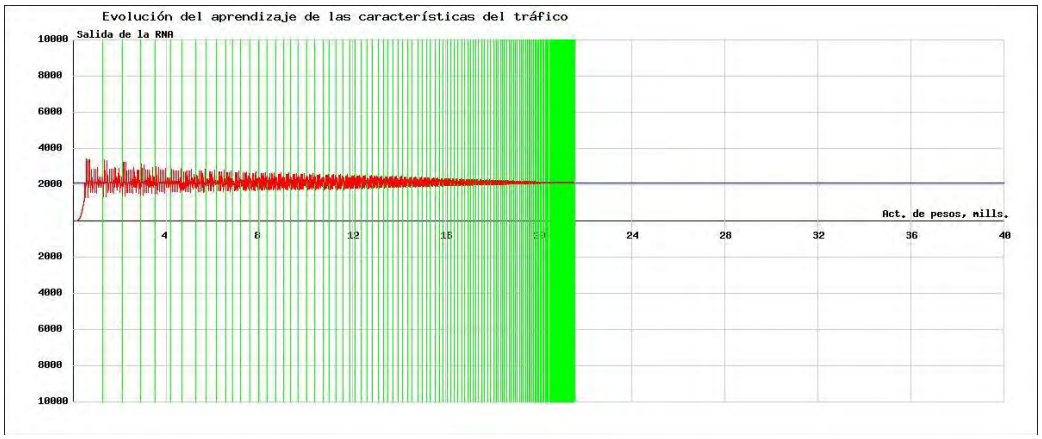


Figura 89. Proceso de aprendizaje con 500 muestras en 200 iteraciones para trafico de intrusión al servicio SSH.

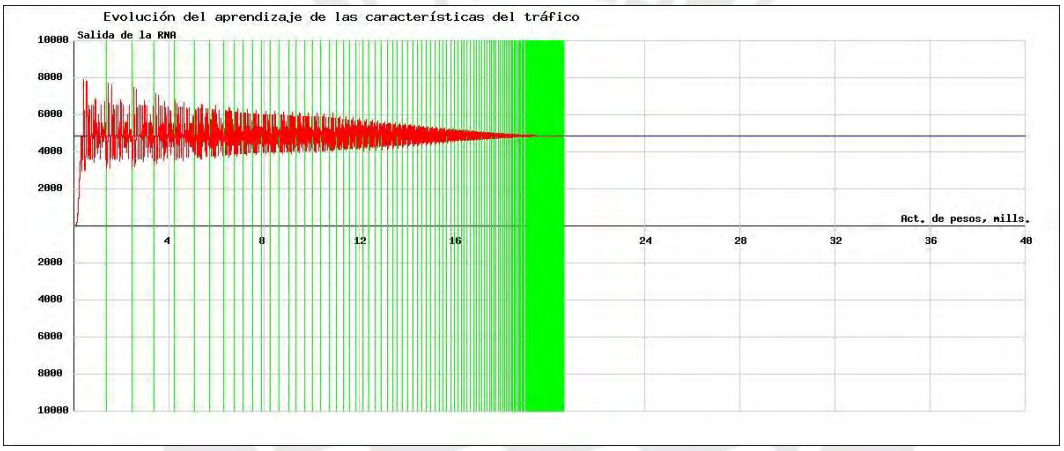


Figura 90. Proceso de aprendizaje con 1000 muestras en 200 iteraciones para trafico de intrusión al servicio SSH.

7.3. Pruebas de reconocimiento de criterios de seguridad generando ataques simulados

Las realizó el proceso de aprendizaje con dos mil muestras de intrusiones al servicio del SSH y se probo con tres mil muestras y se obtuvo un 99.8% de reconocimiento como se presenta en la Figura 105. Con los otros tipos de trafico se obtuvo comportamiento semejante.

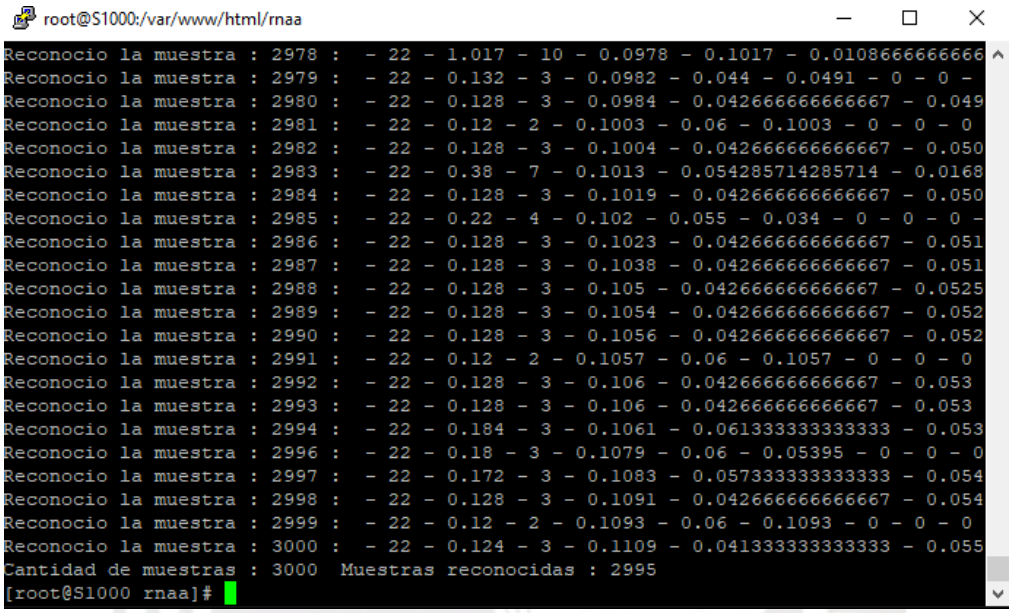


Figura 91. Resultado de pruebas con tres mil muestras.

7.4. Pruebas de configuración automática con las reglas de los nuevos criterios de seguridad del firewall

Las conexiones de tráfico son analizadas por la RNA de reconocimiento de intrusiones y cada una de ellas marca la conexión si reconoció o no reconoció y se bloquea las conexiones que estén marcado como intrusión. El bloqueo se realiza a la IP fuente si la conexión se origino desde internet o a la IP destino si la conexión se origino en la red interna, mediante una regla en iptables.

En la Figura 94, se presenta conexiones marcadas con el campo *rnan* de trafico normal en estado 22 que representa no reconocido como trafico normal y el campo *rnai* marcado con 01 que representa reconocido como conexión de intrusión, con esta información el modulo de auto configuración crea una

regla que bloquea el IP 218.92.0.121 como se muestra en la Figura 95 con una regla en iptables y almacena la reglas en el archivo firewall.

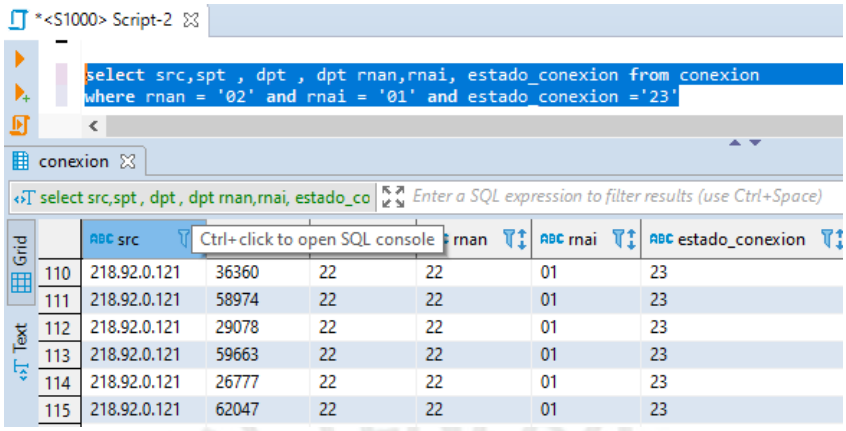


Figura 92. Resultado de pruebas con tres mil muestras.

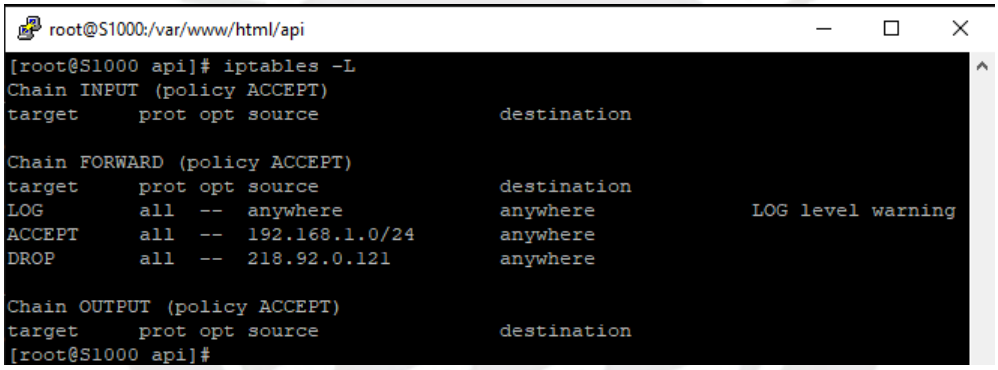


Figura 93. Canal FORWARD con la regla de bloqueo al IP 218.92.0.121.

7.5. Prueba final del sistema con ataques simulados

El sistema completo se implementa mediante un script como el que se muestra en la Figura 96, que se ejecuta cada minuto mediante la programación de tareas con *crontab*, en le cual se se hace una copia de los logs con el programa *copialog.php*, se remueve los archivos *kernelsub* y *kernel* y se resetea el sistema *rsyslog* de captura de logs, seguido el programa *subelogpg.php* sube los logs a la base de datos, luego el programa *extrae_caracter.php* procesa los logs para extraer las características de las conexiones, conexiones que son analizadas por el programa *operacionRNA_I.php* para la detección de intrusiones mediante el uso de RNAs y finalmente el programa *autoConfig.php* crea reglas de iptables para bloquear el trafico si hay intrusiones.

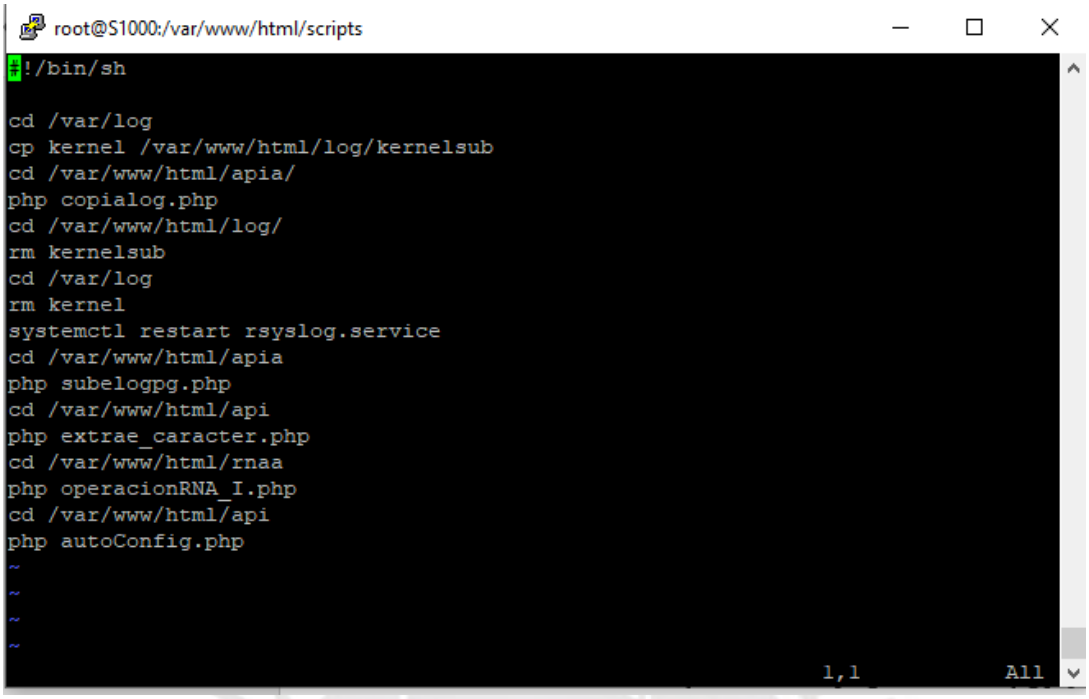
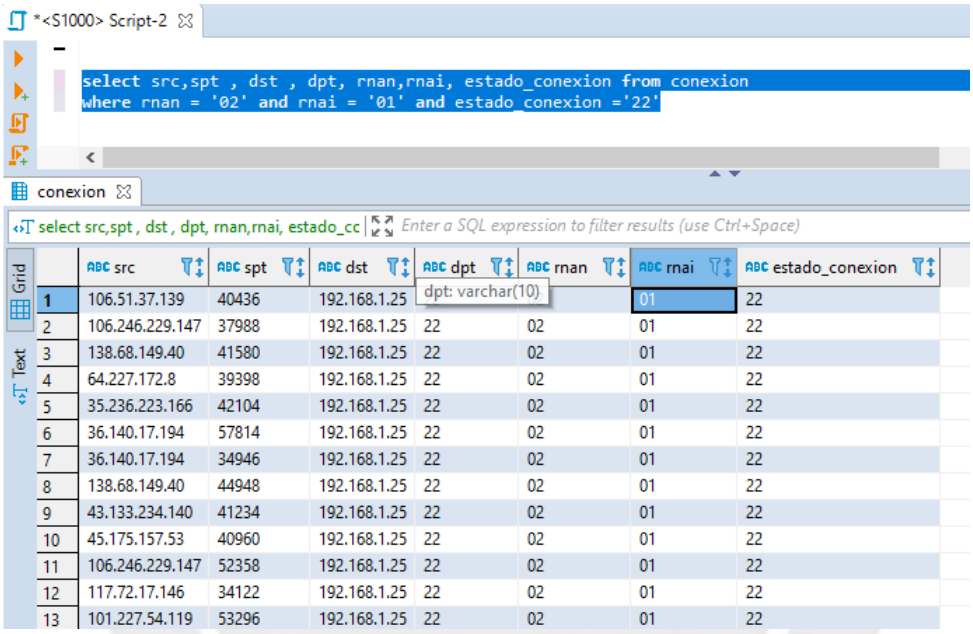


Figura 94. Script de ejecución de los modulos del sistema.

Las pruebas se realizan para los ataques de fuerza al que estan expuestos constantemente los servidores en le servcio SSH como el servidor interno con el IP 192.168.1.25 como se muestra en las conexiones de la Figura 97, las cuales son bloquedas por el sistema con reglas de iptables como se presenta en la Figura 98, donde en la cadena FORWARD se puede ver las diferentes reglas de bloqueo.



	src	spt	dst	dpt	rnan	rnai	estado_conexion
1	106.51.37.139	40436	192.168.1.25			01	22
2	106.246.229.147	37988	192.168.1.25	22	02	01	22
3	138.68.149.40	41580	192.168.1.25	22	02	01	22
4	64.227.172.8	39398	192.168.1.25	22	02	01	22
5	35.236.223.166	42104	192.168.1.25	22	02	01	22
6	36.140.17.194	57814	192.168.1.25	22	02	01	22
7	36.140.17.194	34946	192.168.1.25	22	02	01	22
8	138.68.149.40	44948	192.168.1.25	22	02	01	22
9	43.133.234.140	41234	192.168.1.25	22	02	01	22
10	45.175.157.53	40960	192.168.1.25	22	02	01	22
11	106.246.229.147	52358	192.168.1.25	22	02	01	22
12	117.72.17.146	34122	192.168.1.25	22	02	01	22
13	101.227.54.119	53296	192.168.1.25	22	02	01	22

Figura 95. Conexiones al servicio SSH en el puerto 22 del servidor con el IP 192.168.1.25.

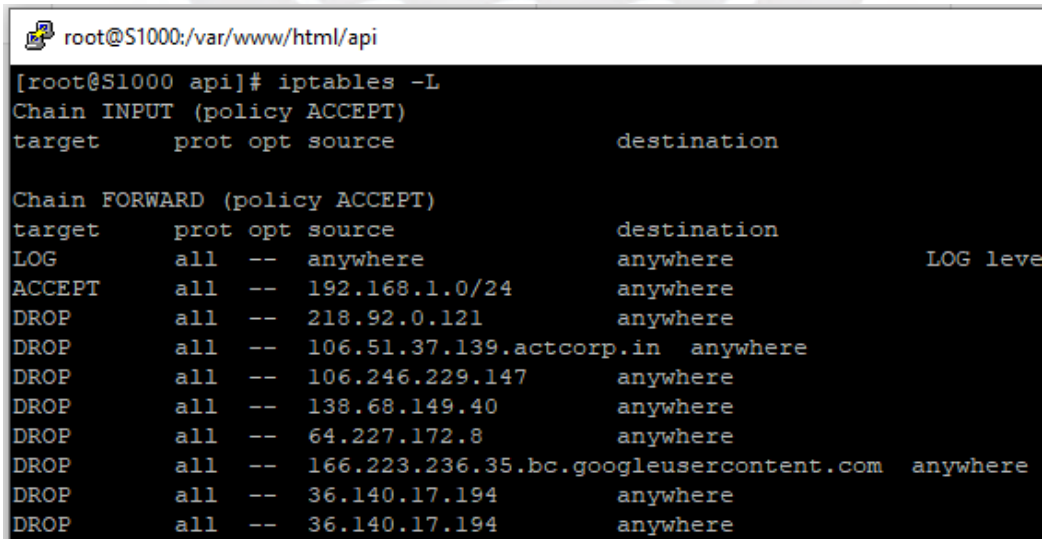


Figura 96. Reglas en la cadena FORWARD que bloquea ips externas.

8. Conclusiones y trabajos futuro

8.1. Conclusiones

Para la implementación del sistema de seguridad centralizado en el firewall de la red a proteger se realizó el ensamblaje de hardware físico del servidor con procesador Core I7, 32 GB de memoria RAM, 1 TB de dispositivo de almacenamiento y dos tarjetas de red, el cual fue conectado a la red separando la red local interna de la red externa internet.

Se realizó la implementación de Linux Fedora con soporte de PostgreSQL para bases de datos, Apache para servcio web con soporte de PHP y se configuro la captura de registros de paquetes IP con *iptables* en un archivo para el posterior procesamiento.

Con los resultados obtenidos, se logró cumplir el objetivo específico de implementar el sistema de seguridad centralizado en el firewall de la red que se desea asegurar.

La implementación de módulo de reconocimiento de criterios de seguridad mediante patrones de trafico anómalo se realizó diseñando la RNA, diseñando el algoritmo de aprendizaje de la RNA, diseñando el algoritmo de operación de la RNA, implementando el algoritmo de aprendizaje de la RNA, implementando el algoritmo de operación de la RNA.

Se implementó la auto configuración del firewall mediante un programa en php que leyendo la información de las conexiones de intrusiones, genera una regla con iptables y ejecuta en comando en el Shell de Linux para cargarla y que empiece ha actuar inmediatamente.

Finalmente se realizó las pruebas individuales con el tráfico anómalo de intrusiones al servicio de SSH y las pruebas finales con el sistema implementado para el mismo tipo de intrusión obteniendo un resultado de auto configuración del firewall creando reglas de bloqueos a las conexiones de intrusiones.

La captura de muestras de trafico anómalo mediante la captuta de logs del trafico de ataques simulados garantiza que se obtiene los patrones del trafico anómalo de intrusiones de red, que al ser utilizados para entenar las RNA que reconocen el trafico anómalo y bloquean a todos los hosts que generen este tipo de tráfico, aunque dichos host aatquen por primera vez a la red.

La propuesta es de aplicación practica en soluciones reales ya que se implementó y se probó en un entorno real con herramientas de uso común.

8.2. Trabajos futuros

Para el sistema implementado, completar el aprendizaje de la RNAs para cubrir todos los tipos de ataques a la red y mejorar la detección y bloqueo de conexiones de intrusión.

También se debe desarrollar módulos de análisis de datos de conexiones de trafico utilizando otros principios inteligentes basados en algoritmos bioinspirados e integrar con los módulos existentes para mejorar el desempeño del sistema.

Referencias

Ali Hussein Shamman Al-Safi, Z. I. (2021). Using A Hybrid Algorithm and Feature Selection for Network Anomaly Intrusion Detection. *Journal of Mechanical Engineering Research and Developments*, 253-262.

Almomani, O. (2021). A Hybrid Model Using Bio-Inspired Metaheuristic Algorithms for Network Intrusion Detection System. *Computers, Materials & Continua*, 409-429.

Assiri, A. (2020). Anomaly Classification Using Genetic Algorithm-Based Random Forest Model for Network Attack Detection. *Computers, Materials & Continua*, 767-778.

Bukac, V. T. (2012). Advances and Challenges in Standalone Host-Based Intrusion Detection Systems. In: Fischer-Hübner, S., Katsikas, S., Quirchmayr, G. (eds) Trust. *Privacy and Security in Digital Business*.

D. Bringhenti, G. M. (2020). Introducing programmability and automation in the synthesis of virtual firewall rules. *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, 473-478.

D. Stiawan, A. H. (2010). The trends of Intrusion Prevention System network. *2010 2nd International Conference on Education Technology and Computer*, 217 - 221.

Glesias, F. Z. (2015). Analysis of network traffic features for anomaly detection. *Machine Learning*, 59–84.

Gundecha, P. a. (2011). Exploiting vulnerability to secure user privacy on a social networking site. *Association for Computing Machinery*, 511–519.

Gustavo González-Granadillo, S. G.-Z. (2021). Security Information and Event Management (SIEM): Analysis, Trends, and Usage in Critical Infrastructures. *Sensors*, 4759-4769.

H. Shiravi, A. S. (2011). A Survey of Visualization Systems for Network Security. *IEEE Transactions on Visualization and Computer Graphics*, 1313-1329.

Jakub Svoboda, I. G. (2015). Network Monitoring Approaches: An Overview. *International Journal of Advances in Computer Networks and Its Security– IJCNS*, 88-93.

Kim, J. L. (2022). Evolution of Firewalls: Toward Securer Network Using Next Generation Firewall. *2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)*, 0752-0759.

Kitchenham, B. a. (2007). Guidelines for performing systematic literature reviews in software engineering version 2.3. *Engineering*, 1051.

Krupa C. Patel, D. P. (2017). A Review paper on pfsense – an Open source firewall introducing with different capabilities & customization. *IJARIIE* , 636-641.

Marin, G. A. (2005). Network security basics. *IEEE Security & Privacy*, 68-72.

Movva Sai Chaithanya, S. N. (2021). Intelligent IDS: Venus Fly-trap Optimization with Honeypot Approach for Intrusion Detection and Prevention. *Research square*.

Paulikas, D. M. (2021). Analysis of Linux OS security tools for packet filtering and processing. *SFL*.

Reza Rafeh, M. N. (2022). A Hybrid Evolutionary Algorithm for Anomaly Detection in Computer Networks. *Research Square*.

S. Helal, S. S. (2002). The Internet enterprise. *Proceedings 2002 Symposium on Applications and the Internet (SAINT 2002)*, 54-62.

S. Jin, Y. W. (2009). A review of classification methods for network vulnerability. *2009 IEEE International Conference on Systems, Man and Cybernetics*, 1171-1175.

Server, W. (24 de 11 de 2022). *Install Apache httpd*. Obtenido de https://www.server-world.info/en/note?os=Fedora_37&p=httpd&f=1

Server, W. (24 de 11 de 2022). *Install PHP*. Obtenido de https://www.server-world.info/en/note?os=Fedora_37&p=nginx&f=7

Server, W. (24 de 11 de 2022). *MariaDB 10.9 Install*. Obtenido de https://www.server-world.info/en/note?os=Fedora_37&p=mariadb&f=1

Soodeh Hosseini, B. M. (2020). New hybrid method for attack detection using combination of evolutionary algorithms, SVM, and ANN,. *Computer Networks*.

Tanenbaum, A. S. (2005). *Redes de Computadoras.* Madrid: Imprenta.

Watkins, L. (2020). Bio-Inspired, Host-based Firewall. *2020 IEEE 23rd International Conference on Computational Science and Engineering (CSE)*, 86-91.

Zouhair Chiba, M. S. (2022). Automatic Building of a Powerful IDS for The Cloud Based on Deep Neural Network by Using a Novel Combination of Simulated Annealing Algorithm and Improved Self-Adaptive Genetic Algorithm . *International Journal of Communication Networks and Information Security*, 93-117.

Apéndices

APÉNDICE A: Programa en php estraer caracteridticas del trafico anómalo desde los logs.

```
function extrae_caracter_conexion($databaseC) {
        $num_conex=0;
        $flag_log=1;
        while ($flag_log=='1'){
                $flag_log=0;
                $j=0;
                try {
                    $cListConex = $databaseC->prepare("select count(*) as conta, src, spt, dst, dpt from logs where chain='FORWARD'
and
                     syn = '1' and estado_log ='1' and proto ='TCP' group by src, spt, dst, dpt order by conta desc limit 10000");
                    $cListConex->execute();
                     while ($filaListConex = $cListConex->fetch(PDO::FETCH_ASSOC)) {
                            $flag_log=1;
                            $Cant = $filaListConex["conta"];
                            $ipFu   = $filaListConex["src"];
                            $ipDe   = $filaListConex["dst"];
                            $puFu   = $filaListConex["spt"];
                            $puDe   = $filaListConex["dpt"];
                            $j++;
                            if ($j%1000==0){
                                    printf("%10s%10s%20s%20s%10s%10s\n\n",$j,$Cant, $ipFu, $ipDe, $puFu, $puDe);
                            }
                            $i=0;
                            $vol_tra_conex=0;
                            $num_pqt_conex=0;
                            $tam_med_paket=0;
                            $duracin_conex=0;
                            $tmp_pro_conex=0;
                            $tiempo_anterior=0;
                            $tiempo_actual  =0;
                            $fecha_ini = "";
                            $hora_ini ="";
                            $fecha_fin = "";
                            $hora_fin ="";
                            try{
                                    if ($num_conex%1000==0){
                                      printf("%10s%10s%20s%20s%10s%10s%15s%20s%5s%5s%5s%5s%5s%5s%5s%8s\n","No","Id
                                      Log", "Ip Fte", "Ip Dst", "Pto Fte", "Pto Dst", "Fecha", "Hora", "SYN", "ACK", "FIN", "PSH",
                                      "RES", "URG", "LENGTH");
                                    }
                                    $cConex = $databaseC->prepare("select id_log, src, dst, spt, dpt,fecha, hora, syn,
ack,fin,psh,res,urgp,
                                    lenght, diezmilesima from logs where chain='FORWARD' and ((src='$ipDe' and dst='$ipFu')
                                    or (dst='$ipDe' and src='$ipFu')) and estado_log ='1' and proto ='TCP' order by fecha, hora");
                                    $cConex->execute();
                                    while ($filaConex = $cConex->fetch(PDO::FETCH_ASSOC)) {
                                            $i++;
                                            $idL = $filaConex["id_log"];
                                            $ipF   = $filaConex["src"];
                                            $ipD   = $filaConex["dst"];
                                            $ptF   = $filaConex["spt"];
                                            $ptD = $filaConex["dpt"];
                                            $fec = $filaConex["fecha"];
                                            $hor = $filaConex["hora"];
                                            $dmsg = $filaConex["diezmilesima"];
                                            $synf   = $filaConex["syn"];
                                            $ackf   = $filaConex["ack"];
                                            $finf = $filaConex["fin"];
                                            $pshf = $filaConex["psh"];
                                            $resf = $filaConex["res"];
                                            $urgf   = $filaConex["urgp"];
                                            $logi = $filaConex["lenght"];
                                            if ($synf==1 and $ackf==0){
                                               if ($num_pqt_conex==0){
```

```
                    $tam_med_paket=0;
                }
                else{
                    $tam_med_paket=$vol_tra_conex/$num_pqt_conex;
                }
                if ($num_pqt_conex==1){
                    $tmp_pro_conex=0;
                }
                else{
                    $tmp_pro_conex=$duracin_conex/($num_pqt_conex-1);
                }
                $tiempo_anterior=strtotime($fec." ".$hor)*10000 + (int)substr($hor,9,4);
                if ($vol_tra_conex>0){
                    $num_conex++;
                    if ($num_conex%1000==0){
                        printf("%30s%10s\n","Volumen : ",$vol_tra_conex);
                        printf("%30s%10.2f\n","Tam. medio paket : ",$tam_med_paket);
                        printf("%30s%10.2f\n","Duracion conex : ",$duracin_conex);
                        printf("%30s%10.2f\n","Tiempo pro paket : ",$tmp_pro_conex);
                        printf("%30s%10s\n\n","Num. paquetes : ",$num_pqt_conex);
                    }
                    try{
                        $stCpru = $databaseC->prepare("insert into conexion (src , dst ,
spt ,dpt
                        ,tamano ,cantidad_paquetes , duracion ,tamano_medio_paquetes ,
                        tiempo_promedio_entre_paquetes, fecha_inicio, hora_inicio,
                        fecha_fin, hora_fin)
                        values('$ipF','$ipD','$ptF','$ptD','$vol_tra_conex',
                        '$num_pqt_conex','$duracin_conex','$tam_med_paket','$tmp_pro_
                        conex','$fecha_ini','$hora_ini','$fecha_fin','$hora_fin')");
                        $databaseC->beginTransaction();
                        $stCpru->execute();
                        $databaseC->commit();
                    } catch (Exception $e) {
                        $databaseC->rollBack();
                        echo "Failed: " . $e->getMessage();
                    }
                }
                try{
                    $cActEs = $databaseC->prepare("update logs set estado_log ='2' where id_log
                    ='".$idL."'");
                    $databaseC->beginTransaction();
                    $cActEs->execute();
                    $databaseC->commit();
                } catch (Exception $e) {
                    $databaseC->rollBack();
                    echo "Failed: " . $e->getMessage();
                }
                if ($num_conex%1000==0){
                printf("%10s%10s%20s%20s%10s%10s%15s%20s%5s%5s%5s%5s%5s%
                5s%5s%8s\n",$i,$idL, $ipF, $ipD, $ptF, $ptD, $fec, $hor, $synf, $ackf,
                $finf, $pshf, $resf, $urgf, $logi);
            }
            $vol_tra_conex=$logi;
            $num_pqt_conex=1;
            $duracin_conex=0;
            $fecha_ini = $fec;
            $hora_ini = $hor;
        }
        else{
            //$tiempo_dms=strtotime($fec." ".$hor)*10000 + substr($hor,9,4);
            if ($num_conex%1000==0){
            printf("%10s%10s%20s%20s%10s%10s%15s%20s%5s%5s%5s%5s%5s%5s%8s\n",$i,$idL,
            $ipF, $ipD, $ptF, $ptD, $fec, $hor, $synf, $ackf, $finf, $pshf, $resf, $urgf, $logi);
            }
            try{
                $cActEs = $databaseC->prepare("update logs set estado_log ='2' where id_log ='".$idL."'");
                $databaseC->beginTransaction();
                $cActEs->execute();
                $databaseC->commit();
```

```
                                        } catch (Exception $e) {
                                                $databaseC->rollBack();
                                                echo "Failed: " . $e->getMessage();
                                        }
                                        $vol_tra_conex=$vol_tra_conex+$logi;
                                        $num_pqt_conex++;
                                        $tiempo_actual=strtotime($fec." ".$hor)*10000 + (int)substr($hor,9,4);
                                        $duracin_conex=$duracin_conex + $tiempo_actual-$tiempo_anterior;
                                        $tiempo_anterior=$tiempo_actual;
                                        $fecha_fin = $fec;
                                        $hora_fin = $hor;
                                        }
                                }
                                } catch (Exception $e) {
                                        echo "Failed: " . $e->getMessage();
                                }
                        }
} catch (Exception $e) {
        echo "Failed: " . $e->getMessage();
}
}
```

APÉNDICE B: Programa en php para crear la RNA.

```
function creaRNA($databaseC){
	$layer=0;
	$neuron_number=0;
	$entry_number=0;
	try {
		$d_weight = $databaseC->prepare("delete from peso");
		$d_neuron = $databaseC->prepare("delete from neurona");
		$databaseC->beginTransaction();
		$d_weight ->execute();
		$d_neuron ->execute();
		$databaseC->commit();
		} catch (Exception $e) {
			$databaseC->rollBack();
			echo "Failed: " . $e->getMessage();
		}
	try {
		$sdw = $databaseC->prepare("select id_estructura_neurona, dato_estructura_neurona, id_rna from estructura_neurona");
		$sdw->execute();
		while ($output = $sdw->fetch(PDO::FETCH_ASSOC)) {
			$indEs =$output['id_estructura_neurona'];
			$EstrDato = $output['dato_estructura_neurona'];
			$idrna	= $output['id_rna'];
			//print("RNAs : ".$idrna."\n")
			if ($indEs ==1){
			  $numero_capas = $EstrDato;
			  $layer=0;
			  $neuron_number=0;
			}
			if ($indEs==2){
			  print("*****  ANR : ".$idrna."	Capa : ".$layer." *******\n");
			 $layer++;
			 $entry_number=10;
			  for ($i=1; $i<=$datoEstruct; $i++){
			  $neuron_number++;
			  print("Neurona : ".$neuron_number."\n");
			 try {
				$m[$neuron_number] = $entry_number;
				$caparna[$neuron_number] = $layer;
				$n[$neuron_number] = 0;
				$a[$neuron_number] = 0;
				$k[$neuron_number] = 1;
				$u[$neuron_number] = rand(800, 1800);
				$statement = $databaseC->prepare("insert into neurona (id_rna,id_neurona,m,capa,n,a,k,u)
				values('$idrna','$neuron_number','$entry_number','$layer','0','0','1','$u[$neuron_number]')");
				$databaseC->beginTransaction();
				$statement->execute();
				$databaseC->commit();
				for ($l=1; $l<=$entry_number; $l++){
					try {
					    $weight = rand(0, 10000) / 10000000;
					     $st_weigth = $databaseC->prepare("insert into peso
					     (id_rna,id_peso,id_neurona,valor_peso)
					     values('$idrna','$j','$neuron_number','$weight')");
					    $databaseC->beginTransaction();
					    $st_weigth->execute();
					    $databaseC->commit();
					    $pesorna[$neuron_number][$j] = $weight;
					    print("		Weigth : ".$j." Value : ".$weight."\n");
					 } catch (Exception $err) {
					  $databaseC->rollBack();
					  echo "Failed: " . $err->getMessage();
					 }
				}
			    } catch (Exception $err) {
```

```
                                        $databaseC->rollBack();
                                        echo "Failed: " . $err->getMessage();
                                }
                        }
                        $entry_number=$datoEstruct;
                }
                if ($idEstruc>2){
                        print("******  ANR : ".$idrna."       Layer : ".$layer." ******\n");
                        $layer++;
                        for ($l=1; $l<=$datoEstruct; $l++){
                                $neuron_number++;
                                print("Neuron : ".$neuron_number."\n");
                                try {
                                        $m[$neuron_number] = $entry_number;
                                        $caparna[$neuron_number] = $layer;
                                        $n[$neuron_number] = 0;
                                        $a[$neuron_number] = 0;
                                        $k[$neuron_number] = 1;
                                        $u[$neuron_number] = rand(800, 1800);
                                        $statement = $databaseC->prepare("insert into neurona
                                        (id_rna,id_neurona,m,capa,n,a,k,u)
                                        values('$idrna','$neuron_number','$entry_number','$layer','0','0','1','$u[$neuron_number]')
                                        ");
                                        $databaseC->beginTransaction();
                                        $statement->execute();
                                        $databaseC->commit();
                                        for ($k=1; $k<=$entry_number; $k++){
                                                try {
                                                        $weight=rand(0, 10000) / 10000000;
                                                        $st_weigth = $databaseC->prepare("insert into peso
                                                        (id_rna,id_peso,id_neurona,valor_peso)
                                                        values('$idrna','$j','$neuron_number','$weight')");
                                                        $databaseC->beginTransaction();
                                                        $st_weigth->execute();
                                                        $databaseC->commit();
                                                        $pesorna[$neuron_number][$j] = $weight;
                                                        print("              Peso : ".$j." Valor : ".$weight."\n");
                                                } catch (Exception $err) {
                                                        $databaseC->rollBack();
                                                        echo "Failed: " . $err->getMessage();
                                                }
                                        }

                                } catch (Exception $err) {
                                        $databaseC->rollBack();
                                        echo "Failed: " . $err->getMessage();
                                }
                }
                $entry_number=$datoEstruct;
        }
      }
    }
    catch (Exception $err) {
    Echo "Failed: " . $err->getMessage();
    }
}
```

APÉNDICE C: Programa en PHP para realizar el entrenamiento de la red.

```
function entrenamiento($databaseC,$databasePG){
        $CalANR = new calculoRNA();
        $layer=0;
        $neuron_number=0;
        $entry_number=0;
        print "******** ANN Learning *******\n\n";
        try {
            $cadc = $databaseC->prepare("select id_rna, id_estructura_neurona, dato_estructura_neurona from ESTRUCTURA_NEURONA
            order by id_rna,id_estructura_neurona");
            $cadc->execute();
            while ($output = $cadc->fetch(PDO::FETCH_ASSOC)) {
                    $indANR = $output['id_rna'];
                    $indEst = $output['id_estructura_neurona'];
                    $EstDato = $output['dato_estructura_neurona'];
                    $StrANRMtr[$indANR][$indEst-1] = $EstDato;
            }
        try {
            $cdneu = $databaseC->prepare("select id_neurona, u, k, a, n, capa, m, id_rna from NEURONA order by id_rna, id_neurona");
            $cdneu ->execute();
            $neuron_number=0;
            while ($outneu = $cdneu ->fetch(PDO::FETCH_ASSOC)) {
                    numNeuro++;
                    $caparna[$outneu['id_neurona']] = $outneu['capa'];
                    $n[$outneu['id_neurona']] =$outneu['n'];
                    $a[$outneu['id_neurona']] =$outneu['a'];
                    $k[$outneu['id_neurona']] =$outneu['k'];
                    $u[$outneu['id_neurona']] = $outneu['u'];*/
                    $cmMtr[$outneu['id_rna']][$outneu['id_neurona']] = $outneu['m'];
                    $layANRMtr[$outneu['id_rna']][$outneu['id_neurona']] = $outneu['capa'];
                    $nMtr[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['n'];
                    $avMt[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['a'];
                    $kvMtr[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['k'];
                    $uvMtr[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['u'];
            }
        } catch (Exception $err) {
                Echo "Failed: " . $err->getMessage();
        }
        try {
            $st_weigth = $databaseC->prepare("select id_rna,id_neurona,id_peso,valor_peso from PESO order by
            id_rna,id_neurona,id_peso");
            $st_weigth->execute();
            while ($outweigth = $st_weigth->fetch(PDO::FETCH_ASSOC)) {
                    $weiANRMt[$outweigth['id_rna']][$outweigth['id_neurona']][$outweigth['id_peso']] = $outweigth['valor_peso'];
             }
        }catch (Exception $err) {
                Echo "Failed: " . $err->getMessage();
        }
        print "Cantidad de neuronas: ".$neuron_number."\n";
        print "\n Arreglo: "."\n\n";
        for ($iANR=1; $iANR<sizeof($weiANRMt)+1; $iANR++){
            $id_ANR=str_pad($iANR, 5, "0", STR_PAD_LEFT);
            print "\n**************ANR :".$id_ANR."*********************\n\n";
            for ($ineur=1; $ineur<=sizeof($weiANRMt[$in_arn]); $ineur++){
                $id_neur=str_pad($ineur, 1, "0", STR_PAD_LEFT);
                print "\nNeuron : ".$id_neur."--------------------------- \n\n";
                for ($nw=1; $nw<=sizeof($weiANRMt[$in_arn][$id_neur]); $nw++){
                    $nwe=str_pad($nw, 1, "0", STR_PAD_LEFT);
                    $wVal=$weiANRMt[$in_arn][$id_neur][$nwe];
                    print sprintf("%15f",$wVal)
                }
                print "\n\n ";
            }
            print "\n\n ";
        }
        print "\n\n ";
```

```
/******** Entrenamiento **********/
try {
    try {
        $cadConf = $databaseC->prepare("select maximo_valor_variable_x, maximo_valor_variable_y, maximo_valor_x_pixel,
        maximo_valor_y_pixel, centro_coordenadas_x, centro_coordenadas_y, numero_iteraciones, velocidad_aprendizaje,
        coeficiente_x_sigma, coeficiente_y_sigma, umbral_salida from CONFIGURACION where codigo_configuracion='00001'");
        $cadConf->execute();
        if ($outconf = $cadConf->fetch(PDO::FETCH_ASSOC)) {
           $xValMaximo =$outconf['maximo_valor_variable_x'];
           $yValMaximo =$outconf['maximo_valor_variable_y'];
           $xMaximo    =$outconf['maximo_valor_x_pixel'];
           $yMaximo    =$outconf['maximo_valor_y_pixel'];
           $xCentro    =$outconf['centro_coordenadas_x'];
           $yCentro    =$outconf['centro_coordenadas_y'];
           $numIter    = $outconf['numero_iteraciones'];
           $VelApr     =$outconf['velocidad_aprendizaje'];
           $coefSigy    =$outconf['coeficiente_x_sigma'];
           $coefSigy    =$outconf['coeficiente_y_sigma'];
           $umbSal     = $outconf['umbral_salida'];
         }
    } catch (Exception $err) {
            Echo "Failed: " . $err->getMessage();
    }
    $yCentro = $yMaximo/2;      $xZonGraf = $xMaximo - 120;
    $yZonGraf = $yMaximo/2 - 40;            $escax = $xZonGraf/$xValMaximo;
    $escay = $yZonGraf/$yValMaximo;      $inXcrd = $xZonGraf/10;
    $inVeXM = $xValMaximo/10000000; $inYcrd = $yZonGraf/5;
    $inVeYM = $yValMaximo/5;              $nAtaPs = 0;
    $CntAtuPs = 0; $inNrn = 1;
    $SumaANR = 0;  $outANR = 0;           $outANRa = 0;
    $imagenC = imagecreatetruecolor($xMaximo, $yMaximo);
    $Columbral = imagecolorallocate($imagenC, 0, 0, 255);
    $iteraC = imagecolorallocate($imagenC, 0, 255, 0);
    $fonCol = imagecolorallocate($imagenC, 255, 255,255);
    $ColorTex = imagecolorallocate($imagenC, 233, 14,91);
    $ColLinea = imagecolorallocate($imagenC, 0, 0, 0);
    $outcolor = imagecolorallocate($imagenC, 255, 0, 0);
    $reja_clr = imagecolorallocate($imagenC, 205, 205, 205);
    imagefilledrectangle($imagenC,0,0,$xMaximo, $yMaximo,$ColLinea);
    imagefilledrectangle($imagenC,1,1,$xMaximo-2, $yMaximo-2,$fonCol);
    imagestring($imagenC, 5,$xCentro+35, 5, 'Entrenamiento de características del tráfico de red', $ColLinea);
    imageline ($imagenC, $xCentro, $yCentro, $xCentro+$xZonGraf, $yCentro,$ColLinea);
    imageline ($imagenC, $xCentro, $yCentro-$yZonGraf, $xCentro, $yCentro+$yZonGraf,$ColLinea);
    imagestring($imagenC, 3, $xCentro+5, $yCentro-$yZonGraf-13, "Salida de la RNA", $ColLinea);
    for ($lx=1; $lx<=10; $lx++){
        imageline ($imagenC, $xCentro+$lx*$inXcrd, $yCentro+$yZonGraf, $xCentro+$lx*$inXcrd, $yCentro-
        $yZonGraf,$reja_clr);
        imagestring($imagenC, 3, $xCentro+$lx*$inXcrd-5, $yCentro+10,$lx*$inVeXM, $ColLinea);
    }
    for ($ly=1; $ly<=5; $ly++){
        imageline ($imagenC, $xCentro, $yCentro-$ly*$inYcrd, $xCentro+$xZonGraf, $yCentro-$ly*$inYcrd,$reja_clr);
        imageline ($imagenC, $xCentro, $yCentro+$ly*$inYcrd, $xCentro+$xZonGraf, $yCentro+$ly*$inYcrd,$reja_clr);
        imagestring($imagenC, 3, $xCentro-40, $yCentro+$ly*$inYcrd-8,$ly*$inVeYM, $ColLinea);
        imagestring($imagenC, 3, $xCentro-40, $yCentro-$ly*$inYcrd-8,$iy*$inVeYM, $ColLinea);
    }
    imagestring($imagenC, 3, $xCentro+$xZonGraf-140, $yCentro-15, "Act. de pesos, mills.", $ColLinea);
    print "Cantidad de iter : ".$numIter."\n\n";
    $neuron_number=0;
    for ($i_e=1; $i_e<=sizeof($StrANRMtr[str_pad(1, 5, "0", STR_PAD_LEFT)])-1; $i_e++){
        $neuron_number = $neuron_number + $StrANRMtr[str_pad(1, 5, "0", STR_PAD_LEFT)][$i_e];
    }
    print "Cantidad de neu: ".$neuron_number."\n";
    $in_arnN = 1;
    $in_arn = str_pad($in_arnN, 5, "0", STR_PAD_LEFT);
    $nNeARN = sizeof($weiANRMt[$in_arn]);
    $umbSal = $uvMtr[$in_arn][$nNeARN];
    imageline ($imagenC,$xCentro+round($escax*$CntAtuPs),$yCentro-round($escay*$umbSal),$xCentro+$xZonGraf,$yCentro-
    round($escay*$umbSal), $Columbral);
    $numIter=5000;
    for ($initr=0; $initr<$numIter; $initr++){
```

```
$nMtra=0;
print "\n\n****************************** Itera : ".$initr."*********************************\n";
$stTraC = $databasePG->prepare("select spt,dpt,tamano,cantidad_paquetes,duracion , tamano_medio_paquetes,
tiempo_promedio_entre_paquetes from conexion where src = '192.168.1.16' and tiempo_promedio_entre_paquetes<5 order
by id limit 30");
$stTraC->execute();
while ($rTraC = $stTraC->fetch(PDO::FETCH_NUM)) {
        $nMtra++;
        print "\n\n*********************      Muestra ".$nMtra."          ***********\n\n";
        print " ANR : ".$in_arn."\n";
        $long = sizeof($rTraC);
        print "Longitud : ".$long."\n";
        for ($cv=0; $cv<$long; $cv++){
             $inVec[$cv]=$rTraC[$cv];
        }
        for ($cv=$long; $cv<10; $cv++){
             $inVec[$cv]=0;
        }
        for ($cv=0; $cv<10; $cv++){
             print " - ".$inVec[$cv];
        }
        print "\n";
        $outANR = $ANRcal->calculoRNAM($inVec, $nMtr, $cmMtr, $layANRMtr, $avMt,$kvMtr, $uvMtr,$weiANRMt,
        $neuron_number, $StrANRMtr,$in_arn);
        print " Init Val ANR : ".$outANR."\n";
        print " Umbral : ".$umbSal."\n";             $inNrn=1; $inRp=1;
        do{
             $outANR = $ANRcal->calculoRNAM($inVec, $nMtr, $cmMtr, $layANRMtr, $avMt, $kvMtr, $uvMtr,
             $weiANRMt, $neuron_number, $StrANRMtr,$in_arn);
             $incP = $umbSal+10-$outANR;
             if ($layANRMtr[$in_arn][$inNrn]==1){
                if ($incP>0){
                    if ($incP>1000){
                        $incP=1000;
                    }
                    for ($inps=1; $inps<=$cmMtr[$in_arn][$inNrn]; $inps++){
                        $nAtaPs++;
                        $weiANRMt[$in_arn][$inNrn][$inps] = $weiANRMt[$in_arn][$inNrn][$inps] +
                        $VelApr*$incP*$inVec[$inps-1];
                        if (($nAtaPs%1000000)==0) {
                          print sprintf("%15d : %15d : %20f",$inRp,$nAtaPs,$outANR)."\n";
                        }
                    }
                 }
                else{
                        if ($incP<-1000){
                            $incP=-1000;
                        }
                        for ($inps=1; $inps<=$cmMtr[$in_arn][$inNrn]; $inps++){
                            $nAtaPs++;
                            $weiANRMt[$in_arn][$inNrn][$inps] = $weiANRMt[$in_arn][$inNrn][$inps] +
                            $VelApr*$incP*$inVec[$inps-1];
                            if (($nAtaPs%1000000)==0) {
                              print sprintf("%15d : %15d : %20f",$inRp,$nAtaPs,$outANR)."\n";
                            }
                        }

                  }
             }
             else{
                    if ($incP>0){
                      if ($incP>1000){
                        $incP=1000;
                      }
                      for ($inps=1; $inps<=$cmMtr[$in_arn][$inNrn]; $inps++){
                          $nAtaPs++;
                      $weiANRMt[$in_arn][$inNrn][$inps] = $weiANRMt[$in_arn][$inNrn][$inps] + $VelApr*$incP;
                          if (($nAtaPs%1000000)==0) {
                            print sprintf("%15d : %15d : %20f",$inRp,$nAtaPs,$outANR)."\n";
                          }
```

```
                                    }
                                }
                                else{
                                    if ($incP<-1000){
                                      $incP=-1000;
                                    }
                                    for ($inps=1; $inps<=$cmMtr[$in_arn][$inNrn]; $inps++){
                                        $nAtaPs++;
                                        $weiANRMt[$in_arn][$inNrn][$inps] = $weiANRMt[$in_arn][$inNrn][$inps] +
                                        $VelApr*$incP;
                                        if (($nAtaPs%1000000)==0) {
                                          print sprintf("%15d : %15d : %20f",$inRp,$nAtaPs,$outANR)."\n";
                                        }
                                    }
                                }
                            }
                            if ($inNrn<$neuron_number){
                              $inNrn++;
                            }
                            else{
                                $inNrn=1; $inRp++;
                            }
                            imageline ($imagenC,$xCentro+round($escax*$CntAtuPs),$yCentro-
                            round($escay*$outANRa),$xCentro+round($escax*$nAtaPs),$yCentro-round($escay*$outANR),$outcolor);
                            $CntAtuPs=$nAtaPs;
                            $outANRa = $outANR;
                        }while(abs($umbSal+10-$outANR)>10);
                        print "Actualizando la Neurona : ".$inNrn." Interacion : ".$initr." Valor Red : ".$outANR." Valor suma :
                        ".$SumaANR."\n";
                    }
                    imageline($imagenC,$xCentro+round($escax*$nAtaPs),$yCentro+$yZonGraf,$xCentro+round($escax*$nAtaPs),$yCentro-
                    $yZonGraf,$iteraC);
                }
                print "Num de iterac : ".$numIter."\n\n";
                print "Num de actua : ".$nAtaPs."\n\n";
                imagejpeg($imagenC, 'images/aprendeRNA.jpg');
                imagedestroy($imagenC);
            }catch (Exception $e) {
                Echo "Failed: " . $e->getMessage();
            }
            print "********      Guarda los pesos **********\n\n";
            for ($idANRF=1; $idANRF<sizeof($weiANRMt); $idANRF++){
                $in_arn=str_pad($idANRF, 5, "0", STR_PAD_LEFT);
                for ($idNro=1; $idNro<=sizeof($weiANRMt[$in_arn]); $idNro++){
                    $inNuro=str_pad($idNro, 1, "0", STR_PAD_LEFT);
                    for ($cpso=1; $cpso<=sizeof($weiANRMt[$in_arn][$inNuro]); $cpso++){
                        $nupso=str_pad($cpso, 1, "0", STR_PAD_LEFT);
                        $VlPso=$weiANRMt[$in_arn][$inNuro][$nupso];
                        try {
                            $VlPso=$weiANRMt[$in_arn][$idNro][$cpso];
                            $acPso = $databaseC->prepare("update PESO set valor_peso='$VlPso' where id_peso='$nupso' and
                            id_neurona='$inNuro' and id_rna='$in_arn'");
                            $databaseC->beginTransaction();
                            $acPso->execute();
                            $databaseC->commit();
                        } catch (Exception $err) {
                            $databaseC->rollBack();
                            echo "Failed: " . $err->getMessage();
                        }
                    }
                }
            }
            catch (Exception $err) {
                    Echo "Failed: " . $err->getMessage();
            }
}
```

APÉNDICE D: Programa en PHP para la operación de la RNA.

```
function operacion($databaseC,$databasePG){
	$ANRcal = new calculoRNA();
	try {
		$cadCst = $databaseC->prepare("select id_rna, id_estructura_neurona, dato_estructura_neurona from ESTRUCTURA_NEURONA
		order by id_rna,id_estructura_neurona");
		$cadCst->execute();
		while ($output = $cadCst->fetch(PDO::FETCH_ASSOC)) {
			$cdANR = $output['id_rna'];
			$cdStrc = $output['id_estructura_neurona'];
			$dtstrc =$output['dato_estructura_neurona'];
			$StrANRMtr[$cdANR][$cdStrc-1] = $dtstrc;
		}
	} catch (Exception $err) {
		Echo "Failed: " . $err->getMessage();
	}
	try {
		$cdNro = $databaseC->prepare("select id_neurona, u, k, a, n, capa, m, id_rna from NEURONA order by id_rna, id_neurona");
		$cdNro->execute();
		$neuron_number=0;
		while ($outneu = $cdNro->fetch(PDO::FETCH_ASSOC)) {
			$neuron_number++;
			$cmMtr[$outneu['id_rna']][$outneu['id_neurona']] = $outneu['m'];
			$layANRMtr[$outneu['id_rna']][$outneu['id_neurona']] = $outneu['capa'];
			$nMtr[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['n'];
			$avMt[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['a'];
			$kvMtr[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['k'];
			$uvMtr[$outneu['id_rna']][$outneu['id_neurona']] =$outneu['u'];
		}
	} catch (Exception $e) {
		Echo "Failed: " . $e->getMessage();
	}
	try {
		$st_weigth = $databaseC->prepare("select id_rna,id_neurona,id_peso,valor_peso from PESO order by id_rna,id_neurona,id_peso");
		$st_weigth->execute();
		while ($outweigth = $st_weigth->fetch(PDO::FETCH_ASSOC)) {
			$weiANRMt[$outweigth['id_rna']][$outweigth['id_neurona']][$outweigth['id_peso']] = $outweigth['valor_peso'];
		}
	}catch (Exception $err) {
		Echo "Failed: " . $err->getMessage();
	}
	print "Cant de neur : ".$neuron_number."\n";
	print "\nDts de la arreglo : "."\n\n";
	for ($idANRF=1; $idANRF<sizeof($weiANRMt)+1; $idANRF++){
		$in_arn=str_pad($idANRF, 5, "0", STR_PAD_LEFT);
		print "\n**********RNA : ".$in_arn."**********\n\n";
		for ($idNro=1; $idNro<=sizeof($weiANRMt[$in_arn]); $idNro++){
			$inNuro=str_pad($idNro, 1, "0", STR_PAD_LEFT);
			print "\nNeurona : ".$inNuro."---------- \n\n";
			for ($cpso=1; $cpso<=sizeof($weiANRMt[$in_arn][$inNuro]); $cpso++){
				$nupso=str_pad($cpso, 1, "0", STR_PAD_LEFT);
				$VlPso=$weiANRMt[$in_arn][$inNuro][$nupso];
				print sprintf("%15f",$VlPso);
				if ($cpso%10==0){
					print "\n";
				}
			}
			print "\n";
		}
	print "\n";
	}
	$neuron_number=0;
	for ($i_e=1; $i_e<=sizeof($StrANRMtr[str_pad(1, 5, "0", STR_PAD_LEFT)])-1; $i_e++){
		$neuron_number = $neuron_number + $StrANRMtr[str_pad(1, 5, "0", STR_PAD_LEFT)][$i_e];
	}
```

```
    $in_arnN = 1;
    $in_arn = str_pad($in_arnN, 5, "0", STR_PAD_LEFT);
    $nNeARN = sizeof($weiANRMt[$in_arn]);
    $umbSal = $uvMtr[$in_arn][$nNeARN];
    $nMtra=0;
    try {
        $stTraC = $databasePG->prepare("select spt,dpt,tamano,cantidad_paquetes,duracion , tamano_medio_paquetes,
        tiempo_promedio_entre_paquetes from conexion where src = '192.168.1.16' and tiempo_promedio_entre_paquetes<5 order by id
        limit 50");
        $stTraC->execute();
        while ($rTraC = $stTraC->fetch(PDO::FETCH_NUM)) {
            $nMtra++;
            $long = sizeof($rTraC);
            for ($inV=0; $inV<$long; $inV++){
                $inVec[$inV]=$rTraC[$inV];
            }
            for ($inV=$long; $inV<10; $inV++){
                $inVec[$inV]=0;
            }
            $muestra="";
            for ($inV=0; $inV<10; $inV++){
                $muestra=$muestra." - ".$inVec[$inV];
            }
             $outANR = $ANRcal->calculoRNAM($inVec,$nMtr ,$cmMtr,$layANRMtr,$avMt,$kvMtr,
            $uvMtr,$weiANRMt,$neuron_number, $StrANRMtr,$in_arn);
            if (abs($umbSal-$outANR)<50){
                print "Reconocio la muestra : ".$nMtra." : ".$muestra."\n";
            }
        }
    }catch (Exception $e) {
        Echo "Failed: " . $e->getMessage();
    }
}
```

APÉNDICE E: Programa en PHP para la autoconfiguración del firewall.

```
function generaReglas($databasePG,$databaseC){
        $fire = "/etc/rc.d/init.d/firewall";
        $file =fopen($fire,"a");
        try {
            $cadCst = $databasePG->prepare("select id_intrusion , src , dst , spt , dpt from intrusion where estado = '01'");
            $cadCst->execute();
            if ($res = $cadCst->fetch(PDO::FETCH_ASSOC)) {
              $idItr = $res['id_intrusion'];
              $srcIt  = $res['src'];
              $dstIt  = $res['dst'];
              $sptIt  = $res['spt'];
              $dptIt  = $res['dpt'];
              $regla = "iptables -A FORWARD -s ".$srcIt." -j DROP";
              shell_exec($regla);
              fwrite($file, $regla."\n");
              fclose($file);
              try {
                  $upItr = $databasePG->prepare("update intrusion set estado ='02' where id_intrusion ='$idItr'");
                  $databasePG->beginTransaction();
                  $upItr->execute();
                  $databasePG->commit();
              } catch (Exception $err) {
                        $databasePG->rollBack();
                        echo "Failed: " . $err->getMessage();
              }
            }
        } catch (Exception $err) {
        echo "Failed: " . $err->getMessage();
        }
        }
```