UNIVERSIDAD SEÑOR DE SIPÁN

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS

COMPARACION DE ALGORITMOS DE REDES NEURONALES PARA MEJORAR LA DETECCIÓN DE INTRUSOS EN REDES DE AREA LOCAL

PARA OPTAR EL TITULO PROFESIONAL DE INGENIERO DE SISTEMAS

Autor:

Bach. Guevara Palomino Nilton

ORCID: https://orcid.org/0000-0002-3609-100X

Asesor:

Mg. Tuesta Monteza Víctor Alexci

ORCID: https://orcid.org/0000-0002-5913-990X

Línea de Investigación:

Infraestructura, Tecnología y Medio Ambiente

Pimentel-Perú 2021

APROBACIÓN DE TESIS

COMPARACION DE ALGORITMOS DE REDES NEURONALES PARA MEJORAR LA DETECCIÓN DE INTRUSOS EN REDES DE AREA LOCAL

_____

Guevara Palomino Nilton

Autor

_____

G. Samillan Ayala Alberto Enrique

Asesor

_____ _____

Mg. Mejia Cabrera Heber Ivan

Presidente de Jurado

Mg. Sanchez Guevara Omar Antonio

Secretario de Jurado

_____

Mg. Tuesta Monteza Victor alexci

Vocal de Jurado

DEDICATORIA

Dedico este trabajo a mi esposa Maritza, a mis hijos y padres por su esfuerzo, dedicación y cariño hacia mi persona y poder cumplir mi meta.

AGRADECIMIENTO

Quiero empezar agradeciendo a mi bella Esposa Maritza Siesquén Bances, a mis adorables hijos Fiorella, Alonso y Anthony porque son la inspiración y el motivo de cada una de mis acciones y sueños.

A mi Asesor el Ing. Alex Coronado por su tiempo y constante apoyo, al Ing. Arthur Huamaní por responder a mis interrogantes y orientaciones para el desarrollo de esta tesis.

A todos los docentes que me brindaron sus experiencias y conocimientos.

A mis compañeros: Emilio, Milagros, Zoila Milagritos, Luis Emir, María, Irene, Juan Carlos, Henry, Johnny, Luis y Magali con quienes compartir muchas experiencias inolvidables en el transcurso de nuestra etapa universitaria.

RESUMEN

Esta presente investigación sobre sistemas de detección de intrusos usando algoritmos de inteligencia artificial, el cual representan hoy en día un factor muy importante que abarca en la seguridad informática, teniendo como objetivo principal la detección de actividades que no han sido autorizadas, por lo que se debe de realizar la identificación de los ataques realizados a los sistemas de flujo de datos en una red. En esta tesis se describe y propone el estudio de tres redes neuronales: RNA FeedForward y Elman usando algoritmo de aprendizaje Backpropagation y la Red Neuronal Recurrente (RNN) usando algoritmo RTLR, con el fin de realizar una comparación en la detección de intrusos y obtener cuál de ellas es la mejor en el monitoreo de una red de datos, donde se captura los paquetes que circulan hacia el protocolo HTTP (Hipertexto Transfer Protocol). El sistema fue diseñado y simulado mediante las herramientas del toolbox de MATLAB permitiendo a la red neuronal demostrar el alto rendimiento y desempeño en la detección de intrusos mostrando al usuario información relevante en los ataques detectados. Los resultados finales de la investigación fueron que el tipo de red neuronal recurrente mostró superioridad a las demás redes en velocidad, convergencia y efectividad, alcanzando más del 90% en porcentaje de clasificación correcta en un tiempo de 60 épocas.

PALABRAS CLAVES: Algoritmos, Anomalías, IDS, Redes Neuronales, Inteligencia artificial, Ataques, Seguridad informática.

ABSTRACT

This present investigation on intrusion detection systems using artificial intelligence algorithms, which today represent a very important factor that encompasses computer security, having as its main objective the detection of activities that have not been authorized, so it is It must carry out the identification of the attacks carried out on the data flow systems in a network. This thesis describes and proposes the study of three neural networks: RNA FeedForward and Elman using the Backpropagation learning algorithm and the Recurrent Neural Network (RNN) using RTLR algorithm, in order to make a comparison in the detection of intrusions and obtain which one of them is the best in monitoring a data network, where the packets that flow towards the HTTP protocol (Hypertext Transfer Protocol) are captured. The system was designed and simulated using the MATLAB toolbox tools, allowing the neural network to demonstrate high performance and performance in intrusion detection, showing the user relevant information on the attacks detected. The final results of the research were that the recurrent type of neural network showed superiority to the other networks in speed, convergence and effectiveness, reaching more than 90% in percentage of correct classification in a time of 60 epochs.

KEY WORDS: Algorithms, Anomalies, IDS, Neural Networks, Artificial Intelligence, Attacks, Computer Security.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

I. INTRODUCCION

1.1. Realidad Problemática.

El hackeo es la intrusión de una persona u organización en un sistema informático con el objetivo de obtener información privada y confidencial.

Actualmente se ha dado un profundo cambio en los ataques cibernéticos, debido a que las capacidades actuales de cómputo son más sofisticadas; el cual hacen uso de herramientas de inteligencia artificial para realizar ataques teledirigidos unos tras otros y amenazas avanzadas persistentes. (Alarcón, 2005)

Con los grandes avances tecnológicos y las nuevas aplicaciones que se han desarrollado actualmente es posible capturar, analizar y procesar grandes cantidades de información para la protección de sistemas y recursos, por eso los Sistemas de Detección de Intrusos – Intrussion Detecction Systems (IDS) están teniendo una mayor importancia en el ámbito de la seguridad informática. Estos sistemas pueden utilizar diversas técnicas para prever un ataque, entre ellas tenemos las Redes Neuronales Artificiales (RNAs).

La amplia mayoría de datos que son sensibles, en el mundo se almacena en diversos gestores de datos como por ejemplo Microsoft SQL Server, Oracle, etc. Por lo tanto, el objetivo principal de los delincuentes cibernéticos es atacar a este tipo de base de datos.

Esto explica porque los ataques externos a las aplicaciones web como por ejemplo la inyección SQL se registraron en 27%, XSS en 4.7% durante el cuarto trimestre del año 2015, datos reportados por la empresa líder en servicios de CDN a nivel mundial. (Technologies, 2015)

Los eventos de ataques cibernéticos más destacados en el mundo fueron descubiertos en primavera del año 2015 por el equipo de GReAT en Kaspersky en donde la banda de ciberdelincuentes Carnabak robó mil millones de dólares de instituciones financieras de todo el mundo. (Alexander Gostev, 2015)

Según estudios de analistas de seguridad de Kasperksy, usuarios de internet en américa latina sufren 10 ataques cibernéticos por segundo, debido a esto en Brasil, julio del 2015 una investigación de la RSA descubrió que ciberdelincuentes robaron mediante el pago de Boleto bancario posiblemente unos 3,750 millones de dólares, siendo el atraco más grande de la historia del Brasil.

En Perú, la empresa Digiware recopilo la información de sus centros de operaciones de seguridad en más de 13 mil dispositivos distribuidos en diversos sectores de américa latina, Perú está ubicado en puesto número 5 por ser un país que ha sido muy atacado por los cibernéticos, después de Colombia, Brasil, México y Argentina. En mayo del 2015 el grupo de hackers peruanos LulzSecPeru se hizo responsable del ataque al portal web del Gobierno Peruano.

Bajo este contexto de inseguridad y persistencia de ataques informáticos, cibercriminales debemos tomar medidas de seguridad para proteger nuestra información tanto a nivel personal como empresarial, aquí es donde muchas soluciones se han planteado desde programas antivirus, firewalls, IDS, IPS y otras; sin embargo, la brecha de ataques continua por las nuevas técnicas empleadas por los delincuentes informáticos. Es aquí donde planteamos el uso de algoritmos de inteligencia artificial haciendo uso de redes neuronales para mejorar la detección de intrusos.

Esta investigación busca mejorar la detección de anomalías o los ataques a las aplicaciones web, en donde se realizará la comparación y el análisis de tres algoritmos de entrenamiento de redes neuronales para analizar el tráfico de red o las peticiones hechas por los intrusos. El sistema identificará comportamientos anómalos en la red y emitirá una alarma o mensaje para tomar alguna acción apropiada.

Este proyecto tendrá un enfoque en la información de la capa de Aplicación, en particular nos centramos nuestra atención en el protocolo HTTP y los ataques que se lanzan sobre él.

En este trabajo realizamos un diagnóstico de los diversos IDS que actualmente se encuentran vigentes. Para el entrenamiento y la prueba en la comparación de diversos

algoritmos de la Red Neuronal, se hizo uso de datos que han sido recolectados o clasificados de un proyecto desarrollado por la Universidad Javeriana de Bogotá, Colombia. (Alarcón, 2005)

En el desarrollo del proyecto se estudió la posibilidad de poder detectar nuevos ataques (Zero day attack), teniendo como objetivo principal lo siguiente: Comparar algoritmos de redes neuronales para poder detectar anomalías en un sistema a través de la red basándose en diversos modelos de inteligencia artificial como las Redes Neuronales.

Para obtener los resultados finales se utilizó Matlab en el entrenamiento y prueba de varias redes con distintos algoritmos de entrenamiento y finalmente se muestran los resultados que sugieren una detección de intrusos de alrededor del 94%, valor máximo de efectividad que la naturaleza de la red neuronal permitió generar.

1.2. Antecedentes del estudio

Yu, Luo, & Min, (2016), realizo la investigación "Intrusion detection in wireless sensor networks for destructive intruders" la detección de intrusiones en una red de sensores inalámbricos (WSN) ha causado mucha atención últimamente debido a sus extensas aplicaciones. Varios problemas en la detección de intrusiones, como el despliegue de sensores, la movilidad de los sensores y la fusión de datos, se han investigado exhaustivamente. En cambio, rara vez se ha investigado el comportamiento del intruso. En esta investigación presentan una situación novedosa en la que el intruso puede deshacer los sensores encontrados, en esta situación se analiza teórica y experimentalmente bajo nuestro modelo de sistema, el punto clave es dialogar el problema de detección de intrusos de manera diferente según la velocidad del intruso, el modelo de detección que disponen incluye un modelo de detección única y un modelo de detección múltiple. Asi como también se consideran algunos factores interesantes en la detección de intrusiones, como el período de transmisión, el tiempo de entrada aleatoria del intruso y el período de muestreo.

Zhang, Cao, Ding, Zhuang, & Yao, (2016), realizo la investigación “An intruder detection algorithm for vision based sense and avoid system “donde dice que la detección de intrusos es un problema impotante en el sistema de detección y precaución centrado en la visión (SAA). En esta investigación se propone un algoritmo de detección de intrusos basado en aprendizaje profundo de funciones. El algoritmo de detección de intrusos consiste de cuatro partes: obtención de las muestras de prueba, creación del diccionario demasiado completo, aprendizaje profundo de las características y determinación de la región del intruso. Se implanta la técnica de ventana deslizante para alcanzar las muestras de prueba. La descomposición de valores singulares de K-means (K-SVD) se usa para el entrenamiento de diccionario demasiado completo. Emplearon el método de aprendizaje profundo de características sobre la base del diccionario para la clase de características. La máquina de vectores de soporte (SVM) se usa para seleccionar la región de interés (ROI), y la región del intruso finalmente se determina fusionando las ROI superpuestas.

Kalnoor & Agarkhed, (2016), realizo la investigación “Preventing attacks and detecting intruder for secured Wireless Sensor Networks” donde dicen que la protección de la red es el tema más desafiante para prevenir los tipos dañinos de ataques y plantea un problema de seguridad en WSN junto con los dominios de aplicaciones de seguridad de la información. Algunas de las arquitecturas y pautas se proponen para proteger las redes de sensores inalámbricos (WSN) contra diferentes ataques de intrusos. WSN tiene una gran variedad de aplicaciones que incluyen público masivo, militares, etc. Los sensores en WSN se implementan de tal manera que su tecnología de detección tiene poder de procesamiento y comunicación inalámbrica. El uso del análisis de tráfico para detectar anomalías es una de las técnicas más eficientes y efectivas para detectar un intruso y prevenirlo. En muchas aplicaciones de WSN, un problema crítico es proteger la red de sensores de adversarios y ataques fuertes maliciosos.

Nagpal & Manojkumar, (2016), realizo la investigación “Hardware implementation of intruder recognition in a farm through Wireless Sensor Network” en donde dice que, en las zonas rurales de la India, los robos en las granjas son más comunes. Los propietarios de las granjas luchan mucho por obtener un alto rendimiento de varias maneras. Pero su rendimiento se ve limitado debido a la interferencia de animales y humanos no autorizados. Los humanos no autorizados ingresan a la granja y roban los productos agrícolas o causan algún daño a los cultivos. Mientras que los animales domésticos como la cabra, la vaca causan daños a los cultivos, ya sea consumiéndolos o dañándolos. Estos dan como resultado rendimientos deficientes que a su vez reducen su ganancia, lo que conduce a la frustración. Cercar la finca no es fácil y asequible, especialmente cuando el campo es grande. Incluso si está vallado, las entradas humanas no autorizadas son posibles. Por lo tanto, es muy esencial monitorear los límites de la granja para detectar el movimiento de entradas no autorizadas a la granja. Para la implementación de esto, se desarrolla un sistema basado en la red de sensores inalámbricos. Los sensores de movimiento se colocan en varios lugares de la granja. Estos sensores detectan continuamente el movimiento y se comunican con el coordinador a través del transceptor de radiofrecuencia. El coordinador de detección genera una alerta y se realiza una llamada al móvil del propietario de la granja a través del Sistema Global para Móvil. Además, para diferenciar entre entradas autorizadas y no autorizadas, se utilizan etiquetas de identificación por radiofrecuencia.

Shahid, y otros, (2017), realizo la investigación “Computer vision based intruder detection framework (CV-IDF)” la vigilancia se ha convertido en un tema fundamental en los últimos años después de la comida y la ropa, el enfoque es cómo evitar ataques repentinos para garantizar nuestras vidas. En este artículo, proponen un algoritmo fuerte que tendrá como finalidad detectar y rastrear múltiples intrusos en un área prohibida. Las adiciones significativas de este artículo son que proponen un marco verídico y eficiente que utiliza el método de sustracción de fondo utilizando modelos de mezcla gaussianos seguidos por el filtro de Kalman para encontrar intrusos en un

área restringida, lo que disminuye las falsas alarmas, lleva a cabo un seguimiento de múltiples intrusos que se mueven dentro de una región restringida empleando el filtro de Kalman, las asociaciones de datos de las predicciones alcanzadas por el filtro de Kalman y el seguimiento de un intruso individual se alcanza mediante un algoritmo húngaro. Las alarmas se ocasionan después de conseguir detecciones para comunicar al personal de seguridad que tome las medidas preventivas.

Jiang Landa; Chu Jun; Miao Jun, (2017), realizo la investigación “Implementation of a Remote Real-Time Surveillance Security System for Intruder Detection” La seguridad familiar es uno de los objetivos clave de un sistema doméstico digital. Actualmente las cámaras web se usan de más en los sistemas domésticos de monitorización remota. Esto se debe al hecho de que su naturaleza de supervisión pasiva deja un problema de seguridad potencial y no se puede utilizar para proporcionar seguridad proactiva. Para resolver este problema, este trabajo propone el diseño de un sistema de monitoreo de seguridad inteligente embebido remoto basado en el algoritmo de modelado de fondo en visión por computadora que puede detectar intrusos de manera proactiva. El sistema usa el algoritmo ViBe para modelar la imagen de fondo que se consigue de la cámara y luego lleva a cabo la detección de objetos en el entorno monitoreado. Si detecta un objeto en movimiento (incluidas personas), el sistema hará sonar automáticamente una alarma, mientras envía un mensaje o llama al usuario para que tome medidas preventivas. Los usuarios pueden iniciar sesión en el servidor a través de la aplicación móvil para detectar al intruso a partir de la imagen interceptada y comprender mejor la situación. Después de la prueba, el sistema puede determinar con precisión si hay algo que se entromete en el área de monitoreo o no. Es robusto, en tiempo real y con un buen valor de aplicación práctica.

Quwaider, (2017), realizo la investigación “Real-time intruder surveillance using low-cost remote wireless sensors “donde se dice la detección de intrusos es necesaria para monitorear pasajes o puertas de enlace sensibles. Estos pasajes suelen existir en

aeropuertos, comisarías, edificios gubernamentales, tribunales o incluso en laboratorios especiales, como laboratorios nucleares o químicos. Por motivos de seguridad, normalmente no se permite que nadie pase por estos pasajes sin permiso y durante el horario laboral. Los sistemas de monitoreo de transmisión de video suelen ser costosos en términos de comunicación de datos y análisis de datos, lo que puede requerir un análisis adicional de sujetos humanos. Además, al límite de la escena del área visible de la cámara. En esta investigación se propone un mecanismo de detección de intrusos en tiempo real utilizando redes de sensores inalámbricos remotos. Un par de sensores inalámbricos se despliegan en el pasaje monitoreado. Para detectar el movimiento de intrusos, el mecanismo propuesto adopta el indicador de intensidad de señal recibida (RSSI) medido de la señal de radiofrecuencia (RF) entre los sensores desplegados. Los resultados mostraron que el RSSI de la señal recibida se reduce significativamente cuando hay intrusos en el paso y la cantidad de reducción depende del número de intrusos.

Tian, Li, Zhou, & Li, (2018), realizo la investigación “WiFi-Based Adaptive Indoor Passive Intrusion Detection” La detección de intrusión pasiva, que es una técnica emergente para detectar si existe algún intruso en el área monitoreada, se usa ampliamente en la seguridad del hogar y el hogar inteligente, etc. sido propuesto. Sin embargo, esos sistemas de detección existentes se basan principalmente en un elaborado proceso de capacitación fuera de línea, lo que dificulta la implementación rápida de dispositivos inalámbricos y también reduce la solidez del sistema. Para responder a esos problemas, en este documento proponemos APID, un sistema para la detección de intrusión pasiva en interiores adaptativa, que permite la detección de intrusiones humanas adaptativa y sin dispositivos en entornos interiores utilizando información de estado del canal (CSI) de señales WiFi. En primer lugar, APID evalúa la dispersión de la amplitud del CSI, que no se ve afectada por la amplitud media. En segundo lugar, APID extrae la relación de dispersión de amplitud de CSI entre dos ventanas de tiempo adyacentes como métricas sensibles para la detección de intrusiones. Luego, la prueba de hipótesis se utiliza para lograr la detección de

movimiento humano sin calibración. Finalmente, implementamos APID en los dispositivos WiFi básicos y lo evaluamos en dos escenarios interiores típicos. Los resultados experimentales muestran que APID puede lograr una precisión de detección promedio de más del 96%.

Xenya, Kwayie, & Quist-Aphesti, (2019), realizo la investigación “Intruder Detection with Alert Using Cloud Based Convolutional Neural Network and Raspberry Pi” presentan un sistema de detección de intrusos con una implementación de red neuronal convolucional (CNN) utilizando raspberry pi, los sistemas en la nube Azure y Twilio de Microsoft. El algoritmo de CNN que se acumula en la nube se pone en marcha para coordinar fundamental los datos de entrada como intrusos o usuarios. Al usar la raspberry pi como middleware y la cámara raspberry pi para la adquisición de imágenes, se realiza una ejecución competente de las operaciones de aprendizaje y clasificación utilizando recursos más altos que la computación en la nube. El sistema en la nube también está preparado para alertar a los usuarios designados a través de servicios de mensajería multimedia (MMS) cuando se detectan intrusos o usuarios.

Chaturvedi, Kumar, & Sharma, (2020), realizo la investigación” Proposing Innovative Intruder Detection System for Host Machines in Cloud Computing” se dice que la virtualización tiene un papel muy fundamental en la computación en la nube, se sabe que el entorno de la nube se reparte por naturaleza y de modo que será más dispuesto a diferentes tipos de ataques de intrusión que incorporan la instalación de software malicioso y la generación de puertas traseras. En un entorno de nube, donde las organizaciones han hospedado datos importantes y críticos, la seguridad de las tecnologías subyacentes se vuelve crítica. En esta investigación, proponen un modelo innovador para el sistema de detección de intrusiones para proteger las máquinas host en la infraestructura de la nube. El IDS planteado tiene dos características importantes: sistema de alerta rápida y basado en firmas, que servirá para moderar el peligro de los

entornos de nube, los sistemas de detección de intrusiones (IDS) son una cubierta de defensa.

Mallikarjun, Kiranmayi, Lavanya, Prateeksha, & Sushmitha, (2020), realizo la investigación “Intruder Detection System - A LoRa Based Approach” se dice que hoy en todos los países enfrentan amenazas del intruso. Desafiando la seguridad interna de la nación. En consiguiente, las aplicaciones militares precisa ubicar a los intrusos en las áreas fronterizas. En esta aplicación, el seguimiento del objetivo, el procesamiento de los datos y el análisis de los datos alcanzados tiene un rol importante. El sistema propuesto pretende el seguimiento del objetivo en el área segura y el análisis de los datos recibidos. Los sensores juegan un papel importante en la detección del intruso. En esta investigación, se ha debatido la propuesta, diseño e implementación de un sistema que hace uso de Sensor Infrarrojo Pasivo (PIR) para detectar al intruso y el entorno circundante. La información recopilada de la red de sensores se envía a la estación base para su procesamiento utilizando tecnología de largo alcance (LoRa). Los resultados de la implementación justifican la importancia de la metodología utilizada. Los resultados de desempeño justifican la efectividad del sistema propuesto.

1.3. Teorías relacionadas al tema

1.3.1. Sistemas para detección de intrusos.

En los años 70 el Departamento de Defensa de los EE. UU invirtió recursos en investigar las diversas cláusulas de seguridad y pautas de control, culminando con una iniciativa de seguridad en 1977 llamados sistemas de confianza. Posteriormente se desarrolló IDS para que sean utilizados en diferentes métodos y así poder analizar los sistemas de comunicación entorno a su comportamiento de intrusos, en 1980 se redactó el primer informe sobre la detección de intrusos presentado por James P. Anderson, el cual sería el inicio para futuros trabajos.

Entre 1984 y 1986 Dorothy Denning y Peter Nuemann desarrollaron un IDES (Intrusion Detection Expert System) fue el primer sistema de intrusiones en tiempo real el cuál se enviaba una correspondencia entre una actividad anómala y de uso indebido. Posteriormente el National Computer Security Center (NCSC) en 1989, desarrollo el famoso MIDAS que utilizó un sistema hibrido, realizando una combinación de las estadísticas anómalas como regla de seguridad en un sistema experto. MIDAS ayudo a la protección de diversos ataques externos y a la autenticación de los usuarios (Garcia, 2008). Kathleen Jackson, en los años 80 creo un sistema llamado Network Audit Director and Intrusion Reporter (NADIR) que consiste en el uso de técnicas de detección similares MIDAS.

La universidad de California desarrollo el primer IDS para monitorear el tráfico que pasaba por la red, al cual se le denomino NetworkSystem Monitor (NSM), este IDS se trabajó en UNIX de Sun.

En 1988 el Centro Criptológico de la fuerza aérea de los EE. UU se unió con academias para poder darle solución al problema que había respecto al gusano de internet. Diversos laboratorios como por ejemplo Lawrence Livermore, Haystak implementaron procedimientos de seguridad uniendo a una red DIDS con sistemas de detección basados en una maquina formando asi el primer sistema que pudo lograr monitorear las intrusiones de seguridad en las redes.

A comienzos del año 1990 empezaron a aparecer los primeros sistemas de detección de intrusos comerciales como: Compter Watch fue creado por la empresa AT&T, el Information Security Officer´s Assistan (ISOA) de PRC y Clyde Digital desarrollo el Clyde Vax Audit

Entre los años 1994 y 1996 la Universidad Tecnológica de Brandemburgo implemento un Sistema de Detección de Intrusos Adaptativo (AID). (Vidal, 2012)

¿Qué es un Sistema de Detección de Intrusos?

Permite monitorear los sucesos de un sistema informático para identificar las intrusiones. Consiste de tres elementos: La fuente de información provee los eventos del sistema, el motor realiza un análisis de información de las variables que indican el uso inusual o erróneo y al final se informa el resultado de la detección.

Las ventajas que proporcionan los IDS son numerosas, por lo que se justifica su uso en las organizaciones:

a) Posibilitan el descubrimiento de atacantes al sistema, lo que da posibilita de ser descubiertos y penalizados.
b) Detectan ataques y otras vulneraciones de la seguridad que otros sistemas de protección no previenen.
c) Detectan preámbulos de ataques, es decir el atacante suele examinar y realizar pruebas antes de realizar el ataque.
d) Justifican y documentan el riesgo de la organización.

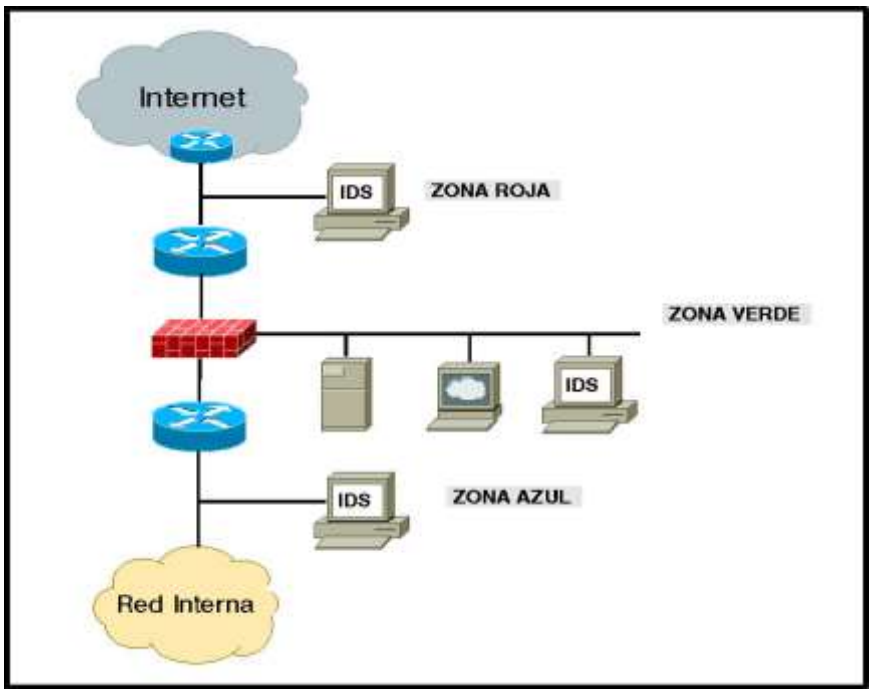


Figura 1. Zonas de un IDS dentro de una organización Fuente: *(Mira Alfaro, 2002)*

Arquitectura de los IDS.

Con el pasar de los años surgieron diferentes propuestas para definir la arquitectura de los IDS por diferentes fabricantes, sin embargo, en la actualidad no hay ninguna de ellas que se utilice de modo estándar. (Tejada, 2015)

No obstante, hay ciertas características comunes en las distintas arquitecturas de los IDS:

a) Fuente de recogida de datos: Consiste en los periféricos de red, Logs o un conjunto de datos.
b) Las políticas son definidas por patrones o directrices para poder realizar la detección de las anomalías que podría presentar un sistema
c) En las reglas se definen los filtros que permiten comparar los logs que se obtuvieron por los patrones
d) Si se encuentra algún tipo de intrusión o ataque, el sistema generará informes y alarmas.

A pesar de los rasgos comunes son muchas las diferencias que tienes las arquitecturas de los IDS. A continuación, detallamos las más importantes del mercado actual.

CIDF (Common Intrusion Detection Framework)

Consta de un generador, analizador y un banco de datos de eventos además de unidades de respuesta ante la aparición de incidentes. Tuvo escasa aceptación en el mercado.

CISL (Common Intrusion Specification Language)

Une diversos dispositivos que conforman la arquitectura CIDF y facilita información sobre información de eventos en bruto, resultados de los análisis y prescripciones de respuestas.

AusCERT

Arquitectura simple que facilita la información de las incidencias en muy pocas líneas. Es muy limitada si se pretende obtener información detallada.

IDWG (Intrusion Detection Working Group)

Facilita la reciprocidad de la información cuando se realiza un incidente de seguridad, esto ayuda a que se defina diversos protocolos y formatos para el intercambio de la información que tiene de por medio un IDS.

Clasificación de los IDS.

Se pueden clasificar de varias formas, donde la más aceptada y completa es la que se muestra en la siguiente figura; sin embargo, la más conocida se basa en el origen de los datos como son: Los NIDS (Network Intrusion Detection System), HIDS (Host Intrusion Detection System) e Híbridos. Cada uno de ellos se diferencian por la forma de monitoreo y análisis. (Tejada, 2015)

Figura. 2. Clasificación de los IDS - Estado del Arte Sistemas De Detección De Intrusos. Fuente: *(Tejada, 2015)*.

IDS basados en Red (NIDS)

Son los que detectan los ataques mediante la captura y análisis de los diversos paquetes que viajan en la red. Una vez capturados y analizados los paquetes de la red los IDS se encargan de buscar patrones que semejen algún tipo de ataque. Además, emiten alertas cuando hay intentos de acceso o análisis externo de alguna vulnerabilidad del sistema.

Su funcionamiento consiste en:

Sensores o agentes que se sitúan en varios puntos de la red para monitorizar el tráfico en busca de tráfico sospechoso. Lo habitual es que estos sensores analicen los paquetes en modo oculto para no ser descubiertos por los atacantes.

Una consola que recibe alarmas emitidas por los sensores o agentes y dependiendo el tipo de alarma se producirá algún tipo de respuesta.

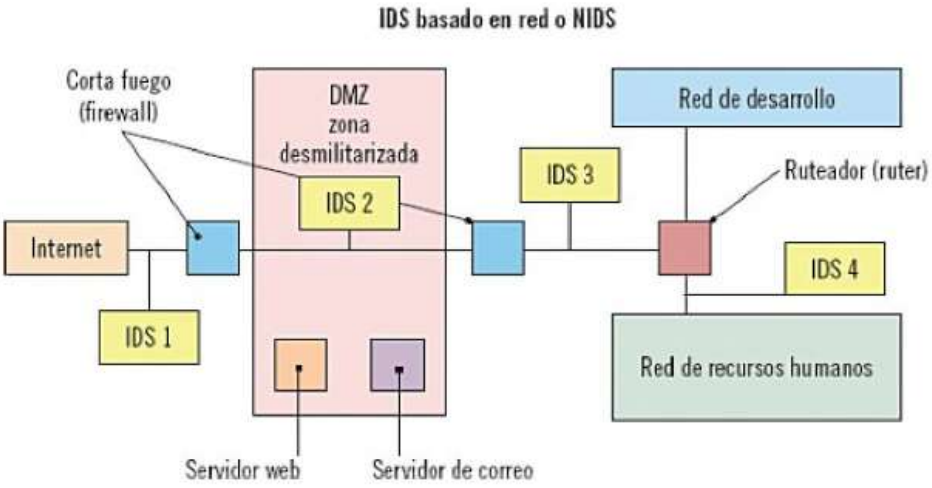


Figura 3. Ejemplo de un NIDS - Estado del Arte Sistemas De Detección De Intrusos. Fuente: (Tejada, 2015).

En la figura anterior se puede observar que hay varios IDS (sensores) situados a lo largo de toda la red, de este modo se monitorizan el tráfico de la red entre ellos.

Ventajas de los NIDS.

Detectan accesos no deseados en la red.
No requieren de utilización de otro software adicional en los servidores para poder funcionar.
Son sistemas de fácil instalación y actualización.
Tiene un bajo impacto en la red al no intervenir en sus operaciones habituales.
Pueden monitorizar redes de grandes dimensiones siempre que haya capacidad suficiente para analizar todo su tráfico.

Desventajas de los NIDS.

A pesar de poder monitorizar grandes redes pueden presentar dificultades en su procesamiento y fallar en el reconocimiento de ataques producidos en momentos de elevado tráfico de la red.

Presentan dificultades para detectar ataques con información cifrada.
También pueden presentar problemas cuando tienen que detectar ataques que viajan en paquetes fragmentados.

1.3.2. Diagnostico Actual de los IDS en Red (NIDS).

Con la expansión progresiva del Internet, la investigación y el desarrollo (I+D) los IDS han surgido en los últimos años un gran crecimiento y a la vez una diversa variación de los productos para poder detectar y prever las intrusiones (Lago, 2015). Destacando más las soluciones comerciales por brindar una gran cantidad de información y características de los ataques que las no comerciales.

IDS Comerciales (PADILLA, 2015)**.**

Dragon- Enterasys Networ.

Este IDS toma la información de actividades no autorizadas a través de un sensor llamado Dragon Sensor y de un módulo DragonSquire, Además se encarga de monitorear los Logs de los Firewalls y otros sistemas. Todos los datos son enviados a un módulo principal llamado Dragon Server para poder analizarlos y correlacionarlos. Sus debilidades del sensor son incapaces de analizar o interpretar tráfico que está codificado en una sesión web SSL.

Net Ranger - Cisco Systems.

Es un IDS fabricado por Cisco para realizar una detección y responder frente a una actividad no autorizada por medio de la red.

Se compone de un sensor y un director que son los 2 principales dispositivos de Cisco Secure y por medio de su velocidad alta permite realizar el análisis de los paquetes que viajan por la red y así determinar si se accede o se deniega el mismo. Si se ocasiona algún ataque, los sensores realizan la detección y envía una señal al director. Después se estima la ubicación del lugar donde se a realizado el ataque para posteriormente retirara al atacante de la red

Existen diversos sensores de Cisco Systems, el cual se divide en 4 categorías según la necesidad del usuario:

Módulo IDS Catalyst® 6000: Es usado para redes que son conmutadas, con capacidad de monitorear 100Mbps de tráfico, y con un aproximado de 47.000 paquetes por segundo.

IDS-4250: Tiene un soporte hasta 500Mb/s sin paralelizar y tiene la capacidad de monitorizar el tráfico en una red Gigabit.

IDS-4235: Puede monitorear 200 Mb/s y los puertos SPAN (Switched Port Analyzer). Además, se adecua para diversos entornos T3.

IDS-4210: Está optimizado para su uso en redes de 45 Mb/, T1/E1, T3 y entornos Ethernet.

RealSecure® - Internet Security Systems (http://www.iss.net)
La empresa Internet Security Sistems desarrollo este sistema para poder detectar, prever los ataques originados desde cualquier parte de la red. Se obtiene respuestas automáticas que el mismo sistema proporciona como por ejemplo almacenar los sucesos en un banco de datos, realizar el bloqueo de conexiones, remitir email, suspender o deshabilitar cuentas que ya han sido definidas por el cliente.

Tiene herramientas que funcionan para renovar patrones de firmas y poder hacer uso de las novedades recientes para evitar los ataques.

Los sensores se gestionan por medio de la consola central que incluye la instalación de forma automática para la actualización y el desarrollo.

Además, detecta en tiempo real y se realiza por medio de las capturas de paquetes con velocidad alta para que funcione sin realizar alguna complicación en la red. Existe una disponibilidad en full duplex, multipuesto y Gigabit.

NFR– NFR Security (http://www.nfr.com/)
El sistema **NFR** tiene las siguientes propiedades:

Detecta altamente la intrusión.

Gama de sensores de alto rendimiento.

Su configuración y modificación es fácil.

Es una opción para el uso en redes de disponibilidad alta

Administración y operatividad flexible.

Tiene una ejecución y mantenimiento rápido.

El tiempo de respuesta es rápida frente a los ataques.

Rodríguez, (2015), **IDS no comerciales**

Snort (http://www.snort.org)

Está desarrollado por Sourcefire y hace uso de la librería libpcap, que permite la captura de paquetes, su funcionamiento es por medio de conjuntos de reglas de las cuales pattern-matching realiza la circulación de los paquetes en la red.

La ventana que tiene Snort es que es opensource, su código fuente es gratuito por lo que se puede realizar algunas modificaciones de tal manera que hay una adaptación del programa a la necesidad especifica del usuario y la definición de reglas es sencilla.

Snort es considerada como la tecnología IDS/IPS que tiene un máximo despliegue mundial por tener múltiples beneficios de la firma y sobre todo en la inspección que se basa en las anomalías dadas en la red con más de 400 mil usuarios registrados según Webimprintsy empresa especializada en seguridad. En cuanto a su arquitectura, esta está basada en capas, en donde los paquetes viajan desde la capa inferior a la superior.

Bro NIDS

Es un framework para el análisis de redes de código abierto para sistemas Unix. Es comparado a un NIDS, pero es mucho más. Es clasificado como un IDS basado en firmas y anomalías.

Puede ser usado para recolectar métricas de redes, favorecer investigaciones forenses, entre otros usos. Es comparado a tcpdump, Snort, netflow y Perl, pero en uno solo.

Prelude Hybrid IDS

Es un SIEM (Security Information and Event Management) capaz de inter operar con todos los sistemas disponibles en el mercado. Es compatible de forma nativa con: AuditD, Nepenthes, NuFW, OSSEC, Pam, Samhain, Sancp y Snort, ofrece la posibilidad de escribir sensores propios o utilizar de terceros.

La versión gratuita (Prelude OSS) está destinada para pruebas y propósitos educacionales, por lo que sus características son limitadas.

Suricata

Es un IDS/IPS de código abierto basado en Snort, tanto la sintaxis como las herramientas de configuración de Snort son compatibles en Suricata. Posee varias características que lo hacen superior a Snort: multihilo (puede ser ejecutado en cada uno de los núcleos de la CPU); permite aceleración de hardware (uso de GPU); extracción de archivos (permite analizar los archivos cuando alguien está bajando un malware). Este IDS no solamente analiza los paquetes, sino que también permite realizar una revisión a las diversas peticiones DNS, solicitud HTTP y los diversos certificados TLS/SSL.

Kismet

Es un sistema detector de redes inalámbricas 802.11, sniffer y detector de intrusiones que funciona con una tarjeta de red inalámbrica que pueda soportar el modo rfmon. Si el Hardware es apropiado se puede realizar un análisis del tráfico 802.11b, 802.11a, 802.11g, y 802.11n. Soporta una gran variedad de complementos que le permiten analizar otros medios como DECT (Digital European Cordless Telecommunications). Mediante la recopilación pasiva de paquetes es capaz de detectar redes ocultas. Posee características básicas de IDS como la detección activa de sniffers de redes WIFI, incluyendo NetStumbler, así como un número significativo de ataques WIFI.

OpenWIPS-NG

Es un IDS/IPS con una arquitectura modular compuesta por un servidor, sensores e interfaces.

Creado por el autor de Aircrack-NG, emplea la mayoría de las funciones y servicios de esta aplicación para el análisis, detección y prevención de intrusiones. OpenWIPS-NG permite la descarga de complementos que habilitan características adicionales. Es un proyecto relativamente nuevo en comparación con Kismet y actualmente se encuentra en fase beta.

Security Onion

Este IDS está basada en Ubuntu para realizar el monitoreo de red y detección de intrusiones. La imagen puede ser distribuida como sensores dentro de la red para monitorear múltiples VLANs y subredes, funciona en VMware y entornos virtualizados. Actualmente no es posible configurarlo como IPS, pero si es posible ejecutarlo como HIDS y NIDS. Emplea servicios como Squil, Bro IDS y OSSEC para realizar las funciones de IDS. No posee buena documentación. Es una vía de tener varias herramientas integradas en un solo sistema de forma fácil.

OSSIM - Open-Source Security Information Management

Es un conjunto de herramientas que tienen licencia GPL, está diseñada para poder brindar ayuda al administrador de redes en seguridad de los dispositivos y así evitar los ataques.

1.3.3. Protocolo HTTP.

Es un protocolo que no almacena información de las conexiones, se emplea entre el cliente y el servidor web. Además, establece sesiones para cada tipo de información ya sea texto, sonido, imagen y culmina el proceso con una solicitud.

En la versión 1 se incorporó MIME (Multimedia Internet Mail Extensions), esta versión tiene soporte para negociar diferentes tipos de datos. Actualmente se encuentra en la versión 2.

La entrada por defecto a este protocolo es mediante el puerto 80, sin embargo, se puede cambiar por otro puerto en redes privadas para poder aumentar las políticas de seguridad.

Solicitudes y respuestas.

El protocolo HTTP tiene 2 tipos de encabezados: Solicitud y respuesta como se detalla a continuación:

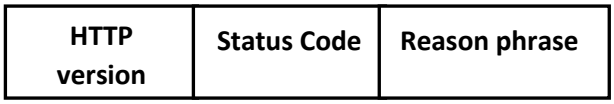
Solicitud

Method: Se establece el método que se ejecutará en el recurso.

Request URI (Uniform Resource Identifier): Es el recurso donde la solicitud se aplica

HTTP version: Versión que va a ser utilizada.

Respuesta



HTTP versión: Versión que va a ser utilizada.

Status Code: Consiste de 3 dígitos enteros

Reason phrase: Describe textualmente el status code

Ejemplo de un dialogo HTTP.

Al requerir ingresar a esta dirección web:

http://www.ejemplo.com/index.html

Primero: Se establece una conexión al host www.ejemplo.com, en el puerto que es por defecto para hacer uso de HTTP (80)

Segundo: Se remite un mensaje de este modo:

GET /index.html HTTP 1.1

Host: www.ejemplo.com

User-Agent: nombre-cliente

Accept:

Los encabezados son secuenciales y forman parte de la respuesta que emite el servidor respecto al recurso solicitado. En la página web se mostrará lo siguiente:

HTTP/1.1 200 OK

Date: Fri, 31 Dic 2015 21:34:45 GMT

Content-Type: text/html

Conetnt-Length: 1456

URL y URI

Una URL (Uniform Resource Locator) significa Localizador Uniforme de Recuro, consiste en una secuencia de caracteres para poder asignarla a una dirección de forma única. Consta de 4 partes:

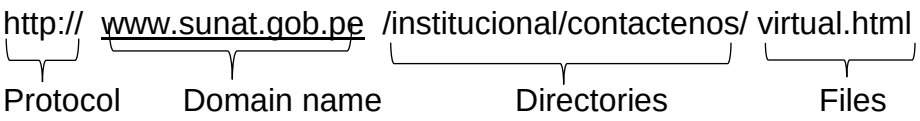
Protocolo: Conocido también como esquema URL, se realiza la especificación del protocolo para tener acceso al documento.

Nombre del ordenador: Se establece el nombre, usualmente la dirección IP o su dominio donde se aloja el contenido.

Directorios: Es una cadena de diversos directorios que son separados por barras en donde se establece la ruta que se debe seguir para obtener el documento

Archivo: Es el alias del archivo donde el recurso se ubica.

Para poder analizar una URL se realiza de la siguiente manera:



Este protocolo está asociada a diversos métodos que permite varias operaciones que se describen en la Tabla 1.

TABLA 1

Comandos HTTP

GET	Permite obtener información del servidor.
HEAD	Es igual al comando GET, solo requiere la lectura del encabezado de un URL.
PUT	Envía datos a la URL especificada.
POST	Tiene un funcionamiento similar a HEAD, además remite datos de información a ser procesados por el servidor.
DELETE	Elimina el recurso que se ubica en el URL

Nota: Comandos HTTP. Fuente: (Rodríguez, 2015)**.**

Categorías de ataques basados en Http

En este mundo actual existen muchísimas técnicas de ataques diferentes, sin embargo, para este proyecto definimos cuatro categorías: que a continuación definiremos:

Inyección de Comandos:

El ataque de inyección de comandos tiene como propósito inyectar y ejecutar comandos especificados por el atacante sobre una aplicación vulnerable. La mayoría

de casos son de este tipo de ataques, debido a la débil o incorrecta de corroborar los datos de entrada y salida.

Por ejemplo: Formato de los datos, Caracteres permitidos, y cantidad de los datos esperada.

¿Cómo se realiza un ataque de inyección de comandos?

La siguiente dirección URL conduce a la función include().

http://testsite.com/index.php?page=contact.php

El registro “evilcode.php” contiene la función phpinfo(), esta función es vital importancia ya que permite obtener información respecto a la configuración del entorno. El atacante solicita al aplicativo la ejecución del código PHP haciendo uso de esta petición. (OWASP, 2015):

http://testsite.com/?page=http://evilsite.com/evilcode.php

Inyeccion SQL:

Consiste en una técnica donde se infiltra código intruso que altera la aplicación en su nivel de validación de entradas al realizar la consulta en el banco de datos.

Este tipo de inyecciones se realizan cuando el atacante ejecuta código SQL malintencionado para poder hacer modificaciones en el banco de datos del aplicativo viendo comprometido la integridad. Con este tipo de acciones se permite a un atacante acceder a información sensible.

¿Cómo es un ataque de inyección sql?

Con la siguiente línea de código o instrucciones:

```
SELECT * FROM Usuario WHERE Usuario=’admin’ AND Contrasena=’sn323@NmsQ’
```

Se puede observar el funcionamiento del ingreso (login) a un aplicativo donde se envía dos tipos de variables “usuario” y “contrasena”

Ahora veamos cómo se puede explotar esta línea de código:

SELECT * FROM Usuarios WHERE Usuario = “OR 1=1; /*’

Cross Site Scripting (Xss):

Son ataques contra aplicaciones web el cual, el intruso controla el navegador del cliente con la finalidad de llevar a cabo una ejecución de código malicioso usualmente escritos en el lenguaje HTML o Javascript).

Se distingue dos categorías dentro de uso posibles fallos de XSS como, por ejemplo:

XSS permanentes y XSS no permanentes.

¿Cómo introducir código Javascript en una página vulnerable a XSS?

Se copia el código entre las etiquetas HTML, siendo esta opción la más práctica.

```
<script>alert("¡Hola Mundo!");</script>
```

Este código se copia en la etiqueta ***value*** que está incluida en una etiqueta <***input***>

```
<input type="text" name="q2 value="[busqueda]" />
```

Modificación de Path:

Esta modificación es un método de ataque que infringe el acceso a los archivos del servidor web que están ubicados fuera de la raíz.
Usualmente las aplicaciones web restringen la entrada a usuarios que no están autorizados para ingresar a los directorios “CGI root”.
Si se desea acceder a los ficheros o realizar una ejecución de comando en el Sistema, el intruso deberá de utilizar una serie de caracteres especiales como "dot-dot-slash" o "../".
Esto conlleva a que el atacante compromete el sistema y por lo tanto puede visualizar todo el esquema de los directorios, obteniendo así el acceso a los códigos fuentes entre otras cosas.

Un ataque DPT tiene dos vulnerabilidades:

Local File Inclusion (LFI): Si detecta que una página web es vulnerable se realiza la inclusión de unos ficheros locales.

Remote File Inclusion (RFI): Se diferencia del LFI por incluir ficheros que se encuentran ubicados en otro servidor

Si se realiza un llamado sin hacer uso del argumento “recurso" la URL seria la siguiente:

http://localhost/lab/dpt-example1.php

Luego se obtiene el siguiente mensaje "No se ha especificado el recurso". Debido a que no se ha realizado el llamado al fichero index.php:

http://localhost/lab/dpt-example1.php?recurso=index.php

Con este llamado se obtiene el mensaje esperado.

1.3.4. Redes Neuronales Artificiales

Existe varias definiciones de las redes neuronales:

Está basado en modelos biológicos, esto conlleva a que sea una nueva forma de computación.
Se compone por varios números de elementos que se procesan por niveles debido a que es un modelo matemático
Están conectadas paralelamente con elementos adaptativos simples y se organiza de forma jerárquica, esto permite conectar con los objetos del mundo real como si fuera un sistema nervioso biológico.
Por lo que se puede concluir con la siguiente definición:

Las RNA intenta realizar el mismo funcionamiento del sistema nervioso por medio de modelos matemáticos. Esto se realiza simplificando el sistema real que se simula para luego tomar las características primordiales y poder tener una tarea determinada.

Arquitectura de Redes Neuronales

La arquitectura es una estructura, patrón o topología de las conexiones que hay en una red neuronal. Los nodos en una RNA realizan una conexión a través de sinapsis, el comportamiento de la red se determina por la conexión según su estructura sináptica. Este tipo de conexiones son de forma direccional, es decir que la información viaja por un solo sentido. Por lo general, estas neuronas se agrupan en lo que se llama unidades estructurales que se denominan capas. El conjunto de esta forma una red neuronal.

Consta de tres tipos de capas:

a. **Capa de entrada. -** Se compone por varias neuronas que decepcionan los datos que proceden del entorno.
b. **Capa de salida. -** Esta capa proporciona respuesta a la red neuronal y se compone neuronas.
c. **Capa oculta. -** La capa oculta no tiene directamente una conexión con el entorno, es decir que no se conecta directamente a sensores, ni a otros elementos. Además, proporciona libertad a la red neuronal por lo que permite determinar características que se intentan modelar del entorno.

Las redes monocapa consideran esta estructura debido a que está compuesta por una capa de neuronas, las redes multicapa se compone de dos o más capas de neuronas.

Las redes unidireccionales (feedforward) consiste en circular los datos en un solo sentido. Mientras que las redes recurrentes o realimentadas (feedback) la información circula entre varias capas de neuronas hacia delante y hacia atrás. Ambas redes permiten el flujo de los datos.

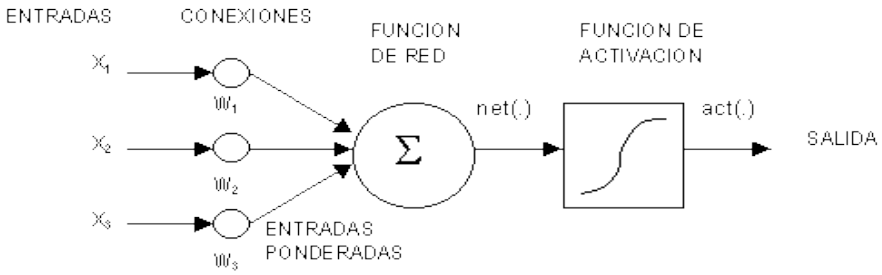


Figura 4. Arquitectura de la Neurona y Función de Activación. Fuente: (Basogain Olabe, 2008)

Redes Feedforward.

Las redes neuronales feedforward (FANN) forman parte de las redes más investigadas y usadas en los diversos campos en el ámbito científico. Esta red neuronal contiene una gran cantidad de neuronas que percibe toda la información para luego procesarla y dar respuestas multivariable. La arquitectura de esta red consiste en una topología de arreglo de neuronas por lo que las interconexiones permiten pasar la información en una sola dirección hasta que se genere la respuesta de salida, esto evita que se pase más de una vez la información a través de la neurona.

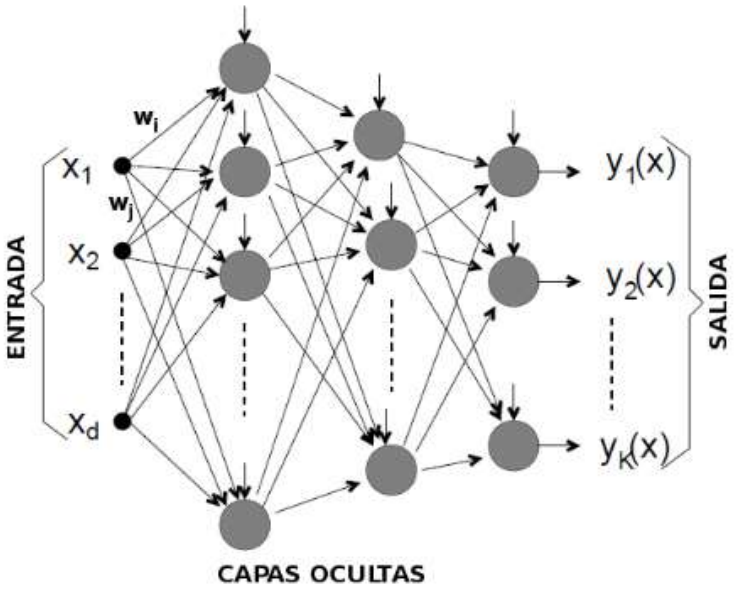


Figura 5. Estructura típica de una RNA tipo Feedforward. Fuente: (Vásquez López, 2014)

A grandes rasgos, podemos decir que la red neuronal realiza su trabajo desde el ingreso de datos a través de la capa de entrada, cuando la información pasa, esta se dirige a la capa oculta y es recepcionada por cada neurona donde se recibe la llamada suma ponderada de todas las entradas conectada a ella. Debido a que cada conexión entre las neuronas simboliza un peso de conexión. Matemáticamente se define así:

$$\sum_j w_{ij} x_j$$

Donde:

***W*ij**: Pesos de cada conexión de las X_j neuronas a la neurona receptora.

Si el estímulo tiene una cierta magnitud se procesa la información, si la magnitud es menor, se ignora la información.

Las RNA Feedforward se obtiene por un umbral m_i, si la suma ponderada supera el valor dado se identifica como una salida efectiva de la información que se procesó. Y se expresa de la siguiente manera:

$$\sum_j w_{ij} x_j - m_i$$

Los algoritmos se emulan agregando funciones de activación. La más conocida que se tiene en consideración revisando la literatura es la tangente hiperbólica, la función sigmoidal que tiene la forma:

$$g: R \rightarrow (0,1) \;/\; g\,(entrada) = \frac{1}{1 + e^{-entrada}}$$

Redes Elman

Esta red posee dos capas, cada capa está compuesta por una red tipo Backpropagation, adicionalmente tiene una conexión de realimentación a partir de la salida de la capa oculta hacia la entrada de la misma capa oculta, esta realimentación permite a la red Elman aprender a generar y reconocer los patrones temporales o variantes con el tiempo.

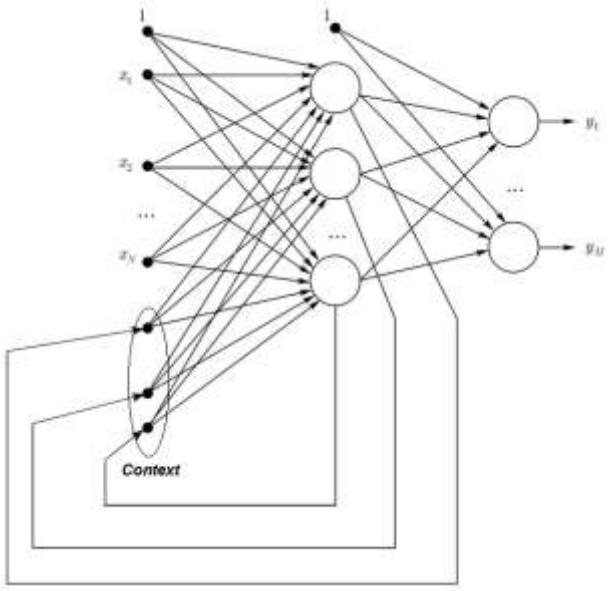


Figura 6. Estructura de una Red Elman. Fuente: (Barber, 2015).

Generalmente poseen neuronas con función transferencia sigmoidal en su capa oculta, en este caso *tansig* y neuronas con función de transferencia tipo lineal en la capa de salida, en este caso purelin, la ventaja que tiene este tipo de redes que constan de dos capas es que permite realizar una aproximación en cualquier función teniendo en consideración la precisión que se desea y esta debe depender de la selección de la cantidad adecuada de las neuronas en la capa oculta

En esta red, la capa oculta vendría a ser la capa recurrente y el retardo que tiene en la conexión al retroalimentarse permitirá almacenar los valores que tienen al realizar las iteraciones previas, por lo cual se usan para las siguientes iteraciones de las dos redes Elman que constan con los mismos parámetros y entradas para poder identificar si se produce salidas diferentes debido al proceso de realimentación.

La red Elman tiene muchas neuronas de contexto como ocultas, no hay un parámetro asociado a la conexión recurrente, la función de activación de las neuronas de contexto es la siguiente:

$$c_i(t) = a_i(t-1)\ para\ i = 1,2,\ldots.,r$$

Las actividades restantes se calculan en la red feedforward, tomando en cuenta el vector de entrada total:

$$u(t) = (x_1(t),\ldots,x_n(t),c_1(t),\ldots,c_m(t))$$

Redes Neuronales Recurrentes

Este tipo de red neuronal se distingue de las redes Elman debido a que se tiene caminos de retroalimentación en los componentes que se establecen. Las neuronas posteriores se relacionan con una sola neurona en la capa siguiente, estas neuronas pasan por medio de vectores de pesos, indicadores que tienen una alteración en cada epoch con la finalidad de obtener los parámetros de la operación

A comparación de la red feedforward, las redes neuronales recurrentes su complejidad es muy alta. Debido a que la red feedforward tiene la capacidad de transmitir la información a las capas siguientes, teniendo como resultado la propagación hacia atrás en el tiempo. La red neuronal recurrente tiene una complejidad mayor al realizar el intercambio de información y dependiendo del algoritmo que se está entrenando se puede realizar la propagación de información hacia delante en el tiempo, con la finalidad de poder predecir eventos.

Este tipo de característica es importante para algunas aplicaciones como por ejemplo los IDS, debido a que su capacidad de predecir eventos se basa en las entradas anteriores que se obtuvo, además este sistema provee un rendimiento muy significativo a la seguridad.

En la Figura 7 se observa la arquitectura elemental de una. La característica significativa es la inclusión de delays (z^{-1}) a la salida de las neuronas en las capas intermedias; las salidas parciales s_{mn} (t+1) se transforma en valores s_{mn}(t), un instante de tiempo anterior, y así poder retroalimentar a los dispositivos de la red y poder almacenar la información de los tiempos previos. Se observa que hay interconexión de todos los nodos entre sí, además de los nodos anteriores a ellos, por medio de conexiones directas. El diagrama se simplificó para no tener mucha complejidad, también se muestra que cada capa está representada por una cantidad de numero de neuronas.

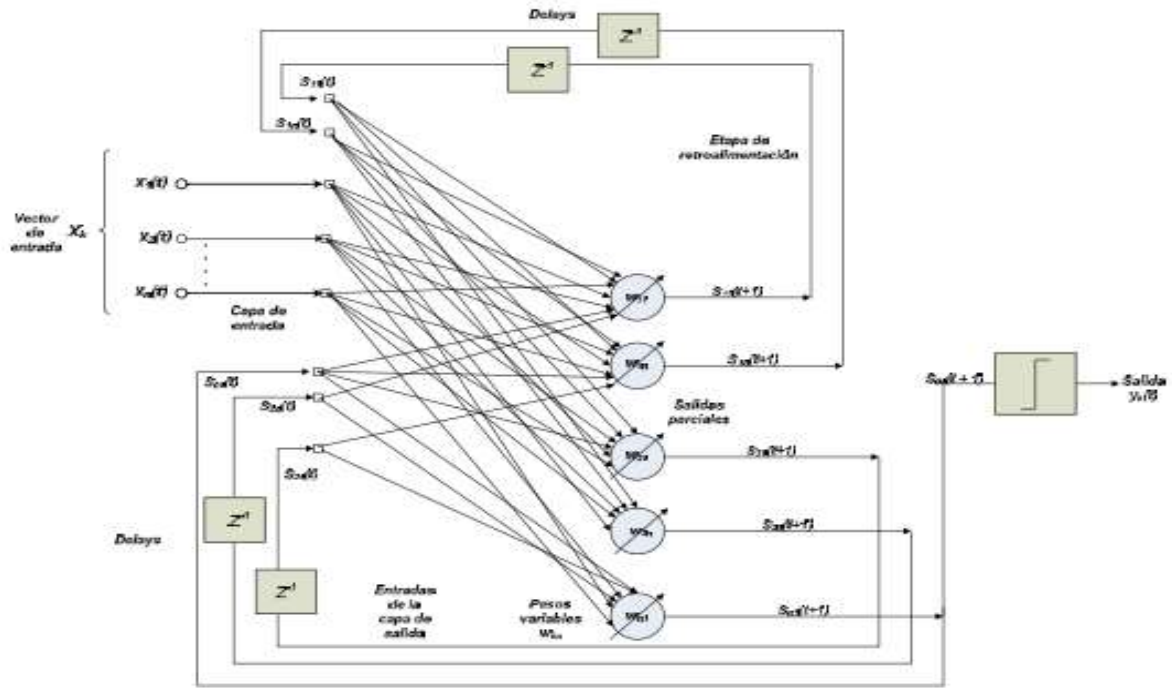


Figura 7. Red Neuronal Recurrente Completamente Conectada. Fuente: (Mejía Sánchez, 2004)

Este tipo de redes neuronales son más eficientes para dar soluciones a los problemas de tipo no linealidades temporales significativas. Son usadas mayormente para reconocer los patrones de tipo secuencial, debido a que tiene la capacidad de predecir y mapear las RNN.

Algoritmos de Aprendizaje.

Los algoritmos de aprendizaje para redes neuronales anteriormente estudiadas son: Backpropagation para las redes feedforward y Elman y para las redes neuronales recurrentes es el algoritmo Real-Time Learning Algorithm (RTRL).

El Algoritmo Backpropagation.

Se basa en la generación de regla delta, también conocido como la propagación del error hacia atrás. Consiste en un aprendizaje que es predefinido por pares de entradas y salidas. Primero se emplea un modelo de entrada como una estimulación para la capa de las neuronas de la red, luego estas se propagan por medio de las capas superiores hasta que se genere una salida. Luego se realiza una comparación con el resultado de las salidas que se desea tener para después realizar un cálculo con el valor de error de cada neurona de salida. Estos errores se transmiten hacia atrás, empezando por la capa de salida y se dirige hacia la capa intermedia que se conecta con la salida. Este proceso se realiza capa por capa hasta que las neuronas de la red reciban un error. Con este error que se recibe se realiza un reajuste de pesos en la conexión por cada neurona, de manera que la próxima salida sea lo más cercano posible, manteniendo el mismo patrón que se utilizó.

Este algoritmo es importante porque tiene la capacidad de adaptarse de forma automática los pesos de cada neurona que se ubican en la capa intermedia para que se aprenda la relación que hay en cada conjunto de patrones, tanto de entrada como de salidas. Además, tiene capacidad de generar y dar facilidad a las salidas satisfactorias de las entradas. Este tipo de redes encuentra una gráfica interna que le permite generar la salida que se desea cuando se le da entradas de entrenamiento.

Esta técnica requiere el uso de neuronas cuya función de activación sea continua, por lo tanto, diferenciable. Usualmente, la función que se utiliza es de tipo sigmoidal.

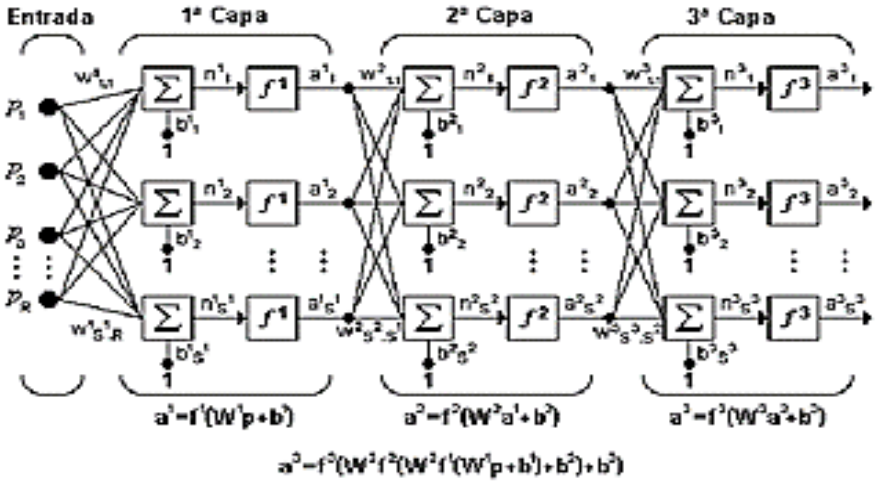


Figura. 8. Red retroalimentada Backpropagation. Fuente: (Castrillón Velasquez, Perlaza Orduz, Van Schoonhoven, & Owen)

El algoritmo de Aprendizaje en Tiempo Real – RTRL

Este algoritmo puede realizar el cálculo de la siguiente salida, teniendo como base los resultados previos de las salidas anteriores. A comparación de la red feedforward que tiene un límite, permite realizar el proceso hasta un cierto punto antes de interrumpirse

$$y(t) = \emptyset\,(x(t), x(t-1), \ldots\ldots x(1), x(0))$$

$$y(t) = \emptyset\,(x(t), x(t-1), \ldots\ldots x(t-h+1))$$

El algoritmo Real-Time Recurrent Learning Algorithm o RTRL fue desarrollado por William y Zipser (S, 1994) . Brinda un complejo calculo para un determinado número de procesadoras directamente conectadas a la red de $O(n^4)$, por lo tanto, involucra a

una gran cantidad de números de cálculos dependiendo de la cantidad de elementos. No obstante, se obtiene una elevada probabilidad de convergencia (Embrechts, 1993).

Para que se active cada nodo en el instante t, s_k, se define de la siguiente manera:

$$s_k(t) = \sum_{peI} w_{kp} x_p(t) + \sum_{qeU} w_{kq} y_q(t) \qquad k \in U$$

U: Conjunto de índices de los nodos procesadores
I: Conjunto de índices de las entradas del sistema
y_q: Salida previa
$w_{kp.q}$: Pesos conectando nodos
sk: Función de f_k de una función sigmoide.

$$y_k(t+1) = f_k(s_k(t))$$

El error cuadrático a cada instante de tiempo t es igual al cuadrado del error e_k(t).

$$E(t) = \sum_{keU} e_k^2\,(t) = \frac{1}{2} \sum_{keT(t)} \{d_k(t) - y_k(t)\}2$$

Sólo si e_k corresponde al conjunto de índices que comprende *U*. De lo contrario, e_k =cero. Los pesos se reestablecen por medio de la regla del descenso en gradiente discutida precedentemente. En donde *I* es el conjunto de índices correspondientes a la entrada del sistema, α : ***learning rate***, tasa de aprendizaje, de la red.

$$\Delta w_{ij}(t) = -\propto \frac{\partial E(t)}{\partial w_{ij}} = \propto \sum_{k \in U} e_k(t) \frac{\partial y_k(t)}{\partial w_{ij}} = \propto \sum_{k \in U} e_k(t)\, P_{ij}^k(t) \quad i \in U, j \in UUI$$

La variable **P** indica la sensibilidad del nodo k en cuestión a una permutación en el peso wij.

$$P_{ij}^k(t+1) = f_k'(s_k(t)) \left\{ z_j(t)\delta_{ki} + \sum_{qeU} w_{kq} P_{ij}^q(t) \right\}$$

$$i, k \in \cup, j \in \cup \cup I$$

En donde:

δki : Delta de Kronecker (con valor de 1 si los subíndices i,j iguales y 0 si son diferentes)

zj: Salida yj.

f'k : Derivada de la función sigmoide f.

Este tipo de Algoritmo se define por las ecuaciones y su ejecución en la práctica está sujeto a la implementación que se realizó. (Embrechts, 1993) (Willams, 1990).

Redes neuronales aplicados a la detección de intrusos.

Fox et al. Realizó el primer modelo para detectar intrusos haciendo uso de las redes neuronales. (Fox, 1990) , El modelo consiste en utilizar este tipo de redes para la creación de perfil de comportamiento del usuario y así poder realizar una predicción de comandos posteriores que el usuario ejecutó. El tipo de redes neuronales fueron las recurrentes por el motivo de que la red es perennemente visualizado y además tiene la capacidad de olvidar los comportamientos anteriores.

Ryan et al. Desarrollo NNID (Neural Network Intrusion Detection) para identificar al usuario fidedigno al distribuir los diversos comandos. Se escogió un diseño de red neuronal multicapa de tipo backpropagation que consiste de tres capas. David Endler hizo uso de un perceptrón multicapa para detectar el uso ilícito y así poder detectar

anomalías teniendo en consideración los datos de la auditoria que provienen BSM (Basic Security Module) de Solaris.

Lippmann y Cunningham desarrollaron un proyecto para mejorar el rendimiento al detectar ataques U2R que se realizan usando palabras claves. Se hizo uso de una red perceptrón multicapa, sin ocultar ni una capa para así poder detectar ataques. Luego se usó otra red neuronal parecida a la anterior para poder clasificar y medir la cantidad de palabras clave y así poder estimar la probabilidad después de un ataque en la sesión telnet. Con la otra red se usó para realizar la clasificación de los ataques y así poder asignar un nombre a ese ataque.

Ghosh y Schwartzbard presento una investigación parecida a los trabajos anteriores, este trabajo se diferencia debido a que utilizó redes neuronales para la creación de perfiles y evaluar el comportamiento de los usuarios, además se trata de diferenciar entre los comportamientos del software normal y el software malicioso. Se Desarrollo una red neuronal backpropagation (perceptrón feed-forward multicapa) con la finalidad de sistematizar la informacion incomplete para luego concluir con la clasificación.

Denning propuso un modelo para detectar intrusos, apareció diversos IDS que se aplica en distintos modelos que a variado a partir de métodos que se basan en conocimiento hasta los métodos estadísticos clásicos y sobre todo la Inteligencia Artificial para poder realizar un aprendizaje automático. (otros, 2005).

En la RNA se constituye de varias técnicas que presentan múltiples beneficios al aplicarlos en la detección de intrusos. Además, se demostró que son buenos al realizar la clasificación con gran capacidad de generalizar y aprender las características de la aplicación en el IDS. A continuación, se puede observar la Tabla 2 con las características y ventajas de las ANN.

TABLA 2

Características y ventajas de las ANN

Características de las ANN	**Ventajas en la Detección de Intrusiones**
Tiene una representación apropiada para la identificación de los problemas según su clasificación.	Los sistemas de detección de intrusos trabajan con problemas de clasificación debido a que posibilita obtener ventajas que permite facilitar la representación del conocimiento.
Permite la clasificación de los datos que están incompletos e identifica la tolerancia a errores.	Los NNIDS y NIDS permiten el análisis de paquetes TCP/IP para observar si presentaron alguna modificación por inconvenientes en la red.
El nivel de seguridad de clasificación se identifica por un valor numérico.	Permite tomar decisiones al administrador debido a que brinda una idea clara.
Pueden clasificar datos desconocidos.	Posibilita la detección de ataques que no fueron localizados anteriormente.

Nota: Características y ventajas de las ANN. Fuente: Elaboración propia

1.3.5. Definición De Términos Básicos

Amenaza. - Escenario que provoca perjuicios en el sistema.

Ataques web. - Este tipo de ataques se ocasionan hacia una aplicación cliente, su origen es a través de un lugar en la Web, por medio de diversos sitios fidedignos que

han sido atacados o a través de sitios maliciosos que se crean con la finalidad de atacar a los usuarios de ese sitio web.

Agente (Sensor): En el ámbito de detección de intrusos, el agente es una persona independiente que se encarga de monitorear y analizar en bajo nivel con el objetivo de enviar los resultados que se obtuvieron a su coordinador o transmisor- receptor (sensor)

Algoritmo. - Es una serie de pasos que tienen un orden y permite transformar los valores de entrada en salida que funciona como soluciones a varios problemas

Anomalía. - No usual o estadísticamente raro.

Bot.- Es un ordenador que es infectada por un programa malicioso que está conformado en una red de bots (bot net).

Capa. - Se le llama capa al conjunto de neuronas que conforman una red neuronal, se encuentra ubicado en el mismo nivel de red.

Cortafuegos. - Es un instrumento de seguridad que provee un límite entre diversas redes manejándolo por niveles de seguridad, utilizando las diversas políticas de control en el acceso a nivel de red.

Código Malicioso. - Este tipo de código origina una vulneración de seguridad para poder realizar algún daño en un el Sistema informático.

Enrutador. - Terminal que remite paquetes de datos que ayuda a que se puedan conectar más de dos hosts en la red. Cada punto de conexión en el router es una puerta de enlace para la red.

Escaneo De Puertos. - Realizar un análisis de los puertos haciendo uso de varias técnicas con la finalidad de eludir las técnicas de detección que se utiliza comúnmente. Algunas técnicas implican un análisis lento dependiendo de los protocolos establecidos.

Firewall. - Es una red que está diseñado para restringir ciertas conexiones dependiendo del puerto del sistema. Independiente si el tráfico que se transmite es bueno o malo.

Intrusión. - Es una infracción que viola las políticas del sistema en el ámbito de la seguridad.

Ids.- Es un procedimiento que monitorea la red de los ordenadores y sistema que busca violaciones de políticas de seguridad. Se compone por 3 elementos básicos como son el motor de análisis, mecanismos de respuesta y las fuentes de información.
Ips.- Este procedimiento hace la combinación de la capacidad de bloquear cortafuegos y analizar un IDS. Se diseñó para interrumpir ataques antes de que se ejecuten.
Neurona. - Unidad básica de constitución de la red neuronal, admite las comunicaciones de los componentes de entrada y salida de la red.
Puerta Trasera. - Permite a un atacante ingresar y que pueda tomar el control del sistema de manera oculta. Esto suele instalarse luego de que el sistema haya sido comprometido.
Zona Desmilitarizada. - Este término es utilizado para precisar un área que está situada entre dos enemigos, además es de origen miliar y está ubicado entre una red LAN y una red externa (Internet). Este tipo de zonas se ubican en los hosts que acceden por medio del internet, como por ejemplo los SMTP o DNS, FTP, servidores para evitar el acceso a través de una red privada.
Vulnerabilidades. - Debilidad del sistema que puede ser utilizado para infringir las diversas políticas de seguridad.

1.4. Formulación del problema

La Problemática antes mencionada atañe a distintos actores tanto del sector de seguridad tradicional y a los que están inmersos en las tecnologías, por lo que se está realizando esfuerzos a nivel mundial para ayudar a contrarrestar esta lacra, es por ello que (Zhai Shuang-can, 2014) han realizado una investigación sobre la base de analizar un IDS existente, en donde proponen un modelo IDS Multiagente Distribuido (DIDS) en base a una red Neuronal BackPropagation (BP). Los experimentos demostraron que el modelo planteado DIDS basado en la Red Neuronal BackPropagation, con 8,600 datos seleccionados de la KDD CUP para el entrenamiento y prueba arrojo los siguientes resultados: Ataques normal 2651, DoS 74, U2R 59, Probe 83, Otros 15.

Otro estudio realizado por (Liang, May2014) permiten mostrar resultados sobre la alta tasa de falsos positivos, como también la baja tasa de detección y otros defectos del IDS propuesto; presenta un nuevo algoritmo de detección de intrusos que utiliza T-S FNN (Takagi-Sugeno Fuzzy Neural Network) para clasificar objetos, divide espacio de los objetos, reconoce comportamientos y las intrusiones normales. Los resultados del experimento muestran que el nuevo método es viable, eficaz y extensible.

Un estudio comparativo de modelos de IDS realizado por (ARAAR & BOUSLAMA) usando seis técnicas de clasificación más representativos como: decision trees (árboles de decisión), BayesNet, NaiveBayes, Rules, SVM y Perceptron multi-layer network (Red Perceptrón Multi capa). En este trabajo presentaron una selección de características utilizando la técnica de bosques al azar (random forest technique) hacia dos reducciones de conjuntos de datos tridimensionales que son eficientes para la formación inicial y continua. Los diferentes resultados y experimentos analizando los dispositivos primordiales y las diversas técnicas de aprendizaje supervisado mejorada son presentados y discutidos a fondo. Demostrando que J48 es el mejor modelo clasificador de IDS con reducido número de características.

El amplio despliegue y propagación de WLAN también ha traído consigo nuevos retos a la seguridad y la privacidad un estudio realizado por (KAVITA & M.USHA, 3/31/2014) investigaron nuevas técnicas para detectar intrusos basado en anomalías en redes inalámbricas; desarrollaron un algoritmo de discriminación mediante el coeficiente de correlación para detectar anomalías combinadas con clasificador bayesiano Naïve en el tráfico inalámbrico y demostró la eficacia utilizando Kyoto 2006 + conjuntos de datos. El experimento se lleva a cabo con el fin de evaluar el rendimiento basado en la precisión, la tasa de detección y tasa de falsos positivos del esquema de clasificación. Los resultados y análisis muestran que el enfoque propuesto ha permitido aumentar la tasa de detección con tasas mínimas de falsos positivos.

Es de vital importancia explorar nuevos algoritmos de inteligencia artificial como las redes neuronales que se puedan aplicar a los IDS para mejorar su funcionamiento.

Es por esto que se han seleccionado las redes neuronales como la herramienta más indicada para realizar esta tarea. Estas redes son capaces de responder a nuevos ataques, de aprender por sí mismas y de ser entrenadas para reconocer patrones normales y anormales, clasificándolos como se deseen. No obstante, las redes neuronales deben ser capaces de actualizarse en tiempo real para ser verdaderamente útiles como un IDS, y es por eso que en esta tesis de investigación se realizará el estudio de comparación de algoritmos de la red neuronal para la mejora en detectar intrusos en redes LAN.

Como base utilizaremos una tesis anterior que desarrolló un IDS que usa redes neuronales recurrentes con wavelets (González, 2011), el cual permitirá comparar los resultados con nuestro modelo propuesto.

1.5. Justificación e Importancia del estudio

Justificación Tecnológica: El uso de algoritmos de redes neuronales permitirá mejorar la detección de intrusos en las aplicaciones web y estar a la vanguardia de la seguridad informática a nivel de red, esto significa que debemos tener una correcta implementación de políticas de seguridad y control para reconocer ataques y tomar decisiones en el momento oportuno.

Justificación social: La realización de esta investigación será beneficiosa para la comunidad estudiantil de la Universidad Señor de Sipan ya que justifica académicamente el cual permitirá conocer los resultados de la comparación del uso de algoritmos de RNAs en la detección de intrusos o ataques informáticos. Además, sentara base para futuros proyectos de investigación.

Justificación económica: El proyecto se justifica debido al uso de aplicaciones de entorno libre y será realizado por el investigador utilizando herramientas de software sin licencia, además los algoritmos de inteligencia artificial se encuentran disponibles en framework en la web.

Las bases de datos y las aplicaciones web hoy en día sufren ataques informáticos a cada minuto, es por eso la importancia de investigar que técnicas o metodologías con

el uso de la I.A, se pueden aplicar para mejorar la detección de intrusos en las redes; para tener una garantía en el funcionamiento correcto de los sistemas y salvaguarda nuestra información tanto empresarial y personal.

1.6. Hipótesis

La implantación de Redes Neuronales permitirá mejorar el Sistema de Detección de Intrusos en redes LAN.

1.7. Objetivos

1.7.1. Objetivo general

Comparar Algoritmos de Redes Neuronales para mejorar la Detección de Intrusos en redes LAN.

1.7.2. Objetivos específicos.

a) Diagnosticar el estado actual de los Sistemas de Detección de Intrusos.
b) Identificar y seleccionar los datos para el entrenamiento y pruebas de detección.
c) Evaluar algoritmos de redes neuronales para mejorar la detección de intrusos en redes LAN.
d) Analizar los resultados.

II. MATERIAL Y MÉTODO

2.1. Tipo y diseño de la investigación.

Esta investigación es del tipo Tecnológica y su diseño Cuasi-Experimental. Se hizo uso del diseño bibliográfico. Los autores Tamayo y Tamayo, (1999), Indican que se debe utilizar información que otros autores obtuvieron para luego evaluarlos y procesarlos dependiendo de la propuesta planteada. Este tipo de diseño bibliográfico ayudó al autor de esta investigación realizar consultas en diversos libros y textos que permitieron posteriormente obtener datos relevantes para luego plasmarlos en el informe.

2.2. Población y muestra

Población:

Se utilizó los archivos de datos utilizados en otros proyectos, el cual consta de un archivo con los diferentes tipos de ataques de un total de 25000 registros.

Muestra:

La muestra será seleccionada utilizando la fórmula estadística de muestra finita.

$$n = \frac{N * Z_a^2\, p * q}{d^2 * (N-1) + Z_{\propto}^2 * p * q}$$

Donde:

N = Total de la población

Zα= 1.96 al cuadrado (si la seguridad es del 95%)

p = proporción esperada (en este caso 5% = 0.05

q = 1 – p (en este caso 1-0.05 = 0.95)

d = precisión (en su investigación use un 5%).

Considerando la población de tramas anteriormente indicado y aplicando la fórmula de muestra finita tenemos un total de 382 tramas o paquetes por clasificar.

2.3. Variables, Operacionalización

2.3.1. Variables:

Variable Independiente:

Algoritmos de Redes Neuronales.

Variable dependiente:

Detección de Intrusos en Redes LAN.

2.3.2. Operacionalización

TABLA 3

Tabla de operacionalización

Variable Dependiente	Dimensiones	Indicadores	Técnicas e instrumentos de recolección de datos
SISTEMA DETECCION DE INTRUSOS EN REDES LAN	Tasa de detección de falsos positivos y negativos	TFP= FP = 1- IE (FP+TN) TFN= FN =1- IS (TP+FN)	Se realizará usando fichas de registro de pruebas de la solución, donde se registrará el desempeño de las técnicas evaluadas.

Fuente: Elaboración propia

Donde:

FP: Número de falsos positivos

FN: Número de falsos negativos

TN: Número de verdaderos negativos

TP: Número de verdaderos positivos

IE: Índice de especificación

IS: Índice de sensibilidad

2.4. Técnicas e instrumentos de recolección de datos, validez y confiabilidad

2.4.1. Análisis.

Con esta técnica se analizó los diversos niveles de ataque en la red local, esto permitirá la selección de la muestra para obtener cual es la más idónea para realizar el entrenamiento en el sistema y así poder lograr óptimos resultados.

2.4.2. Observación.

Es el proceso de observación directa de lo que ocurre durante el trabajo cotidiano asociado, se registra la información que se obtiene al desarrollar la investigación como por ejemplo el tiempo de respuesta, los reportes, etc.

2.4.3. Prueba del Sistema.

Las muestras serán procesadas utilizando algoritmos de redes neuronales para detectar y analizar los ataques o vulnerabilidades. Los datos se almacenaron en un banco de datos para que luego el sistema fuera entrenado con estos datos para que clasifique las muestras dependiendo del tipo de deficiencia que se identifique. Luego se recolectó la información más importante respecto a la eficacia y eficiencia de cada técnica para luego obtener un resultado al final de la fase de prueba.

2.5. Procedimiento de análisis de datos.

Se analizarán los efectos de las pruebas evaluadas y determinaremos que técnicas es la más adecuada para detectar intrusos en las redes de área local.

Será necesario la ayuda de un experto en Seguridad Informática conocedor de los diversos métodos de inteligencia artificial para aplicarlo en la detección de intrusos.

Análisis estadístico e interpretación de los datos.

En el estudio de las técnicas y del algoritmo evaluado se calculó las tasas de detección de intrusos de los 5 niveles de ataques.

Los resultados serán evaluados utilizando los indicadores siguientes:

Tasa de falsos positivos:

$$TFP = \frac{FP}{(FP + TN)} = 1 - IE$$

Tasa de falsos negativos:

$$TFP = \frac{FN}{(TP + FN)} = 1 - IS$$

Además del cálculo de los indicadores anteriores es útil definir dos medidas adicionales. La primera se relaciona con el grado de especificación del modelo a evaluar, es decir, el porcentaje de reconocimiento para los patrones de tráfico de red que no son intrusivos; esta medida se llama índice de especificación y está relacionada con la emisión de falsos positivos. La segunda medida es la de sensibilidad y describe el porcentaje de detección de conductas intrusivas llevadas a cabo correctamente. El cálculo se describe en las siguientes fórmulas:

$$sencibilidad = \frac{numero\ de\ verdaderos\ positivos}{numero\ de\ verdaderos\ positivos + numero\ de\ falsos\ negativos}$$

$$especificacion = \frac{numero\ de\ verdaderos\ negativos}{numero\ de\ verdaderos\ negativos + numero\ de\ falsos\ positivos}$$

2.6. Criterios éticos

a) **Medio ambiente. -** La investigación propiciará el cuidado del medio ambiente.
b) **Confidencialidad. -** La presente investigación cuidará de la identidad de los organismos o entidades donde hubiese detectado vulnerabilidades.
c) **Originalidad. -** Se referencio las fuentes bibliográficas de la información que se recopilo para el desarrollo del proyecto y así evitar plagio intelectual.

d) **Veracidad**. - Toda la información que se adjuntó en el presente informe es verdadera.

2.7. Criterios de rigor científico

a) **Validación. -** Se realizó la validación de los instrumentos para aplicarlos en la propuesta de investigación.
b) **Confiabilidad. -** Se efectuó los cálculos estadísticos de manera correcta para determinar el nivel de la consistencia de los instrumentos para recolectar los datos
c) **Contrastación. -** Para realizar la contrastación en la hipótesis se utilizó métodos estadísticos.
d) **Relevancia. -** Permite evaluar el logro del objetivo planteado en el proyecto.

III. RESULTADOS

3.1. Resultados en tablas y figuras.

Primero se propuso en el diseño que para poder identificar el tipo de ataque se asignaría a la neurona de salida el valor de 1 y serian ceros para las 4 neuronas restantes. Pero al llevarlo en práctica esta cadena de prueba arrojo resultados decimales que comprenden entre -1 y 1, y en el vector de salida incluía valores decimales en el intervalo -1<= x <=1.

Estos resultados no serían óptimos para esta investigación debido a que se necesita valores de 1 y 0 (binarios), para poder obtener estos valores se utilizó una función en MATLAB® llamada función de competencia (Beale M., 2003). Esta función tiene como propósito que, si se le brinda un vector con valores determinados, el valor máximo tendrá el valor de 1, y los restantes tendrán valor 0. Por lo tanto, la salida se transformará tal y como se muestra en la siguiente figura.

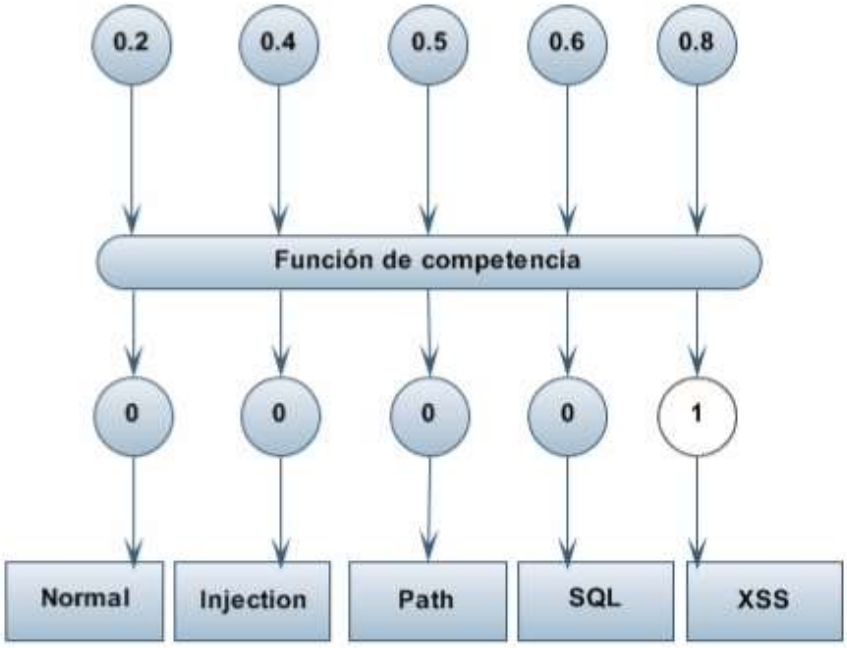


Figura 9. Salida después de la función de competencia. Fuente: (Oropeza Clavel, 2007)

En el vector de salida, la red neuronal identificó un ataque XSS, con este resultado se toma una acción para poder advertir ataques similares, es importante decir que si se usa vectores de tamaño fijo a la entrada de las tres redes (64 elementos), algunas cadenas con elementos que superen a 64 / 8 = 8 elementos se procederán a dividir entre dos o más cadenas y poder representar la cadena de comandos (Alarcón, 2005).

En algunos casos la red realizó la identificación correcta de una cadena como una forma de ataque, pero esta fue errónea al querer clasificarla en las categorías que se tenían. Por ejemplo, si la cadena que se envió era un ataque SQL la red debió identificarla como un ataque de inyección SQL, pero el IDS si logró identificar como una cadena anormal por lo que conlleva a detectar que es un ataque a la red. Para subsanar estos problemas de desempeño de la red, se procedió a definir 2 porcentajes (clasificación e identificación) y poder realizar la evaluación de los resultados. El porcentaje de clasificación será el número de veces que la red neuronal determinó que la cadena de caracteres sea un ataque frente a un comando normal. El porcentaje de

identificación consiste a la cantidad de veces que cualquier red determine que sea un ataque dentro de las 5 categorías. Si la cadena que se representó en la categoría Path se debe de identificar como tal. Esto se visualiza en la Figura10.

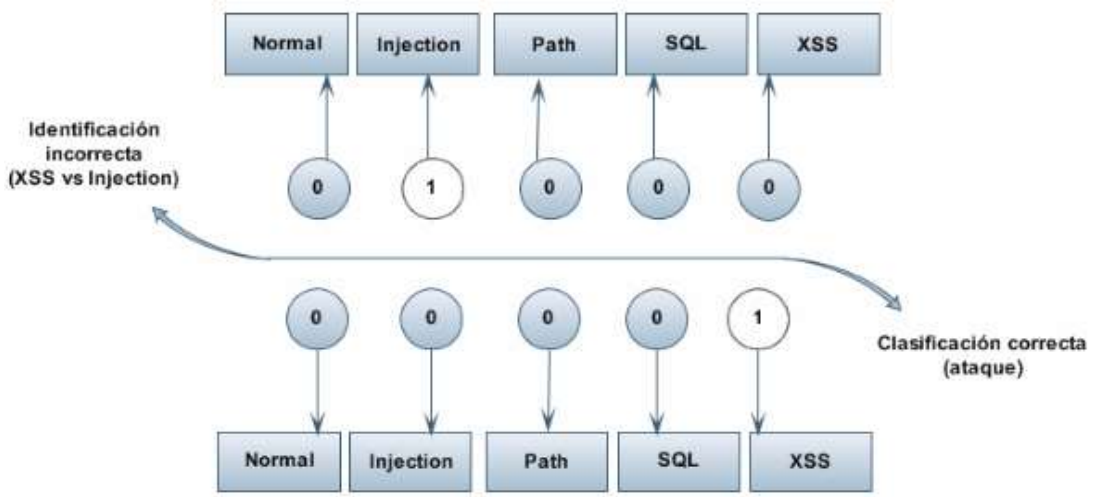


Figura 10. Clasificación e Identificación. Fuente: (Oropeza Clavel, 2007).

El porcentaje de clasificación deberá de ser mayor que el porcentaje de identificación y bastará que la red realice la identificación del vector para determinar si es un ataque o no para poder tener un desempeño correcto con los valores de entrada de formato binario.

Resultados Comparativos.

Los resultados comparativos entre los distintos modelos utilizados en el Sistema de Detección de Intrusos (IDS), se observa en la tabla 4 la primera columna describe el tipo de arquitectura utilizada, así como el número de unidades en cada capa, la segunda columna especifica el número de épocas utilizadas en el entrenamiento, la tercera el error obtenido y las 4 columnas restante el número de falsos positivos, falsos negativos, verdaderos positivos y verdaderos negativos. Como puede observarse en la Tabla 4, la arquitectura con menos falsas alarmas emitidas fue la red con alimentación hacia adelante con solo 8 falsas alarmas, seguida del modelo Elman con

10 y RNN con 20. En términos de falsos negativos la arquitectura con menor número fue la FRNN con 23, seguida de la Elman y FeedForward con 77 y 92 respectivamente. Para la obtención de verdaderos positivos, observamos a la arquitectura RNN obtener el mejor número con 329 ataques detectados seguida por las arquitecturas Elman y FeedForward con 275 y 260. Finalmente, en la detección de patrones normales podemos establecer que las arquitecturas con mejores números fueron la de alimentación hacia adelante FF y Elman con 268 y 266, seguida del modelo RNN con 261 patrones.

TABLA 4

Comparativa de Resultados Finales

Arquitectura	Epocas	MSE	FP	FN	TP	TN
FeedForward 64x15x15x5	239	0.0839	8	92	260	268
Elman 64x30x30x5	106	0.0537	10	77	275	266
RNN 64x30x5	60	0.0044	20	23	329	256

Nota: Comparativa de Resultados Finales. Fuente: Elaboración propia

En la Figura 11 se muestran las cantidades FP, FN, TP, TN, obtenidas para cada arquitectura.

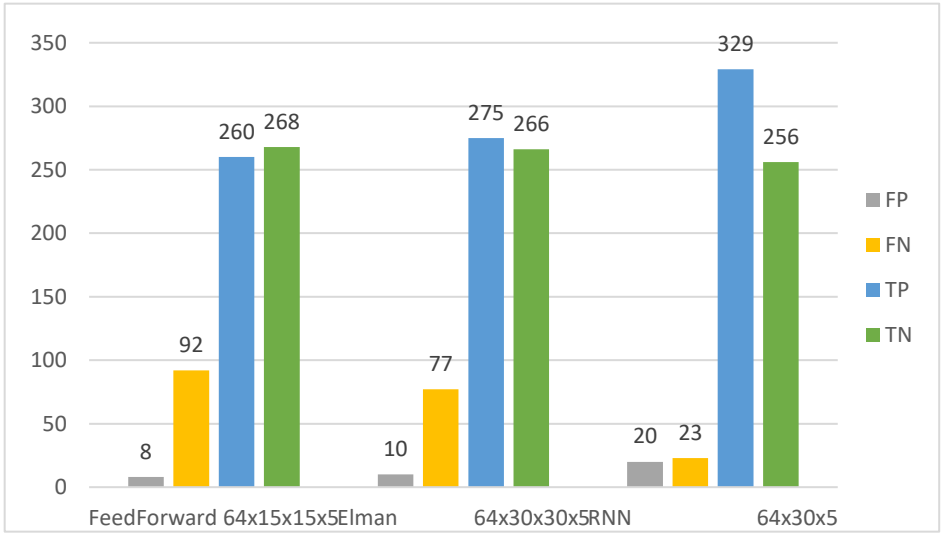


Figura. 11. Número de FP, FN, TP y TN. Fuente: Elaboración propia

En la Tabla 5 se concentran los diversos resultados según el tipo de tasa, así como las medidas estadísticas de desempeño especificación y sensibilidad.

Como se puede observar el modelo con menor tasa de FP corresponde a la red con alimentación hacia adelante seguida de los modelos Elman y RNN. Respecto a la tasa FN el mejor resultado lo obtiene la arquitectura RNN con 6.53%. Los índices de especificación y sensibilidad mejor equilibrados son obtenidos por la arquitectura RNN, seguida de los modelos Elman y FF.

TABLA 5

Especificación y sensibilidad.

Arquitectura	TasaFP	TasaFN	Especificación	Sensibilidad
FeedForward	2.89%	26.13%	97.10%	73.86%
Elman	3.62%	21.87%	96.37%	78.12%
RNN	7.24%	6.53%	92.75%	93.46%

Nota: Especificación y sensibilidad. Fuente: Elaboración propia

La Figura 12 presenta las diferencias entre las medidas de especificación y sensibilidad para cada uno de los modelos estudiados.

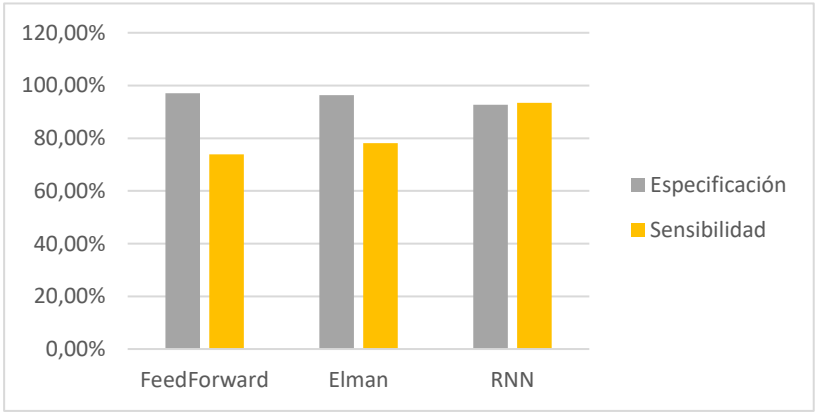


Figura 12. Comparación de especificación y sensibilidad. Fuente: Elaboración propia.

La Tabla 6 figura los errores por categoría que se obtuvo en cada arquitectura, se puede apreciar que para todas las arquitecturas se obtiene un índice mayor de errores en la categoría de inyección y un menor índice de errores en la categoría de path solamente en la RNN.

TABLA 6

Errores por categoría

Arquitectura	Normal	Inyección	Path	SQL	XSS
FeedForward	8	84	8	7	29
Elman	19	78	14	4	8
RNN	20	36	1	5	37

Nota: Errores por categoría. Fuente: Elaboración propia

En la Tabla 7 se observa los porcentajes de aciertos por categoría para cada modelo. Al analizar estos valores encontramos que la categoría path fue correctamente detectada en todos los modelos, y que las categorías con menor índice de detección fueron SQL e Inyección.

TABLA 7

Porcentaje de detección por categoría

Arquitectura	Normal	Inyección	Path	SQL	XSS
FeedForward	97.10%	3.44%	95.89%	0%	53.96%
Elman	96.37%	10.34%	92.82%	42.85%	87.30%
RNN	92.75%	58.62%	99.48%	28.57%	41.26%

Nota: Porcentaje de detección por categoría. Fuente: Elaboración propia

En la Figura 13 se observa el resultado en porcentaje de los aciertos por categoría para cada modelo.

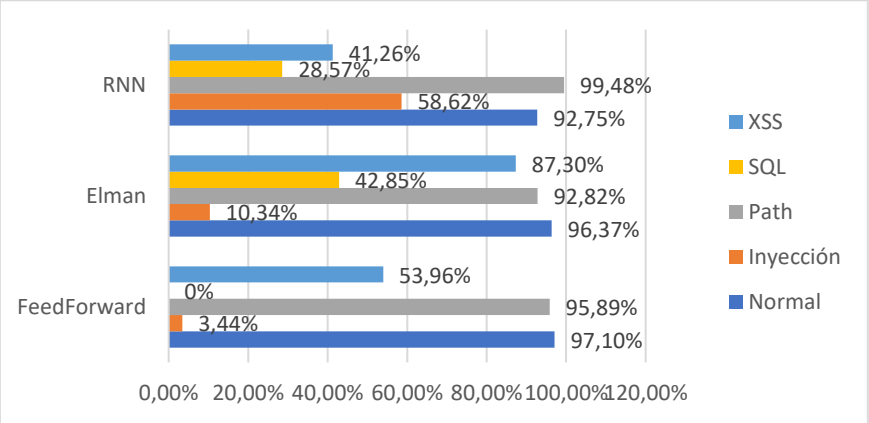


Figura 13. Porcentaje de aciertos por categoría. Fuente: Elaboración del autor.

Como se explicó al principio del capítulo, existen dos cantidades importantes para medir el desempeño de los IDS, llamados porcentajes de clasificación e identificación. El porcentaje de clasificación describe la medida en la que el IDS es capaz de clasificar los patrones de entrada como tráfico normal o intrusivo mientras que el porcentaje de identificación se refiere a la medida en la que el modelo clasifica cada patrón como perteneciente a cada una de las cinco categorías posibles. La Tabla 8 resume los porcentajes de clasificación e identificación para cada modelo.

TABLA 8

Porcentajes de Clasificación e identificación

Arquitectura	Clasificación	Identificación	Complejidad
FeefForward	84.07%	78.34%	n^2
Elman	86.14%	81.84%	n^2
RNN	93.15%	84.23%	n^4

Nota: Porcentajes de Clasificación e identificación.

Fuente: Elaboración propia

Tomando en cuenta los resultados de clasificación e identificación, el modelo con mejores valores es la RNN, en segundo lugar, se ubica el modelo Elman.

En la Tabla 9 se concentran las tasas de falsos positivos y verdaderos positivos.

La idea es que la tasa de falsos positivos debe estar cerca a cero dado que representa la emisión de falsas alarmas, por otra parte la tasa de los verdaderos positivos debe ser cercana al 100% pues este valor representa la detección de patrones intrusivos. Como puede observarse la menor tasa de falsos positivos es obtenida por el modelo FF, seguida de Elman y RNN. El mejor resultado obtenido tomando en cuenta ambos indicadores corresponde a la arquitectura RNN con los valores 7.24% y 93.46% pues mantiene el índice de falsos positivos en un valor bajo y la detección de ataques por arriba del 90%.

TABLA 9

Falsos Positivos vs verdaderos positivos

Arquitectura	Tasa de Falsos Positivos	Tasa de verdaderos positivos
FeefForward	2.89%	73.86%
Elman	3.62%	78.12%
RNN	7.24%	93.46%

Nota: Falsos Positivos vs verdaderos positivos. Fuente: Elaboración propia

3.2. Discusión de resultados

Se demostró que la red neuronal recurrente es mejor en cuanto a la comparación de las demás redes que se utilizaron. Este tipo de redes mostro una cantidad de errores que se mostraron en la medida de porcentaje según su clasificación para poder identificar los falsos positivos y negativos. Se concluyo con un cuadro comparativo para mostrar que la red neuronal recurrente es superior a los demás.

En la salida, las redes mostraron datos innecesarios para esta investigación debido a que se necesita únicamente valores binarios, por ese motivo se incluyó una función de competencia al sistema en el cual permite la asignación de valores 0 y 1 para poder facilitar el procesamiento de los resultados.

La red realiza una clasificación correcta de un vector de entrada como un dato normal, pero hay una dificultad en colocar este vector en la categoría que le corresponde dentro de las cinco posibles.

Existe la ocasión que la red neuronal ignore una intrusión y en este caso se habla de un falso negativo, lo cual conllevaría problemas graves al IDS. En el desarrollo, el resultado de los porcentajes fueron mínimos por lo que no se consideró como grandes complicaciones.

Cuando se obtuvieron los valores finales de las redes luego de su entrenamiento y las pruebas con otros datos, se mostró que la red neuronal recurrente conecta con todos los parámetros a excepción del costo de procesamiento por neurona.

Si se aumenta el número de neuronas a la red, esta funciona rápidamente, pero ocasionaría que el procesador demande más tiempo de lo normal debido a que se aumenta a la cuarta potencia dependiendo del número de elementos que hay en el sistema. Además, se puede decir que el desarrollo es viable porque se demostró que el algoritmo de entrenamiento basado en un IDS no ejercía muchos recursos del procesador para que se ejecute correctamente.

3.3. Aporte práctico

Como se pudo observar en el área de conocimientos, la detección de intrusos y las redes neuronales son muy importantes en la efectividad y eficiencia de los IDS, por lo tanto, en esta presente investigación se propone una solución para poder detectar comportamientos inusuales en la red aplicando técnicas en Inteligencia Artificial específicamente en las redes neuronales artificiales recurrentes (ANN).

Para desarrollar esta propuesta se tomó en cuenta trabajos científicos, para así poder detallar las etapas de la implementación, selección, representación de los datos, la selección del modelo y arquitectura de la red neuronal, la configuración de los parámetros de entrenamiento, la verificación del aprendizaje de la red y las herramientas utilizadas para la simulación de las ANN.

Definición del Método Propuesto.

Los factores tales como disponibilidad, caracterización de datos, topología y el algoritmo de entrenamiento deben de ser analizados muy cuidadosamente para que la red neuronal cumpla su objetivo. (HEADY ,1990). Definiendo el problema que se presenta, la presente investigación brinda una solución que consiste en la comparación de algoritmos de redes neuronales para mejorar los IDS en redes de área local.

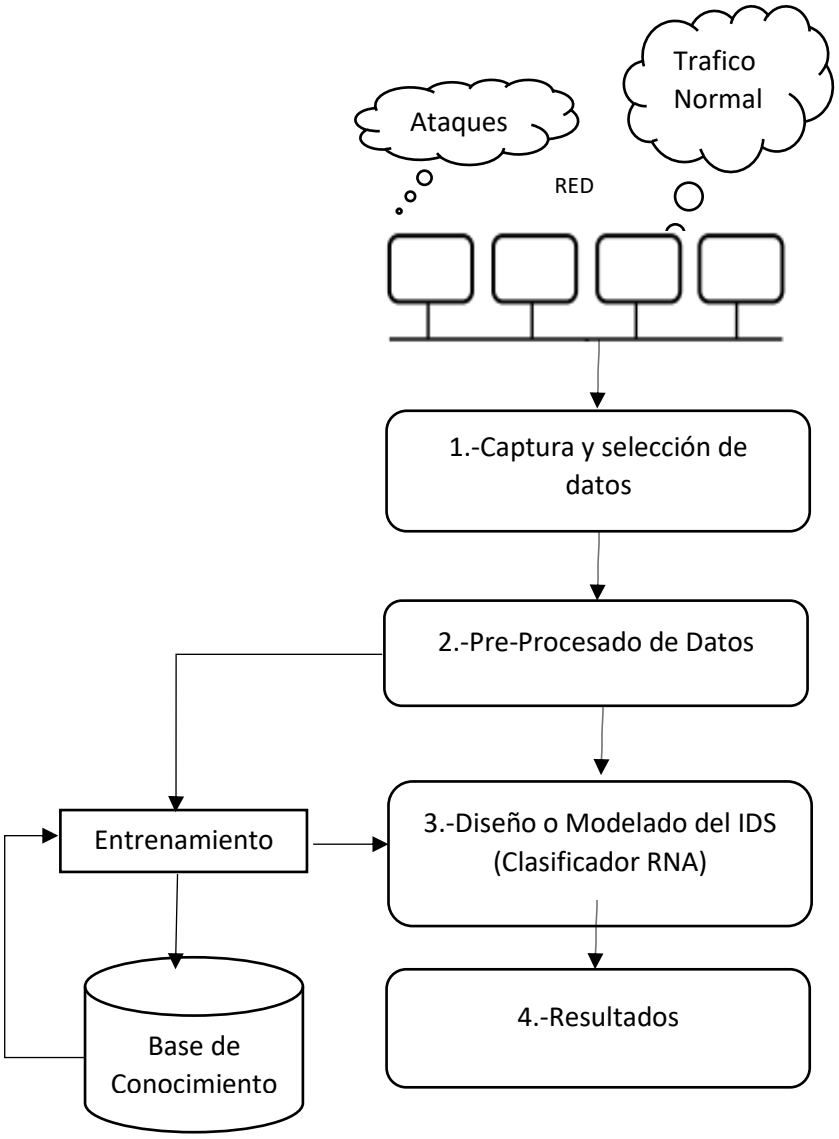


Figura 14. Procesos elaborados por el método propuesto. Fuente: Elaboración propia.

Captura y Selección de Datos.

La captura del tráfico de red se realiza por medio de un sniffer, elaborado en la plataforma Java con la Biblioteca JPCAP la cual permite interactuar con la tarjeta de Red en modo Promiscuo. Luego se realiza una configuración básica como el filtrado de puerto y capturar los paquetes entrantes.

Capturado el paquete correctamente es procesado o analizado para realizar su extracción de características de los campos deseados en cada trama; es decir la línea de requerimiento de petición realizada al protocolo HTTP será almacenada en un banco de datos.

Se ha realizado la investigación de los ataques que se efectúan dentro de la red, en especial aquellos que explotan vulnerabilidades de las aplicaciones web. Los ataques que han sido seleccionados son: Inyección de comandos, ataques a bases sql, cross site scripts y modificación de ruta.

En esta etapa se filtran y se analizan los paquetes, con el propósito de obtener cadenas de caracteres, dentro de los cuales existen palabras claves que identifican a los ataques.

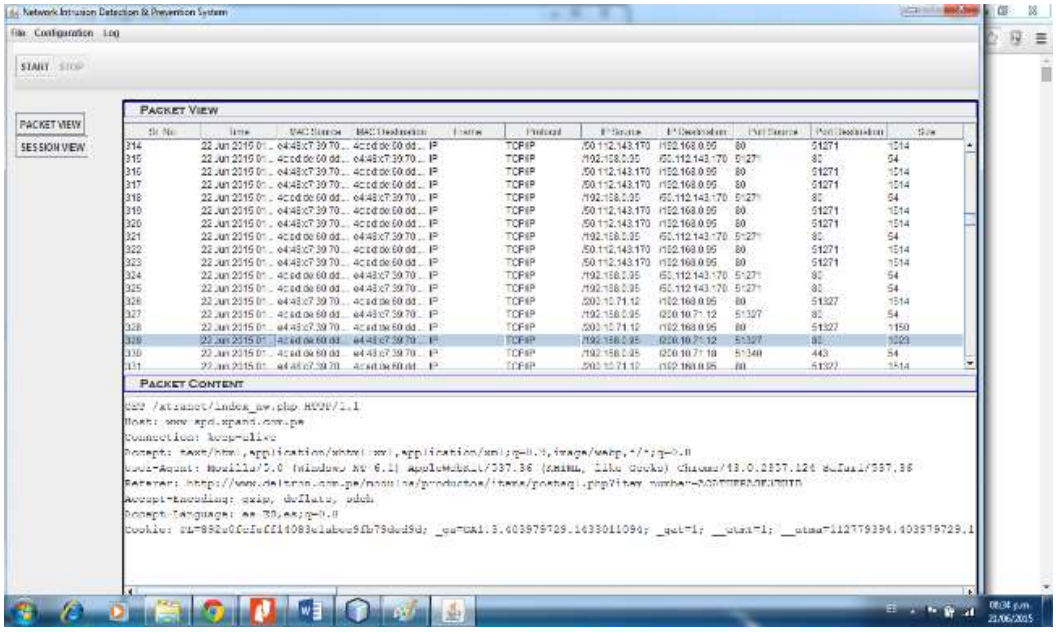


Figura 15. Sniffer captura de paquetes de la red. Fuente: Elaboración propia.

Pre procesamiento de los datos.

Esta etapa consiste en preparar los datos de entrada para aplicarlos a un formato especial para que pueda ser usado en la fase de entrenamiento y prueba de la red neuronal.

Las variables de entrada tienen un estándar para poder transformar los requerimientos HTTP a un formato de 0 y 1. Esta acción se debe detallar adecuadamente el orden secuencial, este orden llevar a cabo un requerimiento HTTP: //nombre.exe?param1=\..\archivo, la estructura de esta cadena consiste en el nombre del archivo, el parámetro que se envía y seguidamente la secuencia alfanumérica, esta última parte varía de acuerdo al equipo. La parte que mas importa en esta cadena es la extensión de archivo y los caracteres especiales.

El primer paso para procesar los datos es realizar el reemplazo de la cadena alfanumérica haciendo uso del carácter "@" : //@.exe?@=\..\@ . Por ejemplo:

TABLA 10

Ejemplo de datos de entrada

CADENA DE ENTRADA	CLASIFICACIÓN
&@_@=@&@_@	NORMAL
")</@>/@.@	XSS
'--@	SQL
"./."./@@	PATH
&@=";@";	INJECTION

Nota: *Ejemplo de datos de entrada.* Fuente: (Oropeza Clavel, 2007)

Cuando se obtiene cada una de estas partes que son vitales para los requerimientos, el paso que sigue es realizar la conversión de la cadena de caracteres en ASCII

decimal que debe corresponder por cada carácter para luego transformarlo a un formato binario.

Para poder realizar un ejemplo se tomarán las cadenas antes mencionadas, por lo cual se representa en formato ASCII de la siguiente manera:

TABLA 11

Conversión a formato ASCII

CADENA ASCII	CLASIFICACION
38 64 95 64 61 38 64 95 64	NORMAL
34 41 60 47 64 62 47 64 45 64	XSS
39 45 45 64	SQL
34 46 47 46 34 46 47 64 47 64	PATH
38 64 61 34 59 64 61 34	INJECTION

Nota: Conversión a formato ASCII. Fuente: (Oropeza Clavel, 2007)

En la etapa del entrenamiento para las redes neuronales se va a utilizar MATLAB, se debe procesar los datos para una entrada válida para el Framework Matlab, para el cual utilizamos un script en R para dividir los datos en dos partes: input y output.

Además, como Matlab no usa nombres de atributos, se tuvo que eliminar manualmente los valores de los encabezados y convertir los valores de clase en valores numéricos.

Al finalizar el pre procesado se realiza la conversión de cadenas de valores decimales en los valores binarios para así poder hacer la alimentación de las entradas a la red neuronal

Diseño o Modelado del IDS.

En la etapa anterior los datos ya fueron preparados, ahora se realiza el diseño de la red neuronal para poder hacer el entrenamiento y probar los datos. El primer problema que se encontró es determinar el número de neuronas que se debe tener, también el número de capas ocultas si así fuera el caso y el número de neuronas de salida.

Para poder determinar la cantidad de datos de entrada se basó en la ecuación de Williams y Zipser (Willams, 1990), $N \geq W/\varepsilon$, en donde W es el número total de pesos presentes en la red y ε es un parámetro constante de error cuyo valor es 1%. Esto permitirá administrar los datos de entrada para poder obtener resultados que tengan un margen de error a 1%.

El número de neuronas de salida debe de ser igual al número de tipos de comportamiento normal debido a que el IDS debe de enviar una señal en formato binario para que se pueda reconocer mediante un elemento complementario y así poder determinar las medidas correctas en caso de un ataque.

Cuando se realice la identificación de la cadena de caracteres, la primera neurona del sistema tendrá el valor 1 y las restantes 0. Si es un ataque, la neurona 5 tendrá el valor 1 y se identificará por ser un ataque XSS. Además, se podría diseñar una red con dos neuronas en la capa de salida, con las siguientes categorías: normal y ataque, pero esto conllevaría a no tener una claridad al momento de identificar y solo permitiría identificar si es un ataque o no. Por estos motivos se optó por cinco neuronas en la capa de salida.

La cantidad de neuronas en la capa oculta dependerá de que tipo de red se va a realizar el modelado. La primera prueba se realizó en una red feedforward simple se tuvo un diseño con un tamaño de 41x15x15x5, en la capa principal tiene 41 neuronas, en la capa oculta tiene 15 y en el vector y_k 5 , estas corresponde a la salida de cada uno según el tipo de ataque. Esta red se observa en la siguiente figura.

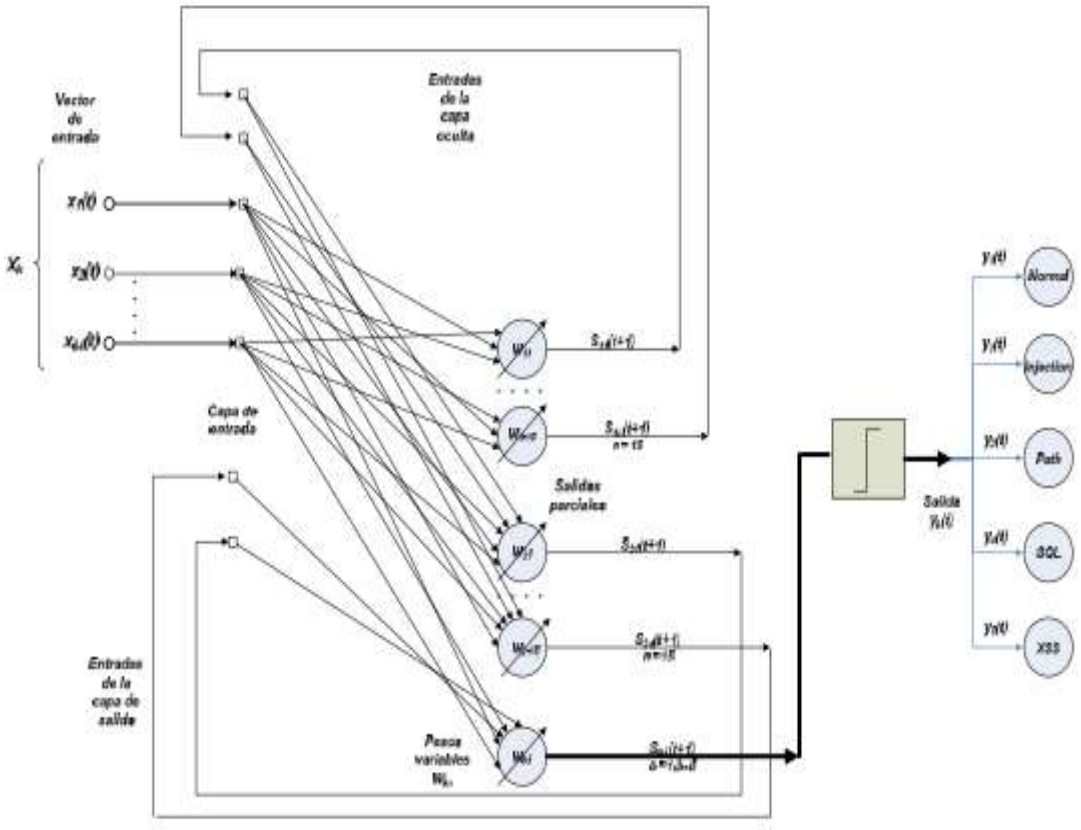


Figura 16. Red feedforward, 41x15x15x5. Fuente: (Willams, 1990)

Las neuronas de una capa están conectadas con las de la siguiente capa empezando en la capa principal y los elementos del vector X_k, suministrando la información (las salidas s_{mn} (t+1)) hacia delante dentro de la red. El vector de los pesos W_k es reestablecido conforme los epochs pasan y al final del último los pesos individuales $w_{11}...w_{1\text{-}15}$, $w_{21}...w_{2\text{-}15}$, etc, asumirán sus valores finales cuando llegue a las cinco salidas globales son (t+1) que pasarán por la función de umbral signum. Un epoch es un ciclo completo de procesamiento en la red, desde la entrada hasta la salida para un *input nuevo*.

Para determinar la cantidad de neuronas en la capa oculta se tomó en consideración los siguientes criterios: Según los autores (Willams, 1990) (Embrechts, 1993) indican que la cantidad máxima de capas que debe tener una red neuronal sin cometer un sobre procesamiento es de dos capas.

El número de neuronas serían las mismas por capa para una red feedforward como la que tenemos aquí ha sido definido por

$$(Ni - No)/\ 4$$

Donde:

Ni: Número de neuronas a la entrada

No: Número de neuronas a la salida de la red,

Para el desarrollo de esta investigación las variables antes mencionadas tendrían los siguientes valores: Ni =41 y No = 5, donde las capas ocultas deberán tener 15 neuronas. Por este motivo es que la red Feedforward fue diseñada de este modo.

En la segunda red, se diseñó como una opción para que sea una base de un IDS. Se eligió una SNR Elman. Este tipo de redes tiene caminos que son de retroalimentación, lo que conlleva a que tenga una cantidad elevada de neuronas para que se pueda utilizar toda la capacidad de poder recordar el estado interno inmediato anterior de la red.

Si se tiene una red Elman con la misma cantidad de neuronas en la capa oculta frente a una red feedforward se observaría una diferencia mínima en cuanto a su desempeño. Pero si el diseño es el apropiado puede haber el caso de que la sobrepase fácilmente en cuanto a velocidad de convergencia y porcentajes de error.

El criterio que se tomó en consideración para el diseño fue que el número máximo de capas ocultas sin ocasionar problemas en el procesamiento. Para determinar el número de neuronas que se necesitan en la capa oculta se aplicó la siguiente formula: (Ni – No) / 2 (Embrechts, 1993), las variables son las mismas que se detallaron anteriormente. En la aplicación que se basó en HTTP, se mostró un resultado de un aproximado de 30 neuronas por capa (ver Figura 17). Que sería el doble de las neuronas que se necesita a comparación de la red anterior y esto permitiría que se podrá compensar con la rapidez y exactitud del desempeño de la red.

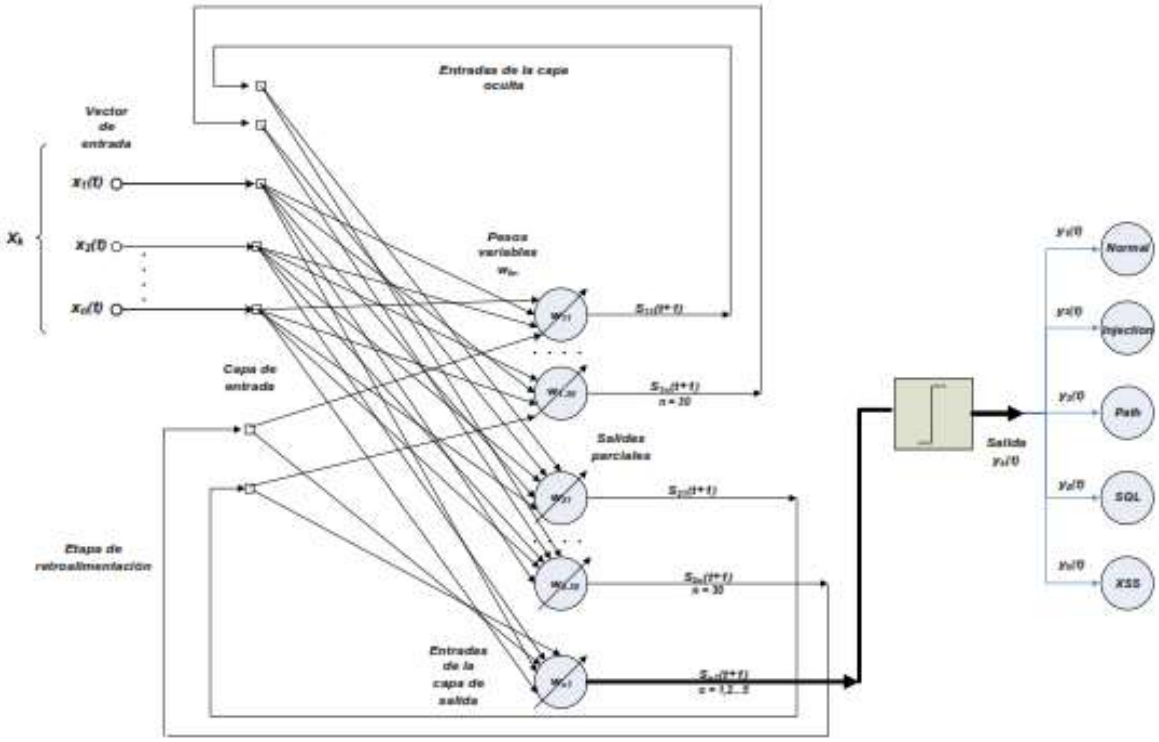


Figura 17. Red Elman 41x30x30x5. Fuente: (Willams, 1990).

Las RNN tiene índices de convergencia elevados y el porcentaje de error es bajo a comparación de las demás redes. En cuanto a la complejidad de las conexiones entre las capas se observó tiempos de procesamiento del orden de $O(n^4)$. Se utilizó un algoritmo de aprendizaje en tiempo real. Por este motivo es importante encontrar el tamaño necesario que debe tener la red y tratar de no exceder con la cantidad de

neuronas o capas ocultas que se debe utilizar en una RNN sin hacer un alto procesamiento de la red.

Se utilizó la siguiente estructura de la red para llevar a cabo un IDS se puede visualizar en la Figura 18, consiste de 30 elementos en la capa oculta, los delays a la salida de dicha capa y la clasificación final.

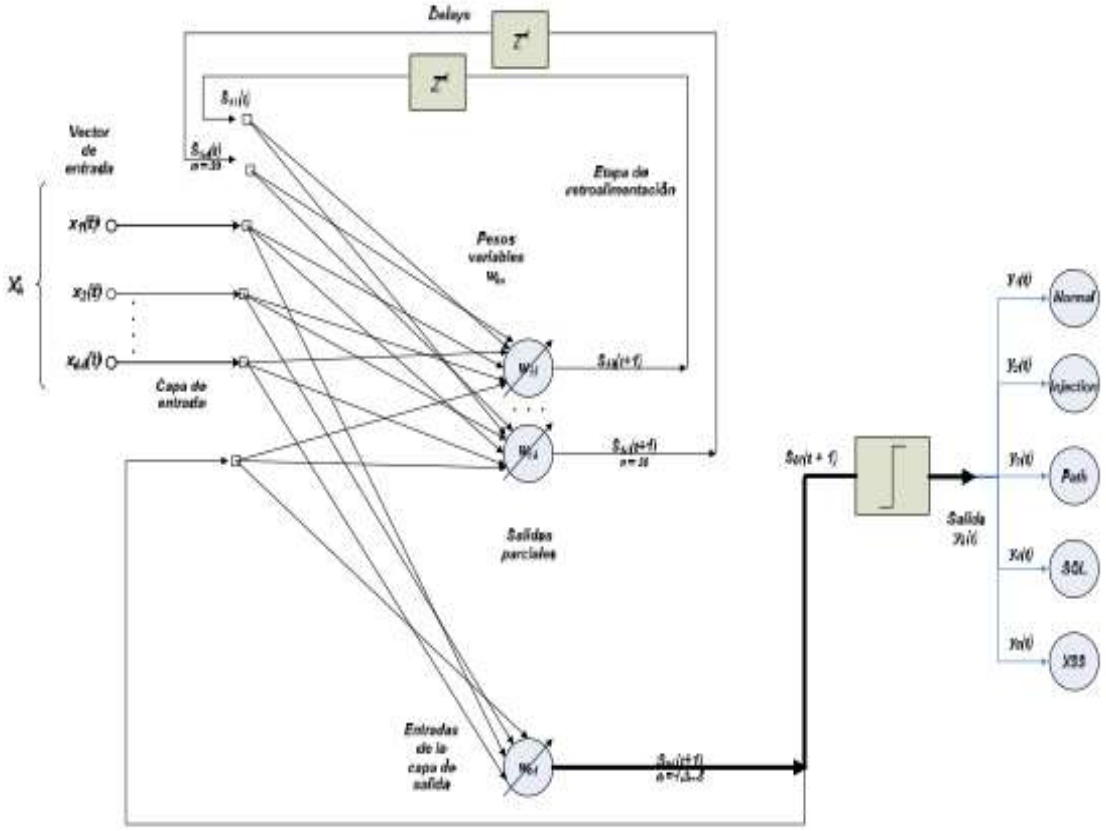


Figura 18. Red Recurrente completamente conectada 41x30x5. Fuente: (Embrechts, 1993)

Según (Embrechts, 1993), la cantidad de capas de una RNN debe de ser una. Si se agrega una segunda capa la red mejora un poco en cuanto a su desempeño, pero esta pequeña variación ocasionaría márgenes de error. Por eso es mejor trabajar con una capa en la red.

El RNN sobrepasa en exactitud a una SNR Elman o red feedforward debido a que se utiliza una sola capa oculta ya que esto es ventaja para el tipo de diseño que tiene la

red. Finalmente, para el cálculo de la cantidad de neurona para las capas se aplicó la formula anterior que se usó en la red Elman, se obtuvo un total de 30 neuronas en la capa oculta.

SIMULACIÓN DE LAS REDES NEURONALES

Después de que se determinaron los parámetros, el paso siguiente es realizar el entrenamiento de las redes neuronales en MATLAB ®. Para realizar esta acción se debe de fijar un valor máximo de error permitido y usar algoritmos BPTT y RTRL, se realizó el entrenamiento de las redes para definir en qué situación se alcanza el valor que se espera del error y con esto señalar que estos valores que se estableció en los pesos de cada capa fueron los mejores para la sesión de prueba

La red que se probo fue la red feedforward. Para llevar a cabo el entrenamiento se usó una variante del algoritmo BPTT que está disponible en el ToolBox de MATLAB® para el uso en redes neuronales, propagación elástica. Esta variante permite el cálculo del gradiente al entrenar la red, se toma en consideración el signo del parámetro para indicar que la superficie de error aumenta o disminuye, esto permite tomar la decisión de cambiar o no los pesos para poder acercarse al mínimo global (Beale M., 2003).

Al ejecutar la simulación del código fuente que esta detallado en el anexo de esta investigación, el desempeño de la red feedforward fue el siguiente:

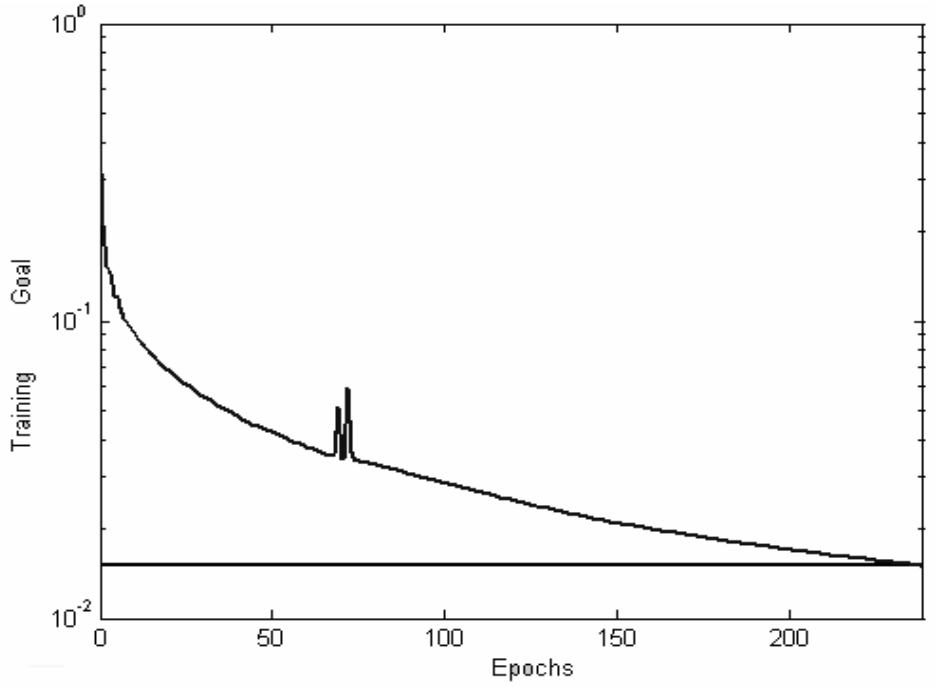


Figura 19. Simulación de la red feedforward, 41x15x15x5. Fuente: (Beale M., 2003).

La gráfica anterior muestra un valor máximo de error permitido (0.015) en un total de 239 epochs (Figura 19). Por el momento, no se cuenta con un parámetro de comparación, por lo que se debe esperar a las simulaciones siguientes de las otras redes para poder identificar si el valor es bueno o malo.

Para la simulación de red SNR Elman se usó el mismo algoritmo de entrenamiento que la red feedforward. De esta forma se pudo visualizar como estas arquitecturas mostraban resultados distintos a pesar de que ambos estén ejecutándose en las mismas condiciones iniciales, ya sea en la cantidad y en la base de datos que se ingresó a la entrada cuando comenzó su entrenamiento.

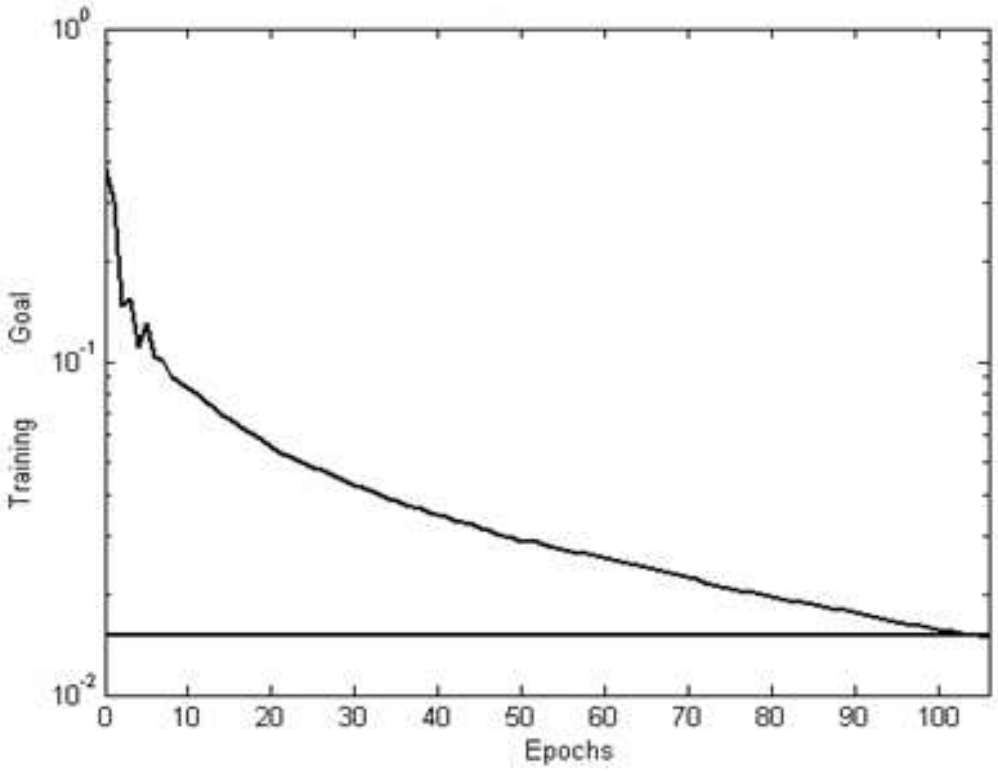


Figura 20. Simulación de la Red Elman 41x30x30x5. Fuente: (Beale M., 2003).

Al realizar la comparación de las simulaciones de la red feedforward, se observa que incremento la velocidad de la red en cuanto al margen de error permitido por los requerimientos del IDS. A comparación de los 239 epochs de la simulación primera, esta red logró la meta en sólo 106, lo que muestra una mejoría import

ante en el desempeño que se esperó en la red neuronal recurrente simple.

Como se puede observar el aumento de la cantidad de neuronas en la capa oculta tuvo el efecto que se predijo, a pesar de tener pocas oscilaciones no deseables cerca de los 60 epochs, estos forman parte del comportamiento de las redes neuronales ya que reaccionan al máximo o mínimo locales antes de estabilizarse a los valores que se requieren.

La red que se consideró como la mejor para el IDS es la RNN porque esta red usa menos capas ocultas a comparación de las demás, además de la capacidad efectiva que tiene en cuanto a la predicción, procesamiento y generalización a través del tiempo.

Para el entrenamiento esta red se usó el algoritmo *Real Time Recurrent Learnign Algortihm* o RTRL, se observó diferencias importantes en cuanto a las simulaciones anteriores en MATLAB® por no contar con las funciones que son necesarias dentro de su ToolBox de redes neuronales para llevar a cabo el entrenamiento de estas redes de forma automática. Se visualizo en los casos anteriores que solo basto el aprendizaje de la sintaxis de las funciones para su entrenamiento, pero para el último caso se dedicó más tiempo de lo debido para el desarrollo de la rutina de la creación, entrenamiento y prueba para el RTRL adaptando el formato, el tamaño de los datos y haciendo uso de las funciones necesarias. El código fuente de todas las funciones que fueron desarrolladas para realizar el entrenamiento se especifica en el anexo 1. La gráfica que se obtuvo en el desempeño de la última red confirmó la expectativa que se esperaba en cuanto al funcionamiento, debido a su rapidez y a la cantidad mínima de iteraciones que se estimaba:

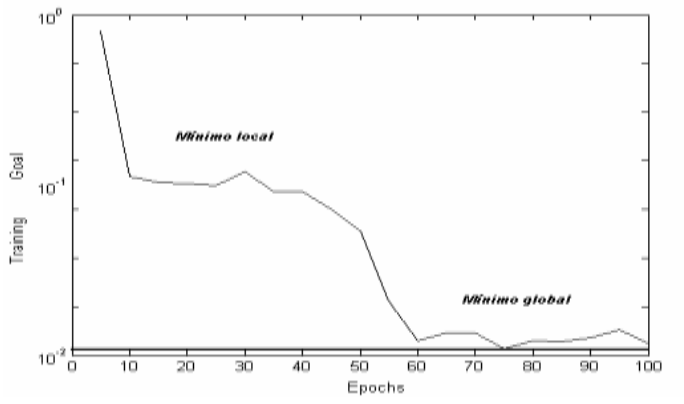


Figura 21. Red Neuronal Recurrente Completamente Conectada 41x30x5. Fuente: (Beale M., 2003)

Esta red obtuvo el parámetro de error esperado en un número de epochs significativamente menor a las otras dos: únicamente 60 epochs comparados con 106 y 239 respectivamente. Tuvo una variación en el intervalo 10 – 40 epochs debido a que el RTRL ha encontrado un mínimo local, pero al no ser el valor deseado (0.38 vs 0.015), no es el valor permanente de la red y luego se encuentra el verdadero mínimo al llegar a 60 epochs, después se encontró pequeñas variaciones pequeñas en la salida de la red. A pesar de tener sólo una capa oculta se obtuvo una mejora en velocidad de alrededor del 40%.

IV. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

Como se puede apreciar, en la red neuronal recurrente completamente conectada es de mejor desempeño en cuanto a su velocidad de procesamiento con un total de 60 épocas, como también su error es menor a las demás redes FerdForward y Elman.

En cuanto al índice de identificación y clasificación, con los falsos positivos y negativos los resultados se observan que la RNN también fue superior a las demás, debido al uso del algoritmo de entrenamiento RTRL demostrando que es superior al Backpropagation.

El porcentaje de detección por categoría de ataques se observa que la red RNN tiene un aceptable porcentaje del 92.75% de datos normales y la categoría de ataque de SQL es bastante baja en comparación a la red Elman.

Es precisamente por lo expuesto anteriormente se puede determinar que los algoritmos de Rede Neuronales pueden ser muy útiles en la detección de intrusos.

4.2. Recomendaciones

Descifrar los paquetes antes que sean analizados y procesados por el Sistema de Detección de Intrusos.

Utilizar hilos o sensores para monitorear varios dispositivos de red a la vez y así poder incrementar la red de los datos en su capacidad de control.

Seleccionar cuidadosamente los datos por ser la etapa más complicada al entrenar y al realizar las pruebas en las redes neuronales haciendo uso del Framework de Matlab.

Realizar una actualización periódica de la red neuronal, que permita incrementar el aprendizaje de la misma, para la clasificación de otros tipos de ataques.

El estudio de este proyecto podría considerarse para futuras estudios y comparar nuevos algoritmos de inteligencia artificial con otros Frameworks.

REFERENCIAS

A. Nieto, a. G. (2012). Sistema Colaborativo de Deteccion y Reaccion ante Intrusiones basado en Intel vPro. *RECSI 2012*.

Acosta, J. (2015). *¿Qué tan valiosa es su información.* Lima: Gestion. Recuperado el 25 de Abril de 2015, de http://gestion.pe/tecnologia/que-tan-valiosa-considera-informacion-su-empresa-2130035

Alarcon, C., Medina, F., & Villarroel, R. (2014). *Finding Usability and Communicability Problems for Transactional Web Applications.*

Alarcón, V. M. (2005). *Detecting and Classifying Attacks in Computer Networks Using Feed-Forward and Elman Neural Networks.* Springer Verlag. Wales, U.K.: Proceedings of the First European Conference on Computer Network Defense.

Alexander Gostev, A. N. (03 de Diciembre de 2015). *Securelist*. Obtenido de BOLETÍN DE SEGURIDAD DE KASPERSKY: https://securelist.lat/analysis/boletin-de-seguridad-de-kaspersky/82250/kaspersky-security-bulletin-20152016-die-top-security-stories/

AMAYA, C. G. (18 de 07 de 2014). *Troyano Neurevt ahora afecta a usuarios en Chile*. Obtenido de Welivesecurity: http://www.welivesecurity.com/la-es/2014/07/18/troyano-neurevt-afecta-usuarios-chile/

ARAAR, A., & BOUSLAMA, R. (Jun2014). A COMPARATIVE STUDY OF CLASSIFICATION MODELS FOR DETECTION IN IP NETWORKS INTRUSIONS. *Journal of Theoretical & Applied Information Technology.*, Vol. 64 Issue 1, p107-114. 8p.

Avendaño, M. A. (2014). Metodología de Medición y Evaluación de la Usabilidad en Sitios Web Gubernamentales. *Revista PGI, investigación, ciencia y tecnología*, 73-77.

AXINTE, S.-D., & BACIVAROV, I. C. (2018). *Improving the Quality of Web Applications Through Targeted Usability Enhancements.*

Barber, X. (16 de 11 de 2015). *Introduciondonos en la Inteligencia Artificial.* Obtenido de Redes Neuronales: http://umh1480.edu.umh.es/wp-content/uploads/sites/44/2013/02/teim_sesion_11_2015_16.pdf

Basogain Olabe, X. (2008). Redes Neuronales Artificiales y sus Aplicaciones. *Open Course Ware*, 79.

Beale M., D. H. (2003). *Neural Network Toolbox.* Massachusetts.: Toolbox, MathWorks, Inc.

Becerra, F. (2019). *Diseño de prototipos para el proyecto de Zappy.* Chiclayo.

Bessghaier, N., & Soui, M. (2017). Towards Usability Evaluation of Hybrid Mobile User Interfaces . *14th International Conference on Computer Systems and Applications*, 895-900.

Castrillón Velasquez, C. E., Perlaza Orduz, J. F., Van Schoonhoven, A., & Owen, E. (s.f.). LA RED NEURONAL BACKPROPAGATION COMO INTERPOLADOR. Cali, Valle del Cauca, COLOMBIA. Obtenido de http://www.bdigital.unal.edu.co/10623/14/19259573.Parte4.pdf

Chaturvedi, A. K., Kumar, P., & Sharma, K. (2020). *Proposing Innovative Intruder Detection System for Host Machines in Cloud Computing.*

CROCFER, M. A.-S.-N. (2011). *Seguridad informática - Ethical Hacking.* España: Ediciones ENI.

D.*, G., & V.** , M. (2014). ARQUITECTURA DISTRIBUIDA PARA LA RESPUESTA AUTOMÁTICA FRENTE A INTRUSIONES EN UN IRS BASADO EN ONTOLOGÍAS. *REVISTA EPN, VOL. 33, NO. 3*, 1-10.

Dias, F., & Paiva, A. (2017). Pattern-based usability testing. *IEEE Iternational Conference on Software Testing, verification and validation workshops*, 366-371. doi:10.1109/ICSTW.2017.65

Embrechts, M. (1993). *MetaNeural Hands-on.* New York.: Rensselaer Polytechnic Institute, .

Experienceux. (2019). *Sometimes questions are more important than answers*. Obtenido de https://www.experienceux.co.uk/faqs/what-is-usability-testing/

Fang, D. C. (2009). *Anomaly Program Behavior Detection Based on Neural Network.* Washington: Fourth International Conference on Innovative Computing.

Fernández, P., & Díaz, P. (2002). *Investigación cuantitativa y cualitativa.* Coruña, España.

Ferreira, A. (2013). *Diseño de un modelo de evaluación de entornos virtuales de enseñanza aprendizaje basado en la usabilidad.* La Plata, Argentina.

Fonseca, I. L. (2010). *Modelo de detección de intrusos basado en técnicas de reducción de características.* Universidad de Alicante. España.

Fox, K. H. (1990). *A Neural Network Approach Towards Intrusion Detection.* Washington,: National Computer Security Conference.

Galindo, C. J. (2009). *Diseño y Optimización de un Sistema de Detección de Intrusos Hibrido.* Almería: http://www.adminso.es/recursos/Proyectos/PFC/PFC_carlos.pdf.

Garcia, M. I. (2008). *Utilización de Sistemas de Detección de Intrusos como elemento de seguridad perimetral.* Universidad de Almeria.: {trabajo de grado.

González, Y. G. (2011). *Modelos y Algoritmos para redes neuronales recurrentes basadas en wavelets aplicados a la detección de intrusos.* Cholula, Puebla, México.

Hakami, B. J. (2013). *A Distributed Intrusion Detection System Using Cooperative Agents.* Qazvin, Iran: Life Science Journal 2013;10(8s) .

Hall, J. (20 de Setiembre de 2017). *Usability testing for early-stage software prototypes*. Obtenido de https://opensource.com/article/17/9/paper-based-usability-testing

Heady, R. L. (1990). *The architecture of a network level intrusion detection system.* Technical Report, Albuquerque.

Hotjar. (31 de Octubre de 2019). *A beginner's guide to usability testing*. Obtenido de https://www.hotjar.com/usability-testing

ISO 9241-2010. (2010). International standar ISO 9241-2010.

José, H. G. (1995). *Redes Nueronales Artificiales. Modelos Fundamentos y Aplicaciones.* España: Ra-ma.

Kalnoor, G., & Agarkhed, J. (2016). *Preventing attacks and detecting intruder for secured Wireless Sensor Networks.*

Kaskaloglu, K., & Herbert, S. (2011). Web Usability Guidelines For Smartphones: A Synergic Approach. *International Journal of Information and Electronics Engineering,* , 33-37.

Kaur, R., & Sharma, B. (2018). *Comparative Study for Evaluating the Usability of Web Based Applications.*

KAVITA, P., & M.USHA. (3/31/2014). ANOMALY BASED INTRUSION DETECTION IN WLAN USING DISCRIMINATION ALGORITHM COMBINED WITH NAÏVE BAYESIAN CLASSIFIER. *Journal of Theoretical & Applied Information Technology.*, p646-653.

Lago, A. F. (30 de 05 de 2015). *Estado Actual de los IDS.* Obtenido de Cesga: archivo.cesga.es/component/option,com_docman/task,doc.../lang,es

Landa, J., Jun, C., & Jun, M. (2017). *Implementation of a Remote Real-Time Surveillance Security System for Intruder Detection.*

Lau, C. (1992). *Artificial neural networks: paradigms, applications, and hardware implementations.* New Jersey: IEEE Press.

Liang, H. (May2014). An Improved Intrusion Detection based on Neural Network and Fuzzy Algorithm. *Journal of Networks.*, p1274-1280.

Long, Y., & Sun, J. O. (2013). *Network Intrusion Detection Model based on Fuzzy Support Vector Machine.* JOURNAL OF NETWORKS, VOL. 8, NO. 6, .

López, P. A. (2010). *Seguridad informática.* Madrid: Editex.

Mallikarjun, B., Kiranmayi, K., Lavanya, N., Prateeksha, K., & Sushmitha, J. (2020). *Intruder Detection System - A LoRa Based Approach.*

Medina, R., & Morales, R. (2015). Usability Evaluation by Experts of a Learning Management System. *Revista Iberoamericana de tecnologías del aprendizaje, Vol. 10, No. 4*, 197-203.

Medium. (30 de Abril de 2019). *Pruebas con Usuarios #1 — ¿Qué, cuándo y para qué testeamos?* Obtenido de https://medium.com/@eugeniacasabona/pruebas-con-usuarios-1-qu%C3%A9-cu%C3%A1ndo-y-para-qu%C3%A9-testeamos-7c3a89b4b5e7

Mejía Sánchez, J. A. (2004). *Sistema de detección de intrusos en redes de comunicaciones utilizando redes neuronales.* México.

Mira Alfaro, E. J. (13 de Enero de 2002). *Introducción a los IDS*. Obtenido de Sistemas de Detección de Intrusos: http://mural.uv.es/emial/informatica/html/IDS.html

Murillo, J. (s.f). *Métodos de investigación de enfoque experimental.*

Nagpal, S. K., & Manojkumar, P. (2016). *Hardware implementation of intruder recognition in a farm through Wireless Sensor Network.*

Oropeza Clavel, C. A. (2007). *Modelado y Simulación de un Sistema de Detección de Intrusos Utilizando Redes Neuronales Recurrentes.* Universidad de las Américas Puebla, México.

Ortega, U. Z. (2004). *Estado del Arte SISTEMAS DE DETECCIÓN DE INTRUSOS.* Mondragón.: Escuela Politécnica Superior Mondragón.

otros, P. R. (2005). Aplicación de Redes Neuronales Artificiales para la detección de intrusos en redes y sistemas de información. *Scientia Et Technica*, 230.pdf.

OWASP. (30 de Julio de 2015). *Inyección de Código: OWASP*. Obtenido de https://www.owasp.org/index.php/Inyeccion_de_Codigo

PADILLA, G. G. (2015). *Sistema de Detección de Intrusos –INPEC.* Tunja. Colombia: Trabajo de Graduación: Especialización Seguridad Informática.

Paz, F., & Pow, J. A. (2016). A systematic mapping review of usability evaluation methods for software development process. *International Journal of Software Engineering and Its Applications*, 165-178.

QuestionPro. (s.f). *Usabilidad web, un test que te dará grandes resultados*. Obtenido de https://www.questionpro.com/blog/es/usabilidad-web-test/

Quwaider, M. (2017). *Real-time intruder surveillance using low-cost remote wireless sensors.*

Ramos, A. G.-C. (2011). *Seguridad Informática .* Madrid, España: Ediciones Parininfo S.A.

Rivero Peréz, J. L. (2014). *Técnicas de aprendizaje automático para la detección de intrusos.* Universidad de Cienfuegos. Cuba: Revista Cubana de Ciencias Informáticas.

Rivero, L., & Conte, T. (2014). *Improving the Quality of Web Applications Through Targeted Usability Enhancements.*

Rodríguez, C. M. (2015). *Sistema de Detección de Intrusos de alerta temprana en la red UCLV.* Santa Clara, cuba: Tesis presentada en opción al Título Académico de Máster en Telemática .

Rodriguez, J. I., Sierra, E., & Jaramillo, L. K. (2015). Model for measuring Usability of Survey Mobile Apps,by analysis of Usability evaluation methods and attributes. *10th Iberian Conference on Information Systems and Technologies (CISTI)*.

S, H. (1994). *Neural Networks.* Ontario, Canada: McMaster University .

Salman, H., Sulaiman, S., & Wan, F. (2018). Usability Evaluation of the Smartphone User Interface in Supporting Elderly Users from Experts' Perspective.

Sarwar, T., Habib, W., & Arif, F. (2013). Requirements Based Testing of Software. 347-352.

Sauro, J., & Lewis, J. R. (2016). *Quantifying the User Experience (Second Edition).* doi:https://doi.org/10.1016/B978-0-12-802308-2.00008-4

Shahid, A., Tayyab, A., Mehmood, M., Anum, R., Jalil, A., Ali, A., . . . Ahmed, J. (2017). *Computer vision based intruder detection framework (CV-IDF).*

Sobri, A., & Xing, V. J. (2018). Usability of ShopCart among Customers at Shopping Malls . *IEEE Conference on e-Learning, e-Management and e-Services (IC3e)*, 140-144.

Technologies, A. (2015). *Informe sobre el estado de Internet en materia de seguridad del cuarto trimestre de 2015.* EE.UU: Akamai Faster Forward.

Tejada, E. C. (2015). *Gestión de Incidentes de Seguridad Informática .* Antequera España: IC EDITORIAL.

Telecomunicaciones, U. I. (05 de 05 de 2014). *La UIT publica las cifras de TIC de 2014*. Obtenido de http://www.itu.int/net/pressoffice/press_releases/2014/23-es.aspx#.VYii7xt_Oko

Tian, Z., Li, Y., Zhou, M., & Li, Z. (2018). *WiFi-Based Adaptive Indoor Passive Intrusion Detection.*

tools, P. b. (19 de 02 de 2013). *http://pumawifi.org/index.php?q=node/145*. Recuperado el 10 de 06 de 2015, de IDS: Evolución y época actual: http://pumawifi.org/index.php?q=node/145

Torres, E. (2003). *Sistema Inmunológico para la Detección de Intrusos a Nivel de Protocolo HTTP.* Bogotá: Proyecto de grado. Pontificia Universidad Javeriana.

Tullis, T., & Albert, B. (2013). *Measuring the User Experience (Second Edition).* doi:https://doi.org/10.1016/B978-0-12-415781-1.00006-6

Usability. (s.f). *Prototype*. Obtenido de https://www.usability.gov/how-to-and-tools/methods/prototyping.html

Usabilitybok. (s.f). Obtenido de http://www.usabilitybok.org/

Uxpanol. (25 de Febrero de 2017). *Sistema de Escalas de Usabilidad: ¿qué es y para qué sirve?* Obtenido de https://uxpanol.com/teoria/sistema-de-escalas-de-usabilidad-que-es-y-para-que-sirve/

Vallejo Pérez, D. &. (2012). *Minería de datos aplicada en detección de intrusos.* Medellin: Ubicación en Biblioteca USB Medellín (San Benito): CD-2031t.

Vargas, S., & Pow-Sang, J. A. (2018). Mapeo sistemático de la literatura sobre técnicas de evaluación de usabilidad en aplicaciones educativas en dispositivos móviles. *7th International Conference on Software Process Improvement (CIMPS)*, 59-68.

Vásquez López, J. P. (2014). RED NEURONAL FEEDFORWARD COMO ESTIMADOR DE PATRONES DE CORRIENTES EN EL INTERIOR DEL PUERTO DE MANZANILLO SUJETO A LA ACCIÓN DE TSUNAMIS. *Instituto Mexicano del Transporte*, 50. Obtenido de Instituto Mexicano del Transporte.

Veldsman, A., & Van Greunen, D. (2017). Comparative usability evaluation of a mobile health app. *International information management corporative*, 1-8.

Vidal, J. M. (2012). *Sistema de Detección de Anomalías de red basado en el procesamiento de Payload.* Universidad Complutense de Madrid.: Trabajo de grado .

Vineeth, Radhika, & Vanitha. (2015). *Intruder Detection and Prevention in a Smart Grid Communication System.*

Wang Jin-Song, Z. L.-h. (2014). *A Small-time Scale Netflow-based Anomaly Traffic Detecting Method Using MapReduce.* Tianjin 300384: International Journal of Security and Its Applications.

Wang, L. (JUNE 2013). An Attribute-weighted Clustering Intrusion Detection Method . *JOURNAL OF NETWORKS, VOL. 8, NO. 6,* .

Webdesing. (10 de Julio de 2017). *3 Métricas para Medir y Cuantificar la Usabilidad*. Obtenido de https://webdesign.tutsplus.com/es/tutorials/3-metrics-for-quantifying-usability--cms-29150

Weichbroth, P. (2018). Usability attributes revisited: a time-framed knowledge map . *Proceedings of the Federated Conference on Computer Science and Information Systems*, 1005-1008.

Welie, M., Veer, G., & Eliëns, A. (2011). Breaking down Usability .

Wich, M., & Kramer, T. (2015). Enhanced Human-Computer Interaction for Business Applications on Mobile Devices: A Design-Oriented Development of a Usability Evaluation Questionnaire. *48th Hawaii International Conference on System Sciences*, 472-481.

Willams, J. Z. (1990). *Gradient-Based Learning Algorithm for Recurrent Connectionist Networks.* California: La Jolla, CA Press.

Xenya, M. C., Kwayie, C., & Quist-Aphesti, K. (2019). *Intruder Detection with Alert Using Cloud Based Convolutional Neural Network and Raspberry Pi.*

Yu, Q., Luo, Z., & Min, P. (2016). *Intrusion detection in wireless sensor networks for destructive intruders.*

Zhai Shuang-can, H. C.-j.-m. (2014). *Multi-Agent Distributed Intrusion Detection System Model Based.* International Journal of Security and Its Applications.

Zhang, Z., Cao, Y., Ding, M., Zhuang, L., & Yao, W. (2016). *An intruder detection algorithm for vision based sense and avoid system.*

ANEXO

ANEXO 1: Resolución de Aprobación

FACULTAD DE INGENIERÍA, ARQUITECTURA Y URBANISMO

RESOLUCIÓN Nº 0878-A-2018/FIAU-USS

Chiclayo, 01 de octubre de 2018

VISTO:

El Acta de Reunión Nº de fecha 01 de octubre de 2018,, para la ejecución de la Tesis titulada: *"COMPARACIÓN DE ALGORITMOS DE REDES NEURONALES PARA MEJORAR LA DETECCIÓN DE INTRUSOS EN REDES DE ÁREA LOCAL"*, presentada por el(los) estudiante(s) **GUEVARA PALOMINO NILTON** de la Escuela Académico Profesional de **INGENIERÍA DE SISTEMAS** y;

CONSIDERANDO:

Que, de conformidad con la Ley Universitaria Nº 30220 en su artículo 48º que a letra dice: *"La investigación constituye una función esencial y obligatoria de la universidad, que la fomenta y realiza, respondiendo a través de la producción de conocimiento y desarrollo de tecnologías a las necesidades de la sociedad, con especial énfasis en la realidad nacional. Los docentes, estudiantes y graduados participan en la actividad investigadora en su propia institución o en redes de investigación nacional o internacional, creadas por las instituciones universitarias públicas o privadas."*;

Estando a lo expuesto, y en uso de las atribuciones conferidas y de conformidad con las normas y reglamentos vigentes;

SE RESUELVE:

ARTÍCULO 1º: **APROBAR**, el Proyecto de Tesis denominado *"COMPARACIÓN DE ALGORITMOS DE REDES NEURONALES PARA MEJORAR LA DETECCIÓN DE INTRUSOS EN REDES DE ÁREA LOCAL"*, perteneciente a la Línea de Investigación **CIENCIAS DE LA COMPUTACIÓN - SISTEMAS INTELIGENTES**, a cargo del(los) estudiante(s) **GUEVARA PALOMINO NILTON**, de la Escuela Académico Profesional de **INGENIERÍA DE SISTEMAS**.

ARTÍCULO 2º: **ESTABLECER**, que la inscripción de la Tesis se realice a partir de emitida la presente resolución, y tendrá una vigencia máxima de 02 años.

REGÍSTRESE, COMUNÍQUESE Y ARCHÍVESE

Cc: Dirección de Investigación, CPGTT, Interesados, Archivo

ANEXO 2 Código Fuente.

En esta sección se detalla el proceso de creación del IDS basado en los tres tipos de arquitectura utilizados: feedforward, Elman y finalmente RNN. También se explica la función de todas las rutinas y subrutinas involucradas, las cuales fueron programadas utilizando MATLAB®.

TABLA 12

Código fuente

```
1   #ids neural network project
2   #author: Nilton Guevara Palomino
3   #date: Nov 23 2016
4   setwd("C:/neuralnet") #set working directory
5   #load data
6   anomaly<-read.csv('anomaly_shuffle.csv')
7   #change factor to character
8   anomaly$Class<-as.character(anomaly$Class)
9   attacks<-data.frame(unique(anomaly$Class))
10  colnames(attacks)<-c('Class')
11  #data preperation
12  int_class<-c('INJECTION','PATH','SQL','XSS')
13  r<-list()
14  for (i in 1:4) {
15    chk<-anomaly
```

```
16        id<-which(chk$Class!=int_class[i])
17      chk$Class[id]<-paste0('Not',int_class[i])
18      filename<-paste0(int_class[i],'.csv')
19      write.csv(chk,filename)
20      input<-as.matrix(chk[,-42])
21      output<-as.character(chk$Class)
22      f1<-paste0('input_',int_class[i],'.txt')
23      f2<-paste0('output_',int_class[i],'.txt')
24      write.table(input, f1, sep="\t",row.names = FALSE,col.names
25    = FALSE)
        write.table(output, f2, sep="\t",row.names =
      FALSE,col.names = FALSE) }
26
      chk<-anomaly
27
      #replace<-paste0('Not',int[i])
28
      chk$Class[!chk$Class %in% int_class]<-c('Normal')
29
      write.csv(chk,'Special.csv')
30
      input<-as.matrix(chk[,-42])
31
      output<-as.character(chk$Class)
32
      write.table(input, "input_Special.txt", sep="\t",row.names =
33    FALSE,col.names = FALSE)
```

```
write.table(output, "output_Special.txt", sep="\t",row.names =
FALSE,col.names = FALSE)
```

Nota: Código fuente. Fuente: Elaboración Propia