Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

ANÁLISIS SOBRE LA IMPLEMENTACIÓN DE REDES NEURONALES APLICADAS A SISTEMAS DE DETECCIÓN DE INTRUSOS

Juan Carlos Maeda Juárez
Asesorado por el Ing. David Estuardo Morales Ajcot

Guatemala, abril de 2021

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

ANÁLISIS SOBRE LA IMPLEMENTACIÓN DE REDES NEURONALES APLICADAS A SISTEMAS DE DETECCIÓN DE INTRUSOS

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

JUAN CARLOS MAEDA JUÁREZ
ASESORADO POR EL ING. DAVID ESTUARDO MORALES AJCOT

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, ABRIL DE 2021

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Christian Moisés de la Cruz Leal
VOCAL V	Br. Kevin Armando Cruz Lorente
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANA	Inga. Aurelia Anabela Cordova Estrada
EXAMINADOR	Ing. César Augusto Fernández Cáceres
EXAMINADOR	Ing. Herman Igor Véliz Linares
EXAMINADOR	Ing. César Rolando Batz Saquimux
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

ANÁLISIS SOBRE LA IMPLEMENTACIÓN DE REDES NEURONALES APLICADAS A SISTEMAS DE DETECCIÓN DE INTRUSOS

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha 21 de agosto de 2019.

Juan Carlos Maeda Juárez

Guatemala 31 de agosto 2020.

Ingeniero
Carlos Alfredo Azurdia Morales
Coordinador del Área de Trabajos de Graduación

Respetable Ingeniero Azurdia:

Por este medio informo que he revisado y aprobado el trabajo de investigación titulado: **" ANÁLISIS SOBRE LA IMPLEMENTACIÓN DE REDES NEURONALES APLICADAS A SISTEMAS DE DETECCIÓN DE INTRUSOS",** desarrollado por el estudiante **Juan Carlos Maeda Juárez**, quien se identifica con el número de carné **199712370**, ya que considero que cumple con los requisitos establecidos, por lo que el autor y mi persona somos responsables del contenido y conclusiones del mismo.

Agradeciendo su atención a la presente.

Atentamente.

Ing. David Estuardo Morales Ajcot
Asesor
No. Colegiado 10993

Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 3 de septiembre de 2020

Ingeniero
Carlos Gustavo Alonzo
Director de la Escuela de Ingeniería
En Ciencias y Sistemas

Respetable Ingeniero Alonzo:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación del estudiante **JUAN CARLOS MAEDA JUÁREZ** con carné **199712370 y** CUI **1610 41701 0101** titulado **"ANÁLISIS SOBRE LA IMPLEMENTACIÓN DE REDES NEURONALES APLICADAS A SISTEMAS DE DETECCIÓN DE INTRUSOS"** y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo aprobado.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,

Ing. Carlos Alfredo Azurdia
Coordinador de Privados
y Revisión de Trabajos de Graduación

ESCUELA DE INGENIERÍA EN CIENCIAS Y SISTEMAS

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERÌA
ESCUELA DE INGENIERÍA EN
CIENCIAS Y SISTEMAS

El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del asesor con el visto bueno del revisor y del Licenciado en Letras, del trabajo de graduación ***"ANÁLISIS SOBRE LA IMPLEMENTACIÓN DE REDES NEURONALES APLICADAS A SISTEMAS DE DETECCIÓN DE INTRUSOS",*** *realizado por el estudiante, JUAN CARLOS MAEDA JUÁREZ aprueba el presente trabajo y solicita la autorización del mismo.*

"ID Y ENSEÑAD A TODOS"

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION DE INGENIERIA EN CIENCIAS Y SISTEMAS

Msc. Carlos Gustavo Alonzo
Director
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 24 de marzo de 2021

Decanato
Facultad de Ingeniería
24189101- 24189102

DTG. 130.2021.

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al Trabajo de Graduación titulado: **ANÁLISIS SOBRE LA IMPLEMENTACIÓN DE REDES NEURONALES APLICADAS A SISTEMAS DE DETECCIÓN DE INTRUSOS,** presentado por el estudiante universitario: **Juan Carlos Maeda Juárez,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANA
FACULTAD DE INGENIERÍA

Inga. Anabela Cordova Estrada
Decana

Guatemala, abril de 2021.

AACE/asga

ACTO QUE DEDICO A:

Dios	Por ser mi fortaleza, mi soporte, por llenarme de fe y esperanza en momentos difíciles.
Mis padres	Por ser mi hogar durante mi carrera.
Mi esposa	Por ser una importante influencia en mi carrera.
Mis hijos	Pablo Andrés y María Vianey Maeda, su amor ha sido el motor de mi vida, por ser mis razones para alcanzar las metas.
Mis hermanos	Por creer en mí y sus porras que nunca faltaron. Por ser parte importante en mi vida.
Mis amigos	Luis Salazar, Roberto Zahabedra, Julio Rodríguez, Luis Díaz, Hugo Huard, Alfredo Ortiz y todo aquellos que estuvieron a mi lado y ser parte importante en mi carrera.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por ser una importante influencia en mi carrera, entre otras cosas.
Facultad de Ingeniería	Por ser mi hogar durante esta importante etapa de mi vida.
Mis amigos de la Facultad	Ing. Miguel Marín, Ing. Luis Espino, Ing. Carlos Alonzo, Alan Bautista, Haroldo Arias, Nelson González.
Mi asesor	David Morales, por su apoyo y conocimiento brindados en la elaboración de mi trabajo de tesis.
Tutores y profesor	Por el conocimiento brindado y por tomarse el tiempo para impulsar mi desarrollo profesional.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

LISTA DE SÍMBOLOS

Símbolo	Significado
z-1	Delay
hi(t) = $\sum$(xjwi)^2	Distancia euclidiana
ek(n)	Error de salida
ai(t)=f(ai(t-1)	Función de activación
Fi(t)	Función de salida
wij	Pesos sinápticos
d(wij,xj(t))	Regla de propagación
yi(t)	Salida
yk(n)	Salida obtenida
Smn(t+1)	Salidas parciales
dk(n)	Valor de destino esperado

GLOSARIO

Cross-Site Scripting	Secuencia de comandos entre sitios.
Delay	Tiempo de espera.
Denial of Service	Negación de servicio.
Epoch	Época, ciclo.
Frame	Paquete de datos.
Hoaxes	Engaños.
Host	Anfitrión.
IA	Inteligencia artificial.
Junk Mail	Correo basura.
Logs	Registros.
Malicious Software	Software malicioso.
Phishing	Suplantación de identidad.
Script	Conjunto de Instrucciones, código útil.

SDI	Sistemas de detección de intrusos.
spyware	Software espía.

RESUMEN

La presente tesis realiza el análisis y la evaluación sobre las técnicas utilizadas por la seguridad informática para proteger el activo más preciado de las empresas e instituciones gubernamentales: la información; así mismo, comparar las técnicas de inteligencia artificial que pueden apoyar a fortalecer los sistemas de detección de intrusos. Por otro lado, es necesario establecer el nivel de conocimiento que tienen los egresados y estudiantes de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, sobre la seguridad informática y su forma de aplicar en las empresas.

En el primer capítulo se describe cómo se aplican las reglas de la seguridad informática en las empresas, se establecen los objetivos general y específicos de la presente tesis; también, se analizan las técnicas utilizadas por la seguridad informática para defenderse de los ataques y las diferentes formas que utilizan los intrusos para romper los protocolos de seguridad establecidos.

En el capítulo dos se hace una comparación entre las diferentes ramas de la inteligencia artificial que permita tomar una decisión sobre la que puede ser más efectiva en la seguridad informática.

En los capítulos tres y cuatro se presenta un análisis comparativo entre los diferentes tipos de redes neuronales como la opción optima por sus características para apoyar el funcionamiento de los sistemas de detección de intrusos (SDI) para elegir la que mejor se adapte.

En el capítulo cinco se redactan las conclusiones y los resultados de la encuesta, se responden los objetivos y se realizan las recomendaciones para aumentar la eficiencia de los SDI en la seguridad informática.

OBJETIVOS

General

Identificar las técnicas de inteligencia artificial que pueden ser aplicadas para la detección de intrusos en sistemas de información y cuáles son más eficientes en costos, beneficios, ventajas y desventajas por medio de un análisis.

Específicos

1. Comparar y analizar los factores que influyen en la implementación de las diferentes técnicas inteligencia artificial en los métodos de detección de intrusos: costos, beneficios, ventajas y desventajas.

2. Definir las ventajas y desventajas de implementar técnicas de inteligencia artificial como apoyo a la detección de intrusos.

3. Definir las tendencias de la implementación de la inteligencia artificial como apoyo a la seguridad informática.

4. Análisis del conocimiento del egresado de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería en temas de fortalecimiento de la seguridad informática con la implementación de técnicas de inteligencia artificial.

INTRODUCCIÓN

Normalmente las áreas tecnológicas y principalmente la seguridad informática no se consideran algo productivo para la empresa; son áreas que a la vista de los altos mandos e inversionistas implican gastos y no generan ningún valor. Esta es una de las principales causas por la que la inversión en seguridad de la información no es la adecuada, las organizaciones no han caído en cuenta de que su mayor activo es la información y velar por la seguridad, protección y custodia de ese activo debe ser realmente un tema prioritario.

La inteligencia artificial ha sido fundamental para lograr grandes avances en el mundo de los negocios, una de sus principales aportaciones ha sido en el área de *big data*, para identificar patrones, para analizar grandes volúmenes de información. La seguridad informática ha aprovechado estas bondades para implementar algoritmos de IA en sistemas de identificación de intrusos, buscando patrones en las acciones de los usuarios, identificación anomalías, todo este esfuerzo con el fin de ser un área preventiva, más que reactiva.

Las redes neuronales, sistemas basados en casos y la regresión multivariada son las técnicas más utilizadas en los sistemas de identificación de intrusos (SDI), en el presente trabajo se busca realizar un análisis comparativo de estas técnicas y responder las siguientes preguntas: ¿cuál es la técnica más eficiente y factible de implementar?, ¿existen nuevas técnicas que pueden ser utilizadas para mejorar la seguridad?

1. MARCO TEÓRICO

1.1. Seguridad informática

Se define como el proceso de prevenir y detectar el uso no autorizado de un sistema informático, incluye el proceso de proteger contra intrusos el uso de recursos informáticos, acceso a la información con intenciones maliciosas o con intención de obtener ganancias, incluso el acceso por accidente, la seguridad informática abarca mucho más que la seguridad de la información, aunque se menciona indistintamente ambos términos.

La seguridad informática es una disciplina que cubre una serie de medidas de seguridad, desde programas de software, *firewall*, hasta procesos bien definidos, formularios para registrar y auditar correctamente los accesos. Se puede definir la seguridad informática como el cumplimiento de confidencialidad, integridad y disponibilidad en un sistema informático.

La seguridad informática juega un papel muy importante en las organizaciones por ejemplo para prevenir el robo de datos tales como números de cuentas bancarias, información de tarjetas de crédito, contraseñas, seguro social, teléfonos, direcciones, entre otros. Es de apoyo fundamental para cumplir sus objetivos protegiendo sus recursos financieros, sus sistemas, su reputación.

1.1.1. Objetivos de la seguridad informática

La seguridad informática cubre 4 objetivos principales (figura 1):

1.1.1.1. Confidencialidad

Solo los usuarios autorizados pueden acceder a los recursos, los datos e información que necesitan. La confidencialidad requiere que la información sea accesible únicamente para las personas que estén autorizados.

1.1.1.2. Integridad

Solo los usuarios autorizados deben hacer las modificaciones de los datos que sean necesarios. Garantiza que la información se mantenga inalterada ante accidentes o intento maliciosos.

1.1.1.3. Disponibilidad

Los datos deben estar disponibles cuando los usuarios los requieran. Los sistemas informáticos se deben mantener trabajando sin sufrir ninguna degradación en cuanto a accesos y provea los recursos que requieran los usuarios autorizados cuando estos los necesiten.

1.1.1.4. Autenticación

Las personas que se están comunicando realmente son las personas con las que se deben comunicar.

Figura 1. **Objetivos de la seguridad informática**

Fuente: SEGU. *Objetivo de la seguridad informática*. https://www.uv.mx/personal/llopez/files/2011/09/presentacion.pdf. Consulta: 29 de agosto de 2019.

1.2. Aplicación en las empresas

Las empresas deben definir procesos o estándares para poder aplicar las técnicas de la seguridad informática eficientemente y obtener, por lo tanto, el mayor provecho.

1.2.1. Políticas de seguridad informática

Las políticas de seguridad informática surgen como herramientas organizacionales para aplicar los objetivos de la seguridad informática en las empresas y concientizar a los colaboradores sobre la importancia y sensibilidad de la información, así como los servicios críticos que permiten a la empresa crecer y mantenerse competitiva. Las políticas son una descripción técnica de mecanismo, no es una expresión legal que involucre sanciones o conductas de los empleados, es más bien una descripción de los que se desea proteger, el

por qué y el cómo se debe hacer. Cada política es una invitación a reconocer la información como uno de los principales activos.

Las políticas de seguridad informática deben considerar principalmente los siguientes elementos:

- Alcance de las políticas, que incluye facilidades, sistemas y personal a quienes va dirigido.

- Objetivos de la política y descripción clara de los elementos involucrados en su definición.

- Responsabilidades por cada uno de los servicios y recursos informáticos.

- Requerimientos mínimos para la configuración de la seguridad de los sistemas.

- Definición de violaciones y sanciones por no cumplir con las políticas.

- Responsabilidades de los usuarios respecto a sus accesos y la información que manejan.

- Deben seguir un proceso de actualización periódica.

1.3. Cómo atacan

Diariamente se realizan millones de intentos de ataques cibernéticos, muchos de ellos logran su objetivo, ya sea a personas individuales o empresas,

se aprovechan de vulnerabilidades en el software e incluso de la ingenuidad de las personas.

1.3.1. Ciberataque

En los últimos años la reputación de las organizaciones es un tema fundamentalmente importante, se invierte mucho dinero anualmente en la preparación de los colaboradores para cuidar la reputación, para volver a las empresas un lugar agradable para trabajar; pero ¿cuánto se invierte en la seguridad informática, en la prevención de ataques?

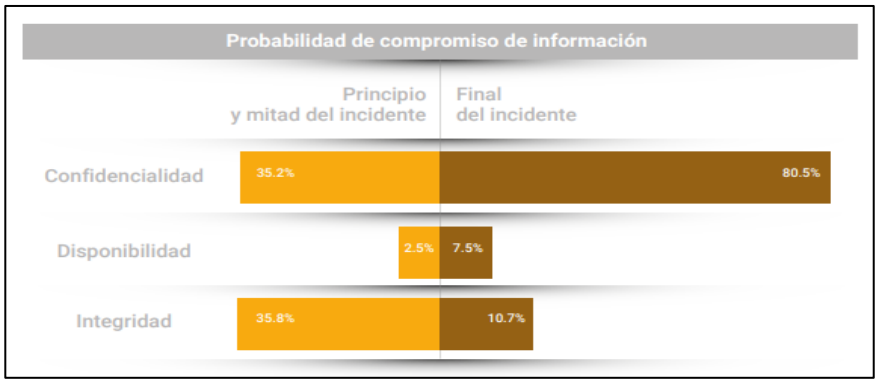
Un ciberataque es la explotación deliberada de sistemas informáticos, empresas y redes dependientes de la tecnología. Estos ataques utilizan códigos maliciosos para alterar la lógica o los datos del ordenador, lo que genera consecuencias perjudiciales que pueden comprometer información y provocar delitos cibernéticos, como el robo de identidad.

Un ciberataque puede implicar un equipo de *hackers* de elite que trabajan bajo el mando de un estado nación o bien un *hacker* individual. Su intención es aprovechar fallas desconocidas en el software para insertar un programa que les permita filtrar datos confidenciales, dañar infraestructura clave o desarrollar una base para futuros ataques.

Los grupos de piratería más peligrosos se conocen como ‘amenazas persistentes avanzadas’. Algo muy importante que recalcar de los ciberataques es que se dan a objetivos que están menos defendidos y no que en realidad sean tan valiosos.

En los últimos años, los ataques contra los datos personales se han incrementado, y no solo por parte de los ciberdelincuentes o grupos hacktivistas, sino también por los Estados. El principal objetivo es el robo de identidad (credenciales), la suplantación o el espionaje, como lo muestra la figura 2.

Figura 2. **Probabilidad de compromiso de información**

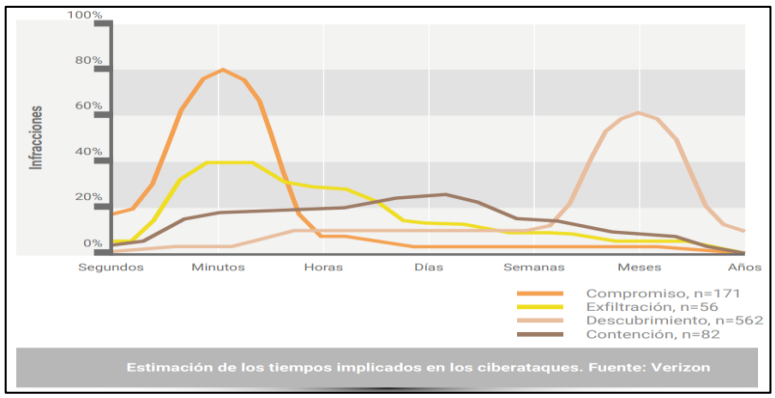


Fuente: Centro Criptológico Nacional, Gobierno de España. *CCN-CERT_IA1319 Informe Amenazas y Tendencias Resumen Ejecutivo 2019*. https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/3767-ccn-cert-ia-13-19-ciberamenazas-y-tendencias-resumen-ejecutivo-2019.html. Consulta: 29 de agosto de 2019.

1.3.1.1. Tiempos implicados en el ciberataques

El tiempo medio entre la primera acción *host*il hasta el compromiso de un activo se mide en términos de segundos o minutos, sin embargo, el plazo para su descubrimiento o detección, que depende mucho del tipo de ataque, suele expresarse en días, semanas o meses (figura 3).

Figura 3. **Estimación de los tiempos implicados en los ciberataques**



Fuente: Centro Criptológico Nacional, Gobierno de España. *CCN-CERT_IA1319 Informe Amenazas y Tendencias Resumen Ejecutivo 2019*. https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/3767-ccn-cert-ia-13-19-ciberamenazas-y-tendencias-resumen-ejecutivo-2019.html. Consulta: 29 de agosto de 2019.

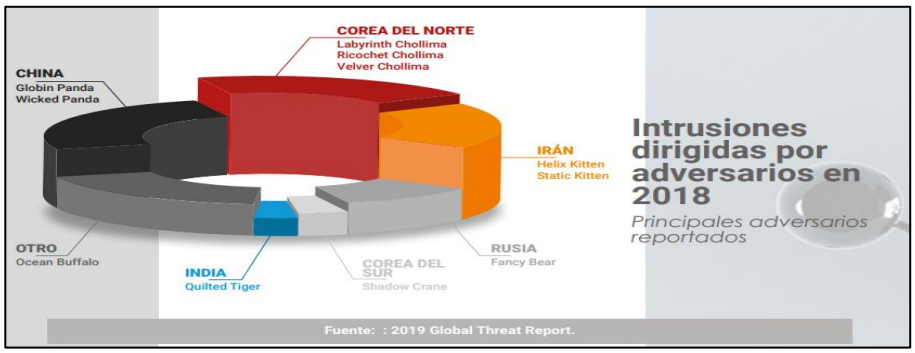
1.3.1.2. Actores de las amenazas

Un actor de amenazas es un atacante individual o estatal, delincuentes con un objetivo claro o personas que actúan por negligencia o errores, puede ser un empleado disgustado, con pocas habilidades o exceso de trabajo.

Más del 60 % del tráfico mundial de correo electrónico en 2018 contenía carga dañina y estuvo involucrado en más del 90 % de los ciberataques.

El número de actores ha aumentado considerablemente debido al fácil acceso a nuevas herramientas de ataque y a lo difícil que es identificar a los actores de los ataques. Se ha evidenciado un incremento en el uso de código dañino por parte de los Estados, dirigidos a aprovechar las vulnerabilidades de los sistemas de información de las infraestructuras críticas, sobre la base de los datos presentados por el Centro Criptológico del Gobierno de España (figura 4).

Figura 4. **Global Threat Report**



Fuente: Centro Criptológico Nacional, Gobierno de España. *CCN-CERT_IA1319 Informe Amenazas y Tendencias Resumen Ejecutivo 2019*. https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/3767-ccn-cert-ia-13-19-ciberamenazas-y-tendencias-resumen-ejecutivo-2019.html. Consulta: 29 de agosto de 2019.

1.3.2. Tipos de ciberataques

En realidad cuando un pirata o delincuente cibernético intenta piratear una organización, no crea una nueva forma para hacerlo, normalmente recurren a un arsenal común de ataques conocidos por ser altamente eficaces. Estos tipos de ataques pueden ser:

1.3.2.1. Ingeniera social

Una de las principales preocupaciones de los profesionales de la seguridad informática son los problemas ocasionados por PEBCAK, acrónimo para describir ‘problema existe entre la silla y el teclado’. Es una forma de decir averías de seguridad causadas por los usuarios. Muchas personas son susceptibles a ataques de ingeniería y correos electrónicos de engaño, tienen malos hábitos como el uso de contraseñas débiles, y se convierten en blancos

ideales para los ataques cibernéticos. Los ciberdelincuentes utilizan ciertos principios básicos para manipular a sus víctimas, por ejemplo:

- Respeto a la autoridad: por lo general, las personas como trabajadores o ciudadanos en general, respetamos la autoridad de nuestros superiores, dentro de la organización o en la vida cotidiana. Este tipo de ataque se basa en ese respeto que tenemos a nuestros superiores o cuerpos de seguridad del Estado.

- Voluntad de ayudar: sobre todo, en los entornos laborales, los trabajadores cuentan con la voluntad de ayudar a los compañeros en todo lo posible. Es aprovechándose de esta buena voluntad que los ciberdelincuentes se pueden hacer pasar por un falso colaborador. Otra forma es hacerse pasar por un técnico para instalar herramientas de acceso remoto no autorizado.

- Respeto social: en algunos, el ataque va dirigido al miedo que tiene los usuarios a no ser socialmente aceptado o a perder su reputación. Esto es común en los correos de sextorsión, donde los delincuentes amenazan con difundir un supuesto video privado que en realidad no existe.

Luego de conocer los principios que los ciberdelincuentes utilizan para un ataque, se enumeran algunas de las técnicas que implementan para violar la seguridad.

1.3.2.2. Phishing

Conocido como suplantación de identidad, es un término informático que denomina un modelo de abuso informático y que se comete mediante el uso de

un tipo de ingeniería social, caracterizado por intentar adquirir información confidencial de forma fraudulenta.

1.3.2.3. Correo no deseado

Es el correo que no es esperado y por qué no decirlo no querido, por el usuario que lo recibe. También, puede llamarse correo basura (Junk Mail).

Los correos no deseados se pueden definir también como *spam*, el *spam* se caracteriza por ser publicitario, existe un fin de venta detrás de ellos, mientras que el correo no deseado puede incluir, bromas, suscripciones, virus, también existen grymail (correo gris) que no es *spam* ni tampoco correo no deseado. Los objetivos que persiguen los correos no deseados:

- El fraude económico. Engañar a la gente y obtener a consecuencia del engaño dinero.

- La diseminación de *malware* a gran escala. Generalmente, con fines económicos. El caso más relevante es el robo de identidad bancaria a través de la diseminación de troyanos de captura de credenciales.

- Publicación fraudulenta, *spam* puro y duro, aquel que persigue efectos comerciales, normalmente ventas de forma ilícita.

- La saturación y denegación de servicios de un proveedor determinado.

- Recopilación de direcciones de correo, para confeccionar listas de recipientes, que serán empleadas posteriormente para cualquier de los propósitos anteriores.

1.3.2.4. *Hunting*

Este tipo de ataques buscan afectar al mayor número de usuarios realizando, únicamente una comunicación. Son comunes en campañas de phishing, como los realizados contra entidades energéticas o bancarias, por ejemplo: un correo en donde indique que se ha recibido un reembolso de un banco. También, son utilizados en ataques cuyo objetivo es realizar una campaña de infección por *malware*, enviando por ejemplo un presupuesto falso en Excel, u oleadas de correos con facturas.

1.3.2.5. Software malicioso

En inglés *malicious software*, programa malicioso o programa maligno, hace referencia a cualquier tipo de software maligno que trata de afectar a un ordenador, a un teléfono celular. El término *malware* es muy utilizado por profesionales de IT para referirse a una variedad de software *host*il, intrusivo o molesto, entre ellos están:

1.3.2.5.1. Virus

El virus permanece inactivo hasta que un usuario lo ejecuta. En este momento el virus comienza a infectar los archivos extendiéndose por todo el equipo, muchas veces inhabilitan el equipo, corrompen archivos del sistema operativo o archivos del usuario.

1.3.2.5.2. Worms

También conocidos como gusanos, el objetivo de los gusanos informáticos es infectar los archivos del equipo para difundirlos. Tienen la capacidad de extenderse a otros equipos sin necesidad de que un usuario lo ejecute.

1.3.2.5.3. Troyanos

Muestra la apariencia de un programa fiable, pero esconden otro tipo de *malware* que es instalado automáticamente con el objetivo de tomar el control del equipo, secuestrar información, obtener datos importantes de la red.

1.3.2.5.4. Keyloggers

Tienen la capacidad de registrar las pulsaciones del teclado. Esta información es utilizada para conseguir contraseñas y datos críticos de la víctima.

1.3.2.5.5. *Ransomware*

Este tipo de ataque es el más popularidad ha alcanzado en la actualidad. Se basa en el cifrado de datos, restringiendo el acceso a los archivos del equipo para pedir un pago por el rescate de los mismos, la mayoría de las veces en bitcoins.

1.3.2.6. Inyección de código

La inyección de código es otro tipo de ataque que aprovecha vulnerabilidades principalmente en las páginas web para ingresar código

ejecutarlo y así obtener información principalmente del entorno de ejecución de la aplicación, por ejemplo servidores de seguridad, código de base de datos.

1.3.2.6.1. Inyección de código SQL

Es un método de infiltración de código intruso que se vale de una vulnerabilidad informática presente en una aplicación en el nivel de validación de las entradas para realizar operaciones sobre una base de datos. El origen de la vulnerabilidad radica en la incorrecta comprobación o el filtrado de las variables utilizadas en un programa que contiene o bien genera código SQL.

1.3.2.7. Cross-Site Scripting (XSS)

Es un tipo de vulnerabilidad informática o agujero de seguridad típico de las aplicaciones Web, que puede permitir a una tercera persona inyectar en páginas web visitadas por el usuario código Java*Script* o en otro lenguaje similar (VBScript), se puede evitar usando medidas como CSP política del mismo origen. Es posible encontrar una vulnerabilidad en aplicaciones que tengan entre sus funciones presentar la información en un navegador web u otro contenedor de páginas web. XSS es un vector de ataque que puede ser utilizado para robar información delicada, secuestrar sesiones de usuario, y comprometer el navegador, subyugando la integridad del sistema. Las vulnerabilidades XSS han existido desde los principios de la Web.

El ataque XSS se divide en dos: el ataque XSS persistente o directo y el segundo XSS reflejado o indirecto.

- El ataque persistente o directo consiste en invadir código HTML mediante la inclusión de *script* y *frame* en sitios vulnerables.

- El ataque indirecto o reflejado funciona modificando valores que la aplicación web pasa de una página a otra (parámetros), se envían normalmente en la cabecera HTTP.

1.3.2.8. Denegación de servicio (DOS)

Denial of Service es un ataque a un sistema de computadoras o red que causa que un servicio o recurso sea inaccesible a los usuarios legítimos. Provoca la pérdida de la conectividad con la red por el consumo del ancho de banda de la red de la víctima o en todo caso sobrecarga los recursos del sistema que han atacado.

Los ataques DoS se generan mediante la saturación de los puertos con múltiples flujos de información, que provoca que deje de prestar el servicio; justamente de allí su nombre, hace que el servidor o la red deje de funcionar.

Este problema ha ido creciendo con los años, esto es por la facilidad para crear ataques y por la cantidad de equipos que pueden ser atacados. Una ampliación del ataque DoS es el llamado ataque de denegación de servicio distribuido; este tipo de ataque se lleva a cabo generando un gran flujo de información desde varios puntos de conexión hacia un mismo punto de destino. El método más común es el ataque por medio de una red de *bots* (red de ro*bots* que se ejecutan de manera autónoma y automática).

1.4. ¿Cómo defendemos?

Hemos revisado las técnicas más populares que los ciberdelincuentes utilizan para robar, secuestrar y dañar la información de organizaciones y

personas individuales, ahora se enumeran algunas de las técnicas que permiten prevenir estos ataques.

1.4.1. Antivirus

Los antivirus son programas que tienen como único objetivo detectar y eliminar virus informáticos. Con el transcurso del tiempo, la aparición de sistemas operativos más avanzados, los antivirus han evolucionado hacia programas más fuertes que además de buscar y detectar virus informáticos, consigue bloquearlos, desinfectar archivos y prevenir una infección de los mismos. Así como los ataques han evolucionado los antivirus han tenido que evolucionar, ahora son capaces de detectar otros tipos como *malware*, *spyware*, gusanos troyanos, *rootkits*, pseudovirus.

1.4.2. Antiransomware

Para proteger los equipos contra los ataques de ransomware se debe tener un *antiransomware* que puede estar o no incorporado a un antivirus, y que permita hacer copias de seguridad que incluya almacenamiento en la nube.

Estos programas supervisan activamente el sistema para detectar procesos sospechosos y detenerlos. Si el ransmware logra cifrar los archivos, el *antiransomware* debe ser capaz de restaurarlos desde una caché local o desde la nube. Los diferentes antivirus existentes en el mercado no cubren todas las necesidades para estar completamente protegido, por lo que es ideal tener una combinación entre antivirus, *antiransomware* y copia de seguridad.

1.4.3. Análisis de perfil de riesgo

Esta técnica está enfocada directamente a mitigar el riesgo de los ataques de ingeniería social, en este caso muchos han sido los esfuerzos que se han hecho por empresas, por ejemplo, programas constantes de formación y herramientas que utilizan correos de *phishing* falsos, políticas de longitudes mínimas de contraseña, es decir, inversiones millonarias en sensibilización, pero el problema persiste.

Es por esto que se han diseñado un conjunto de algoritmos que permiten identificar a los usuarios susceptibles a los ataques en función a sus rasgos de personalidad y conductas del uso de computadora. Esta tecnología puede adaptar contramedidas de seguridad para los usuarios individuales, por ejemplo, él envió de mensajes de advertencia a los individuos que acostumbran hacer clic en los URL en los mensajes de correo, o elevar el nivel de amenaza de correos electrónicos sospechosos enviados entre los departamentos por usuarios con características o historial de malas prácticas.

Los algoritmos de análisis de perfil de riesgo han identificado entre muchas cosas que las personas que dan prioridad a los beneficios más que los riesgos están en mayor riesgo de un ataque de virus, mientras que aquellos con una mayor confianza en sus habilidades informáticas son un mayor riesgo en fuga de datos. Si bien es cierto que estos algoritmos necesitan de muchas configuraciones para limitar a las personas identificadas, son un gran avance en tema de prevención

1.4.4. Sistemas de detección de intrusos

Desde 1980 la investigación sobre detección de intrusos tomo revuelo, el primer estudio fue realizado por el gobierno norteamericano por James P. Anderson, trataron de mejorar la complejidad de los sistemas de auditoría de la seguridad informática. Anderson presento la hipótesis de que el comportamiento normal de un usuario podría caracterizarse mediante el análisis de su actividad en los registros de auditoría. De esta manera, los intentos de abuso podrían descubrirse detectando actividades anómalas que se desviaran significativamente de ese comportamiento normal.

El término IDS (sistema de detección de intrusiones) hace referencia a un mecanismo que, sigilosamente, escucha el tráfico de red para detectar actividades anormales o sospechosas, y de este modo, reducir el riesgo de intrusión.

1.4.4.1. Clasificación

La clasificación de los IDS se realiza de acuerdo con diferentes criterios (figura 5):

- La fuente de información que utilizan para detectar la intrusión. Se refiere al origen de los datos que se usan para determinar si una intrusión se ha llevado a cabo, y se dividen en dos:

    - N-IDS: sistema de detección de intrusiones de red, garantiza la seguridad dentro de la red. Son los más populares en el área de detección de intrusos, utilizan el tráfico de red (paquetes TCP/IP) como fuente de información.

- H-IDS = sistema de detección de intrusiones en el *host*, garantiza la seguridad en el *host*. Protegen únicamente a la máquina en la que son instalados. Utilizan como fuente de información los datos generados por la computadora en la que operan, directamente a nivel de sistema operativo. Dentro de los HIDS existen subgrupos que utilizan diferentes vías para detectar las intrusiones. Algunas de estas son:

    - Sistemas verificadores de integridad (SIV): lo que hace es monitorizan los sistemas de archivos en busca de cambios relevantes.

    - Analizadores de archivos log (LFM): verifica los archivos logs en busca de patrones sugerentes.

    - Sistemas víctima (*honeypots*): sistemas que aparentan servidores o computadoras vulnerables y que se ofrecen como señuelos para desviar la actividad de los intrusos.

- La estrategia de análisis. Se refiere a la técnica utilizada por el IDS, una vez recuperada la información, para identificar si se produjo o no una intrusión. La información que se ha recopilado en el proceso anterior puede ser analizado mediante dos estrategias: una basada en uso indebido y la otra basada en anomalías.

    - Las IDS basadas en uso indebido, tiene entre sus grandes ventajas la amplia seguridad que ofrece al detectar una intrusión. Cuando clasifica una actividad como intrusiva significa que la correspondencia con un patrón de la base de datos de ataques

anteriores ha sido exitosa. De esta manera la cantidad de faltos positivos disminuye considerablemente. Dentro de las desventajas esta, la incapacidad para detectar nuevos ataques debe mantener constantemente actualizada su base de datos de patrones. Y en la actualidad en donde los ataques cada vez son más frecuente y original. Otra desventaja es que hay que adaptar manualmente el IDS al sistema es el que se adapta. Esta estrategia es la más utilizada en los IDS comerciales y por la que apuestan los fabricantes. Básicamente la técnica de uso indebido contiene dos componentes principales:

- Un lenguaje o modelo para describir o representar las técnicas utilizadas por los atacantes.

- Programas de monitorización para detectar la presencia de un ataque basado en las representaciones o descripciones dadas.

- Basados en la detección de anomalías, funciona de forma contraria a la estrategia de uso indebido, esta técnica parte del conocimiento de lo normal y toda actividad que se aleje de ese comportamiento es considerada una intrusión. Con este tipo de detección se evita el proceso de actualización de una base de datos de patrones de intrusión, por lo tanto, esto permite identificar ataques nuevos, porque no requiere una base de datos actualizada. Pero una de sus desventajas por lo que ha perdido popularidad es que generar muchos falsos positivos, porque el comportamiento de los usuarios no es fácil de modelar. Otra de sus debilidades es que el periodo de entrenamiento previo a su

uso es muy largo. Básicamente, un sistema basado en detección de anomalías se clasifica en sistemas basados en conocimiento por ejemplo sistemas expertos, sistemas basados en métodos estadísticos y sistemas basados en aprendizaje automático.

- Según el tipo de respuesta. La respuesta que proporcione un IDS pueden ser pasivas o activas:

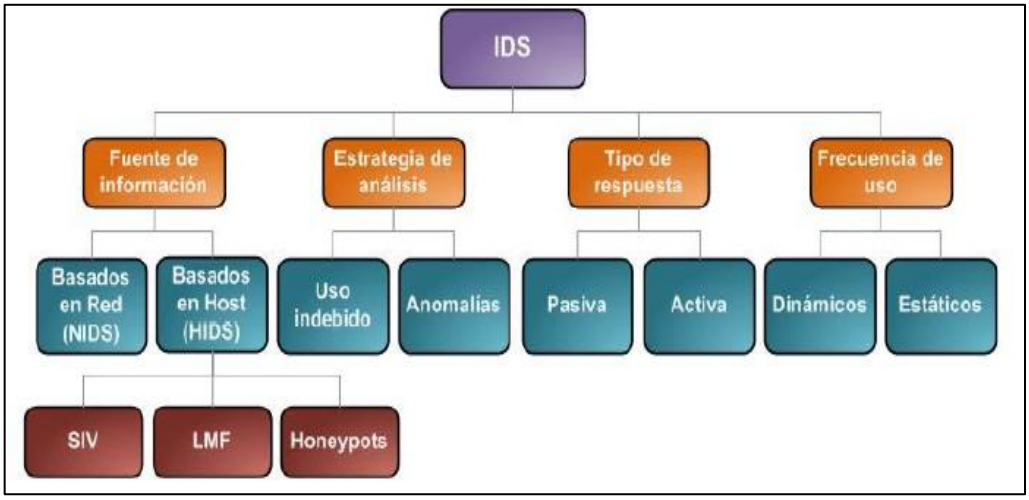
  - Las respuestas pasivas se refieren a emisiones de informes, sonidos o el registro de las acciones ocurridas. Las activas son las que desencadenan alguna acción en particular como: la ejecución de programas, el cierre de una cuenta o la reconfiguración del *firewall*.

  - Las activas son las que desencadenan alguna acción, como la ejecución de programas, el cierre de una cuenta o la reconfiguración del *firewall*.

- Según la frecuencia de uso. Según esta categoría los IDS se clasifican en dinámicos y estáticos. Los IDS dinámicos son capaces de analizar la actividad en tiempo real. Esto permite que se pueda responder adecuadamente ante un ataque cuando está ocurriendo y evitar sus efectos. La dificultad con este tipo de herramientas que tienen un costo operativo alto.

Los IDS estáticos trabajan fuera de línea a intervalos de tiempo definidos. Estos no ofrecen seguridad entre intervalos de ejecución y en casos de ataques exitosos solo pueden ser utilizados para análisis forense.

Figura 5. **Clasificación de IDS**



Fuente: Departamento de Tecnología Informática y Computación, Universidad de Alicante. *Método para la detección de intrusos mediante redes neuronales basado en la reducción de características*. p. 3.

1.5. Otras herramientas para defendernos

Existe diversidad de herramientas que pueden apoyar a proteger los equipos y por lo tanto la información, aunque por el constante avance e ingenio que los ciber-delincuentes han demostrado es importante adaptar nuevas técnicas que no solo protejan sobre ataques o virus conocidos, sino que sean capaces de predecir un ataque, es acá en donde aparece la inteligencia artificial como una herramienta útil.

1.5.1. inteligencia artificial

La inteligencia artificial (IA) es una de las ramas de la informática, con fuertes raíces en otras áreas como la lógica y las ciencias cognitivas. La inteligencia artificial tiene características importantes que se deben cumplir:

- Actuar como las personas
- Razonar como las personas
- Razonar racionalmente
- Actuar racionalmente

- Inteligencia artificial débil: se considera que los equipos únicamente pueden simular que razonan, y únicamente pueden actuar de forma inteligente. Los precursores de la IA débil consideran que no será nunca posible construir ordenadores conscientes, y que un programa es una simulación de un proceso cognitivo pero no un proceso cognitivo en sí mismo.

- inteligencia artificial fuerte: en este caso se considera que un ordenador puede tener una mente y unos estados mentales, y que por tanto un día será posible construir uno con todas las capacidades de la mente humana. Este ordenador será capaz de razonar, imaginar, entre otros.

Aunque son varios los puntos de vista que definen la inteligencia artificial, hay un consenso respecto a cuáles son los campos de la IA:

- Resolución de problemas y búsqueda.
- Representación del conocimiento y sistemas basados en el conocimiento.

- Aprendizaje automático.
- Inteligencia artificial distribuida.
- Lenguaje natural.
- Visión artificial.
- Robótica.
- Reconocimiento del habla.

Por estas razones, la inteligencia artificial se ha convertido en un importante aliado para fortalecer la seguridad informática. En el siguiente capítulo se analizan diferentes técnicas de inteligencia artificial que pueden ser por sus características las adecuadas para apoyar los sistemas de detección de intrusos.

2. COMPARACIÓN DE TÉCNICAS DE INTELIGENCIA ARTIFICIAL

La inteligencia artificial es una rama importante y relativamente nueva de la computación; en la última década el crecimiento de las aplicaciones de la IA ha sido impresionante y la seguridad informática no es la excepción, es por ello que trataremos de concluir la pregunta: ¿cuál de las diferentes técnicas se puede adaptar mejor a la seguridad informática?

2.1. Técnicas de inteligencia artificial

Existen diferentes técnicas de inteligencia artificial que pueden ser utilizadas por sus características como fuertes aleados en la defensa de la información y principalmente con los sistemas de detección de intrusos (SDI); el objetivo de esta sección es conocer las diferentes técnicas, su funcionamiento, ventajas, desventajas para luego poder tomar una decisión sobre cuál puede ser la técnica más eficiente, la más factible de implementar.

2.1.1. Sistemas basados en casos

Es el proceso de solucionar nuevos problemas basándose en las soluciones de problemas anteriores, es un sistema basado en el conocimiento, parte de la experiencia, del conocimiento adquirido. Se considera que el razonamiento basado en casos no solo es un método poderoso para el razonamiento de computadoras, sino que también es utilizado por las personas. Los casos son almacenados como unidades de conocimiento. Para indexar la

gran cantidad de casos se puede utilizar un vocabulario prefijado o bien cualquier palabra del propio vocabulario.

- Funcionamiento: el ciclo de razonamiento es dividir el razonamiento en diferentes subprocesos:

- Recordar: los casos similares al que se analiza.

- Reutilizar: información y conocimiento que se obtiene en el caso específico para resolver el problema.

- Revisar: la solución propuesta.

- Retener: se agrega la experiencia a la base de conocimiento para resolver futuros problemas.

Este tipo de funcionamiento es el razonamiento del proceso en forma de ciclo. Los puntos anteriores se repiten una y otra vez hasta encontrar la mejor solución (figura 6).

La estructura de un sistema basado en el conocimiento es:

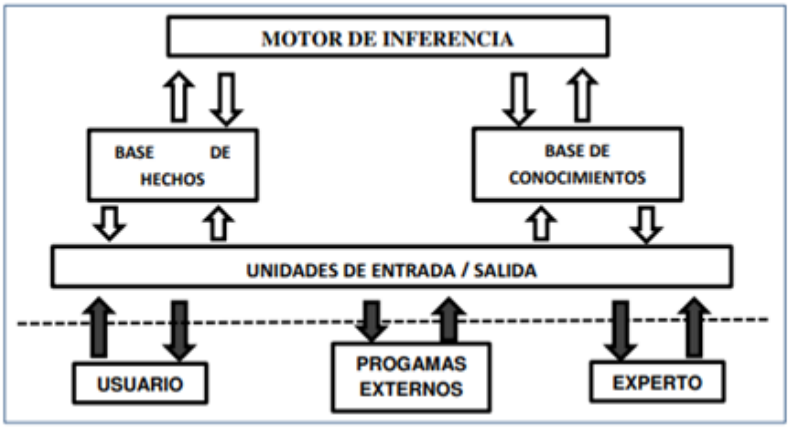
- Base de conocimiento: contiene los conocimientos relativos a la tarea. Es llamada memoria a largo plazo, en las que se guarda los hechos (base de hechos) y los conocimientos acerca del dominio.

- Motor de inferencias: método por el cual controlan y aplican los conocimientos. Permite que el sistema razone a partir de los datos,

noticias o conocimientos de entrada para producir los resultados de salida.

- Interfaz de E/S: no solo par que el usuario proporcione hechos y datos y el sistema responda.

Figura 6. **Estructura de un sistema basado en el conocimiento**



Fuente: WINOGRAD, Terry; FLORES, Fernando. *Understanding Computer and Cognition Norwood*. https://es.scribd.com/doc/64479271/Understanding-Computers-and-Cognition-Winograd-Flores. Consulta: 2 de septiembre de 2019.

2.1.2. Redes neuronales

Es un campo sumamente importante dentro la IA, se inspira en el comportamiento conocido del cerebro humano, trata de crear modelos artificiales que solucionen problemas difíciles de resolver mediante técnicas algorítmicas convencionales. Una red neuronal es la unión de varias redes, una neurona es la analogía de una neurona biológica, tal como se observa en la figura 7, está formada por:

- Conjunto de entradas o vector de entrada x, de n componente.

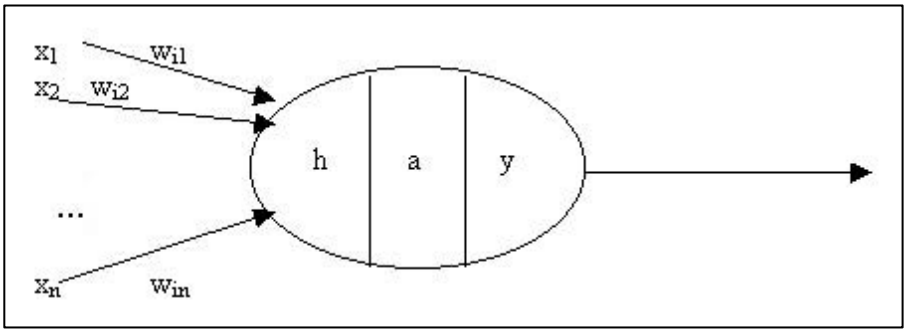
- Conjuntos de pesos sinápticos $w_{ij.}$ representan la interacción entre la neurona presináptica j y la possináptica i.

- Regla de propagación $d(w_{ij,}x_j(t))$: proporciona el potencial possináptico, $h_i(t)$.

- Función de activación $a_i(t)=f(a_i(t-1), h_i(t))$: proporciona el estado de activación de la neurona en función del estado anterior y del valor possináptico.

- Función de salida $F_i(t)$: proporciona la salida $y_i(t)$, en función del estado de activación.

Figura 7. **Estructura de una red neuronal**



Fuente: Redes neuronales. *Modelo de neurona artificial*. http://avellano.fis.usal.es/~lalonso/RNA/index.htm. Consulta: 2 de septiembre de 2019.

Las señales de entrada y salida pueden ser valores binarios (0,1 – neuronas de Mcculloch y Pitts), bipolares (-1,1), números enteros o continuos,

variables borrosas. La regla de propagación normalmente es una suma ponderada del producto escalar del vector de entrada y el vector de pesos: $h_i(t) = \sum w_{ij} x_{i.}$

Se utiliza normalmente la distancia euclidiana entre ambos vectores:

$$h_i(t) = \sum (x_j w_i)^2$$

La función de activación no suele tener en cuenta el estado anterior de la neurona, sino solo el potencia $h_i(t)$. Suele ser una función determinista, continua y monótona creciente. Las funciones más utilizadas de activación son:

2.1.2.1. *Sigmoid – Sigmoide*

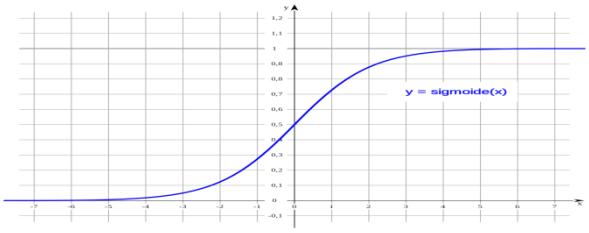
Transforma los valores introducidos a una escala (0,1), donde los valores altos tienen de manera asintótica a 1 y los valores muy bajos tienden a 0 (figura 8).

- Características
    - Satura y mata el gradiente
    - Lenta convergencia
    - No está centrada en el cero
    - Está acotada entre 0 y 1
    - Buen rendimiento en la última capa

Figura 8. **Gráfica *Sigmoide***

$$f(x) = \frac{1}{1 - e^{-x}}$$

Función Sigmoide



Fuente: elaboración propia, empleando Geogebra.

2.1.2.2. *Tanh – Tangent Hyperbolic-*

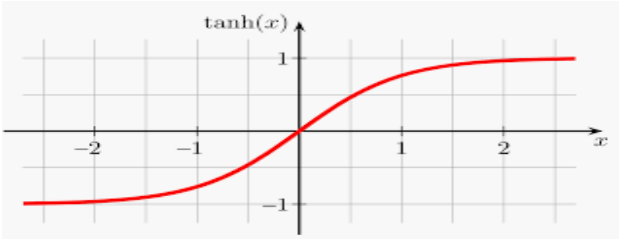
Tangente hiperbólica: la función tangente hiperbólica transforma los valores introducidos a una escala (-1,1), donde los valores altos tienen de manera asintótica a 1 y los valores muy bajos tienden a -1 (figura 9).

- Características
  - Similar a la *sigmoide*
  - Satura y mata el gradiente
  - Lenta convergencia
  - Centrada en 0
  - Está acotada entre -1 y 1
  - Se utiliza para decidir entre una opción y la contraria
  - Buen desempeño en redes recurrentes

Figura 9. **Gráfica Función *Tanh***

$$f(x) = \frac{2}{1 + e^{-2x}} - 1$$

Función tangente hiperbólica



Fuente: elaboración propia, empleando Geogebra.

2.1.2.3. *ReLU – Rectified Lineal Unit*

Transforma los valores introducidos anulando los valores negativos y deja los positivos tal como ingresan a la neurona (figura 10).

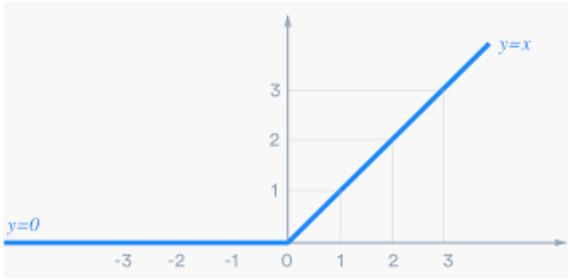
- Características

  - Activación Sparse – solo se activa si son positivos
  - No está acotada
  - Se pueden morir demasiadas neuronas
  - Se comporta bien con imágenes
  - Excelente desempeño en redes convolucionales

Figura 10. **Gráfica de funciones *ReLU***

$$f(x) = \max(0, x) = \begin{cases} 0 \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases}$$

Función ReLU



Fuente: elaboración propia, empleando Geogebra.

2.1.3. ¿Qué función de activación utilizar?

Para aprovechar la capacidad de las redes neuronales de aprender relaciones complejas o no lineales entre variables, es recomendable utilizar funciones no lineales al menos en las neuronas de la capa oculta. Por tanto, en general se utilizará una función *sigmoidal* como función de activación en las neuronas de la capa oculta. La elección de la función de activación en las neuronas de la capa de salida dependerá del tipo de tarea impuesto. En tareas de clasificación, las neuronas normalmente toman la función de activación *sigmoidal*. En cambio, en tareas de predicción o aproximación de una función, generalmente las neuronas toman la función de activación lineal (*ReLU*).

2.1.4. Sistemas expertos

También se les conoce como sistemas basados en conocimiento, porque razonan como uno o un grupo de expertos en una determinada área o especialidad. Estos sistemas intentan imitar las acciones que realizaría un experto frente a un problema que se haya programado, con el fin de resolverlo.

Es un método costoso, pero el mantenimiento y el precio de su uso es relativamente bajo. Para que sea efectivo debe reunir dos capacidades:

- Explicar sus razonamientos o base del conocimiento: estos sistemas deben crearse siguiendo ciertas reglas compresibles de manera que se genere una explicación basada en hechos reales para estas reglas.

- Adquisición de nuevos conocimiento o integrador del sistema: estos son mecanismos de razonamiento que debe tener el usuario para modificar los conocimientos anteriores.

Los sistemas expertos se clasifican de la siguiente manera:

- Deterministas: se basan en conjuntos de reglas correctamente definidas, también se los conocen como sistemas basados en reglas porque obtienen los resultados de un mecanismo de razonamiento lógico que analiza el conjunto de reglas en el que se basan, algunos ejemplos de estos sistemas son las transacciones bancarias y control de tráfico.

- Estocásticos: utiliza medios para tratar la incertidumbre en entornos dudosos; algunos procedimientos expertos hacen uso de la estructura de los sistemas basados en reglas, estos valores pueden ser la probabilidad en que la distribución de un conjunto de variables se utiliza, para describir las relaciones de dependencia. También, se les conoce como sistemas probabilísticos.

2.1.4.1. Estructura básica de un sistema experto

Un sistema experto está conformado básicamente por:

- Base de conocimientos: se debe suministrar una base de conocimientos para diseñar el sistema basado en conocimiento, la base de datos debe estar ordenada y estructurada, además de un grupo de relaciones definidas y explicadas.

- Memoria de trabajo: contiene los datos obtenidos de un problema durante el análisis.

- Motor de inferencia: es el motor de todo sistema experto, su principal objetivo es el de sacar conclusiones aplicando el conocimiento a los datos.

- Módulo de justificación: realiza la explicación del proceso que siguió el motor de inferencia para llegar a una conclusión.

- Permanencia: los sistemas expertos no envejecen por lo tanto no sufre perdida de sus facultades a largo tiempo.

- Rapidez: los sistemas expertos pueden obtener información de una base de datos y hacer cálculos numéricos mucho más rápido que un humano.

2.2. Comparación de técnicas de inteligencia artificial

Una vez hecho el análisis de algunas técnicas de inteligencia artificial, se realiza un cuadro comparativo valuando la factibilidad y viabilidad de implementación.

Tabla I. **Análisis comparativo técnicas de inteligencia artificial**

	Sistemas basados en casos	Redes neuronales	Sistemas expertos
Factibilidad	Se requiere de grandes recursos de almacenamiento e información. Se crea una importante base de conocimiento.	Requiere mucho tiempo para entrenamiento y grandes recursos, Una vez entrenado es portable y fácil de utilizar.	Igual que los otros métodos requieren información previa para el cálculo, depende de elegir las variables adecuadas y la salida correcta para encontrar la relación.
Viabilidad	Muy efectivo para identificar casos documentados, deficiente en casos nuevos.	Una vez entrenada la neurona puede reconocer nuevos casos sin problema.	Requiere que la que base de conocimientos obtenga la información exacta para el análisis.

Fuente: elaboración propia.

Una vez analizadas las características y funcionalidades de las diferentes tecnologías descritas se analiza para seleccionar la que mejor se adapte.

En la tabla II se muestra la comparación de las tecnologías de inteligencia artificial, las características que se analizan permiten colocar las tres tecnología en un mismo nivel de comparación.

Tabla II. **Selección de tecnológica de inteligencia artificial**

Característica	Sistemas basados en casos	Sistemas expertos	Redes neuronales artificiales
Facilidad de diseño del sistema	No	No	Sí
Capacidad de aprender	No	No	Sí
durante la marcha			
Consumo bajo de recursos	No	No	Sí
Capacidad de abstracción y generalización de la información	No	Sí	Sí
Clasificación de los datos	No	No	Sí
Extracción de características de datos clasificados	No	No	Sí

Fuente: elaboración propia.

En la tabla III se realiza un análisis comparativo de las ventajas y desventajas de utilizar las diferentes técnicas de inteligencia artificial.

Tabla III. **Ventajas y desventajas de técnicas de inteligencia artificial**

Método	Ventajas	Desventajas
Sistemas basados en casos	• Disminuye la cantidad de falsos positivos. • Método de razonamiento ha sido demostrado como eficiente. • Incrementa la -difusión de la experiencia escasa. • La manipulación de los casos es sencilla.	• La cantidad de información a indexar es muy grande. • Cuando no se encuentra un caso parecido le es muy difícil encontrar una solución. • Mantenimiento elevado de la base de conocimientos. • Ausencia de sentido común.

Continuación de la tabla III.

Redes neuronales	• Tiene la habilidad de aprender. • Crea su propia representación de la información en su interior. Almacena la información de forma redundante, ésta puede seguir respondiendo de manera aceptable aun si se daña parcialmente. • Pueden trabajar en ambientes donde los datos a procesar no se encuentran bien definidos o contienen un alto grado de ruido.	• Complejidad de aprendizaje, utiliza grandes tareas, mientras más cosas se necesiten que aprenda una red, más complicado será enseñarle. Tiempo de aprendizaje elevado. • Elevada cantidad de datos para el entrenamiento, cuanta más experta se requiere que sea la red, más información tendrá que enseñarle para que realice de forma adecuada la identificación.
Regresión multivariada	• Enorme capacidad de determinar la influencia relativa de una o más variables predictoras para el valor de criterio. • Es útil para identificar valores atípicos o anomalías.	• Difícil de trabajar cuando tiene datos incompletos. • Puede fallar y presentar una conclusión falsa de que una correlación es una causa.

Fuente: elaboración propia.

Con base en los análisis anteriores se puede definir que la técnica recomendada como apoyo a los SDI son las redes neuronales; la principal razón de la selección es que aunque se debe invertir mucho tiempo para el entrenamiento de la red neuronal y se requiere una base de datos de conocimiento bastante grande, la efectividad en la predicción pueden hacer la diferencia entre detectar o no a un intruso.

Otra razón por la que la selección son las redes neuronales es su habilidad para predecir un posible ataque, han sido probadas y constantemente renovadas; a continuación se hace un análisis de ventajas que pueden aportar las redes neuronales a los sistemas de detección de intrusos.

Tabla IV. **Ventajas de las redes neuronales aplicadas en los SDI**

Características de las redes neuronales	Ventajas en la detección de intrusiones
Forma de representación del conocimiento muy apropiado para problemas de clasificación	En los IDS se trabaja precisamente con un problema de clasificación, permite tomar ventajas de la facilidad de presentación del conocimiento.
Alta tolerancia a errores, siendo capaces de clasificar datos que representa ruidos o están incompletos.	Esta característica es de gran importancia sobre todo en los IDS basados en red que analizan paquetes TCP/IP que pueden haber sufrido algunas modificaciones por problemas en la red.
Devuelven un valor numérico que da idea del nivel de seguridad de la clasificación realizada. Pueden clasificar datos desconocidos.	Da una idea más clara al administrador acerca de la decisión a tomar. Brinda la posibilidad de detectar ataques de los cuales aún no se tiene conocimiento.

Fuente: elaboración propia.

Ahora se realiza un análisis comparativo con diferentes tipos de Redes Neuronales, para encontrar cual puede ser la que mejor se adapte a los sistemas de seguridad informática.

2.3. Selección del tipo de red neuronal a implementar

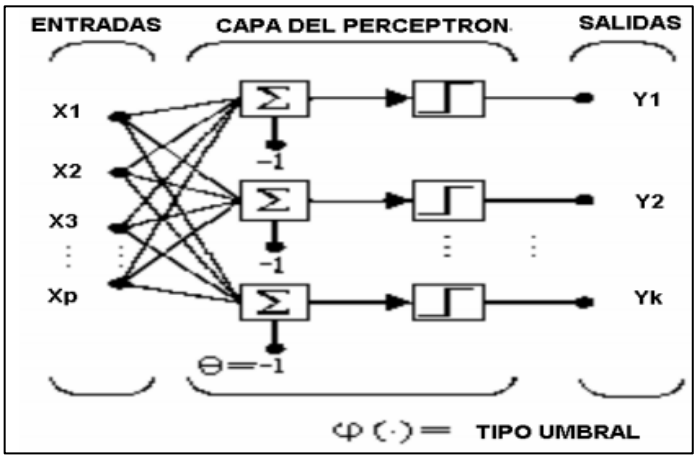
Las neuronas entonces se vuelven la herramienta idónea para aumentar el valor y eficiencia de los SDI, con su funcionamiento de 1 entrada, N capas ocultas cuando así están diseñadas y una salida. Dentro de las redes neuronales existen diferentes diseños que es interesante analizar, se analizan las redes multicapa Perceptron por la estructura de diferentes capas que la hacen ser más eficiente en la reducción del error, las redes recurrentes que por su sistema de bucle permite emular una memoria que permite una mejor predicción. Las redes FeedForward que aunque pueden ser más lentas pueden hacer un análisis 1 a 1 de elementos ingresado y aunque no mantienen una

memoria es la base de las redes Elman y pueden ser utilizadas como una técnica para implementar cuando no se cuente con una base de datos de información muy robusta y se requiera actualizar frecuentemente, las redes de base radial que su diseño simple permite una programación más rápida y mantienen una buena eficiencia.

2.3.1. Multicapa Perceptrón

Según el modelo de una neurona artificial de Rosenblatt (1958,1962) nace el concepto de Perceptrón, este modelo básicamente consiste en capas de neuronas con pesos y umbrales ajustables. En la figura 11 se muestra la estructura de una red Perceptrón.

Figura 11. **Perceptrón**



Fuente: MEJÍA SÁNCHEZ, Juan Arturo. *Perceptrón Multicapa*. http://catarina.udlap.mx/u_dl_a/tales/documentos/lep/mejia_s_ja/capitulo3.pdf. Consulta: 5 de septiembre de 2019.

Este tipo de neuronas utiliza el paradigma de aprendizaje supervisado y algoritmos de correcciones de errores o regla delta, para este aprendizaje se entrega al sistema la información de entrada como la información de salida o destinos que debe tener para dicha entrada. El algoritmo de aprendizaje del Perceptrón sigue los principios de la regla de ajustar los pesos de la neurona para ir optimizando la función de coste, y que el error o la salida sea cero.

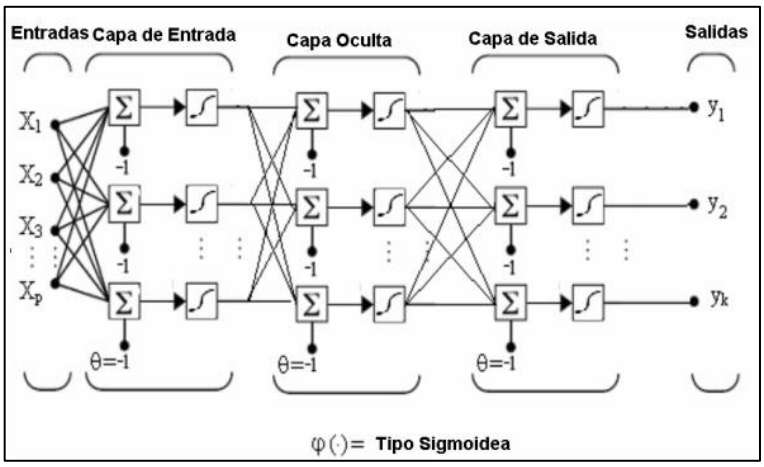
Es por este tipo de aprendizaje que esta neurona es una buena candidata para utilizar en los sistemas de detección de intrusos, porque se le puede definir las salidas necesarias para identificar si un comportamiento es o no normal.

Si se aplica este método al Perceptrón con distintas muestras, hasta que el error sea cero, se tendrá una red que pueda generar exactamente las salidas deseadas para las entradas determinadas.

2.3.1.1. Limitaciones del Perceptrón

Ahora bien, como todo algoritmo de aprendizaje supervisado y no supervisado hay algunas limitaciones, en este caso puede quedarse en un loop a un error cero o nulo. Es más, el algoritmo del Perceptrón es incapaz a converger en funciones que son linealmente separables, es decir, aquellas en las que los elementos pueden ser separados por una línea recta. Ahora bien, esto se soluciona evolucionando el algoritmo al Perceptrón multicapa, como se ve en la figura 12, que consiste en agregar capas ocultas de neuronas con funciones de activación diferentes, por ejemplo, una función Sigmoide.

Figura 12. **Multicapa Perceptrón**



Fuente: MEJÍA SÁNCHEZ, Juan Arturo. *Perceptrón Multicapa.* http://catarina.udlap.mx/u_dl_a/tales/documentos/lep/mejia_s_ja/capitulo3.pdf. Consulta: 5 de septiembre de 2019.

Se infiere de la arquitectura que el algoritmo de entrenamiento de una red de la forma Perceptrón multicapa deberá ser planeado para que los cambios en los parámetros libres sean tales que el error en las unidades básicas de la estructura sea mínimos, esto hace que el conjunto de los cambios produzca un error global, que tiende al mínimo.

2.3.2. Redes neuronales recurrentes

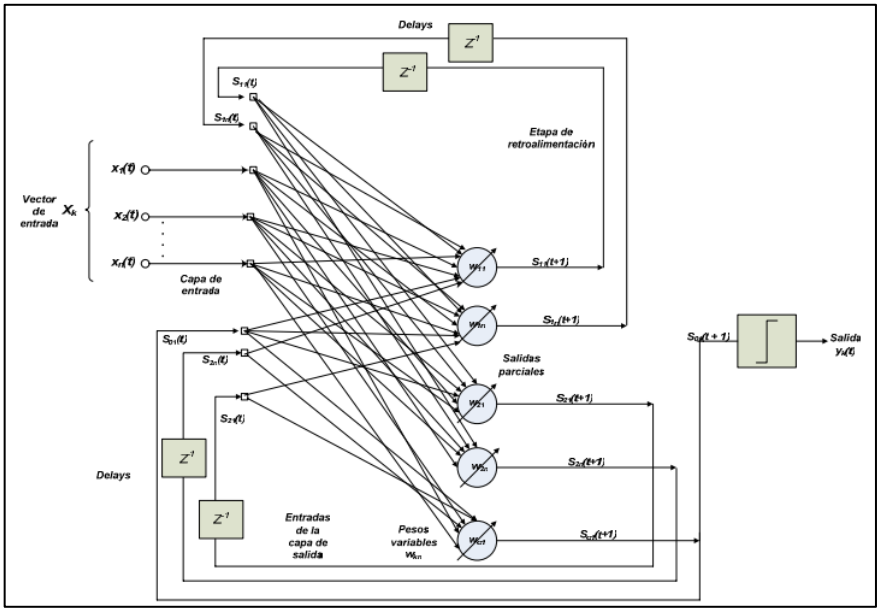
No tiene una estructura de capas definida, permite conexiones arbitrarias entre las neuronas, incluso crea ciclos, con esto consigue crear un estado de temporalidad, haciendo de esta manera que las redes tengan la tan ansiada memoria.

Es por eso que son tan potentes para todo análisis secuencias, por ejemplo análisis de texto, sonido o video. Las redes neuronales recurrentes son la arquitectura base sobre la que se implementan otras topologías, difieren del resto en que estas incorporan la retroalimentación para crear la temporalidad. Una sola neurona está conectada a las neuronas posteriores en la siguiente capa, las neuronas pasadas de la capa anterior y a ella misma a través de vectores de pesos variables que son alteradas en cada *epoch* (iteración) con el fin de alcanzar los parámetros o metas de operación.

Las redes neuronales recurrentes realizan el intercambio de información entre sus neuronas de una manera muy compleja, y dependiendo del algoritmo de entrenamiento que se seleccione pueden propagar la información hacia adelante en el tiempo, lo cual equivale a predecir eventos.

Una característica importante es la inclusión de delays (z^{-1}) a la salida de las neuronas en capas intermedias, las salidas parciales $S_{mn}(t+1)$ se convierten en valores $S_{mn}(t)$, un instante de tiempo anterior, y así se retroalimentan a todos los componentes de la red, guardando información de instantes de tiempo anteriores. Otra característica es la interconexión entre todos los nodos anteriores a través de conexiones directas y también *delays* antes de cada capa o memorias temporales. En la figura 13 se observa la estructura de una red recurrente.

Figura 13. **Redes neuronales recurrentes**



Fuente: MEJÍA SÁNCHEZ, Juan Arturo. *Perceptrón Multicapa.* http://catarina.udlap.mx/u_dl_a/tales/documentos/lep/mejia_s_ja/capitulo3.pdf. Consulta: 5 de septiembre de 2019.

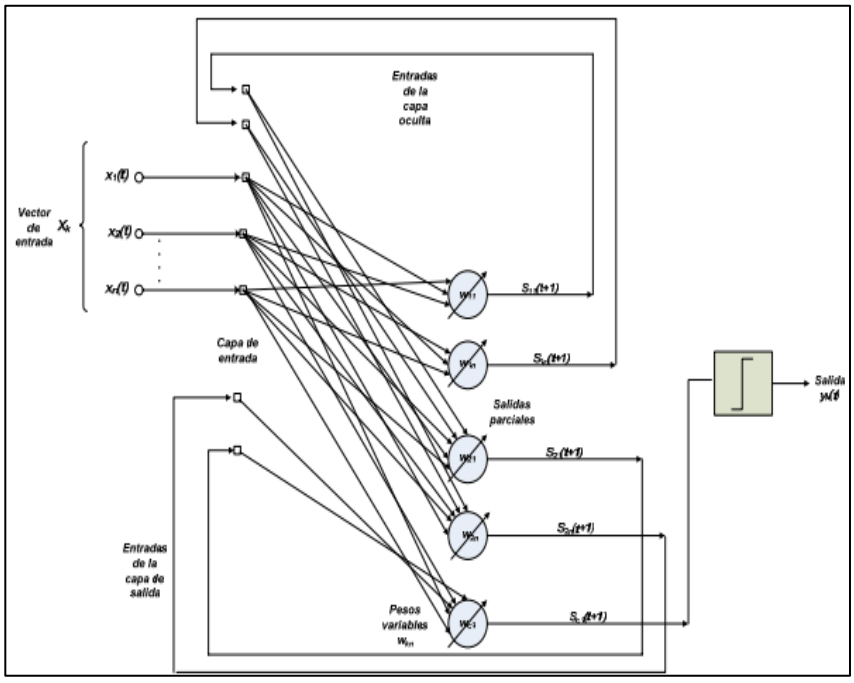
2.3.3. Redes FeedForward

Este tipo de red como su nombre lo indica inicia con un vector de entrada el cual es equivalente en magnitud al número de neuronas de la primera capa de la red, cada elemento del vector es procesado elemento por elemento en paralelo. La información que es modificada por los factores multiplicativos de los pesos, es transmitida hacia adelante por la red pasando por las capas ocultas, para ser procesada por la capa de salida.

Cada vector de entrada que se utilizara como entrenamiento es un proceso aislado del resto, al finalizar las pruebas la red estará lista para comenzar a identificar y clasificar patrones, reconocimiento de imágenes, entre

otros. Este tipo de redes son una opción cuyo balance costo-velocidad y costo-exactitud es tal que da mayor ventaja al costo que a los otros parámetros, no existe una interconexión entre las capas más allá de la conexión directa, por lo tanto la mantienen una memoria en la red (figura 14).

Figura 14. **Redes FeedForward**



Fuente: MEJÍA SÁNCHEZ, Juan Arturo. *Perceptrón Multicapa.* http://catarina.udlap.mx/u_dl_a/tales/documentos/lep/mejia_s_ja/capitulo3.pdf. Consulta: 5 de septiembre de 2019.

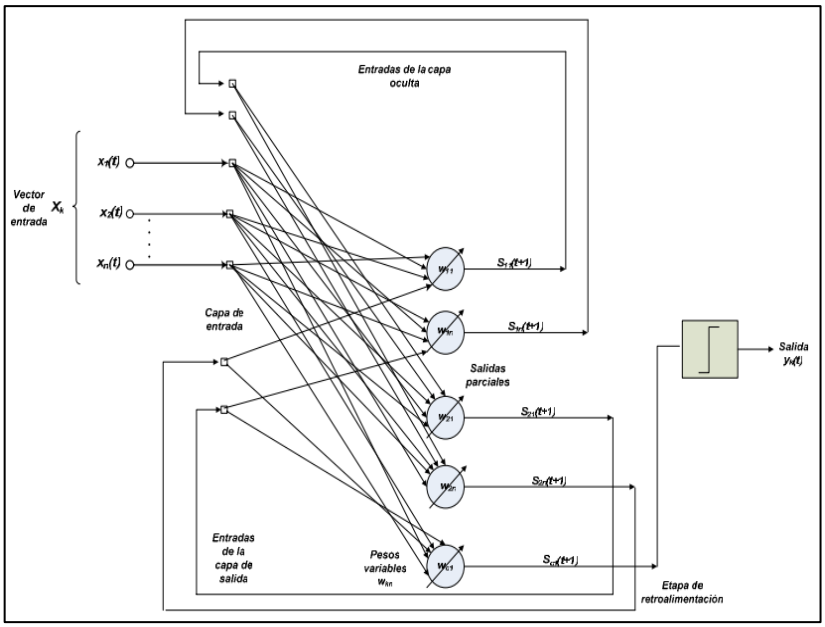
Como se ve en la figura, las neuronas en su totalidad no tienen conexión hacia atrás, solamente hacia las capas siguientes, inicial por el vector X_k, de esta manera la propagación es hacia adelante por medio de las salidas $S_{mn}(t+1)$. El vector de pesos W_k es actualizado en cada epochs (época) que pase mientras el entrenamiento se realiza, los resultados finales de una vez

procesada el vector de entrada $w_{11}\ldots w_{1n}$, $w_{21}\ldots w_{2n}$, asumen sus valores finales para iniciar el trabajo de la neurona como datos de entrada, La función Signum se encuentra al a salida de la red para convertir $S_{on}(t+1)$ en el valor final y_k.

2.3.4. Redes Elman

Las redes feedforward tiene limitaciones propias a su diseño que pueden ser mejoradas con un cambio de arquitectura. Las redes Elman son redes recurrentes simples (SRN) y son una mejora de las redes feedforward, y la mejora es la inclusión de una retroalimentación entre las capas inmediatas contiguas, como se muestra en la figura 15.

Figura 15. **Redes Elman**



Fuente: MEJÍA SÁNCHEZ, Juan Arturo. *Perceptrón Multicapa.* http://catarina.udlap.mx/u_dl_a/tales/documentos/lep/mejia_s_ja/capitulo3.pdf. Consulta: 5 de septiembre de 2019.

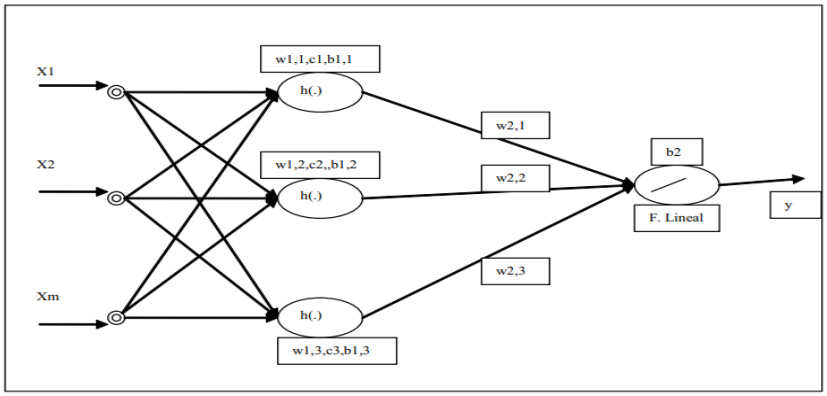
El cambio sobre la red FeedForward es que se crea una memoria, y puede hacer un análisis retroactivo, de esta manera los algoritmos de aprendizaje pueden mejorar el desempeño del sistema de detección de intrusos para conseguir mejores porcentajes generales al final de una sesión de entrenamiento con un conjunto de datos igual al de la red FeedForward.

Las redes Elman tiene varias características, entre ellas es mucho más rápido que una red FeedForward (FFNN), esto quiere decir que el número de iteraciones que se utilizan en el entrenamiento y posterior a ellos es menor. El porcentaje de efectividad de estas redes es menor a las FFNN, debido a que los caminos de retroalimentación generan un mejor comportamiento no-lineal durante el aprendizaje supervisado. Aunque el costo de las ventajas de esta red se ven afectadas en un costo alto de procesamiento. Es por eso que agregar una capa oculta debe ser algo muy bien justificado, porque definitivamente afectara el tiempo de procesamiento.

2.3.5. Redes neuronales de base radial

Son redes multicapa con conexión hacia adelante, que se caracterizan por que están formadas por una única capa oculta y cada neurona posee un carácter local, en el sentido de que cada neurona oculta de la red se activa en una región diferente del espacio de patrones de entrada. Este carácter local está dado por el uso de las funciones de base radial, generalmente funciones gausiana, como funciones de activación. Las neuronas de la capa de salida de las redes de base racial realizan una combinación lineal de las activaciones de las neuronas ocultas. Si la red tiene ‘p’ neuronas en la capa de entrada, ‘m’ neuronas en la capa oculta y ‘r’ neuronas en la capa de salida, las activaciones de las neuronas de salida para el patrón de entrada ‘n’, en la figura 16 se muestra su estructura.

Figura 16. **Arquitectura de una red neuronal de base radial**



Fuente: Redalyc. *Redes neuronales de base radial aplicadas a la mejora de calidad*. https://www.redalyc.org/pdf/816/81619829009.pdf. Consulta: 5 de septiembre de 2019.

El aprendizaje de estas redes es hibrido, la primera fase es no supervisada y la segunda supervisada.

Características:

- Capa de entrada: en la primera capa se tiene como entradas el error, la derivada del error y una constante, reciben las señales del exterior, no realizan ningún preprocesado.

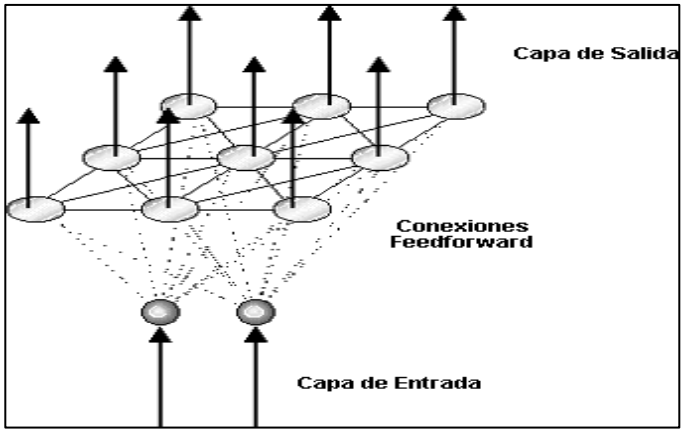
- Capa oculta: la capa oculta está formada por neuronas con funciones de base radial, reciben las señales de la capa de entrada y realizan una transformación local y no lineal sobre dichas señales.

- Capa de salida: se realiza una combinación lineal de las activaciones de las neuronas de la capa oculta y actúa como salida de la red.

2.3.6. Mapas autoorganizados (SOM)

Es un modelo neuronal inspirado en el cerebro, semejando a las neuronas detectoras de rasgos que se encuentra topológicamente ordenadas en el cerebro humano. Desarrollado por el finlandes Teuvo Kohonen. Este modelo ha demostrado ser útil para resolver problemas reales que incluyen tareas de clasificación, reducción de dimensiones y extracción de rasgos. Su utilidad más importante se relaciona con la clasificación de información o el agrupamiento de patrones por tipos o clases. Este modelo utiliza dos capas de neuronas una de entrada y una de salida. Las capas de la primera capa se limitan a recoger y canalizar la información, la segunda capa está conectada a la primera a través de pesos sinápticos y realiza la tarea más importante, una proyección no lineal del espacio multidimensional de entrada, manteniendo las características esenciales de estos datos en forma de relaciones de vecindad, como se observa en la figura 17.

Figura 17. **Arquitectura red neuronal autoorganizado**



Fuente. Unicen. *Mapas aotoorganizados*. http://www.exa.unicen.edu.ar/catedras/dmining/clases/clase_9.ppt. Consulta: 5 de septiembre de 2019.

El mapa autoorganizado está formado por una matriz rectangular de neuronas, de modo que las relaciones entre los patrones de entrada son mucho más fácilmente visibles en forma de relaciones de vecindad. Cada neurona sintoniza o aprende por si misma a reconocer un determinado tipo de patrón de entrada.

Los diferentes tipos de redes neuronales ofrecen diferentes características que las hacen ser la mejor opción para apoyar a los SDI, en realidad todas las opciones son viables, algunas con mejor desempeño que otras, incluso más fáciles de implementar, ese análisis es el que se realiza en el siguiente capítulo.

3. SELECCIÓN E IMPLEMENTACIÓN DE UNA RED NEURONAL Y UN SISTEMA DE DETECCION DE INTRUSOS

Uno de los objetivos de la esta investigación es demostrar cuál de las técnicas de inteligencia artificial presentadas, es la más eficiente de implementar como apoyo a los sistemas de detección de intrusos, de las diferentes técnicas presentadas se define que la mejor es la de redes neuronales; esto por la capacidad que tienen algunos tipos de redes para aprender y poder predecir posibles ataques, por lo fácil de implementar.

Aunque el proceso de entrenamiento puede llevar algún trabajo, no es algo significativo como para optar por otra solución, la tendencia a buscar el apoyo en las redes neuronales está tomando mayor relevancia conforme esta técnica está evolucionando, compañías como Google dedican millones de dólares en investigación para hacer cada vez más eficientes el funcionamiento de las redes neuronales, y que su parecido con el cerebro humano sea cada vez más cercano a la realidad.

Luego de analizar la estructura, utilización y topología de las redes neuronales, queda realizar un análisis sobre las ventajas y desventajas de cada una de ellas para elegir la o las opciones viables a la hora de seleccionar una herramienta como apoyo a los sistemas para la detección de intrusos (SDI).

Tabla V. **Comparativo redes neuronales ventaja vs desventaja**

Red neuronal	Ventajas	Desventaja
Perceptron	Tipo de aprendizaje supervisado, permite que sea muy útil para el tema de detección de intrusos	Se deben agregar varias capas e ir validando, además la selección y búsqueda de la pendiente 0 es un proceso de mucho entrenamiento.
Recurrentes	Por su capacidad de predicción de eventos significativos (por ejemplo ataques a la red) basadas en entradas anteriores al sistema, Mantienen una memoria en la red.	Posee muchos parámetros configurables y esto crea muchos problemas a la hora de hacer finetunning.
FeedForward	Tienen forma sencilla de implementar. Tiene un correcto balance entre costo-velocidad y costo-exactitud. Los modelos más rápidos para ejecutar.	No retiene información de eventos pasados, como ayuda para predecir eventos futuros. No se ha podido desarrollar un algoritmo confiable y lo suficientemente rápido por lo que se vuelve muy lento para entregar. Solo se utiliza con altas velocidades de ejecución, para que los altos tiempos de entrenamiento no sean un problema.
Elman	Son más eficientes para implementar en sistemas de detección de intrusos. Tiene mayor velocidad. Disminuye los falsos positivos. No requieren tanto tiempo de entrenamiento.	El tiempo de procesamiento es mayor. Cuando la cantidad de neuronas es mayor, el procesamiento se hace realmente lento. No es eficiente como las redes neuronales recurrentes.
Base radial	Permite realizar procedimientos constructivos, permite determinar la estructura óptima de una red neuronal.	Es difícil de implementar, por la cantidad de neuronas que utiliza no tiene un buen comportamiento.
Autoorganizados	Su topología ayuda a detectar rasgos. Realiza predicciones del futuro con pocas neuronas.	Las formas te entrenamiento supervisado y no supervisado hace complicada su implementación.

Fuente: elaboración propia.

3.1. Selección del software a utilizar para implementar la red neuronal

En la actualidad el software que implementa redes neuronales es muy amplio, principalmente se ha desarrollado para estudio, en esta investigación se analizaran 3 diferentes software potentes que se pueden utilizar en los SDI. Una de las ventajas es que la mayoría son de software libre, las tres opciones son:

- Tiberius
- Keras
- TensorFlow

Para determinar cuál es el óptimo para el estudio que se realiza, se realiza una breve descripción de cada uno, que muetra principal énfasis en sus fortalezas.

3.1.1. Tiberius

Software desarrollado por Phil Brierley y Anderw Lewis de NeuSolutions, este software permite realizar varios modelos de IA: redes neuronales, árboles de decisiones, por mencionar algunas, una ventaja es que los resultados los almacena en archivos de Excel o base de datos Access para el análisis posterior.

Tiberius ayuda a encontrar relaciones entre conjuntos de datos, en realidad simula una red tipo multicapa perceptrón, es por eso la importancia que representa para este proyecto, porque lo hace ideal para problemas de clasificación y regresión donde se tiene una variable dependiente que es manejada por otras variables conocidas, pero sin saber cuál de todas estas tiene mayor influencia en ella, el software le indicara si existe o no relación entre

esas variables. Generalmente, es utilizado para simular modelos como las tasas de cambio de monedas, prevención de carga eléctrica, o la predicción de pérdida de clientes.

Es muy útil para simular modelos, además es fácil de instalar, requiere licencia y no hay mucha documentación, eso complica de alguna manera su implementación y está muy enlazado a Windows.

3.1.2. Keras

Keras es un *frame* work de alto nivel para el aprendizaje, escrito en Python, capaz de correr sobre otros *frame* works, como TensorFlow, CNTK o Theano. Está especialmente diseñada para posibilitar la experimentación en más o menos poco tiempo con redes de Aprendizaje Profundo. Sus mayores fortalezas son su ambiente amigable para el usuario, modular y extensible. En 2017 Google decidió ofrecer soporte a Keras y eso la ha fortalecido mucho. Es de código abierto y disponible en Github, existe mucha documentación disponible. Ofrece soporte para redes neuronales convolucionales y recurrentes.

3.1.3. TensorFlow

Tensorflow es una biblioteca de software de código abierto, que utiliza gráficos de flujo de datos, en el grafo los nodos representan operaciones matemáticas, mientras que los bordes de la gráfica representan las matrices de datos multidimensionales (tensores) comunicadas entre ellos.

Tensorflow es una gran plataforma para construir y entrenar redes neuronales, que permiten detectar y descifrar patrones y correlaciones,

análogos al aprendizaje y razonamiento usados por humanos. La arquitectura flexible de TensorFlow le permite implementar el cálculo a una o más CPU o GPU en equipos de escritorio, servidores o dispositivos móviles con una sola API. Fue desarrollado originalmente por el equipo de Gooble Brain Team, dentro del departamento de investigación de Machine Intelligence, el sistema es lo suficientemente general como para ser aplicable a una amplia variedad de otros dominios igualmente.

Tensorflow es una plataforma de código abierto para Machine Learning de Google, es la herramienta más utiliza en el mundo del Deep Learning. Parte del éxito es la cultura de Google 'código primero, código siempre'.

Tabla VI. **Comparativo de software para implementar redes neuronales**

	Tiberius	Keras	TensorFlow
Software libre	No	Sí	Sí
Código abierto	No	Sí	Sí
Soporta múltiples redes neuronales	Sí	Sí	Sí
Documentación	No	Sí	Sí
Disponibilidad de SO	No	Sí	Sí
Entrenamiento profundo	Sí	Sí	Sí

Fuente: elaboración propia.

3.2. Tareas requeridas en una red neuronal

Una vez se han analizado diferentes tipos de redes neuronales, se debe enumerar las tareas que deben cumplir cualquiera red neuronal para que se pueda utilizar en un SDI. Estas tareas son recopilación de información (fuente de datos), reducción de datos, análisis de comportamiento, información y respuesta.

3.2.1. Recopilación de información

Se puede clasificar por su localización, *host*, red, aplicación y objetivo.

- Monitores basados en máquina (recoger datos generados de un ordenador).

- Monitores basados en red (recoge paquetes de la red).

- Monitores basados en aplicaciones (registran la actividad de una aplicación).

- Monitores basados en objetivos (generan sus propios registros, usan funciones para detectar alteraciones de sus objetivos, y contrastan los resultados con las políticas, se utilizan en elementos que no puede ser monitoreados de otra forma)

- Monitores híbridos, esto si se combinar varias fuentes.

3.2.2. Reducción y análisis

Luego de realizar el proceso de recopilación de la información, se lleva a cabo el proceso de análisis y detección de ataques. En muchos casos la información se ordena cronológicamente, se clasifica, se evalúa de forma estadística en muchos casos o por otras técnicas, se reduce y se identifica mediante patrones o firmas de actividad relativos a distintos aspectos de seguridad.

Según el objetivo del motor de análisis se pueden definir dos tipos:

- Detección del mal uso ‘misuse’, es la técnica usa por la mayoría de los sistemas comerciales, y analizan la actividad del sistema buscando eventos que coincidan con un patrón predefinido o firma que describe el ataque, de esta manera se comparan firmas con la información recogida en búsqueda de coincidencias. Se lleva a cabo a partir de modelos de ataque bien definidos, que utilizan fallos conocidos del sistema. Es una detección directa.

- Detección de anomalías, en la que el análisis busca patrones anormales de actividad, se centra en identificar comportamientos inusuales, suelen construir perfiles que representan el comportamiento normal, mediante la recolección de datos en intervalos de operación normal, donde se manejan técnicas estadísticas que definen de forma aproximada cual es el comportamiento usual o normal, es decir, cuantifican el comportamiento normal del sistema, las técnicas incluyen:

    - Detección de umbrales
    - Medidas estadísticas
    - Redes Neuronales
    - Algoritmos genéticos
    - Modelos de sistema inmune

3.2.3. Respuesta

Una vez se detecte una anomalía existen dos caminos, la respuesta pasiva no genera ninguna alarma, se registra en alguna bitácora, la otra opción es una acción activa, generando alarmas y reacciona modificando el entorno,

otra forma es alimentar la base de datos de conocimiento para reentrenar la red neuronal y mantenerla actualizada.

Ya se ha visto los beneficios de utilizar redes neuronales en los SDI, ahora veamos algunas herramientas útiles que ya implementan redes neuronales dentro de su análisis.

3.3. Sistemas de detección de intrusos

A continuación, se describen algunos SDI disponibles en el mercado que implementan redes neuronales para fortalecer su desempeño.

3.3.1. Snort

Snort es un SDI en tiempo real desarrollado por Martin Roesch y disponible bajo GPL. Se puede ejecutar en máquinas UNIX y Windows. Es el número uno en sistemas de detección de intrusos. Actualmente contiene 1 600 reglas para el análisis de cadenas. Fue diseñado inicialmente para cumplir requerimientos de un SDI ligero, en algunos casos funcionando solamente como un sniffer, poco a poco ha ido incrementando sus funcionalidades.

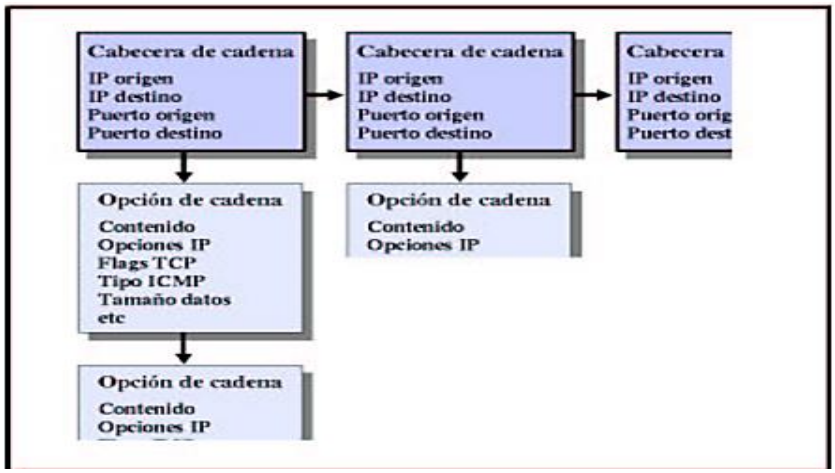
Aunque en Snort no es posible separar el componente de análisis y los sensores en máquinas distintas, si es posible ejecutar Snort en varias interfaces a la vez, cada uno monitoreando diferentes lugares de la red. La arquitectura de Snort se enfocó para ser eficiente, simple y flexible. Snort está formado por tres subsistemas, el decodificador de paquetes, las máquinas de detección y el subsistema de alerta y logs. Estos subsistemas corren sobre la librería libpcap,

la cual es portable y proporciona mecanismos de filtrado y captura de paquetes, como se detalla en la figura 18.

- Decodificador de paquetes: Soporta gran variedad de protocolos de capa de enlace bajo TCP/IP, tales como Ethernet, SLIP, PPP y ATM. Es el encargado de organizar los paquetes conforme van pasando por la pila de protocolos.

- Motor de detección: Snort mantiene sus reglas de detección en una lista enlazada bidimensional. La lista base se denomina ´Chain Header´ y la que deriva de esta se llama ´Chain Option´.

Figura 18. **Motor de detección de Snort**



Fuente: MIRA ALFARO, Emilio José. *Implantación de un sistema de detección de intrusos en la Universidad de Valencia.* p. 42.

Cuando llega un paquete al motor de detección, este busca en la lista ´Chain Header´ de izquierda a derecha la primera coincidencia. Después buscará por la lista ´Chain Option´si el paquete cumple las opciones

especificadas. Si aparece alguna coincidencia no seguirá buscando y se tomarán las acciones correspondientes. El motor de búsqueda tiene capacidad para inserción de módulos plug-in, que puede utilizarse para dos cosas:

- Permiten hacer búsquedas más complicadas en los paquetes o flujos TCP cuando la sintaxis de las reglas no lo permite. A estos módulos se les llama preprocesadores, y los más utilizados son el detector de escaneos de puertos, el desfragmentado, el reconstructor de flujo TCP.

- Permite definir modos de alerta y log no nativos a Snort, como bases de datos (MySQL, Oracle, PostgreSQL, MS—SQL), SNMP traps, CSV o salida de formato XML.

3.3.2. NetRanger – Cisco Systems

El Sistema de detección de intrusos de Cisco, conocido formalmente por Cisco NetRanger, es una solución para detectar, prevenir y reaccionar contra actividades no autorizadas a través de la red.

Es capaz de identificar ataques y prevenir accesos a recursos críticos del servidor antes de que ocurra cualquier transacción no autorizada. Esto lo hace integrando sus capacidades de respuesta con el resto de sus equipos.

Cisco Systems incluye dos componentes: sensor y director. Las ‘herramientas’ de red de alta velocidad analizan el contenido y el contexto de los paquetes individuales para determinar si se autoriza su tráfico. Si se detecta una intrusión, como por ejemplo un ataque de pruebas SATAN (System Administrators Tool for Analyzing Networks), un barrido de pings o si una persona que tiene acceso a información confidencial envía un documento que

contiene una palabra código de propiedad, los sensores de Cisco Secure IDS pueden detectar el uso incorrecto en tiempo real, enviar alarmas a una consola de gestión de Cisco Secure IDS director para la representación geográfica y sacar al agresor de la red.

3.3.3. Internet Security Systemns – RealSecure

RealSecure proporciona detección, prevención y respuestas a ataques y abusos originados en cualquier punto de la red. Entre las respuestas automáticas a actividades no autorizadas se incluyen el almacenar los eventos en una base de datos, bloquear una conexión, enviar un mail, suspender o deshabilitar una cuenta en un *host* o crear una alerta definida por el usuario.

El sensor de red rápidamente se ajusta a diferentes necesidades de red, incluyendo alertas específicas por usuario, sintonización de firmas de ataques y creación de firmas definidas por el usuario. Las firmas son actualizables automáticamente mediante aplicación X-Press Update. El sensor de red puede ser actualizado de una versión a otra posterior sin problema, asegurando así la última versión del producto. Proporciona detección de intrusiones en tiempo real. Tiene mecanismos de respuesta a comportamientos sospechosos en un segmento de red. Los drivers de captura de paquetes a alta velocidad funcionan sin causar el más mínimo impacto en la red. Están disponibles en full dúplex, multipuerto, y Gigabit.

Es un filtro que protege segmentos de red, incluyendo sistemas de producción críticos o conexiones. El tráfico que atraviesa el sistema Guard, es analizado en tiempo real en búsqueda de evidencia de ataques o abusos.

3.3.4. Shadow

Fue desarrollado como respuesta a los falsos positivos de un SDI anterior, NID. La idea era construir una interfaz rápida que funcionara bien en una DMZ caliente (una DMZ que sufre muchos ataques). La interfaz permitiría al análisis evaluar gran cantidad de información de red y decidir de qué eventos informar.

No es en tiempo real. Shadow almacena el tráfico de red en una base de datos y se ejecuta en procesos nocturnos. Los resultados deben esperar para ser analizados el día siguiente, esto coloca a Shadows con una gran desventaja ante otros SDI.

3.4. Consideraciones técnicas

Una vez realizada una descripción rápida de los diferentes sistemas de detección de intruso, en la tabla VI se muestra un análisis en el que se resaltaran las características técnicas más importantes de cada uno de los IDS´s estudiados. Con base en el cumplimiento de las características analizadas se seleccionara un software de código abierto para poder adaptar a las necesidades que se desean demostrar en este trabajo. También, se busca que exista documentación amplia para facilitar su implementación.

Tabla VII. **Comparativo de sistemas de detección de intrusos**

	NetRanger	Iternet Security Systems	Shadow	Snort
Código abierto	No	No	Sí	Sí
Software libre	No	No	Sí	Sí
Comunidad de desarrolladores	No	No	No	Sí
Facilidad de aumentar nuevos módulos	No	No	No	Sí
Popularidad	Sí	Sí	No	Sí
Documentación	Sí	No	No	Sí
Facilidad de instalación	Sí	Sí	No	Sí
Funcionalidad	Sí	Sí	Sí	Sí
Rendimiento	Sí	Sí	No	Sí
Facilidad de administración	Sí	No	No	Sí

Fuente: elaboración propia.

El software IDS que cumple con todas las características necesarias es Snort, es por eso que se elige este software como el SDI para implementar una red neuronal y fortalecer su eficiencia.

3.4.1. Preparación de un SDI con una red neuronal

Como se detalla anteriormente hay ciertas tareas requeridas para que una red neuronal pueda funcionar correctamente en un SDI. Antes del diseño de la red neuronal es importante analizar la forma en que se trabajaran los datos, los datos deben tener el formato necesario para que sea útil como información de entrada de una red neuronal, es la parte más importante para tomar en cuenta al momento de implementar una red neuronal en un SDI.

3.4.1.1. Conversión a formato decimal

Las cadenas de caracteres que se almacenan o se reciben por la herramienta que se utiliza para recibir la información, deben ser convertidas a decimal, de tal forma que se transforme en una nueva cadena de valores ASCCI correspondiente a la cadena original.

Ejemplo: La siguiente cadena representa un ataque de inyección de código ´@.cgi?@= %0a@´, se representan de la siguiente manera:

[64 46 99 103 105 63 64 61 37 48 97 64]

De esta manera todas las cadenas de caracteres pueden ser manipuladas como datos enteros que tengan un significado para la red neuronal.

3.4.1.2. Ventana deslizante

Ahora bien, estas cadenas tienen un tamaño variable, eso quiere decir que la red neuronal debería tener entradas variables, o limitar la red a un número fijo de entradas, esto definitivamente no sería útil, para esto se crea una ventana deslizante. Esta ventana deslizante convierte las cadenas de tamaño variable, en cadenas de tamaño fijo, la técnica es utilizar la longitud de la cadena más pequeña, por ejemplo:

120 33 45 64 32
23 34 45 67
88 90 14 57 10 13 45
24 15 12

La cadena más pequeña es de tres, por lo tanto se limita el resto de las cadenas; quedan de esta manera:

120 33 45
33 45 64
45 64 32
23 34 45
34 45 67
88 90 14
90 14 57
14 57 10
57 10 13
10 13 45
24 15 12

Se han generado cadenas fijas, al costo de tener más cadenas para entrenamiento, aunque este costo cuando se trata de redes neuronales es un beneficio, ya que le permite a la red más patrones de entrenamiento, esto le da, por lo tanto, la oportunidad de generalizar.

3.4.1.3. Conversión a formato binario

Finalmente las cadenas decimales de tamaño fijo, se deben convertir a un formato que pueda generar los cambios suficientes para que en el rango de entrada de una neurona sigmoidea genere mayor cambio posible. Las entradas de una neurona aceptan valores de números reales, sin embargo los cambios más significativos suceden cuando las entradas están entre 0 y 1.
Por ejemplo:

64 37 42

37 42 33

43 28 96

28 96 41

Serán transformadas a formato binario quedando de la siguiente manera:

01000000 00100101 00101010

00100101 00101010 00100001

00101011 00011100 01100000

00011100 01100000 00101001

4. DISEÑO Y ENTRENAMIENTO DE RED NEURONAL

Una vez definido el número de entradas de la red neuronal igual a 64 se puede definir también el número de salidas necesarias para presentar las cinco clasificaciones posibles de la red neuronal debe tener. Se define entonces el número de salidas en 5, tomando en cuenta que cada neurona sea destinada a activarse cuando el dato en la entrada de la red neuronal corresponda a un tipo de clasificación, y que las demás neuronas se encuentren desactivadas cuando el dato en la entrada no pertenezca a una de las cinco posibles clasificaciones de manera que cuando el dato de entrada pertenezca a una de estas clasificaciones, la neurona correspondiente a esta clasificación sea activada y las demás sean deshabilitadas.

Entonces la primera neurona de salida se activa cuando el dato a la entrada sea de tipo NORMAL, la segunda neurona deberá activarse cuando el dato a la entrada sea de tipo ATAQUE POR INYECCION, la tercera neurona deberá activarse cuando el dato de estrada sea de tipo ATAQUE POR MODIFICACION DE PATH, la cuarta neurona debe ser activada cuando el dato a la entrada sea de tipo ATAQUE POR INYECCION SQL, y la quinta neurona debe ser activada cuando el dato a la entrada corresponda a un ATAQUE POR XSS (Cross Site Script). Mientras una neurona esta activa o en valor alto o en uno, las demás deberán estar en valor bajo, o cero.

Durante el presente trabajo se ha analizado diferentes técnicas y herramientas para evitar ataques, se eligen las redes neuronales como la mejor técnica de inteligencia artificial, y los sistemas de detección de intrusos como la herramienta de seguridad informática óptima para combinar y ser más eficientes

en la protección de la información. Para culminar se muestra un cuadro de ventajas y desventajas de la implementación de redes neuronales y sistemas de detección de intrusos.

Tabla VIII. **Ventajas y desventajas de utilizar redes neuronales en los SDI**

Ventajas	Desventajas
La red aprenderá de manera autónoma a partir de los ejemplos, lo que disminuirá el tiempo y esfuerzo del proceso de "finetunning" que se necesita para su correcto funcionamiento.	Variabilidad del sistema.
La red neuronal artificial se puede adaptar a nuevos comportamientos, la cual hace el sistema mucho más flexible a variaciones y modificaciones de los métodos de intrusión actualmente conocidos "Base de conocimiento".	Posee muchos parámetros configurables y esto crea muchos problemas a la hora de hacer finetunning.
Disminuir la cantidad de falsos positivos y falsos negativos que presentan los sistemas IDS basados en firmas.	La representación de los datos para que el sistema opere con ellos, y su adquisición por medio de los elementos de E-box.
La red infiere ataques que no aprendió, y puede adaptarse a los que el administrador de red cual lo convierte en un sistema, de cierta manera, heurístico.	Posible pérdida de información discriminante valiosa al hacer conversión de datos.
El éxito de un sistema de redes neuronales se basa en el entrenamiento y en la selección de un universo adecuado y familiar a la red donde se planea utilizar el sistema.	Poco o nula trazabilidad de los ataques.

Fuente: elaboración propia.

Luego de analizar las ventajas y desventajas de la combinación de una red neuronal y SDI, se puede concluir que dicha combinación puede aumentar considerablemente la efectividad de los ataques a organizaciones y personas individuales, pero los profesionales en tecnología tienen el conocimiento necesario para implementar este tipo de tecnología.

4.1. Integración de la red neuronal artificial en el sistema de detección de intrusos

Luego de seleccionar un SDI y una red neuronal que pueden enlazarse para lograr mayor eficiencia en la detección de intrusos, se detallan a continuación algunos de los pasos para realizar la integración de estas tecnologías

4.1.1. Configuración de Snort

Para integrar una red neuronal artificial a un SDI, primero se debe configurar el SDI, en este caso SNORT se deben seguir algunos pasos, el primero es modificar algunas características de Snort, en la figura 19 se muestra la estructura de módulos de Snort, los módulos que se pueden modificar son:

- Módulo de captura de tráfico
- Decodificador
- Preprocesadores
- Motor de detección
- Archivo de reglas

4.1.2. Módulo de captura de tráfico

Es el modulo encargado de capturar paquetes y analizar los diferentes protocolos desde la capa de enlace de datos hasta la de transporte, utiliza las librerías LibPcap y WinPcap.

4.1.3. Decodificador

Es el encargado de formar las estructuras de datos con los paquetes capturados e identificar los diferentes protocolos.

4.1.4. Preprocesadores

Los preprocesadores pueden desfragmentar paquetes, decodificar URLs HTTP, reensamblar streams TCP, entre otros. Una vez reordenados los paquetes, al pasar por el motor de detección se aplicarán las reglas de patrones de ataques, virus, información, entre otros.

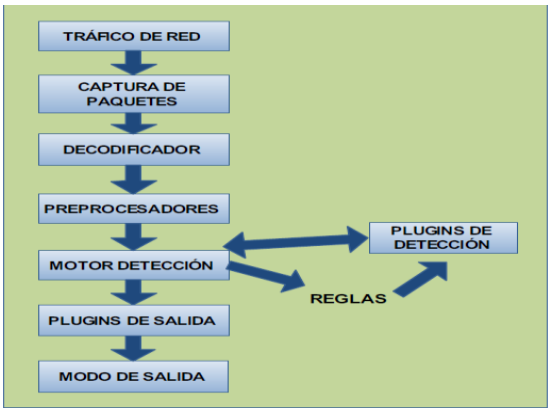
4.1.5. Motor de detección

Es el módulo encargado de analizar y detectar si existe alguna actividad de intrusión en los paquetes según las reglas definidas. En función de esto se determina si continúa con el logueo en archivo o base de datos, o se envía una alerta de administrador. El tiempo de análisis de los paquetes depende básicamente de la cantidad de reglas analizadas.

4.1.6. Archivo de reglas

Define el conjunto de reglas con base enlos cuales se analizará los paquetes detectados, este módulo es al que se le agregarán las reglas que genere la red neuronal cuando detecte una posible intrusión.

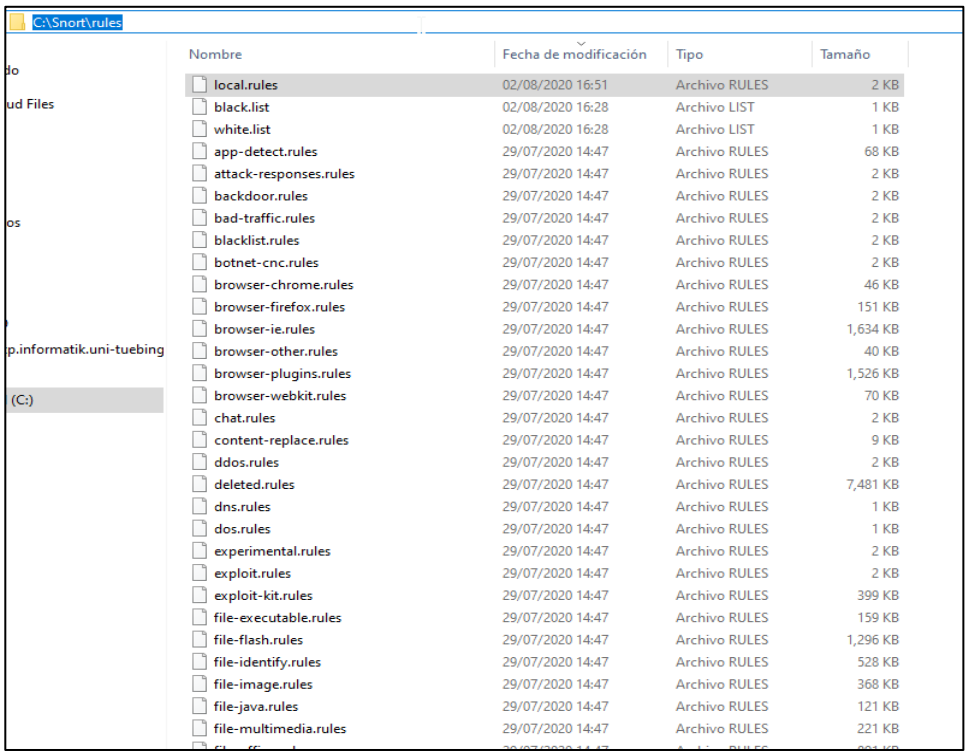
Figura 19. **Módulos de Snort**



Fuente: elaboración propia, con Documentación Snort.

Por defecto se instalan las reglas de la comunicad Snort y se configura el archivo snort.conf, el paquete descargado con las reglas se debe reemplazar en la capeta Rules de la carpeta de instalación de Snort, como se muestra en la figura 20.

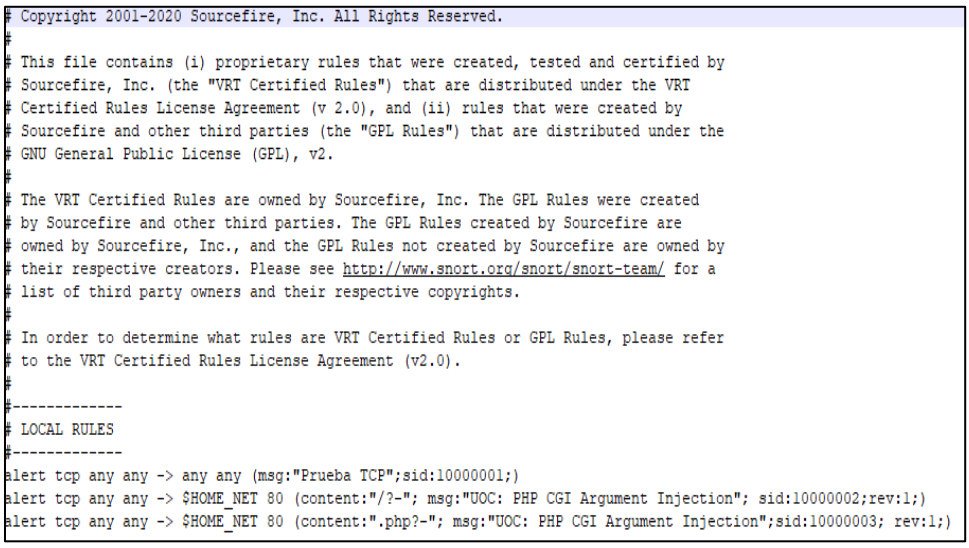
Figura 20. **Copia de reglas a la carpeta de instalación de Snort**



Fuente: elaboración propia.

Además, se debe configurar el archivo local.rules en donde se especifican las reglas que se utilizaran cuando Snort este ejecutándose (figura 21).

Figura 21. **Configuración de archivo local.rules**

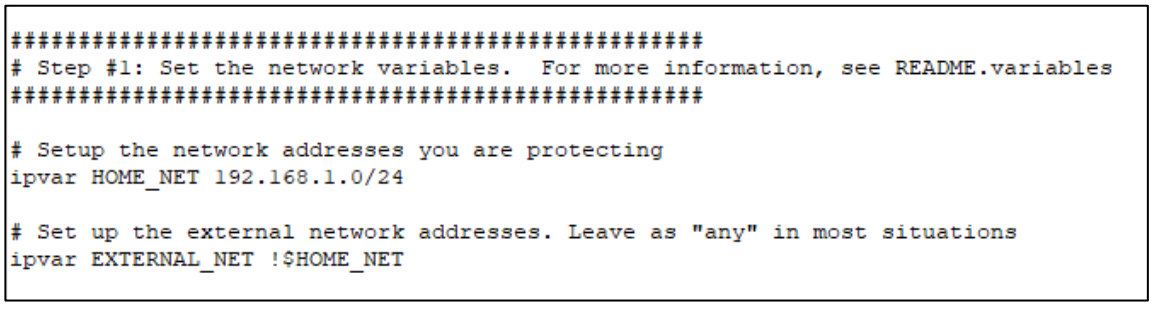


```
# Copyright 2001-2020 Sourcefire, Inc. All Rights Reserved.
#
# This file contains (i) proprietary rules that were created, tested and certified by
# Sourcefire, Inc. (the "VRT Certified Rules") that are distributed under the VRT
# Certified Rules License Agreement (v 2.0), and (ii) rules that were created by
# Sourcefire and other third parties (the "GPL Rules") that are distributed under the
# GNU General Public License (GPL), v2.
#
# The VRT Certified Rules are owned by Sourcefire, Inc. The GPL Rules were created
# by Sourcefire and other third parties. The GPL Rules created by Sourcefire are
# owned by Sourcefire, Inc., and the GPL Rules not created by Sourcefire are owned by
# their respective creators. Please see http://www.snort.org/snort/snort-team/ for a
# list of third party owners and their respective copyrights.
#
# In order to determine what rules are VRT Certified Rules or GPL Rules, please refer
# to the VRT Certified Rules License Agreement (v2.0).
#
#-------------
# LOCAL RULES
#-------------
alert tcp any any -> any any (msg:"Prueba TCP";sid:10000001;)
alert tcp any any -> $HOME_NET 80 (content:"/?-"; msg:"UOC: PHP CGI Argument Injection"; sid:10000002;rev:1;)
alert tcp any any -> $HOME_NET 80 (content:".php?-"; msg:"UOC: PHP CGI Argument Injection";sid:10000003; rev:1;)
```

Fuente: elaboración propia.

El siguiente paso es configurar Snort, se deben aplicar algunos cambios en el archivo snort.conf, primero se cambia la variable HOME_NET; luego, se coloca la IP del equipo que ejecutará Snort, y en la EXTERNAL_NET, que niega la variable HOME_NET, como se muestra en la figura 22.

Figura 22. **Configuración archivo Snort.conf**



```
###################################################
# Step #1: Set the network variables.  For more information, see README.variables
###################################################

# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.1.0/24

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET !$HOME_NET
```

Fuente: elaboración propia.

El siguiente paso es descomentar todas las reglas que se encuentra al final del archivo, estas reglas han sido cargadas previamente en la carpeta de reglas, en el directorio Snort, como se muestra en la figura 23.

Figura 23. **Configuración archivo snort.conf**

```
##################################################
# Step #7: Customize your rule set
# For more information, see Snort Manual, Writing Snort Rules
#
# NOTE: All categories are enabled in this conf file
##################################################

# site specific rules
include $RULE_PATH\local.rules

include $RULE_PATH\app-detect.rules
include $RULE_PATH\attack-responses.rules
include $RULE_PATH\backdoor.rules
include $RULE_PATH\bad-traffic.rules
include $RULE_PATH\blacklist.rules
include $RULE_PATH\botnet-cnc.rules
include $RULE_PATH\browser-chrome.rules
include $RULE_PATH\browser-firefox.rules
include $RULE_PATH\browser-ie.rules
include $RULE_PATH\browser-other.rules
include $RULE_PATH\browser-plugins.rules
include $RULE_PATH\browser-webkit.rules
include $RULE_PATH\chat.rules
include $RULE_PATH\content-replace.rules
include $RULE_PATH\ddos.rules
include $RULE_PATH\dns.rules
include $RULE_PATH\dos.rules
include $RULE_PATH\experimental.rules
include $RULE_PATH\exploit-kit.rules
include $RULE_PATH\exploit.rules
include $RULE_PATH\file-executable.rules
include $RULE_PATH\file-flash.rules
include $RULE_PATH\file-identify.rules
include $RULE_PATH\file-image.rules
include $RULE_PATH\file-multimedia.rules
include $RULE_PATH\file-office.rules
include $RULE_PATH\file-other.rules
include $RULE_PATH\file-pdf.rules
include $RULE_PATH\finger.rules
```

Fuente: elaboración propia.

Con esto se ha configurado Snort y está listo para utilizarse y recopilar información del escaneo de la red. Los resultados de las anomalías detectadas se agregan en el archivo alert.ids, en la carpeta log de Snort.

En la carpeta log también se guardan las bitácoras de Snort como Sniffer, estos archivos guardan todas las cadenas del tráfico de la red, son estos los que serán analizados.

Para ejecutar Snort en modo NIDS se ejecuta la siguiente línea de comandos:

Snort -dev -c c:\Snort\etc\snort.conf -i 2 -l c:\Snort\log

La descripción de los parametros es:

- d: Visualizar los campos de datos que pasan por la interface de red
- e: Muestra información detallada
- v: Inicia Snort en modo Sniffer, muestra la cabecera TCP/IP
- c: Archivo que utilizara Snort como fichero de configuración
- l: directorio donde guardar las alertas y logs
- i: interfaz que se monitoreara, en este caso se coloca 2 que es como ha detectado snort el equipo o se puede colocar la IP.

4.1.7. Preparación de la red neuronal

Una vez configurado el SDI y siguiendo los pases enumerados la sección 3.4 de este trabajo para poder configurar la red neuronal es necesario recopilar la información con la que se entrenara la neurona. Para realizar esta implementación se utilizan las siguientes herramientas:

- Snort
- Visual Studio Code
- Zenmap

- Python
- Tcpdump (se utiliza para leer los archivos binarios generados por snort)

4.1.8. Recopilar información

Para recopilar información se construyen perfiles generados a partir del análisis de asociación de patrones; estos perfiles representan el comportamiento normal de los usuarios, *hosts* o conexión de red. Para obtener esta información existen diferentes fuentes, no existe una fuente específica, ni cantidad de datos reglamentaria para poder realizar esta labor, aunque para que el entrenamiento sea efectivo y se logre el objetivo se recomienda que la cantidad de datos sea considerable.

Para este ejercicio se utilizara el dataset KDD-99, que es un conjunto de datos de ataques con información de ataques 'malos', llamados intrusiones o ataques y conexiones normales 'buenas'. Esta base de datos contiene un conjunto estándar de datos a auditar, que incluye una amplia variedad de intrusiones simuladas en un entorno de red militar, luego se obtendrá información del propio SDI instalado que monitorea la red interna, en este caso SNORT. De forma general la información que se utilizara para el entrenamiento debe tener ciertas características:

- Que sean ataques que ocurren con frecuencia

- Que por sus características este permita identificar, obtener y codificar la información para las entradas de red neuronal artificial.

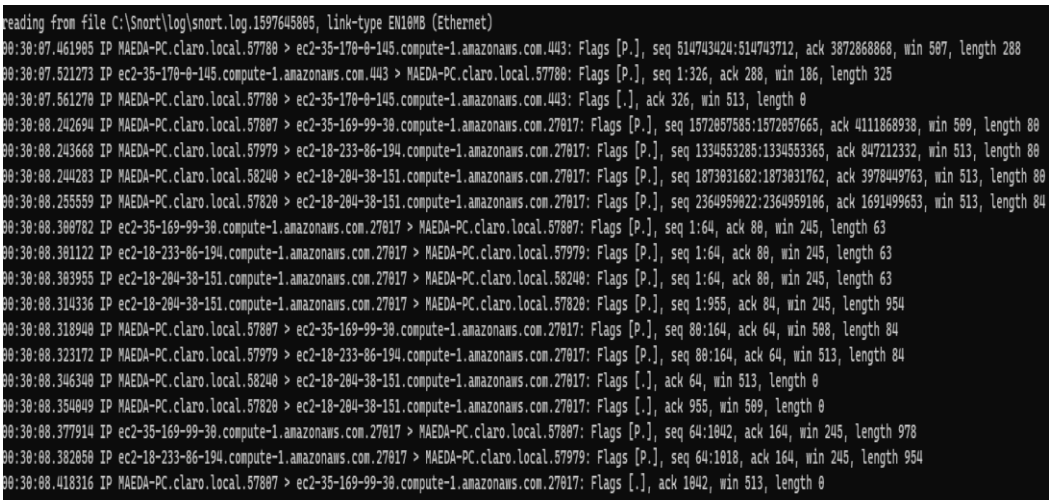
- Que el SDI tenga baja eficiencia en la detección de este tipo de ataque, es decir que presente fasos positivos y falsos negativos, Snort cumple con estas características.

Otra fuente de información es los archivos generados por Snort con la información del tráfico de red, para eso se utiliza el siguiente comando:

tcpdump -r C:\Snort\log\snort.log.1597645805

Y retorna la información como se muestra en la figura 24.

Figura 24. **Muestra archivo .log generado por Snort**



Fuente: elaboración propia.

En la tabla VIII se enumeran algunos de los campos que contiene el data set KDD-99 que serán utilizados.

Tabla IX. **Atributos básicos de conexiones**

Atributo	Descripción	Tipo
Duration	Longitud de la conexión	Continuo
Protocol_type	Tipo de protocolo (tcp)	Discreto
Service	Tipo de servicio de destino (HTTP, Telnet, SMTP)	Discreto
Src_bytes	Número de bytes de datos de fuente a destino	Continuo
Dst_bytes	Número de bytes de datos de destino a la fuente	Continuo
Flag	Estado de la conexión (SF, SI, REJ…)	Discreto
Land	1 si la conexión corresponde mismo *host*/puerto; 0 de otro nodo	Discreto
Wrong_fragment	Número de fragmentos erróneos	Continuo
Urgent	Número de paquetes urgentes	Continuo

Fuente: elaboración propia.

Como lenguaje de programación se utilizará Python, y se obtendrá el data set kddcup.data_10_percent, como se ve en la figura 25 una muestra del código fuente para la descarga del dataset.

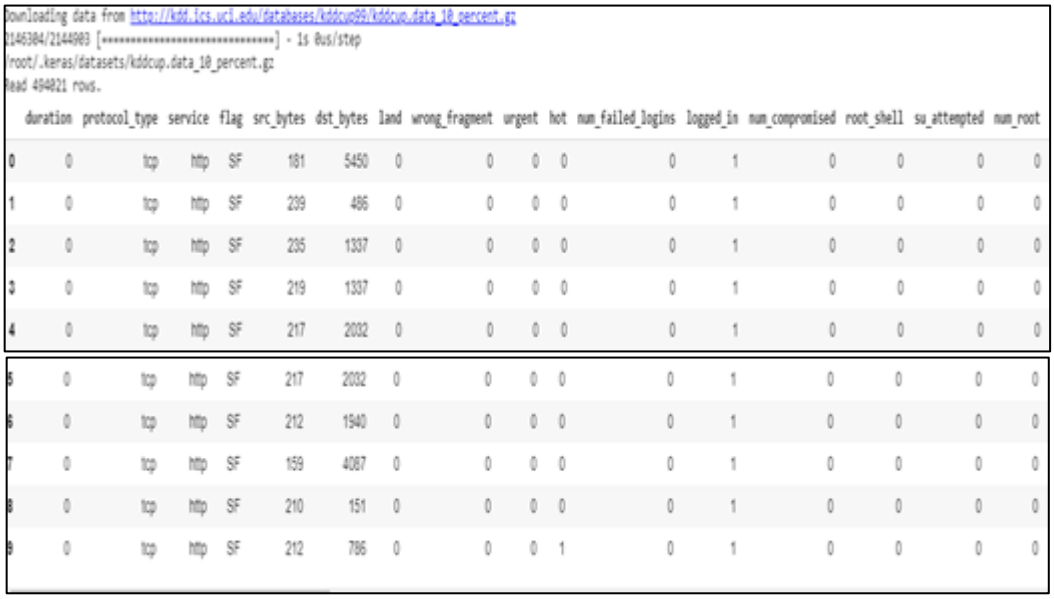
Figura 25. **Código fuente para descargar DataSet**

```
# -*- coding: utf-8 -*-
"""Untitled0.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1II9QHENqExYSm5isyolKL4vqagi9pVw5
"""

import pandas as pd
from tensorflow.keras.utils import get_file

try:
    path = get_file('kddcup.data_10_percent.gz', origin='http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data_10_percent.gz')
except:
    print('Error downloading')
    raise

print(path)

# Este es un archivo CSV sin encabezado
# Se descarga la información de: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
df = pd.read_csv(path, header=None)
```

Fuente: elaboración propia.

En las figuras 26 y 27 se muestra el resultado de la descarga de información cargada a un dataset de Python, para finalizar con el proceso de recopilación de información y pasar a la reducción y análisis.

Figura 26. **Salida de la carga de datos**



```
Downloading data from http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data_10_percent.gz
2146304/2144903 [==============================] - 1s 0us/step
/root/.keras/datasets/kddcup.data_10_percent.gz
Read 494021 rows.
```

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	root_shell	su_attempted	num_root
0	0	tcp	http	SF	181	5450	0	0	0	0	0	1	0	0	0	0
1	0	tcp	http	SF	239	486	0	0	0	0	0	1	0	0	0	0
2	0	tcp	http	SF	235	1337	0	0	0	0	0	1	0	0	0	0
3	0	tcp	http	SF	219	1337	0	0	0	0	0	1	0	0	0	0
4	0	tcp	http	SF	217	2032	0	0	0	0	0	1	0	0	0	0
5	0	tcp	http	SF	217	2032	0	0	0	0	0	1	0	0	0	0
6	0	tcp	http	SF	212	1940	0	0	0	0	0	1	0	0	0	0
7	0	tcp	http	SF	159	4087	0	0	0	0	0	1	0	0	0	0
8	0	tcp	http	SF	210	151	0	0	0	0	0	1	0	0	0	0
9	0	tcp	http	SF	212	786	0	0	0	1	0	1	0	0	0	0

Fuente: elaboración propia.

Figura 27. **Salida de la carga de datos**

dst_host_diff_srv_rate	dst_host_same_src_port_rate	dst_host_srv_diff_host_rate	dst_host_serror_rate	dst_host_srv_serror_rate	dst_host_rerror_rate	dst_host_srv_rerror_rate	outcome
0.0	0.11	0.00	0.0	0.0	0.0	0.0	normal.
0.0	0.05	0.00	0.0	0.0	0.0	0.0	normal.
0.0	0.03	0.00	0.0	0.0	0.0	0.0	normal.
0.0	0.03	0.00	0.0	0.0	0.0	0.0	normal.
0.0	0.02	0.00	0.0	0.0	0.0	0.0	normal.
0.0	0.02	0.00	0.0	0.0	0.0	0.0	normal.
0.0	1.00	0.04	0.0	0.0	0.0	0.0	normal.
0.0	0.09	0.04	0.0	0.0	0.0	0.0	normal.
0.0	0.12	0.04	0.0	0.0	0.0	0.0	normal.
0.0	0.12	0.05	0.0	0.0	0.0	0.0	normal.

Fuente: elaboración propia.

4.1.9. Reducción y análisis

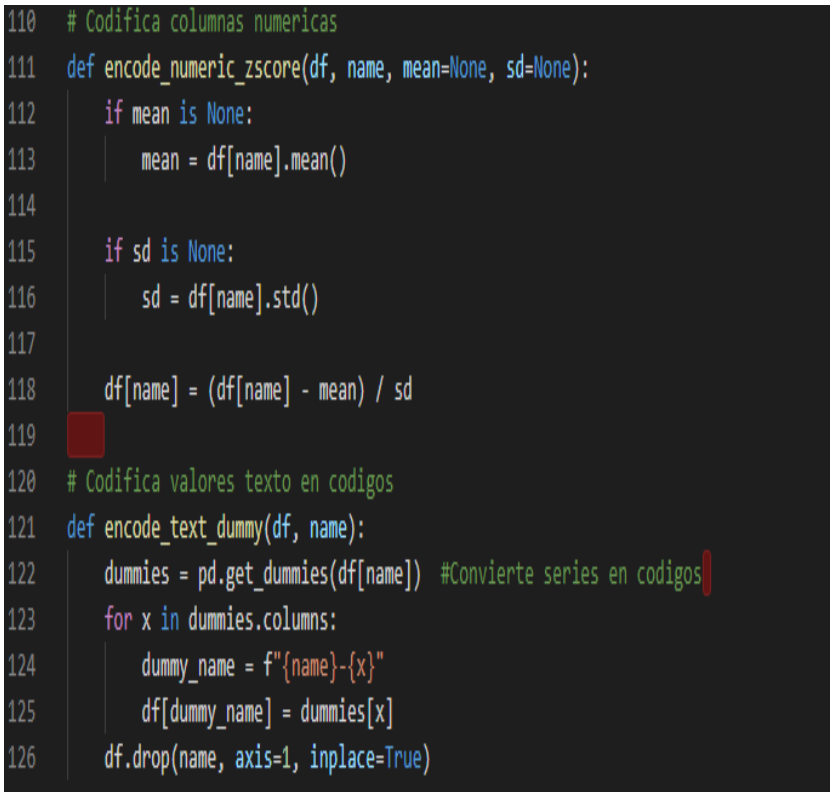
En la sección de reducción y análisis lo que se hace es clasificar la información cargada, para ello se crean dos funciones que permite analizar columna a columna del data set, para obtener catálogos únicos de la data cargada. La información se debe transformar en un dataset que la red neuronal reconozca, y esto son valores numéricos. En las figuras 28 y 29 se muestra el código para hacer la reducción y análisis de los datos.

Figura 28. **Código para análisis de dataset**

```
78  import pandas as pd
79  import os
80  import numpy as np
81  from sklearn import metrics
82  from scipy.stats import zscore
83
84  def expand_categories(values):
85      result = []
86      s = values.value_counts()
87      t = float(len(values))
88      for v in s.index:
89          result.append("{}:{}%".format(v,round(100*(s[v]/t),2)))
90      return "[{}]".format(",".join(result))
91
92  def analyze(df):
93      print()
94      cols = df.columns.values
95      total = float(len(df))
96
97      print("{} rows".format(int(total)))
98      for col in cols:
99          uniques = df[col].unique()
100         unique_count = len(uniques)
101         if unique_count>100:
102             print("** {}:{} ({}%)".format(col,unique_count,int(((unique_count)/total)*100)))
103         else:
104             print("** {}:{}".format(col,expand_categories(df[col])))
105             expand_categories(df[col])
106
107 # Analyze KDD-99
108 analyze(df)
```

Fuente: elaboración propia.

Figura 29. **Funciones para codificar columnas numéricas y texto**

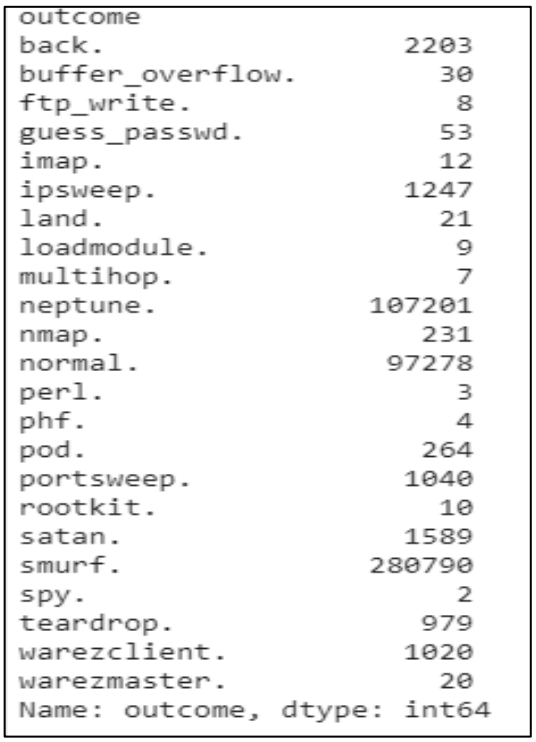


```
110 # Codifica columnas numericas
111 def encode_numeric_zscore(df, name, mean=None, sd=None):
112     if mean is None:
113         mean = df[name].mean()
114 
115     if sd is None:
116         sd = df[name].std()
117 
118     df[name] = (df[name] - mean) / sd
119 
120 # Codifica valores texto en codigos
121 def encode_text_dummy(df, name):
122     dummies = pd.get_dummies(df[name])  #Convierte series en codigos
123     for x in dummies.columns:
124         dummy_name = f"{name}-{x}"
125         df[dummy_name] = dummies[x]
126     df.drop(name, axis=1, inplace=True)
```

Fuente: elaboración propia.

Una vez realizada la codificación de las columnas se ve la clasificación de los datos; en la figura 30 se ve la cantidad de registros que hay para cada estado posible de información en la red. Con esta información se entrenara la neurona para que pueda reconocer entre una entrada normal y un ataque.

Figura 30. **Clasificación del dataset para entrenamiento**



```
outcome
back.                  2203
buffer_overflow.         30
ftp_write.                8
guess_passwd.            53
imap.                    12
ipsweep.               1247
land.                    21
loadmodule.               9
multihop.                 7
neptune.             107201
nmap.                   231
normal.               97278
perl.                     3
phf.                      4
pod.                    264
portsweep.             1040
rootkit.                 10
satan.                 1589
smurf.               280790
spy.                      2
teardrop.               979
warezclient.           1020
warezmaster.             20
Name: outcome, dtype: int64
```

Fuente: elaboración propia.

Ya que se haya clasificado el data set, se procede hacer el entrenamiento de la neurona, del data set que se ha prepara se utilizara 75 % para el entrenamiento de la neurona y el 25 % para validar si la neurona ha sido entrenada correctamente, que porcentaje de efectividad ha logrado. En la figura 31 se muestra la porción de código que prepara la red neuronal y el data set para el entrenamiento, se utilizara una red multicapa, con 3 capas intermedias, que utilizara como funciona de activación una RELU, las capas intermedias tendrá un total de 70 iteraciones para el entrenamiento.

Figura 31. **Código fuente, creación de red neuronal**

```
import pandas as pd
import io
import requests
import numpy as np
import os
from sklearn.model_selection import train_test_split
from sklearn import metrics
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras.callbacks import EarlyStopping

# Crea el data set para entrenamiento 25% del total del data set
x_train, x_test, y_train, y_test = train_test_split(
    x, y, test_size=0.25, random_state=42)
print(x_test)

# Crea una red Neuronal, con una capa de entrada, 3 intermedias
#que utilizara una funcion de activacion Relu, y una capa de salida
model = Sequential()
model.add(Dense(10, input_dim=x.shape[1], activation='relu'))
model.add(Dense(50, input_dim=x.shape[1], activation='relu'))
model.add(Dense(10, input_dim=x.shape[1], activation='relu'))
model.add(Dense(1, kernel_initializer='normal'))
model.add(Dense(y.shape[1],activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
monitor = EarlyStopping(monitor='val_loss', min_delta=1e-3,
                        patience=5, verbose=1, mode='auto')
model.fit(x_train,y_train,validation_data=(x_test,y_test),
          callbacks=[monitor],verbose=2,epochs=1000)
```

Fuente: elaboración propia.

Luego de realizar el entrenamiento (figura 32), se evalúa la presión del entrenamiento con el 25 % de la data almacenada en el data set x_test (figura 33), retorna un 0.99589 de efectividad, esto quiere decir que puede reconocer entre una línea de conexión NORMAL y un ataque, o un posible ataque, basado en las columnas con las que se ha entrenado la red. La red programada se coloca en el módulo de preprocesadores de Snort para que cada paquete que ingrese sea analizado.

Figura 32. **Proceso de entrenamiento de red neuronal**

```
Epoch 1/1000
11579/11579 - 16s - loss: 0.1084 - val_loss: 0.0391
Epoch 2/1000
11579/11579 - 16s - loss: 0.0347 - val_loss: 0.0295
Epoch 3/1000
11579/11579 - 14s - loss: 0.0286 - val_loss: 0.0253
Epoch 4/1000
11579/11579 - 14s - loss: 0.0262 - val_loss: 0.0235
Epoch 5/1000
11579/11579 - 14s - loss: 0.0317 - val_loss: 0.0248
Epoch 6/1000
11579/11579 - 14s - loss: 0.0234 - val_loss: 0.0234
Epoch 7/1000
11579/11579 - 14s - loss: 0.0281 - val_loss: 0.0232
Epoch 8/1000
11579/11579 - 14s - loss: 0.0222 - val_loss: 0.0205
Epoch 9/1000
11579/11579 - 14s - loss: 0.0235 - val_loss: 0.0202
Epoch 10/1000
11579/11579 - 14s - loss: 0.0215 - val_loss: 0.0208
Epoch 11/1000
11579/11579 - 14s - loss: 0.0212 - val_loss: 0.0223
Epoch 12/1000
11579/11579 - 14s - loss: 0.0215 - val_loss: 0.0195
Epoch 13/1000
11579/11579 - 14s - loss: 0.0202 - val_loss: 0.0197
Epoch 14/1000
11579/11579 - 15s - loss: 0.0212 - val_loss: 0.0190
Epoch 15/1000
11579/11579 - 14s - loss: 0.0204 - val_loss: 0.0185
Epoch 16/1000
11579/11579 - 14s - loss: 0.0192 - val_loss: 0.0198
Epoch 17/1000
11579/11579 - 14s - loss: 0.0204 - val_loss: 0.0206
```

Fuente: elaboración propia.

Figura 33. **Medición de precisión**

```
pred = model.predict(x_test)
pred = np.argmax(pred,axis=1)
y_eval = np.argmax(y_test,axis=1)
score = metrics.accuracy_score(y_eval, pred)
print("Efectividad: {}".format(score))
```

```
Efectividad 0.9958949362783994
```

Fuente: elaboración propia.

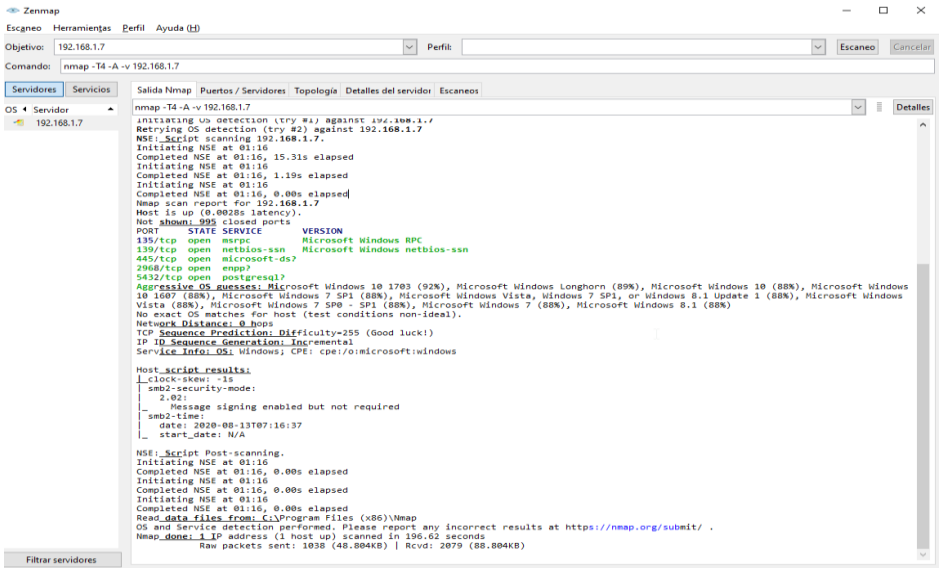
4.1.10. Respuesta

Cuando el preprocesador adaptado con la red neuronal devuelve información al motor de detección y se valida con el archivo de reglas devuelve la información en este caso una alerta se guarda en el archivo alert.ids y Snort en este caso bloquea el acceso.

Para validar el funcionamiento se lanza una revisión de todos los puertos del equipo servidor en donde está instalado snort y se ha preparado la red neuronal, con la herramienta Zemap, como se muestra en la figura 34, previamente se ha configurado el archivo local.rules con la información para alertar sobre un escaneo de puertos. Cuando se ejecuta el escaneo las alertas quedan registradas en el archivo alert.ids, figura 35, con este archivo se muestra la alerta en el pluggis de salida.

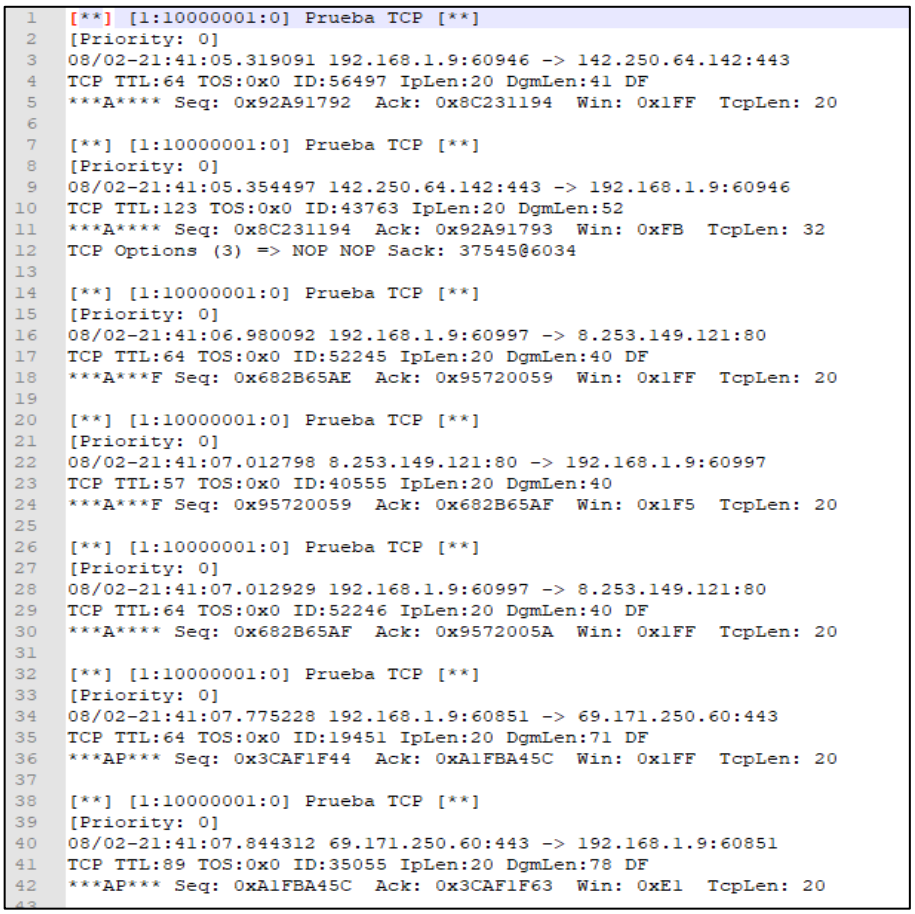
.

Figura 34. **Escaneo de puertos con Zenmap**



Fuente: elaboración propia.

Figura 35. **Muestra archivo alert.ids resultado del escaneo de puertos**



```
[**] [1:10000001:0] Prueba TCP [**]
[Priority: 0]
08/02-21:41:05.319091 192.168.1.9:60946 -> 142.250.64.142:443
TCP TTL:64 TOS:0x0 ID:56497 IpLen:20 DgmLen:41 DF
***A**** Seq: 0x92A91792  Ack: 0x8C231194  Win: 0x1FF  TcpLen: 20

[**] [1:10000001:0] Prueba TCP [**]
[Priority: 0]
08/02-21:41:05.354497 142.250.64.142:443 -> 192.168.1.9:60946
TCP TTL:123 TOS:0x0 ID:43763 IpLen:20 DgmLen:52
***A**** Seq: 0x8C231194  Ack: 0x92A91793  Win: 0xFB  TcpLen: 32
TCP Options (3) => NOP NOP Sack: 37545@6034

[**] [1:10000001:0] Prueba TCP [**]
[Priority: 0]
08/02-21:41:06.980092 192.168.1.9:60997 -> 8.253.149.121:80
TCP TTL:64 TOS:0x0 ID:52245 IpLen:20 DgmLen:40 DF
***A***F Seq: 0x682B65AE  Ack: 0x95720059  Win: 0x1FF  TcpLen: 20

[**] [1:10000001:0] Prueba TCP [**]
[Priority: 0]
08/02-21:41:07.012798 8.253.149.121:80 -> 192.168.1.9:60997
TCP TTL:57 TOS:0x0 ID:40555 IpLen:20 DgmLen:40
***A***F Seq: 0x95720059  Ack: 0x682B65AF  Win: 0x1F5  TcpLen: 20

[**] [1:10000001:0] Prueba TCP [**]
[Priority: 0]
08/02-21:41:07.012929 192.168.1.9:60997 -> 8.253.149.121:80
TCP TTL:64 TOS:0x0 ID:52246 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x682B65AF  Ack: 0x9572005A  Win: 0x1FF  TcpLen: 20

[**] [1:10000001:0] Prueba TCP [**]
[Priority: 0]
08/02-21:41:07.775228 192.168.1.9:60851 -> 69.171.250.60:443
TCP TTL:64 TOS:0x0 ID:19451 IpLen:20 DgmLen:71 DF
***AP*** Seq: 0x3CAF1F44  Ack: 0xA1FBA45C  Win: 0x1FF  TcpLen: 20

[**] [1:10000001:0] Prueba TCP [**]
[Priority: 0]
08/02-21:41:07.844312 69.171.250.60:443 -> 192.168.1.9:60851
TCP TTL:89 TOS:0x0 ID:35055 IpLen:20 DgmLen:78 DF
***AP*** Seq: 0xA1FBA45C  Ack: 0x3CAF1F63  Win: 0xE1  TcpLen: 20
```

Fuente: elaboración propia.

El esquema de conexión para esta prueba se detalla en la figura 36, en este esquema se simula un ataque de escaneo de puertos, y fue detectado con la configuración obtenida del entrenamiento de la red neuronal.

Figura 36. **Esquema de conexión para prueba de ataque**

Fuente: elaboración propia.

4.1.11. Después de la realización de las pruebas con el sistema se concluye que

La selección de Keras y la base KDD-99 para entrenar la red neuronal es eficiente, previa a cargar archivos generados por Snort. Porque permite definir reglas específicas para Snort con base enataques previos, y crea una base para definir la estructura de los archivos que alimentara Snort.

Respecto a la efectividad de la detección con las condiciones que se valido fue muy efectivo, se puede analizar con condiciones de tráfico normal, el performance es el que se ve afectado en un mínimo porcentaje.

Se comprobó que el módulo de preprocesadores no genera falsas alarmas durante el escaneo. Incluso genera menos mensajes que el módulo de Snort, esto se debe a que la red neuronal considera ataque, cuando existe un alto porcentaje de certeza de que realmente sea, el porcentaje promedio fue del 95 %.

Ningún software creado para dar seguridad a una red es 100 % efectivo y este caso no es la excepción, pero se logra aumentar considerablemente su efectividad y con el constante reentrenamiento de la red neuronal, con los archivos generados por el mismo Snort, se puede disminuir considerablemente el riesgo de fallas. Ahora el estudio se enfoca en otro parte fundamental que tan preparados están los profesionales de la seguridad informática o ingenieros en ciencias y sistemas para realizar este tipo de adaptaciones que han estado trabajando juntos desde algún tiempo atrás.

5. DIAGNÓSTICO DE CONOCIMIENTO

5.1. Conocimiento de los estudiantes en técnicas de seguridad con inteligencia artificial

Como complemento a esta investigación se busca evaluar el conocimiento con el que cuentan los profesionales de la tecnología sobre SDI y redes neuronales al finalizar la carrera, y como ven los profesionales a los egresados de la Universidad de San Carlos de Guatemala; esta percepción es muy importante ya que una de las principales causas por las que los ataques cibernéticos tienen éxito es porque las empresas no se han preparado lo suficiente para prevenir este tipo de ataques, la inversión en tecnología de punta y contratar personal capacitado en el tema es fundamental.

¿Pero los profesionales cuentan con el conocimiento necesario para implementar sistemas de seguridad informática que logren repeler ataques?, para responder a esta pregunta se ha preparado una encuesta con preguntas directas, que permitan conocer la percepción de estudiantes y profesionales sobre perciben la preparación en dichas áreas, la cual fue resuelta por estudiantes de la Escuela de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala y profesionales que laboran en instituciones dedicadas a la banca y servicios informáticos.

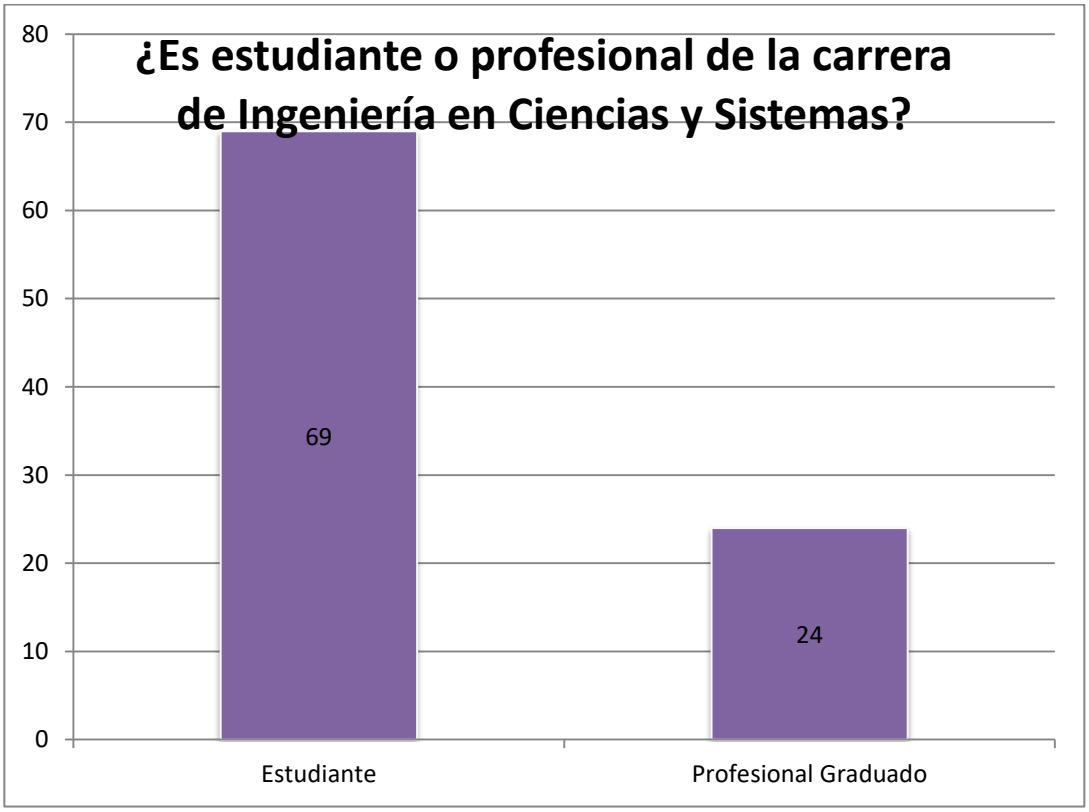
- Evaluación realizada

El objetivo poblacional de la encuesta lo representan los estudiantes de cierre de la Escuela de Ingeniería en Ciencias y Sistemas de la Universidad de

San Carlos de Guatemala y profesionales graduados principalmente encargados de seguridad informática y gerentes de tecnología de diferentes empresas. La población es de 93 individuos, estudiantes y profesionales. A continuación, se analiza las respuestas de los encuestados a cada una de las preguntas.

¿Es estudiante o profesional de la carrera de Ingeniería en Ciencias y Sistemas? (figura 37)

Figura 37. **Resultados, pregunta 1**



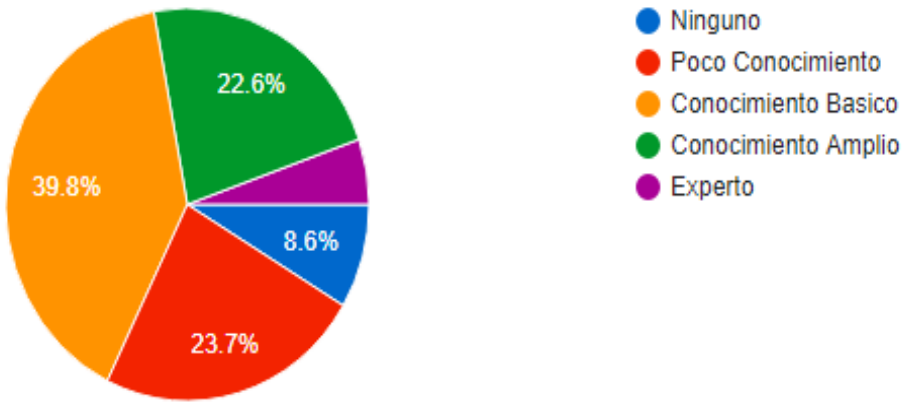
Fuente: elaboración propia.

La encuesta fue orientada a estudiantes y profesionales de la carrera de Ingeniería en Ciencias y Sistemas, principalmente, a estudiantes de cierre y profesionales en el área de seguridad informática. Por lo cual se tuvo una participación de un 74,2 % de estudiantes que equivale a 69 participantes. Y un 25,8 de participantes graduados que equivale a 24 participantes.

La siguiente pregunta pretende medir la percepción de los encuestados respectos su nivel de conocimiento en el área de seguridad informática.

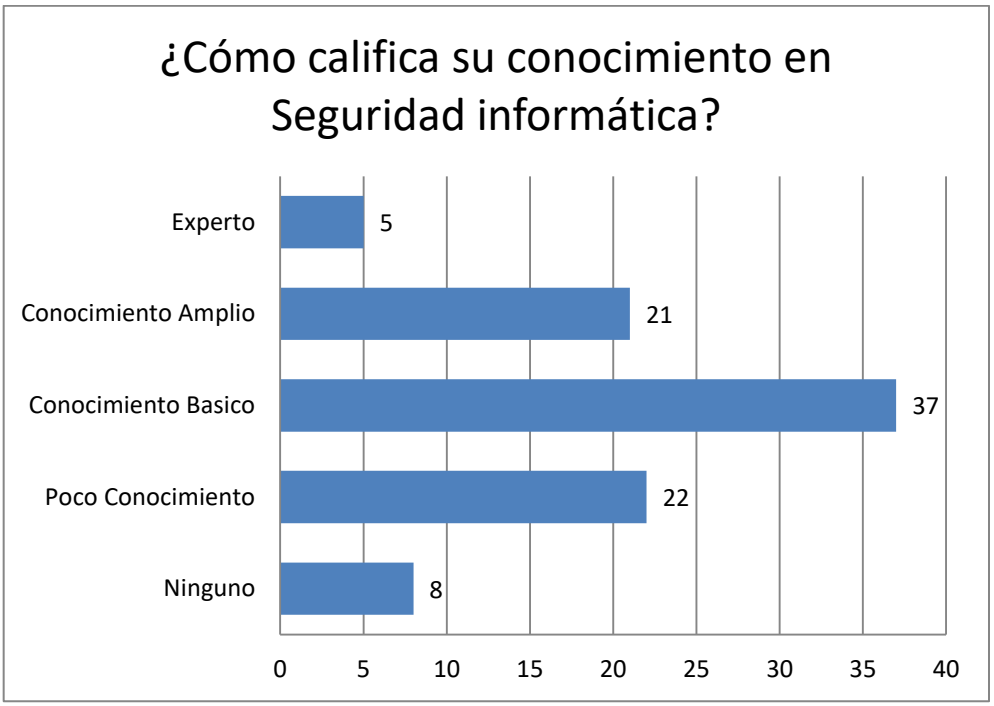
¿Cómo califica su conocimiento en seguridad informática?

Figura 38. **Resultados, pregunta 2**



Fuente: elaboración propia.

Figura 39. **Resultados, pregunta 2**



Fuente: elaboración propia.

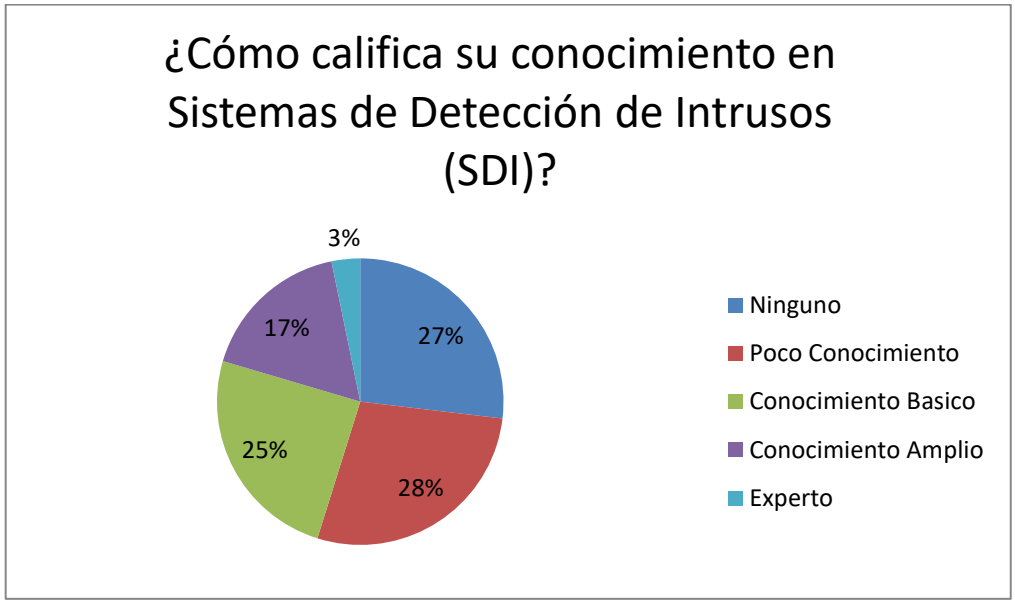
A la pregunta 2 el 5,4 % consideran ser expertos en al área de seguridad informática, esto equivale a 5 encuestados, todos profesionales, como se observa en las figuras 38 y 39.

La mayoría el 37 % de los encuestados considera tener un conocimiento básico, de estos 32 son estudiantes y 5 son profesionales.

En conclusión, el 68 % de los encuestados consideran tener un conocimiento entre básico y experto sobre seguridad informática, un total de 63 encuestados el 32 % considera tener un poco conocimiento o ninguno respecto a esta área, todos estudiantes.

La pregunta 3 está orientado a saber si los encuestados tienen algún conocimiento sobre los sistemas de detección de intrusos. Los resultados se muestran en la gráfica de la figura 40.

Figura 40. **Resultados, pregunta 3**



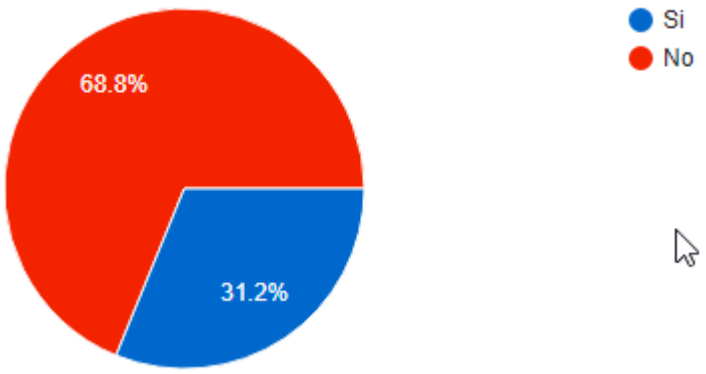
Fuente: elaboración propia.

A esta pregunta 3 profesionales se consideran expertos en SDI, que representa un 3,23 % del total de la muestra, mientras que 25 (26,88 %) todos estudiantes afirman no tener conocimiento sobre que son los sistemas de detección de intrusos, en conclusión y contrario a la pregunta anterior el 45,16 % de encuestados consideran tener un conocimiento entre básico y experto, es decir, aunque conozcan de seguridad informática, los SDI no son tan conocidos, este 45,16 % se distribuyen en 22 Profesionales y 20 estudiantes.

En este caso es recomendable tanto para los profesionales como para estudiantes actualizarse respecto a los SDI, y para cubrir el 54,84 % de encuestados que consideran tener ninguno o poco conocimiento, en su mayoría estudiantes (49 encuestados) es posible considerar por parte de la escuela de sistemas colocarlos como parte del contenido de los cursos de redes 1 o redes 2.

Ahora respecto al 45,16 % de encuestados que consideran su conocimiento entre básico y experto respecto a SDI, solamente el 31,2 % han configurado uno en la práctica, como se muestra en la figura 41; estos resultados refuerzan la recomendación a estudiantes y profesionales sobre ampliar su conocimiento en esta área de la seguridad informática.

Figura 41. **Resultados, pregunta 4**

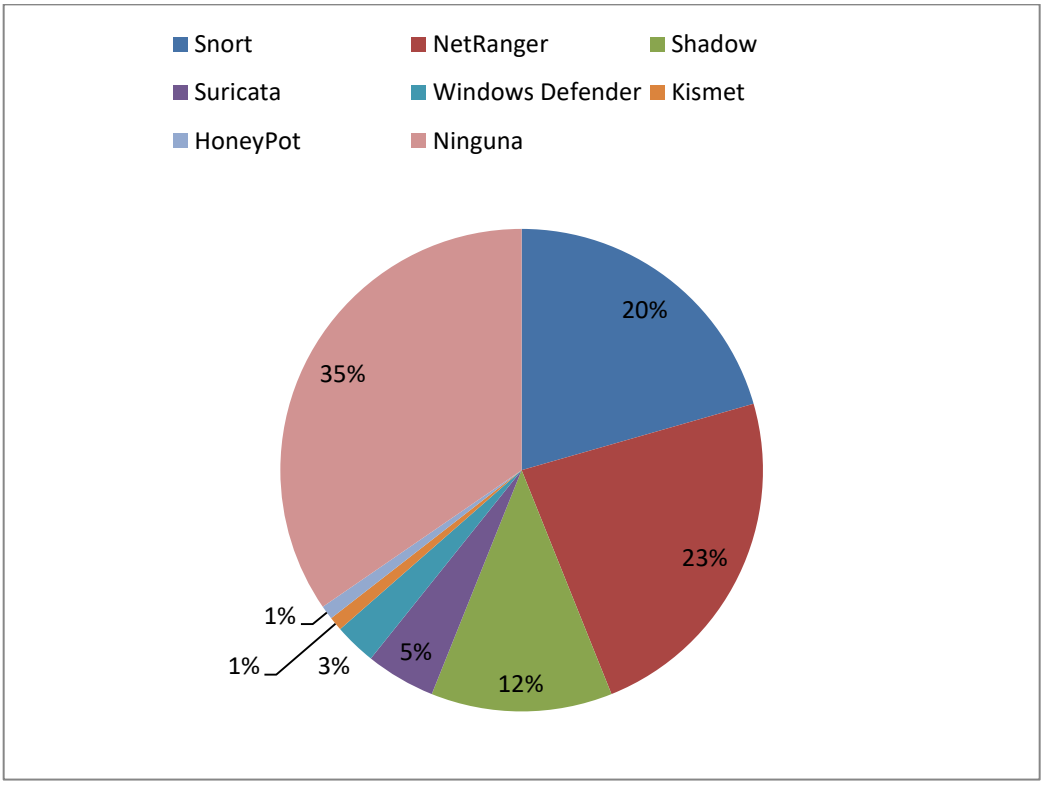


Fuente: elaboración propia.

Respecto a la pregunta 5 (figura 42) sobre que SDI's son más populares; entre los encuestados, los resultados son NetRanger (25) es el más conocido, seguido por Snort (22) y Shadow (13). Aunque la respuesta más marcada fue

ninguna (37), esto corresponde al 54,84 %, de encuestados que no tienen conocimiento o poco conocimiento en SDI.

Figura 42. **¿Conoce algunos de los siguientes SDI?**



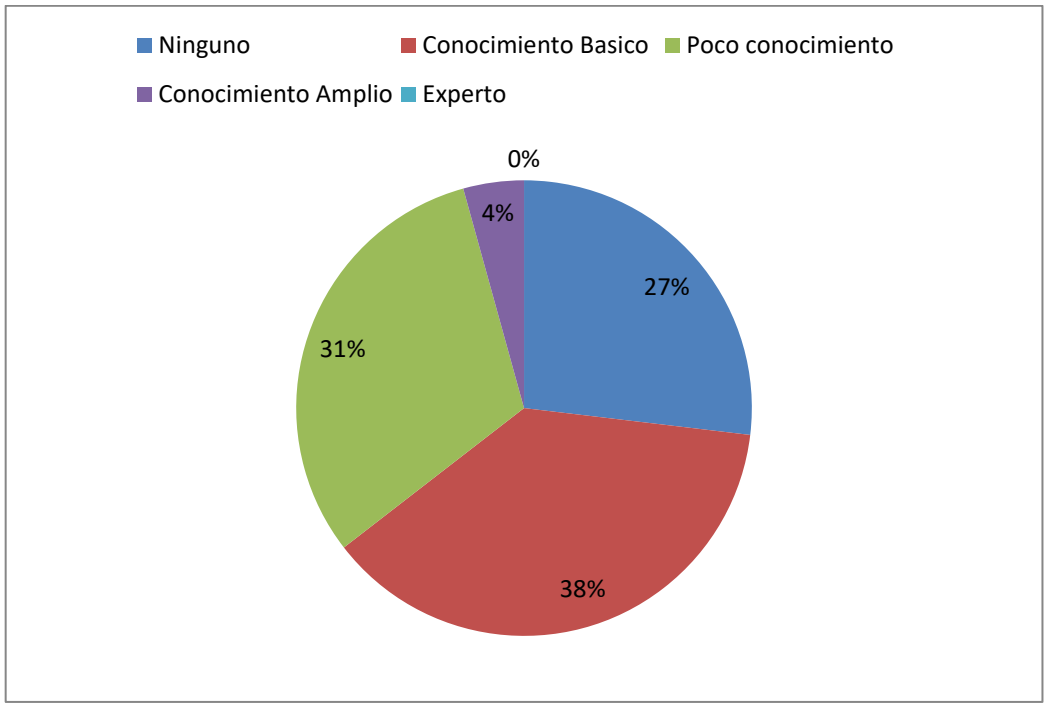
Fuente: elaboración propia.

La pregunta 6 pretende conocer la percepción de los encuestados sobre su conocimiento en redes neuronales, los resultados se muestran en la figura 43, en este caso contrario a la pregunta 2 sobre el conocimiento en seguridad informática, el 58,06 % admiten que no tienen ningún conocimiento o un conocimiento básico respecto a las redes neuronales, de esto 6 son profesionales graduados y 48 estudiantes. Y el 41,94 % se considera entre un conocimiento básico y amplio, ninguno se considera experto en el tema.

Estos resultados llevan nuevamente a la recomendación tanto a profesionales como a estudiantes a reforzar su conocimiento, y en este caso con las redes neuronales. Otra recomendación a los estudiantes es aplicar al curso de inteligencia artificial 2, que, de momento al no ser obligatoria, ninguno la toma, por el lado de la Escuela de Ingeniería en Ciencias y Sistemas podría recomendarse que en la reestructuración de pensum se coloque el curso de inteligencia artificial 2 como obligatorio para poder reforzar temas como Machine Learning y redes neuronales.

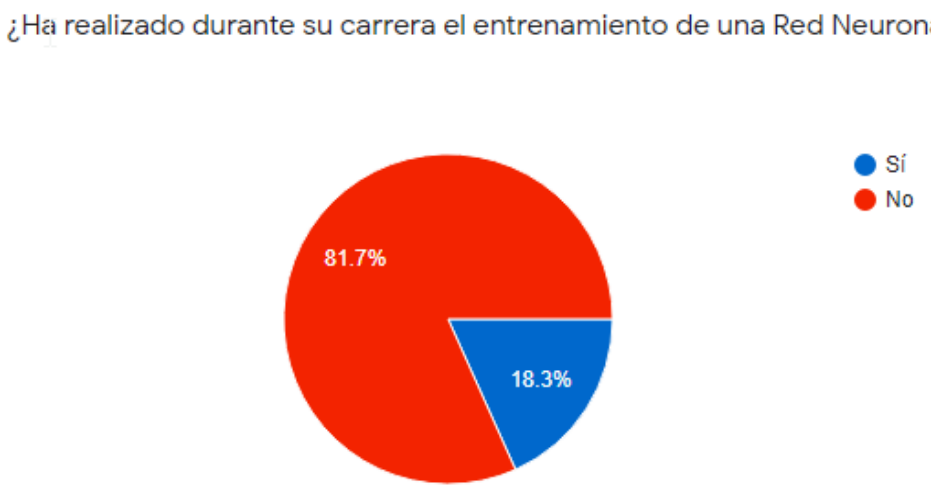
Esta recomendación se refuerza con el resultado de la pregunta 7, ¿ha realizado durante su carrera el entrenamiento de una red neuronal?, (figura 44), el 81,72 % ha respondido que No ha entrenado ninguna red neuronal, esto quiere decir que del 41,94 % que tiene un conocimiento entre básico y amplio solamente tienen la teoría, y no han llevado ese conocimiento a la práctica, del 81,72 % que no han realizado el entrenamiento de una red neuronal 18 son profesionales y 58 son estudiantes

Figura 43. **¿Cómo califica su conocimiento en redes neuronales?**



Fuente: elaboración propia.

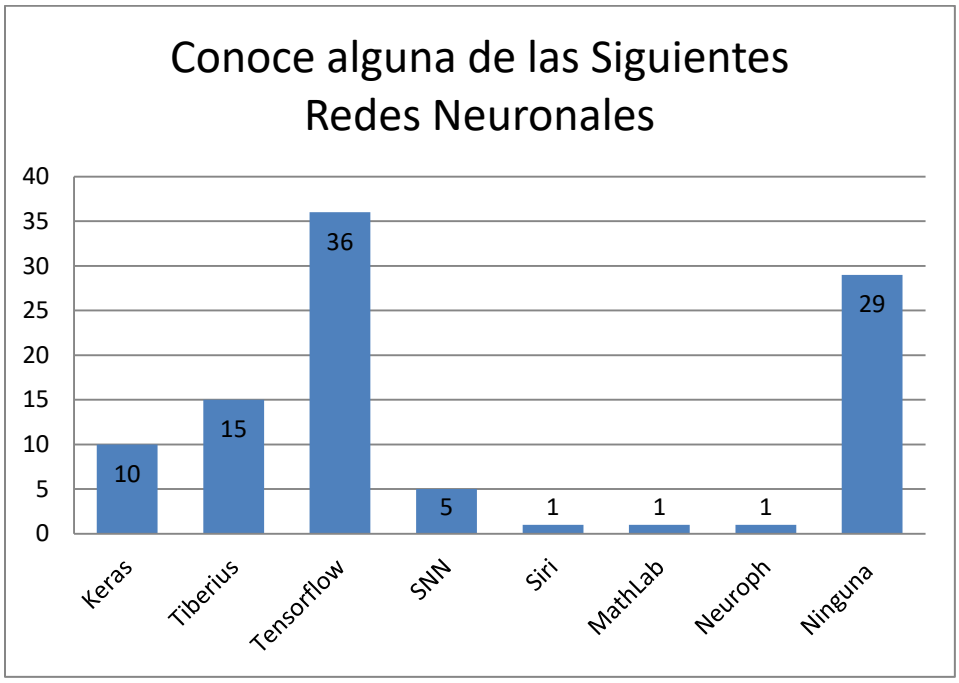
Figura 44. **Resultados, pregunta 7**



Fuente: elaboración propia.

La pregunta 8 cuyos resultados se muestran en la figura 45 pretende saber que red neuronal es más popular entre los encuestados, siento Tensorflow la más conocida, con 36 respuestas a favor equivalente a 36,73 %, se encuentra Ninguna en 2do lugar con un 29,59 %, Tiberius y Keras en 3er lugar con 5 % de diferencia entre ambos, lo interesante de estos resultados es que el porcentaje que respondió Ninguna es menor al que se obtuvo con la pregunta de SDI; es decir, aunque el conocimiento de redes neuronales no es alto, han escuchado de algunas redes neuronales, y Tensorflow puede ser el más conocido por de Google y el que más promoción a recibido, además que, por ejemplo Keras lo utiliza como base para entrenamiento.

Figura 45. **Resultados, pregunta 8**



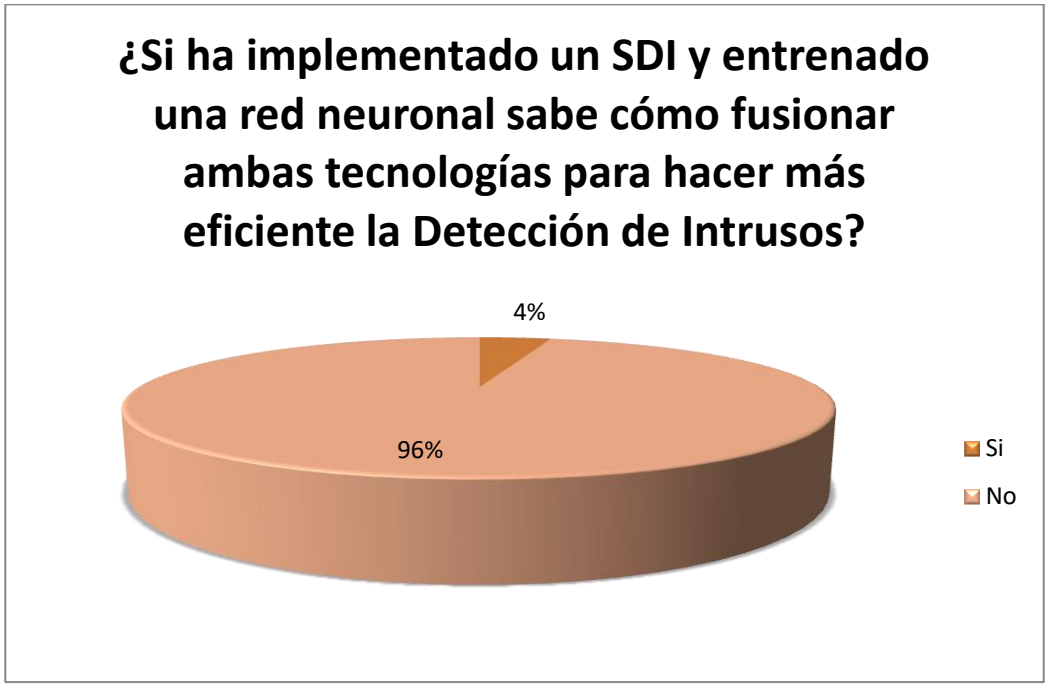
Fuente: elaboración propia.

Los resultados de la pregunta 9 (figura 46), es la principal razón que motivo a la realización de esta tesis, en nuestro país normalmente consumimos los servicios que se desarrollan en otros países, es sumamente importante cambiar esta situación y ser nosotros como profesionales los que podamos proponer soluciones, 4 profesionales de la muestra ha trabajado SDI fortalecidos con redes neuronales, y 89 nunca lo han hecho.

La práctica de utilizar redes neuronales como medio de apoyo en los SDI ha demostrado ser sumamente efectiva, porque permite incluso predecir posibles ataques, es considerable el daño que los ataques pueden hacer a una empresa, y las empresas que han sido dañadas de alguna manera valoran mucho la seguridad, tal es el caso de Equifax que ha invertido millones de

dólares en seguridad luego del ataque sufrido, y ha implementado SDI entrenando redes neuronales que le permiten predecir posibles ataques.

Figura 46. **Resultados, pregunta 9**



Fuente: elaboración propia.

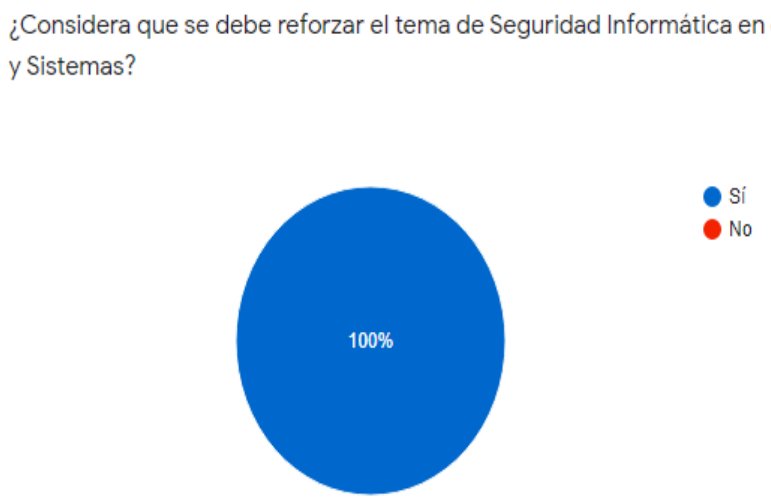
Las siguientes 2 preguntas están orientadas a la opinión de los encuestados sobre la formación que se da los estudiantes en la carrera de Ingeniería en ciencias y sistemas, la pregunta 10 ¿Considera que se debe reforzar el tema de seguridad informática en el pensum de Ciencias y Sistemas?, (figura 47) y la pregunta 11 ¿Considera que es importante ampliar la preparación sobre inteligencia artificial en la carrera de ciencias y sistemas?, (figura 48), para la pregunta 10 el 100 % respondió que si es necesario reforzar el tema de seguridad informática y respecto a reforzar temas de inteligencia

artificial el 97,8 % están de acuerdo en que se debe reforzar y 2,2 % (3 personas) que no les parece que es un tema que no se debe reforzar.

Dentro de la encuesta se hizo una pregunta abierta para complementar las preguntas 10 y 11, y en esta pregunta las 3 personas que no ven necesario reforzar el tema de inteligencia artificial coinciden en que reforzar temas como este dependen del estudiante, otros estudiantes y profesionales coinciden en que a nivel de licenciatura la preparación es básica y que si se desean reforzar los conocimientos se puede tomar un post grado.

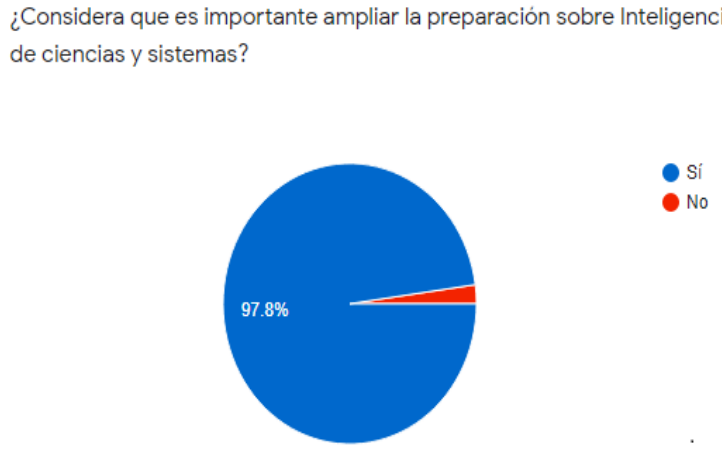
Aunque la mayoría de los estudiantes afirman en la pregunta abierta que estos temas son importantes, pero que no se da el enfoque necesario para conocerlos mejor. Estos resultados permiten recomendar a la Escuela de Ingeniería en Ciencias y Sistemas, colocar como obligatorios los cursos de inteligencia artificial 2 y seguridad de redes, esto permitirá reforzar los conocimientos en estas áreas sin llegar a la especialización, de tal forma que se den los insumos necesarios a los estudiantes para buscar por sus propios medios expandir su conocimiento por medio de la investigación.

Figura 47. **Resultados pregunta 10**



Fuente: elaboración propia.

Figura 48. **Resultados, pregunta 11**



Fuente: elaboración propia.

5.2. Discusión de resultados

Con base en la investigación realizada y las respuestas brindadas por los participantes, se recomienda a la Escuela de Ingeniería en Ciencias y Sistemas evaluar el contenido de los pensum actual e incorporar cursos que refuercen el conocimiento en las áreas de seguridad informática e inteligencia artificial, estos cursos ya existen dentro del pensum, pero al no ser obligatorios los estudiantes no los toman, aunque la mayoría de los estudiantes coinciden en que tienen las nociones básicas sobre seguridad y conocen los conceptos, también reconocen la importancia de reforzar sus conocimientos. Si bien es cierto el pensum de estudio de la carrera de ciencias y sistemas podría cambiar e incluir como obligatorios los cursos de seguridad y auditorías de redes y el curso de inteligencia artificial 2, para reforzar estos temas, es importante que los estudiantes consideren ampliar su conocimiento por medio de postgrados que ayuden a reforzar el conocimiento adquirido en durante su carrera de licenciatura, y que signifique una oportunidad ante la creciente competitividad laboral a nivel nacional e internacional.

CONCLUSIONES

1. Luego de analizar las técnicas de inteligencia artificial que pueden ser implementadas en apoyo a los sistemas de detección de intrusos, se concluye que la técnica más eficiente es la de redes neuronales, por la factibilidad de uso, un costo bajo y son realmente muy eficientes, han logrado hacer una diferencia y ha servido de gran apoyo a la seguridad informática. Existen diferentes tipos de redes neuronales, en este caso queda a discreción del profesional en IT en implementar la que desee según las necesidades y recursos con los que cuente.

2. Que las ventajas de implementar redes neuronales a los SDI son:

- La capacidad de aprender y predecir posibles ataques con base en información pasada, esto las hace mucha más eficiente de cuando se hacen por medio de restricciones de accesos e implementación de hardware de seguridad.

- Su bajo costo de implementación, el costo es realmente reflejado en tiempo de preparación que es considerado como el tiempo de configuración de equipos, de tal cuenta el costo es nulo.

3. Que las desventajas de implementar redes neuronales a los SDI son:

- Se debe tener un conocimiento previo a la selección de una red correcta, existe muchas variantes en el sistema, el código es abierto, pero se

necesita un conocimiento amplio sobre estas para poder implementarlas y que sean eficientes.

4. La tendencia de la implementación de la inteligencia artificial como apoyo a la seguridad informática, en nuestro país es prácticamente nula, el área de seguridad informática como tal está cobrando importancia para las empresas grandes, no es un tema común en empresas medianas y pequeñas. El área de inteligencia artificial es realmente bajo el conocimiento y mucho menos relacionado con los beneficios que se pueden tener al implementarlo en sistemas de detección de intrusos, esto coloca realmente en una desventaja porque empresas multinacionales requieren cada día más personas con experiencia en implementación de sistemas que permitan proteger su activo más preciado.

5. Los estudiantes de la Facultad de Ingeniería de la Escuela de Ingeniería en Ciencias y Sistemas de forma general tienen muy buena reputación a nivel nacional, son reconocidos por sus habilidades técnicas y su capacidad de investigación, sin embargo, es importante brindar los conocimientos básicos sobre inteligencia artificial y seguridad informática, para disminuir la brecha de conocimiento y la búsqueda del refuerzo por medio de cada estudiante sea mejor enfocada.

6. En el estudio se observa que los profesionales aunque se desempeñen laboralmente en áreas de seguridad informática, no cuentan con el conocimiento de temas recientes como inteligencia artificial y las ventajas que pueden brindar a sus labores; la recomendación es buscar los espacios académicos adecuados para poder mantenerse actualizados, y esto puede ser tomado por la Escuela de Ingeniería en Ciencias y

Sistemas como una gran oportunidad para ofrecer maestrías en estos temas.

RECOMENDACIONES

1. La información es un activo para la organización, es por eso la importancia de la seguridad que se implemente para mitigar los riesgos a los que está expuesta. Se debe prestar la importancia necesaria y hacer partícipe a todos los miembros de la empresa en este tema.

2. Dentro del conocimiento que un ingeniero en ciencias y sistemas egresado de la Universidad de San Carlos de Guatemala debe tener, es importante incluir temas de actualidad, la inteligencia artificial es útil no solo enfocada a la seguridad informática, pueden ser utilizadas en diferentes áreas, como en la ciencia de datos.

3. Es importante dar las herramientas necesarias a los egresados, no solo en conocimientos técnicos, sino que en habilidades blandas, que les permitan tener un don de convencimiento para implementar nuevas técnicas; ya que por ejemplo el aseguramiento de la información no se refiere a crear barreras de software que protejan el acceso a los datos, sino que hay que tomarlo de forma, integran e involucrar a todos los colaboradores y altos mandos.

4. Incluir los cursos de Inteligencia Artificial 2 y Seguridad y Auditoría de Redes como obligatorios puede ser una buena opción para reforzar el conocimiento de los estudiantes de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala.

BIBLIOGRAFÍA

1. BUHIGAS, Javier. *Todo lo que necesitas saber sobre TensorFlow, la plataforma para inteligencia artificial de Google*. [en línea]. <https://puentesdigitales.com/2018/02/14/todo-lo-que-necesitas-saber-sobre-tensorflow-la-plataforma-para-inteligencia-artificial-de-google/#:~:text=TensorFlow%20fue%20desarrollado%20originalmente%20por,investigaci%C3%B3n%20de%20redes%20neuronales%20profundas>. [Consulta: 25 de febrero de 2020.

2. FONSECA, Iren; MACIA, Francisco; LAU, Rogelio; MORA, José; GIL, Juan. *Método para la detección de intrusos mediante redes neuronales basa en la reducción de características*. España: Universidad de Alicante, 2008. 29 p.

3. GIMÉNEZ GARCÍA, María Isabel. *Utilización de sistemas de detección de intrusos como elemento de seguridad perimetral*. [en línea]. <www.adminso.es/images/1/1d/PFC_marisa.pdf>. [Consulta: 25 de mayo de 2019].

4. Global infosec. *Global infosec spending to top $ 114bn in 2018, says Gartner. ComputerWeekly.com*. [en línea]. <https://www.computerweekly.com/news/252446970/Global-infosec-spending-to-top-114bn-in-2018-says-Gartner>. [Consulta: 29 de agosto de 2018].

5. IBM Security. *Ponemon Institute: global overview, 2017 cost of data breach study*. [en línea]. <https://www.ibm.com/downloads/cas/ZYKLN2E3>. [Consulta: 2 de junio de 2019].

6. Inteligencia Artificial, Revista Iberoamericana. *Aplicación de técnicas de inteligencia artificial en la seguridad informática: un estudio.* [en línea]. <http://www.revistasbolivianas.org.bo/scielo.php?script=sci_arttext&pid=S1997-40442009000100012&lng=es&nrm=iso>. [Consulta: 11 de mayo de 2019].

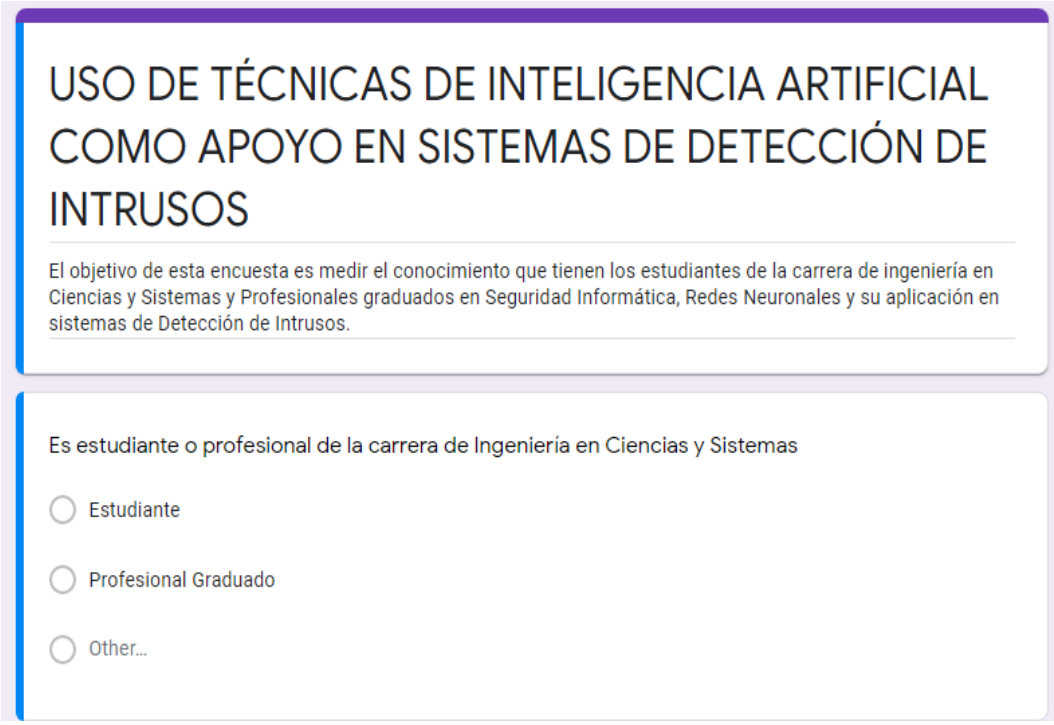
7. Revista de Información, Tecnológica y Sociedad. *Integración de Redes neuronales y Sistema de Detección de Intrusos (IDS), para las amenazas y ataques a los sistemas de información y redes* [en línea]. <http://www.revistasbolivianas.org.bo/scielo.php?pid=S1997-40442009000100008&script=sci_arttext&tlng=es>. [Consulta: 19 de mayo 2019].

8. Small Business Trends. *iber Security Statistics: numbers small businesses need to know* [en línea]. <https://smallbiztrends.com/2017/01/cyber-security-statistics-small-business.html>. [Consulta: 29 de agosto de 2019].

9. Universidad de granada. *Análisis estadístico de distintas técnicas de inteligencia artificial en detección de intrusos*. España: Editorial Universidad de Granada, 2012. 121 p.

APÉNDICES

Apéndice 1. **Encuesta para medir el conocimiento de los estudiantes de cierre y algunos profesionales**



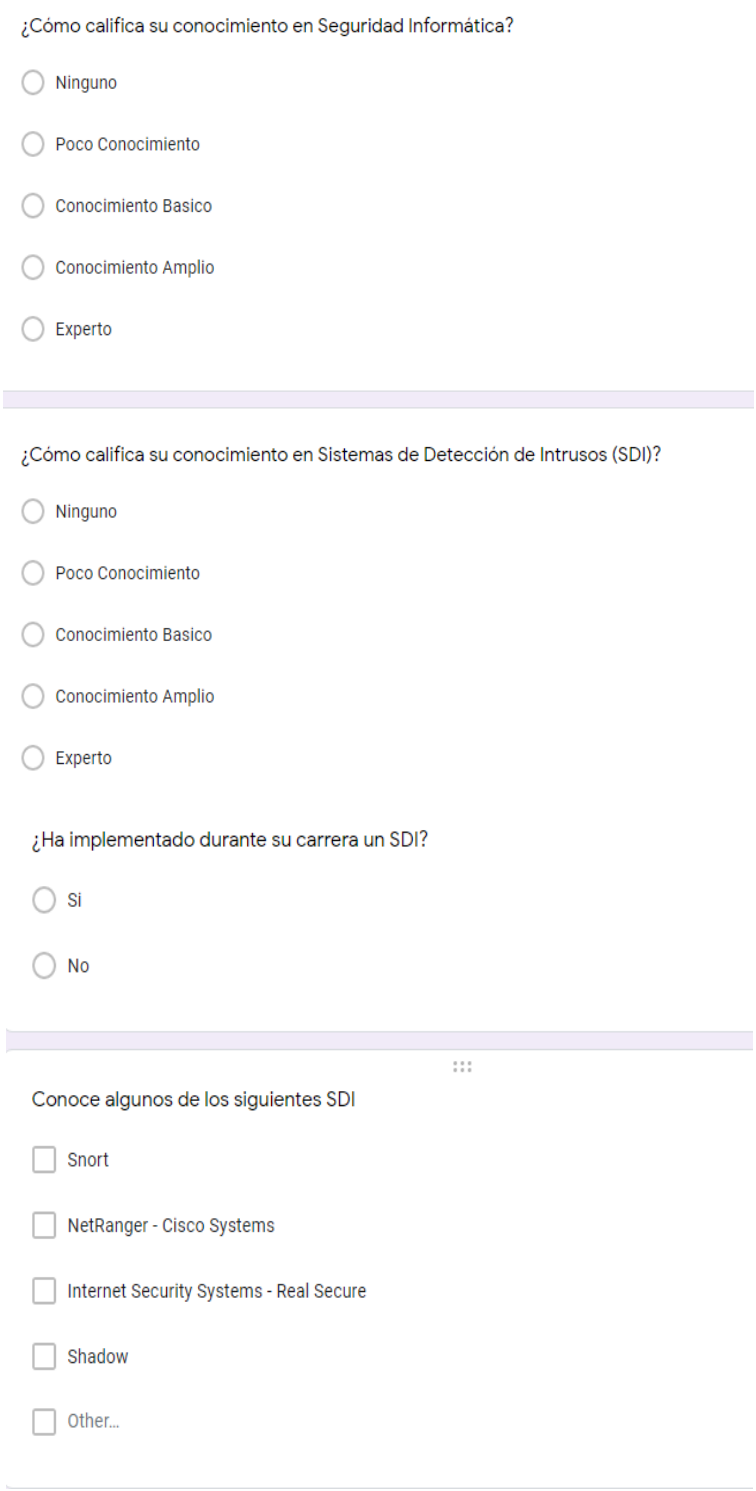
USO DE TÉCNICAS DE INTELIGENCIA ARTIFICIAL COMO APOYO EN SISTEMAS DE DETECCIÓN DE INTRUSOS

El objetivo de esta encuesta es medir el conocimiento que tienen los estudiantes de la carrera de ingeniería en Ciencias y Sistemas y Profesionales graduados en Seguridad Informática, Redes Neuronales y su aplicación en sistemas de Detección de Intrusos.

Es estudiante o profesional de la carrera de Ingeniería en Ciencias y Sistemas

- Estudiante
- Profesional Graduado
- Other...

Continuación del apéndice 1.



Continuación del apéndice 1.

¿Cómo califica su conocimiento en Redes Neuronales?

- ○ Ninguno
- ○ Poco conocimiento
- ○ Conocimiento Basico
- ○ Conocimiento Amplio
- ○ Experto

¿Ha realizado durante su carrera el entrenamiento de una Red Neuronal?

- ○ Sí
- ○ No

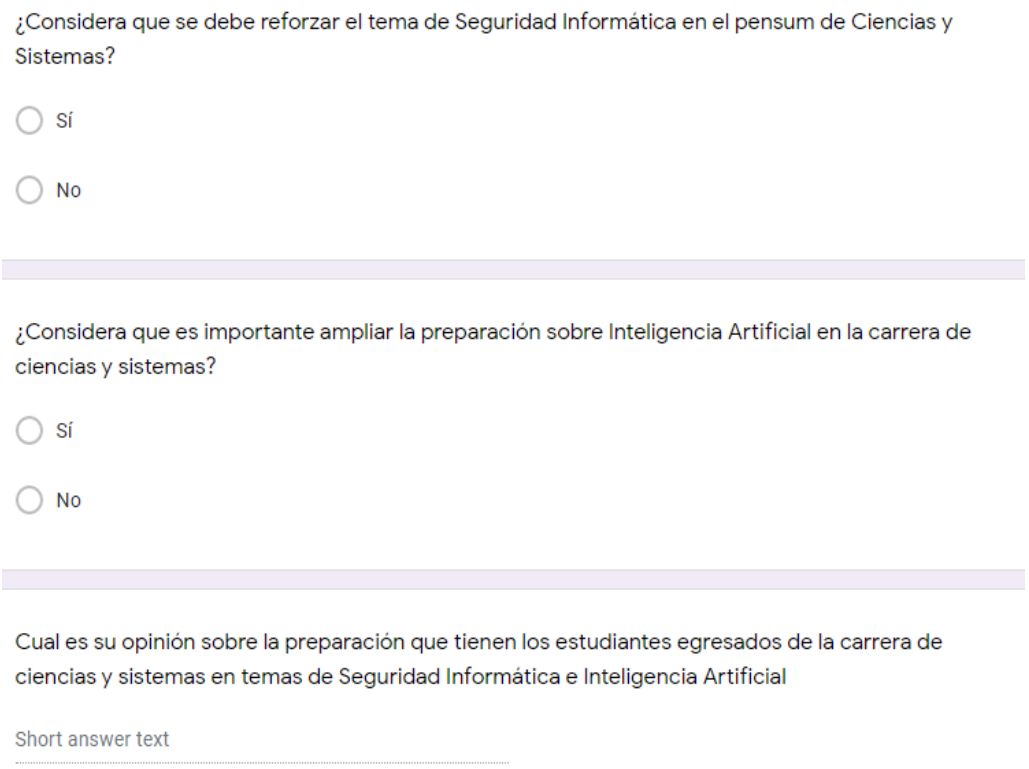
Conoce alguna de las Siguientes Redes Neuronales

- ☐ Keras
- ☐ Tiberius
- ☐ TensorFlow
- ☐ Other...

¿Si ha implementado un SDI y entrenado una red neuronal sabe cómo fusionar ambas tecnologías para hacer más eficiente la Detección de Intrusos?

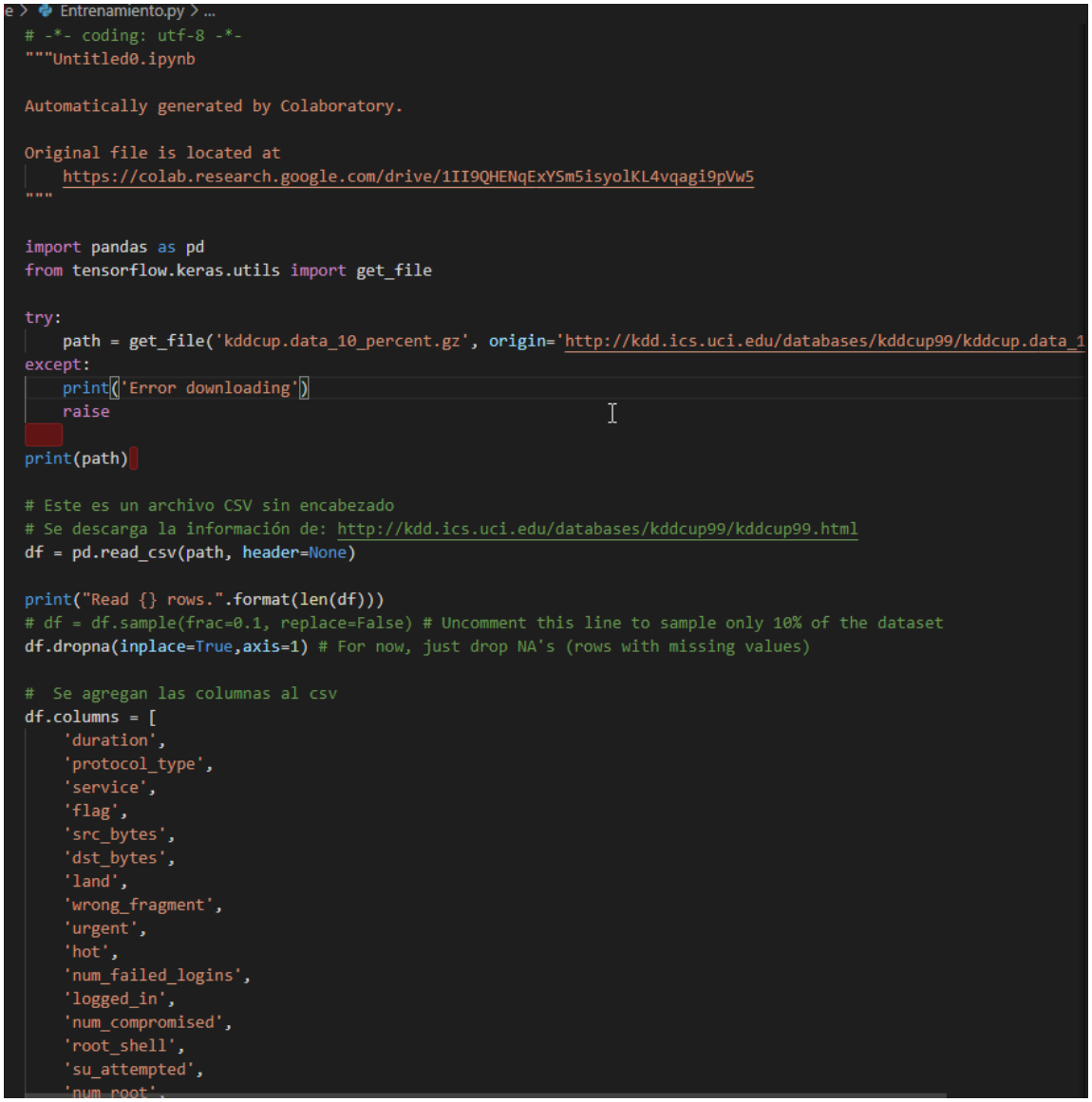
- ○ Sí
- ○ No

Continuación del apéndice 1.



Fuente: elaboración propia.

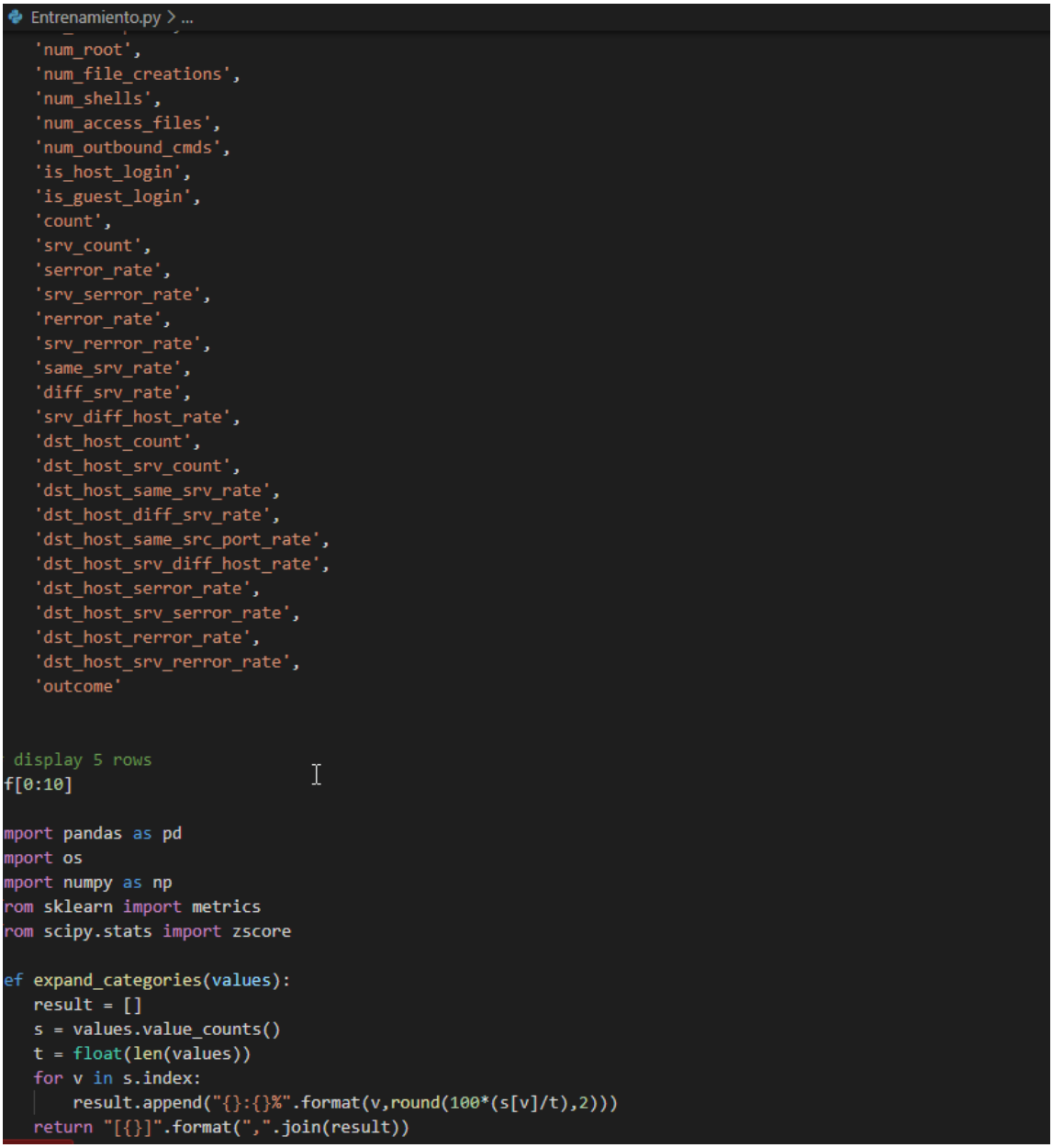
Apéndice 2. **Código fuente para la preparación de datos y entrenamiento de red neuronal, descarga de Dataset y declaración de datos para entrenamiento de red neuronal**



```
e > Entrenamiento.py > ...
# -*- coding: utf-8 -*-
"""Untitled0.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1II9QHENqExYSm5isyolKL4vqagi9pVw5
"""

import pandas as pd
from tensorflow.keras.utils import get_file

try:
    path = get_file('kddcup.data_10_percent.gz', origin='http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data_1
except:
    print('Error downloading')
    raise

print(path)

# Este es un archivo CSV sin encabezado
# Se descarga la información de: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
df = pd.read_csv(path, header=None)

print("Read {} rows.".format(len(df)))
# df = df.sample(frac=0.1, replace=False) # Uncomment this line to sample only 10% of the dataset
df.dropna(inplace=True,axis=1) # For now, just drop NA's (rows with missing values)

#  Se agregan las columnas al csv
df.columns = [
    'duration',
    'protocol_type',
    'service',
    'flag',
    'src_bytes',
    'dst_bytes',
    'land',
    'wrong_fragment',
    'urgent',
    'hot',
    'num_failed_logins',
    'logged_in',
    'num_compromised',
    'root_shell',
    'su_attempted',
    'num_root',
```

Continuación del apéndice 2.

- Definicion las categorias de la data



Continuación del apéndice 2.

- Transformación de datos, en esta sección se tratan los datos para el entrenamiento.

Entrenamiento.py > ...

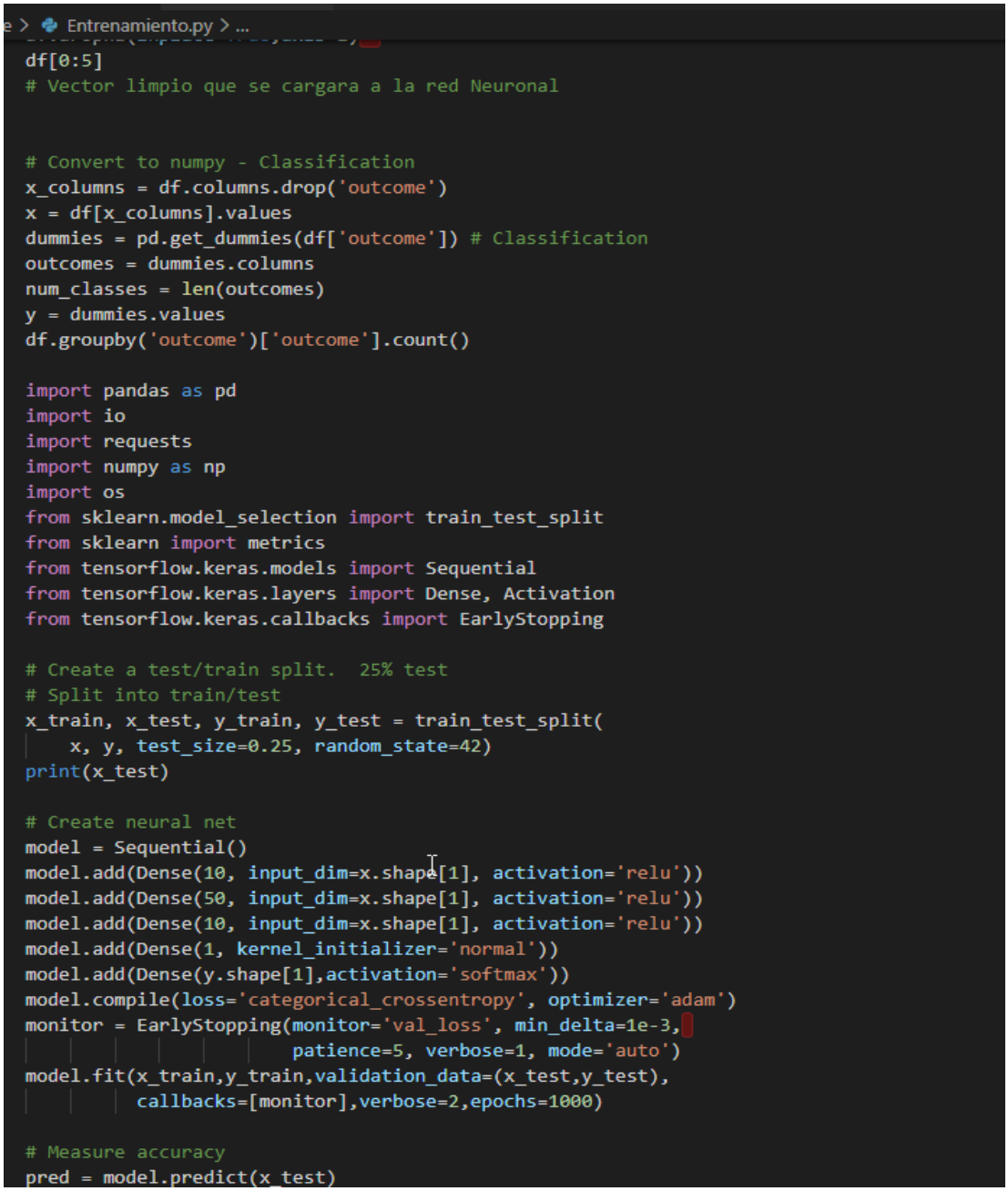
```
    for v in s.index:
        result.append("{}:{}%".format(v,round(100*(s[v]/t),2)))
    return "[{}]".format(",".join(result))

def analyze(df):
    print()
    cols = df.columns.values
    total = float(len(df))

    print("{} rows".format(int(total)))
    for col in cols:
        uniques = df[col].unique()
        unique_count = len(uniques)
        if unique_count>100:
            print("** {}:{} ({}%)".format(col,unique_count,int(((unique_count)/total)*100)))
        else:
            print("** {}:{}".format(col,expand_categories(df[col])))
            expand_categories(df[col])

# Analyze KDD-99
analyze(df)

# Codifica columnas numericas
def encode_numeric_zscore(df, name, mean=None, sd=None):
    if mean is None:
        mean = df[name].mean()

    if sd is None:
        sd = df[name].std()

    df[name] = (df[name] - mean) / sd

# Codifica valores texto en codigos
def encode_text_dummy(df, name):
    dummies = pd.get_dummies(df[name])  #Convierte series en codigos
    for x in dummies.columns:
        dummy_name = f"{name}-{x}"
        df[dummy_name] = dummies[x]
    df.drop(name, axis=1, inplace=True)

# Se codifica el vector

encode_numeric_zscore(df, 'duration')
encode_text_dummy(df, 'protocol_type')
encode_text_dummy(df, 'service')
encode_text_dummy(df, 'flag')
```

Continuación del apéndice 2.

- Categorías en las que se clasifican la base de datos, según la muestra de los datos del análisis de red.

Continuación del apéndice 2.

- Entrenamiento de la red Neuronal y definición de cantidad de capas a utilizar, y función de activación.



Continuación del apéndice 2.

- Resultado del entrenamiento, y prueba del 25 % de la base para validar.

Entrenamiento.py > ...

```
model.add(Dense(10, input_dim=x.shape[1], activation='relu'))
model.add(Dense(50, input_dim=x.shape[1], activation='relu'))
model.add(Dense(10, input_dim=x.shape[1], activation='relu'))
model.add(Dense(1, kernel_initializer='normal'))
model.add(Dense(y.shape[1],activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
monitor = EarlyStopping(monitor='val_loss', min_delta=1e-3,
                        patience=5, verbose=1, mode='auto')
model.fit(x_train,y_train,validation_data=(x_test,y_test),
          callbacks=[monitor],verbose=2,epochs=1000)

# Measure accuracy
pred = model.predict(x_test)
pred = np.argmax(pred,axis=1)
y_eval = np.argmax(y_test,axis=1)
score = metrics.accuracy_score(y_eval, pred)
print("Validation score: {}".format(score))


np.savetxt("pred.txt", pred, delimiter=",")
```

Fuente: elaboración propia.