Método para la Detección de Intrusos mediante Redes Neuronales basado en la Reducción de Características

Iren Lorenzo Fonseca[1], Francisco Maciá Pérez[2],
Rogelio Lau Fernández[1], Fco. José Mora Gimeno[2],
Juan Antonio Gil Martínez-Abarca[2]

[1] Centro de Estudios de Ingeniería y Sistemas
Instituto Superior Politécnico José Antonio Echevarría.
{ilorenzo, lau}@ceis.cujae.edu.cu

[2] Departamento de Tecnología Informática y Computación.
Universidad de Alicante.
{pmacia, fjmora, gil}@dtic.ua.es

Resumen. La aplicación de técnicas basadas en Inteligencia Artificial para la detección de intrusos (IDS), fundamentalmente las redes neuronales artificiales (ANN), están demostrando ser un enfoque muy adecuado para paliar muchos de los problemas abiertos en esta área. Sin embargo, el gran volumen de información que se requiere cada día para entrenar estos sistemas, junto con la necesidad exponencial de tiempo que requieren para asimilarlos, dificulta enormemente su puesta en marcha en escenarios reales. En este trabajo se propone un método basado en la aplicación de una técnica para la reducción de características, denominada Análisis de Componentes Principales (PCA). El PCA permite obtener un modelo para la reducción del tamaño de los vectores de entrada a la ANN, asegurando que la pérdida de información sea mínima y, en consecuencia, disminuyendo la complejidad del clasificador neuronal y manteniendo estables los tiempos de entrenamiento. Para validar la propuesta se ha diseñado un escenario de prueba mediante un IDS basado en ANN. Los resultados obtenidos a partir de las pruebas realizadas demuestran la validez de la propuesta y acreditan las líneas futuras de trabajo.

1 Introducción

A pesar de los avances en las investigaciones referentes a seguridad informática en el mundo, los resultados no cubren las expectativas, siendo cada vez mayor el número de ataques y su impacto económico y social [1], [2], [3].

Los sistemas de detección de intrusos (IDS) surgen para dar solución a los problemas de seguridad existentes, siendo concebidos para detectar las acciones que vayan dirigidas a poner en riesgo la integridad, confidencialidad o disponibilidad de un recurso [4]. Esta tecnología es relativamente joven, pero desde su surgimiento han aparecido una enorme variedad de propuestas que intentan dar solución a este enfoque, tan complicado como rico en posibilidades.

Los IDS más aceptados basan su detección de intrusiones en la comparación de la información a clasificar con los ataques que se encuentran en su base de patrones. Esta estrategia de análisis es conocida como *uso indebido* y su principal ventaja radica en el bajo índice de *falsos positivos* que genera. Por tanto, el buen funcionamiento de estos sistemas depende en gran medida del óptimo nivel de actualización de su base de patrones de ataques. Con la velocidad de aparición de nuevas versiones de ataques en la actualidad, este tipo de sistemas deja de ser funcional, puesto que resulta extremadamente difícil mantener un repositorio de ataques debidamente actualizado [5].

Por su parte, existe otra técnica de detección empleada por los IDS basada en la definición de un comportamiento normal, clasificando como intrusión a todo el que se aleje de él. Esta estrategia es conocida como *anomalías* y tiene como ventaja, a diferencia de lo anterior, que no necesita mantener patrones de ataques para la clasificación. Además, con este enfoque se logra la detección de ataques nuevos sobre los cuales aún no se tiene información. Sin embargo, estos IDS no han logrado una amplia aceptación debido a tres aspectos fundamentales:

- El uso de técnicas de Inteligencia Artificial (AI) lleva implícito la necesidad de un proceso de entrenamiento lento que hace que los administradores de red lo desestimen.
- Son generadores de numerosos *falsos positivos* debido a la dificultad que trae consigo modelar el comportamiento normal en una red o sistema.
- No ofrecen información adicional del ataque detectado.

En la literatura se pueden encontrar diferentes investigaciones que proponen IDS que basan su funcionamiento en Redes Neuronales Artificiales (ANN). Una de las principales limitaciones en este tipo de propuestas radica en que tanto la especificación de la arquitectura como la selección de los datos de entrada se realizan a partir de criterios básicamente empíricos [6], [7], [8], [9], [10], [11], [12], de forma que no se puede asegurar que dicha selección sea efectiva, es decir, que los datos no lleguen a provocar un funcionamiento deficiente del clasificador neuronal. De hecho, uno de los criterios empleados en la mayor parte de los trabajos analizados es limitar dicha selección en función de los tamaños que aseguren que las ANN empleadas puedan soportarlos eficientemente.

En este trabajo se propone un enfoque diferente: analizar con toda libertad cuáles son los datos significativos que intervienen en la detección de intrusos para, en una segunda fase, aplicar técnicas de compactación que permitan la reducción del espacio de entrada sin que se produzca una pérdida de información sustancial. De esta forma, además de mantener los niveles de eficacia alcanzados en las investigaciones revisadas, se mejora sustancialmente la eficiencia. En concreto, uno de los algoritmos que está brindando mejores resultados en este sentido es el de Análisis de Componentes Principales (PCA).

Para sistematizar la aplicación de este algoritmo en este tipo de investigaciones, tras realizar una revisión de los antecedentes (cap. 2) y del estado del arte (cap. 3) en las temáticas relacionadas, se propone un método que engloba las diferentes fases involucradas y la relación entre las mismas (cap. 4). A partir de este método se obtiene y se evalúa un modelo con el que se diseña un filtro PCA para la compactación del tráfico TCP/IP. A continuación (cap. 5) se construye un escenario

de pruebas mediante un IDS basado en una ANN multicapa y, con el fin de validar la propuesta, se analizan los resultados alcanzados comparándolos con los obtenidos antes de aplicar el filtro PCA. Finalmente (cap. 6) se presentan las principales conclusiones que se desprenden de la investigación junto con las líneas futuras a seguir.

2 Antecedentes

Dentro de los antecedentes se exponen los principales conceptos de los Sistemas de detección de Intrusos a través de la clasificación más aceptada de los mismos. Además se abordan temas como: la familia de protocolos TCP/IP, las ANN y el algoritmo de reducción de característica PCA, que están estrechamente relacionados con la propuesta de solución que se brinda en este trabajo. Por otra parte, su muestran las características principales de las bases de datos de pruebas existentes para la evaluación de los IDS.

2.1. Sistemas de detección de intrusos

Los IDS se clasifican de acuerdo a diferentes criterios, tales como: la fuente de información que utilizan para detectar la intrusión, el tipo de detección de intrusión que realizan una vez recopilada la información, el tipo de respuesta que proporcionan después de haber detectado una intrusión y, por último, la frecuencia de uso. En la figura 1 se muestra de forma gráfica una clasificación de los IDS en función de todos estos criterios. Durante los próximos apartados analizaremos cada una de estas propuestas.

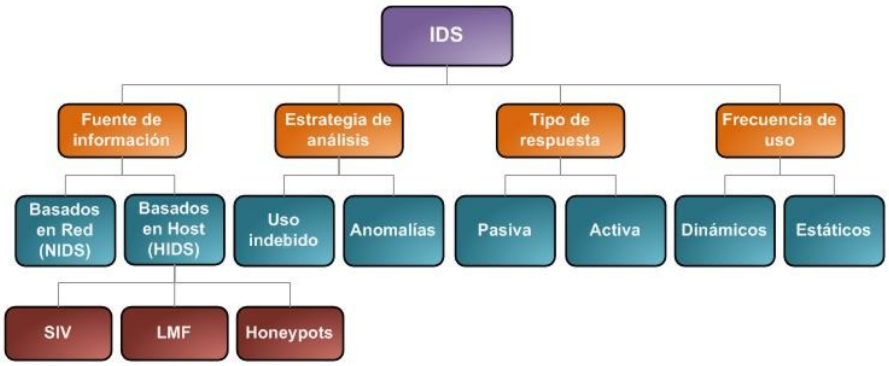

Fig. 1. Clasificación de IDS

Según la Fuente de información

Las fuentes de información clasifican a los IDS en: IDS basados en host (HIDS) e IDS basados en red (NIDS).

Los NIDS son los más populares en el área de detección de intrusos y constituyen el tipo de IDS en el que se centra nuestra investigación debido a su potencial. Utilizan el tráfico de red (paquetes TCP/IP) como fuente de información, revisando los paquetes que circulan por la red en busca de elementos que denoten un ataque contra alguno de los sistemas ubicados en ella. De dichos paquetes, verifican la validez de algunos parámetros y el comportamiento de los protocolos de la familia TCP/IP empleados [4].

Este tipo de IDS tiene la ventaja de no afectar el rendimiento de las computadoras que protege debido a que utiliza como fuente el tráfico de red. No obstante, tiene una gran limitación en las actuales redes conmutadas ya que su ubicación es un elemento clave para su buen funcionamiento. Además, entre sus desventajas se encuentra su incapacidad para detectar intrusiones en paquetes cifrados, puesto que estos pueden ser únicamente descifrados en los nodos extremos de la comunicación y no en un elemento intermedio.

Mientras que los NIDS operan en toda una red, protegiendo a un conjunto de máquinas, los HIDS 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, especialmente a nivel de sistema operativo: archivos de auditoría del sistema, archivos *logs* o cualquier archivo que el usuario desee proteger. Los HIDS tienen como ventaja su capacidad para detectar situaciones como los intentos fallidos de acceso o modificaciones en archivos considerados críticos. Las desventajas principales de este tipo de IDS se encuentran en el consumo de recursos del propio equipo que desea proteger y el hecho de que él mismo está expuesto a ser víctima de algún ataque.

Dentro de los HIDS existen subgrupos que utilizan diferentes vías para detectar las intrusiones. Algunas de estos subgrupos son [13]:

- Sistemas Verificadores de Integridad (SIV): típicamente monitorizan los sistemas de archivos en busca de modificaciones relevantes.
- Analizadores de Archivos Log (LFM): comprueban los archivos *logs* en busca de patrones sugerentes.
- Sistemas Víctima (honeypots): sistemas que aparentan servidores o computadoras vulnerables y que se ofrecen como señuelo para desviar la actividad de los intrusos.

Según la estrategia de análisis

La estrategia de análisis se refiere a la técnica que utiliza el IDS, una vez recuperada la información, para identificar si se produjo o no una intrusión. Estas estrategias dividen a los IDS en dos grandes grupos: los IDS de detección del *mal uso* o *uso indebido* y los IDS de detección de *anomalías*.

Los IDS basados en detección de *uso indebido* cuentan con el conocimiento a priori de las secuencias y actividades que conforman un ataque almacenado en una base de datos. Por tanto, para detectar intrusiones dentro de la información recopilada de la fuente, se hace una comparación de la misma con los patrones de ataques almacenados y, en caso de encontrar similitud, se genera una alarma. Con este método se logra detectar intentos de explotación de vulnerabilidades típicos, de los

cuales ya existe información. Esta estrategia es la más utilizada en los IDS comerciales [14].

La detección de *uso indebido* tiene entre sus grandes ventajas la amplia seguridad que ofrece al detectar una intrusión puesto que clasificar una actividad como intrusiva significa que se correspondió con un patrón (de la base de datos) que fue reconocido con anterioridad como un ataque. De esta forma, el índice de *falsos positivos* es muy bajo.

La desventaja principal de este tipo de detección radica en su incapacidad para detectar nuevos ataques y en la necesidad de mantener constantemente actualizada su base de patrones. Estas debilidades son de vital importancia en la actualidad donde los ataques son cada vez más originales y cada vez más frecuentes. Debido a esto las investigaciones recientes de detección de uso indebido se centrar en lograr patrones más genéricos que permitan que los sistemas de este tipo no puedan ser burlados con mutaciones de ataques conocidos [16], [17], [15].

Los IDS de detección de *anomalías* tienen un conocimiento contrario a los de *uso indebido:* parten del conocimiento de lo *normal* y toda actividad que se aleje de este comportamiento es considerada *una intrusión*. Este tipo de detección evita el proceso de actualización de una base de datos de patrones de intrusión y brinda la posibilidad de detectar ataques nuevos de los cuales no se tenga información alguna. No obstante, este tipo de IDS tiene algunas desventajas que han provocado el rechazo por parte de los administradores de redes. Por ejemplo, son generadores de un gran número de *falsos positivos* ya que el comportamiento normal de los usuarios es extremadamente difícil de modelar, por lo variable que puede llegar a ser y, por tanto, un comportamiento inusual no tiene necesariamente por qué ser ilícito. Otra debilidad importante de este tipo de sistema es que necesita un largo período de entrenamiento, previo a su uso, para detectar los comportamientos normales de los usuarios y sistemas dentro de la red [14].

Ambas propuestas presentan en la actualidad serias carencias. Los IDS basados en *uso indebido* no son funcionales en las condiciones actuales donde se generan nuevos ataques con tan alta frecuencia. Por otra parte, los IDS basados en *anomalías* aunque surgen para eliminar estas limitantes, generan un número excesivo de *falsos positivos*.

Según el tipo de respuesta

La respuesta que proporciona un IDS ante la detección de una intrusión también es un factor determinante en su clasificación. Estas respuestas pueden ser: *pasivas* o *activas*. Las *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 cortafuego.

Un IDS puede soportar ambos tipos de respuesta, no son excluyentes, todo lo contrario, ya que es recomendable reaccionar de manera pasiva ante determinadas anomalías de poca envergadura y activamente ante ataques que puedan poner seriamente en peligro la seguridad de los sistemas.

Según la frecuencia de uso

La frecuencia de uso clasifica a los IDS 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. No obstante, este tipo de herramientas introduce costos operativos significativos.

Los IDS estáticos trabajan fuera de línea a intervalos específicos de tiempo. Éstos no ofrecen seguridad entre intervalos de ejecución y en casos de ataques exitosos sólo pueden ser utilizados para análisis forense [5].

2.2 Familia de Protocolos TCP/IP

Los NIDS utilizan como fuente de información fundamental las características del tráfico TCP/IP y de los paquetes que circulan por la red. Por tanto, resulta de interés el estudio de las principales características de esta familia de protocolos.

Las siglas TCP/IP provienen de *Transmission Control Protocol / Internet Protocol* que se corresponden con los nombres de los protocolos más importantes dentro de la familia

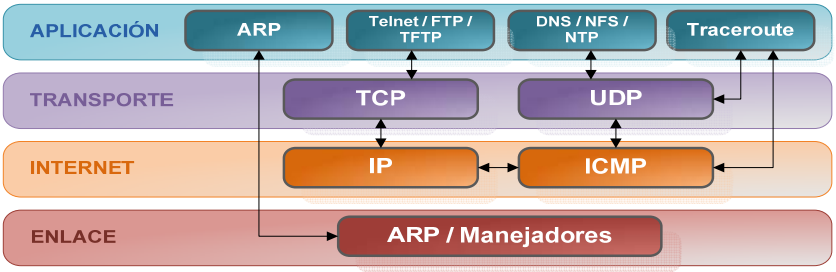

Fig. 2. Capas del protocolo TCP/IP y protocolos asociados

TCP/IP está ampliamente extendido por ser el protocolo de comunicación básico utilizado en Internet. En muchas ocasiones se denomina la pila TCP/IP debido a que, en realidad, está compuesto por varios protocolos [18]:

- **Internet Protocol (IP).** Encargado del transporte del flujo de datos de una máquina a otra.
- **Internet Control Message Protocol (ICMP).** Proporciona varios tipos de soporte de bajo nivel para IP, incluyendo mensajes de error, ayuda al encaminamiento y peticiones de eco.
- **User Datagram Protocol (UDP) y Transmisión Control Protocol (TCP).** Envían datos desde un programa a otro utilizando IP. UDP proporciona transporte sin gestión de errores para mensajes individuales siendo no orientado a conexión, mientras que TCP es orientado a conexión y realiza gestión de errores.
- **Address Resolution Protocol (ARP).** Es el componente de más bajo nivel, encargado de traducir las direcciones IP a direcciones físicas que dependerán del dispositivo real que conecta a la red.

TCP/IP está estructurado por capas. En la bibliografía se reconocen cuatro capas, donde cada una de ellas es la encargada de llevar a cabo una tarea específica de la comunicación. La figura 2 muestra las capas y los principales protocolos asociados con cada una de ellas.

Capa de Enlace: los protocolos de este nivel no pertenecen propiamente a la familia TCP/IP pero son la base sobre la que se sustentan. Esta capa incluye los drivers que controlan las tarjetas de red y toda la gestión de la conexión entre el hardware y los cables y dispositivos de red.

Capa de Internet (o capa de red): se encarga del envío y recepción de los paquetes a través de la red. Es a este nivel que se lleva a cabo el enrutamiento de los paquetes, encaminándolos hacia los diferentes puntos por los cuales debe transitar hasta llegar al destino. El principal protocolo de esta capa es el *Internet Protocol* (IP, que le da nombre), aunque también se encuentran: *Address Resolution Protocol* (ARP), *Internet Control Message Protocol* (ICMP) e *Internet Group Management Protocol* (IGMP).

Capa de Transporte: la principal tarea de la capa de transporte es proporcionar la comunicación entre un programa de aplicación y otro. Esta capa regula el flujo de información. Los dos protocolos posibles de transportes son: *Transmission Control Protocol* (TCP) que ofrece una entrega confiable y *User Datagram Protocol* (UDP) que es no confiable. Se puede utilizar uno u otro protocolo dependiendo del método preferido de envío de datos.

Capa de Aplicación: en la cima de este modelo está la capa de aplicación. Ésta es la capa que las aplicaciones utilizan para acceder a la red. Los protocolos de este nivel interactúan con uno de los protocolos de nivel de transporte para enviar o recibir datos. Existen diversos protocolos de aplicación entre los que se encuentran: HTTP, FTP, Telnet, SNMP, DNS, entre otros.

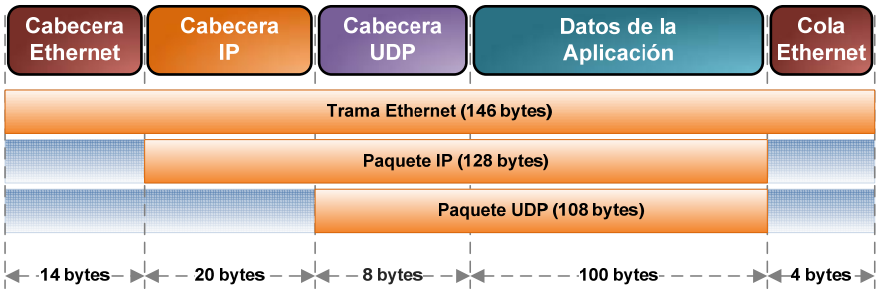

Fig. 3. Estructura del Paquete TCP/IP para un paquete UDP

El protocolo TCP/IP se encarga de dividir la información que debe transmitir en paquetes, a los que les añade información adicional según las capas por las que vaya descendiendo. Finalmente, la última capa debe componer un paquete que entienda la red física por la que van a ser transmitidos (ethernet, token-ring, etc.), en este nivel, al paquete se le suele denominar trama (frame).

La figura 3 muestra de manera gráfica el encapsulamiento de la información de los protocolos en el área de datos del protocolo de nivel inferior. Para ello se presenta un ejemplo con el protocolo de transporte UDP.

Una vez recibido el paquete en la máquina destino, las tramas son recogidas por el dispositivo de red al que van dirigidas gracias a la información a nivel físico que poseen y a su formato compatible. Una vez leída por el dispositivo de red, se la pasará a la pila TCP/IP que irá efectuando el proceso contrario: elimina la información que se le añadió en el origen, hasta conseguir pasar los datos que realmente se desean comunicar a la aplicación o protocolo de aplicación que los esperaba [18].

2.3. Redes Neuronales Artificiales en la detección de intrusos

Desde la propuesta realizada por Denning [19] del primer modelo de detección de intrusos, han aparecido un sinnúmero de IDS que aplican diferentes técnicas que han variado desde métodos basados en conocimientos, hasta métodos de la estadística clásica y la AI para el aprendizaje automático. En [14] se puede encontrar una clasificación detallada de las diferentes técnicas.

Las ANN constituyen una de las técnicas que ha presentado amplias ventajas en su aplicación para la detección de intrusos. Las redes neuronales han demostrado ser potentes clasificadores con grandes capacidades de generalización y aprendizaje que presentan características que hacen muy factible su aplicación en los IDS tal y como se expresa en la tabla 1.

Tabla 1. Ventajas de las ANN para la detección de intrusos**.**

Características de las ANN	Ventajas en la detección de intrusiones
Forma de representación del conocimiento muy apropiada para problemas de clasificación.	En los IDS se trabaja precisamente con un problema de clasificación, posibilitando tomar ventajas de la facilidad de representación del conocimiento.
Alta tolerancia a errores, siendo capaces de clasificar datos que presentan ruidos o están incompletos.	Esta característica es de gran importancia sobre todo en los NIDS y NNIDS 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.	Da una idea más clara al administrador acerca de la decisión a tomar.
Pueden clasificar datos desconocidos.	Brinda la posibilidad de detectar ataques de los cuales aún no se tiene conocimiento.

Las ANN tratan de captar la esencia de procesos biológicos y aplicarla a nuevos modelos de computación. La Neurona Artificial es un modelo simplificado de neurona biológica. La interconexión entre neuronas decide el flujo de información en la red y, junto con los pesos y las funciones de salida de cada neurona definen el comportamiento global de la red neuronal artificial. Esto hace que las redes neuronales estén formadas por un gran número de elementos de cómputo lineales y no lineales (neuronas) complejamente interrelacionados y organizados en capas [20]. En la figura 4 se muestra la estructura de una neurona típica de una red neuronal.

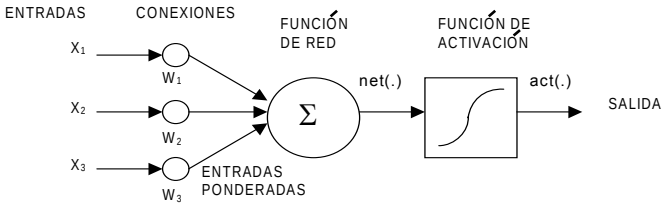

Fig. 4. Unidad de procesamiento típica de una red Neuronal (Neurona)

Las ANN por su estructura distribuida masivamente paralela y la habilidad de aprender y generalizar, están capacitadas para resolver problemas complejos. Son capaces de procesar grandes cantidades de datos y según su diseño hacer predicciones con buen nivel de precisión. Su forma de aprendizaje se produce a través de patrones de entrenamiento que le permiten determinar el comportamiento del sistema.

Una ANN aprende de su ambiente mediante un proceso iterativo de ajuste de los pesos sinápticos y umbrales. Este proceso de entrenamiento tiene alto costo computacional y existen varios tipos de algoritmos de aprendizaje, entendidos como las mejores reglas para la solución del problema de aprendizaje planteado. Normalmente se dan dos condiciones para no continuar iterando [20]:

- Un valor de conformidad para el error global de la iteración.
- Un número máximo de iteraciones.

En el proceso de entrenamiento se actualizan los valores de los pesos buscando que las salidas obtenidas de la red sean las esperadas.

Las ANN, en función del tipo de entrenamiento, se pueden clasificar en supervisadas y no supervisadas. Las supervisadas exigen la introducción de los patrones de entrada así como las respuestas esperada en cada caso. Un ejemplo es el *Perceptrón Multicapa* (MLP) [21] que, a pesar de contar con diversos algoritmos de entrenamiento, resulta bastante sencillo de implementar y utilizar.

Por otra parte se encuentran las *redes auto-organizativas* [20] que se entrenan de manera no supervisada y van agrupando las neuronas en correspondencia con los patrones de entrada. Éstas tienen la capacidad de describir relaciones topológicas entre las señales de entrada, de modo que las relaciones de semejanza más importantes entre las señales de entrada son convertidas en relaciones espaciales entre las neuronas. Algunos ejemplos son los Mapas Auto-organizativos de Kohonen (SOM) [21], las Growing Cell Structure [22], la Neural Gas [23] y la Growing Neural

Gas [23]. Siendo las dos primeras de topología fija y dimensionalidad preestablecida a diferencia de las últimas que varían su estructura durante el proceso de aprendizaje.

2.4. Análisis de Componentes Principales

El Análisis de Componentes Principales (PCA) es un método matemático que permite hacer un análisis exploratorio de los datos y disminuir la cantidad de variables que se utilizan para expresar la información que contienen. Aunque se puede utilizar con conjuntos de datos que contienen información para un número pequeño de variables, es realmente útil cuando el número de variables es grande. El objetivo de PCA es descubrir nuevas variables, llamadas Componentes Principales (PC), que apuntan a la mayor variabilidad en los datos. Esto permite describir la información con una cantidad de variables considerablemente menor que la original.

PCA calcula la base más significativa para re-expresar un conjunto de datos, esperando que ésta elimine la correlacionalidad y revele información útil oculta. Por tanto, PCA se basa en determinar qué dimensiones son importantes y cuáles son redundantes en los datos.

Cada muestra en los datos es un vector en un espacio m-dimensional, donde *m* es el número de variables utilizadas, es decir, cada muestra es un vector que se encuentra en un espacio vectorial de dimensión *m*, expandido por una base ortonormal. Todos los vectores de mediciones en este espacio son combinaciones lineales del conjunto de vectores unitarios de la base. Una elección simple e intuitiva de una base **B**, es la matriz de identidad **I**

$$B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} = I \qquad (1)$$

donde cada fila es un vector base $\mathbf{b_i}$ con *m* componentes.

Se puede declarar ahora con más precisión que PCA busca la combinación lineal de la base original que re-exprese mejor el conjunto de datos. Para ello se basa en la linealidad, logrando simplificar el problema.

Con este presupuesto, PCA está ahora limitado a re-expresar los datos como combinación lineal de los vectores de la base. Sean **X** y **Y** matrices relacionadas por una transformación lineal **P**, siendo **X** el conjunto de datos originalmente archivados y **Y** la re-representación de ese conjunto de datos.

$$PX = Y \qquad (2)$$

Se define además lo siguiente:

- $\mathbf{p_i}$ son las filas de **P**.
- $\mathbf{x_i}$ son las columnas de **X**.
- $\mathbf{y_i}$ son las columnas de **Y**.

La ecuación (2) representa un cambio de base y puede tener diferentes interpretaciones, a saber:

1. **P** es una matriz que transforma **X** en **Y**.
2. Geométricamente, **P** es una rotación que nuevamente transforma **X** en **Y**.
3. Las filas de **P**, $\{p_1, \ldots, p_m\}$ forman un nuevo conjunto de vectores bases para expresar las columnas de **X**.

Esta última interpretación no es obvia, pero puede ser vista escribiendo explícitamente el producto interno **PX**

$$PX = \begin{bmatrix} p_1 \\ \vdots \\ p_m \end{bmatrix} \begin{bmatrix} x_1 & \ldots & x_n \end{bmatrix} \qquad (3)$$

$$Y = \begin{bmatrix} p_1 \cdot x_1 & \cdots & p_1 \cdot x_n \\ \vdots & \ddots & \vdots \\ p_m \cdot x_1 & \ldots & p_m \cdot x_n \end{bmatrix} \qquad (4)$$

puede notarse la forma de cada columna de **Y**, las cuales son el producto interno de $\mathbf{x_i}$ con su correspondiente fila en **P**.

$$y_i = \begin{bmatrix} p_1 \cdot x_i \\ \vdots \\ p_m \cdot x_i \end{bmatrix} \qquad (5)$$

En otras palabras, $\mathbf{y_i}$ es una proyección en la base formada por $\{p_1, \ldots, p_m\}$. Por tanto, las filas de **P** constituyen un nuevo conjunto de vectores que forman una base para representar las columnas de **X**.

Con la aplicación de la linealidad, el problema se reduce a encontrar el cambio de base apropiado. Los vectores fila $\{p_1, \ldots, p_m\}$ en la transformación serán las Componentes Principales (PC) de **X**. Una buena elección de la base P para re-expresar X sería aquella que proporcionase como resultado un grupo de datos donde se disminuya el ruido y la redundancia [24].

2.5 Bases de Datos de prueba

Tanto para el entrenamiento de la ANN como para la obtención del modelo PCA es necesario contar con un conjunto de datos que tenga tráfico de red TCP/IP *normal* y tráfico con *paquetes intrusivos*. La obtención de un conjunto de datos con estas características, que tenga la calidad requerida, es un problema importante dentro de las investigaciones en el campo de la detección de intrusos basado en técnicas de AI.

La bibliografía consultada referencia dos vías fundamentales para obtener estos datos: utilizar bases de datos de paquetes TCP/IP existentes en Internet o capturar tráfico de una red y simular ataques con alguna batería de ataques.

En Internet existen de manera pública recopilaciones de paquetes TCP/IP, entre las que se destaca, por su uso, la base de datos de la agencia DARPA (Defense Advanced Research Projects Agency). Ésta se considera la primera colección de datos que se distribuye oficialmente para lograr que los sistemas de detección de intrusos sean evaluados. Desde el sitio web del Laboratorio de Lincoln del MIT (Massachusetts Institute of Technology) se permite la descarga gratuita de estos archivos con la estructura *tcpdump* correspondiente a ejemplos de ataques y tráfico inofensivo de 1998 y 1999, así como un conjunto adicional y experimental del 2000 [25]. Las bases de DARPA contienen instancias de cuatro clasificaciones de ataques fundamentales: indagación o exploración (probing), Usuario a Raíz (U2R), Remoto a Local (R2L) y Denegación de Servicios (DoS). Esta base de datos es considerada la más grande colección brindad para la evaluación de sistemas de seguridad. No obstante, algunos autores señalan que la selección de ataques en DARPA 1999 puede ser considerada anticuada comparado con las amenazas modernas para la seguridad de computo [26].

Existe otro conjunto de datos importante que fue creado para el Tercer Concurso Internacional de Herramientas de Descubrimiento de Conocimiento y Minería de Datos (KDD Cup) que en el año 1999 fue dedicado a la detección de intrusos. Éste no es más que un subconjunto procesado de los datos DARPA 1998. Cada uno de los registros de esta base de datos contiene valores de 41 variables independientes, los cuales describen diferentes características de una conexión y, además, incluyen otra variable etiquetando la conexión como normal o como un tipo de ataque específico [27].

Estas bases de datos son recopilaciones hasta el año 2000, por lo que no contienen instancias de ataques ocurridos en los últimos años. Por esta razón, muchas veces se utilizan soluciones alternativas para la obtención del conjunto de datos. Una de estas soluciones consiste en capturar paquetes de tráfico real utilizando un *sniffer*. Para incluir dentro del conjunto paquetes de ataques, se puede utilizar una batería de ataques como *Nessus* [28]. *Nessus* cuenta con actualizaciones anuales, siendo capaz de proveer ataques surgidos en los últimos años.

3 Estado del arte

Las características de las ANN para la clasificación y sus ventajas para la detección de intrusos las han convertido en fuentes de numerosas investigaciones en los últimos tiempos. Aunque los resultados brindados muestran las potencialidades de su uso, también evidencian un grupo de limitaciones que aún no han sido resueltas.

Las investigaciones recientes que utilizan redes neuronales artificiales en detección de intrusos se basan en su flexibilidad y adaptación a los cambios naturales que se pueden dar en el entorno y, sobre todo, a la capacidad de detectar instancias de los ataques desconocidos. Las ANN poseen la ventaja de permitir una fácil representación de relaciones no lineales entre las entradas y las salidas y también son capaces de analizar datos incompletos o distorsionados [29]. La mayor deficiencia

que tienen las redes neuronales es que constituyen un modelo no descriptivo, por tanto, actúan como una caja negra sin que se pueda conocer la razón de la decisión tomada.

Los primeros trabajos en detección de intrusos que han utilizado ANN como clasificador, lo han hecho tanto para *uso indebido* como para *anomalías*. Muchos han basado su procesamiento en el uso del *perceptrón multicapas* (MLP) para modelar el comportamiento normal del proceso sobre la base de las llamadas al sistema [6]. Los MLP han sido también utilizados para la detección de *uso indebido* utilizando el tráfico de red como fuente de información, a través del análisis de los diferentes campos de los paquetes TCP/IP. Con el objetivo de reducir el índice de *falsos positivos*, existen trabajos basados en MLP [30] que en lugar de analizar aisladamente los paquetes, analizan una sesión completa de conexión. El presente trabajo propone un IDS basado en una ANN de tipo MLP.

Las técnicas que utilizan algoritmos de aprendizaje no supervisado como los mapas autoorganizados (SOM), han sido utilizados para clasificar los paquetes de red en sistemas que utilizan dos etapas de detección, siendo la segunda basada en un algoritmo tradicional de detección de anomalías. En estos casos la detección se ha llevado a cabo tanto a nivel de paquetes como a nivel de sesión [31]. En [32] y [33] se presentan dos sistemas de detección de anomalías que utilizan SOM a nivel de conexión de red, donde cada conexión es definida por un grupo de características.

En este trabajo se utiliza una selección de campos TCP/IP como entradas del sistema y se realiza una reducción de características a través de la aplicación del algoritmo PCA. Por esta razón, nos centraremos en aquellos trabajos que utilizan el tráfico de red como fuente de información.

La investigación presentada en [10] se apoya en redes de dos arquitecturas diferentes: una MLP, y un SOM con el objetivo de probar la eficiencia de éstas para analizar paquetes de red y clasificarlos como normales o peligrosos. Se tomaron 29 campos como datos de entrada a la ANN y se obtuvieron resultados de veracidad por encima del 90 % para ambas redes.

En [9] se desarrolla un *Sistema Automático para la Detección de Actividades Maliciosas (SADAM)*. El objetivo principal perseguido fue explorar las potencialidades del uso de un SOM como núcleo del sistema de análisis de un IDS. SADAM no analiza el contenido de los paquetes, y por tanto no es capaz de detectar los ataques que puedan realizarse a nivel de aplicación. El propósito es la detección de ataques cuya finalidad sea el escaneo de puertos o la denegación de servicios, sobre la base del análisis de los datos de la cabecera del paquete. Para la detección se tuvieron en cuenta 21 campos, además de una serie de variables estadísticas que muestran la evolución del tráfico en una cierta ventana de tiempo.

[34] propone una arquitectura de red recurrente para detectar intrusos a partir de los datos de los protocolos IP, ICMP, TCP, UDP. Se utiliza una *Time Lag Recurrent Network* con memoria de 10 instancias, basado en el hecho que un ataque no es autocontenido en un paquete, sino que se conforma por un conjunto de ellos. Se obtuvieron resultados favorables alcanzándose un 98.2% de detección de paquetes normales y un 98% para paquetes intrusivos.

En la investigación llevada a cabo en [11] se compara el desempeño de un *perceptrón multicapas* y una *red de Elman*. Para ambas se utilizaron como datos de

entrada 64 campos del protocolo de aplicación HTTP, sin tener en cuenta el resto de los protocolos del paquete. Mejía llega a conclusiones del alto poder de clasificación de ataques ante muestras desconocidas de ambas redes apreciándose una mayor eficiencia en la *red de Elman.*

En [12] se limitó al análisis del tráfico de paquetes TCP, obviando protocolos como UDP e ICMP. La información contenida en el área de datos de los paquetes fue asimismo tomada en cuenta. Por esta razón, además de usar la información más significativa del encabezado TCP, se seleccionaron los primeros 393 caracteres del contenido del paquete. Este límite se debe a que de los paquetes peligrosos que se encontraron a lo largo de la investigación el de mayor longitud era de 393 caracteres en el área de datos. En total fueron seleccionados 402 datos de entradas: 9 campos de la cabecera TCP y los primeros 393 caracteres del contenido del paquete. La red obtenida mostró un poder de detección por encima del 95 % y un alto poder de generalización.

[35] realiza una investigación para detectar intrusos con una ANN del tipo *perceptrón multicapas*. Se utilizaron como entradas al clasificador los 41 datos de la *KKD Cup 1999* obteniéndose un porcentaje de detección igual al 93.51%. Este trabajo no se adentra en los campos del paquete para la detección, sino que utiliza los datos propuestos por la *KDD* que describen el comportamiento del tráfico.

[36] utiliza una red neuronal *Growing Grid* (GG) para detectar intrusos, basándose en las ventajas de este tipo de arquitectura para el entrenamiento, puesto que la introducción de un nuevo conocimiento a través de un patrón no exige el aprendizaje de todos los anteriores, a diferencia del resto de las arquitecturas de ANN. En este caso se detectan ataques a nivel de conexión utilizando 5 entradas que describen la misma. Se realiza un estudio comparativo con un SOM demostrándose las ventajas de la GG.

Existen otros enfoques que utilizan redes neuronales evolutivas para lograr adaptabilidad en los sistemas de detección de intrusos [37].

Los trabajos examinados, a pesar de brindar resultados favorables, presentan importantes limitaciones. Todos estos trabajos se pueden resumir en dos grandes grupos. El primero de ellos se refiere a aquéllos que utilizan un número muy elevado de entradas para la red neuronal que implementan, lo que hace que el entrenamiento e incluso el uso de la misma sean ineficientes. En el segundo grupo, por el contrario, se encuentran aquellos trabajos en los que no se utilizan muchas entradas, para lo cual deben limitar su detección, obviando algunos protocolos o no contemplando los datos del contenido del paquete. De esta forma, la controversia entre cantidad óptima de entradas seleccionadas para el buen funcionamiento de la red y la cantidad necesaria de campos del paquete TCP/IP para detectar el mayor número de intrusiones, es uno de los problemas involucrados en este tipo de investigación que necesita aún ser resuelto. Debido a esto, existe otro grupo de investigaciones recientes encaminadas a la selección y disminución de la cantidad de datos necesarios para la clasificación a partir de los paquetes TCP/IP y las características generales del tráfico.

[38] realiza un trabajo de detección de intrusos utilizando como clasificador una Máquina de Soporte Vectorial (SVM). Las entradas utilizadas son las 41 variables definidas en la base de datos de la *KDD CUP 1999*, referidas a características del tráfico TCP/IP. Debido a que este clasificador sólo ofrece salidas binarias, para la

identificación de las 5 clases dentro de los patrones (normal, probing, DoS, U2R, R2L), se utilizan 5 SVM. Esta investigación tiene como resultado de interés la identificación de los datos más influentes para considerar que ha ocurrido una intrusión, teniendo en cuenta los diferentes tipos de ataques que contiene el conjunto de datos (probing, DoS, U2R, R2L).

[39] propone una arquitectura hardware reconfigurable basado en FPGA, donde se utiliza PCA tanto para reducir características como para clasificar los patrones. La clasificación se realiza basándose en el poder discriminatorio del modelo y definiendo rangos dentro de la nueva representación de las muestras en la dimensión de las PCs. Los resultados mostraron un porcentaje de efectividad en la detección de ataques del 99.2% con una tasa de *falsos positivos* inferior al 12.5%.

Por otra parte en [40], se realiza una investigación acerca de la efectividad del uso del PCA para la reducción de características. Utilizan como datos de entradas los referidos en la base de datos de la *KDD Cup 1999*. Su aporte fundamental radica en la comparación de un modelo de detección basado en el algoritmo del vecino más cercano (KNN) con y sin la aplicación del PCA. La misma comparación es llevada a cabo utilizando como clasificador un árbol de decisión. En ambos casos se presentan datos que demuestran las ventajas de la reducción de características previa a la clasificación.

Otro trabajo de interés en el tema es [41], que propone un clasificador para la detección de intrusos basado en SVM. Para la clasificación, los datos de entrada son sometidos a un redimensionamiento a través de PCA. El vector de entrada original cuenta con los 41 datos de la *KDD Cup 1999*, que describen el tráfico de red. En esta investigación se llegan a conclusiones de interés en cuanto a la conveniencia del uso del PCA. En todos estos trabajos se pone de manifiesto la efectividad de la reducción de los datos de entrada para la detección de intrusos. Aunque algunas de ellas utilizan PCA para reducir y clasificar, otras lo utilizan únicamente para redimensionar el vector de entrada y luego emplean otra heurística para la clasificación. En estos últimos, se deja claro, mediante experimentos, que la aplicación del PCA ofrece resultados de interés. No obstante, la mayoría de las investigaciones toman como datos de entrada las 41 variables referidas en la *KDD Cup 1999*, la cual describe el comportamiento del tráfico, pero no se adentra en el contenido del paquete TCP/IP.

Las investigaciones analizadas evidencian el potencial de las ANN para la detección de intrusos así como de la aplicación de PCA como mecanismo de reducción de características. La combinación de ambas técnicas puede aportar grandes beneficios al campo. Esta fusión permitiría mantener las ventajas de generalización y aprendizaje de las ANN y eliminar el problema de eficiencia introducido por un número grande de entradas.

4 Método propuesto para la detección de intrusos

En este trabajo se propone un método, representado gráficamente en la figura 5, basado en la aplicación conjunta de las técnicas PCA y ANN que va encaminado a resolver los problemas de la detección de intrusos identificados en el apartado anterior. La aplicación de PCA se dirige precisamente a nar la disyuntiva entre la

necesidad de tomar datos suficientes para la clasificación pero asegurando que sea el número mínimo para que el rendimiento del clasificador no se vea comprometido. Por lo tanto, con este método se busca eficiencia y escalabilidad en la aplicación de ANN para la detección de intrusos, manteniendo los niveles de eficacia logrados en trabajos anteriores.

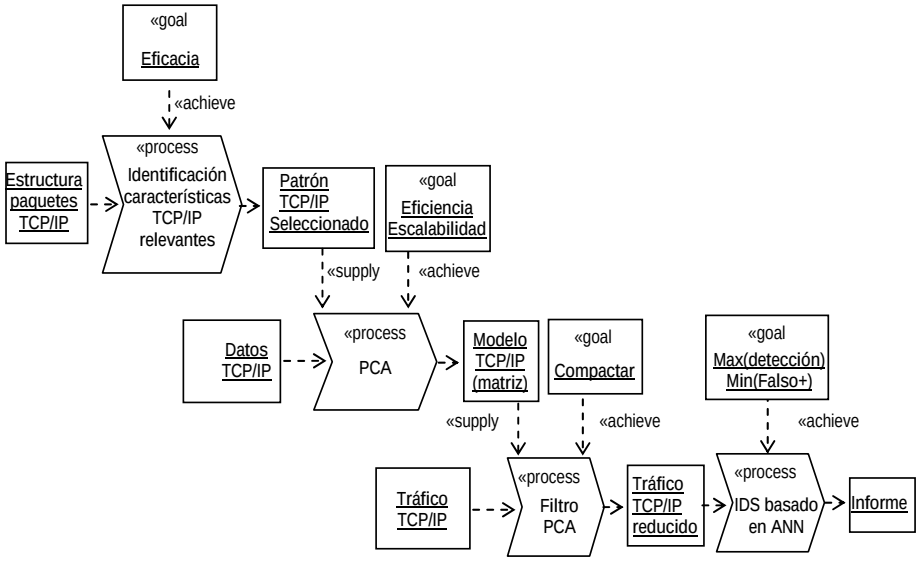

Fig. 5. Método propuesto para la detección de intrusos basado en la reducción de características

El método inicialmente propone el análisis de la estructura del paquete TCP/IP con el objetivo de identificar las características relevantes para la clasificación. Los datos TCP/IP que responden al patrón seleccionado son procesados por PCA para obtener la matriz de cambio de base que permite reexpresar los datos utilizando menos dimensiones, con lo cual se logra eficiencia y escalabilidad. De esta manera se obtiene un filtro PCA que permite compactar el tráfico TCP/IP que una vez reducido se introduce en el IDS basado en ANN para obtener los informes de clasificación.

En los siguientes apartados se estudia con más detenimiento las principales etapas de éste método.

4.1 Análisis y Selección de los Datos

La primera de las etapas del método propuesto consiste en la *Identificación de las características TCP/IP relevantes* (ver fig. 5). Su objetivo fundamental es determinar las características del tráfico de red que tienen importancia para la clasificación.

Cada uno de los protocolos que conforman a la familia TCP/IP incluye un gran número de información necesaria para su homólogo en la máquina receptora [18]. Muchos de esos datos, por su significado dentro de la comunicación, son utilizados

para identificar un paquete como intrusivo. Por tanto, con el objetivo de obtener un clasificador adecuado para detectar anomalías en el tráfico TCP/IP, es necesario realizar un estudio detallado de los campos de las cabeceras de los protocolos: IP, ICMP, TCP y UDP; y analizar su importancia para la clasificación de los paquetes. A partir de este estudio se han identificado 6 campos de la cabecera IP, 2 de ICMP, 7 TCP y 3 de UDP completando las primeras 18 características que se muestran resumidas en la tabla 2 (cada campo TCP/IP, con independencia de su tamaño en bytes, se considera una característica o rasgo diferente para la clasificación).

La capa de aplicación es portadora de numerosos ataques [42], fundamentalmente a través de los protocolos HTTP, NetBios y DNS. A este nivel existe un gran número de protocolos diferentes que incluyen campos disímiles dentro de su encabezado. Debido a esto, resulta complicado precisar los campos que podrían resultar de interés para la clasificación. La solución que se propone es tomar un número de bytes de esta capa por cada paquete TCP/IP que pueden resultar significativos. Existen trabajos recientes que han monitorizado tráfico de red y concluyen como el paquete de ataque más grande encontrado uno de 393 bytes en el área de datos [12] que se corresponde con la capa de aplicación. Siguiendo este criterio la propuesta se concreta en incluir entre los datos importantes para la clasificación los primeros 400 bytes del área de datos de los paquetes TCP/IP (Tabla 2). Cada byte es considerado como una característica diferente.

Tabla 2. Campos TCP/IP seleccionados.

# Campo	# Rasgo	Descripción	Protocolo	Tamaño
1	1	ToS (Tipo de servicio)	IP	1 byte
2	2	Longitud Total	IP	2 bytes
3	3	Banderas	IP	3 bits
4	4	Tiempo de Vida	IP	1 byte
5	5	Protocolo (se refiere al protocolo de nivel superior)	IP	1 byte
6	6	Cantidad de Opciones *(si las hay)*	IP	1 byte
7	7	Tipo	ICMP	1 byte
8	8	Código	ICMP	1 byte
9	9	Puerto Fuente	TCP	2 bytes
10	10	Puerto Destino	TCP	2 bytes
11	11	Número de Secuencia	TCP	4 bytes
12	12	Número de acuse de recibo	TCP	4 bytes
13	13	Banderas	TCP	6 bits
14	14	Ventana	TCP	2 bytes
15	15	Cantidad de Opciones *(si las hay)*	TCP	1 byte
16	16	Puerto Origen	UDP	2 bytes
17	17	Puerto destino	UDP	2 bytes
18	18	Longitud del Mensaje	UDP	2 bytes
19	19-418	Primeros bytes del área de datos (400)	APLIC.	400 bytes
20	419	Tiempo transcurrido desde la llegada del último paquete.	-	4 bytes

Otro dato de suma importancia es el tiempo transcurrido entre la transmisión de un paquete y otro. Esto se debe a que muchos ataques basan su efectividad en la emisión frecuente de información para inhabilitar a la máquina objetivo, como por ejemplo DoS (Denegación de Servicio) y los DDoS (DoS Distribuido). Por tanto, este tiempo es otro de los datos que se empleará en la clasificación (recogido como campo número 20 en la tabla 2).

Como conclusión, se propone un total de 419 características o rasgos diferentes por cada paquete TCP/IP. Estos rasgos conforman el *patrón TCP/IP seleccionado* del método propuesto (ver fig. 5). El elevado número de rasgos diferentes seleccionados para la clasificación hace que el vector de entrada de la ANN sea muy grande y, en consecuencia, también lo sea el tiempo de entrenamiento por ser directamente proporcionales.

En el caso de las ANN, el número de nodos de entrada tiene relación directa con el tamaño de las capas ocultas, ya que cada nodo representa características diferentes del patrón que se utilice. Esto es, una mayor cantidad de nodos de entrada implica registrar en la red una mayor cantidad de información para distinguir las clases entre sí en el espacio de características. El tiempo requerido para el entrenamiento de la ANN se incrementa en forma cuadrática con la cantidad de datos de entrada [43]. Por tanto, es recomendable que la incorporación de nuevos datos de entrada a la ANN se realice sólo si contribuyen a mejorar la calidad de la clasificación de manera significativa.

4.2 Aplicación de PCA

Para lograr la disminución de características del vector de entrada, en una segunda fase, se propone aplicar un método matemático capaz de reducir el tamaño del vector de entrada sin que se produzca apenas pérdida de información. Este método se denomina análisis de componentes principales (PCA). El objetivo fundamental de este método es obtener un modelo de los datos (plasmado en una matriz de transformación) que permita eliminar dimensiones redundantes y ruidosas.

Para la obtención del modelo es preciso, previamente, contar con un conjunto de datos que respondan al *patrón TCP/IP seleccionado* y que sirvan de entrenamiento al modelo. Este conjunto debe incluir tanto paquetes intrusivos como normales. En este trabajo se utiliza la variante de capturar paquetes reales para obtener los conjuntos de entrenamiento y prueba. Utilizando *Nessus* [28] para los paquetes de ataques y *Ethereal* [44] como *sniffer* para la captura de paquetes. Mediante estas herramientas se capturaron paquetes normales representativos del funcionamiento de la red en una máquina y se utilizó un servidor Nessus para explorar la misma máquina y capturar los paquetes de ataques. De esta manera se obtuvieron varios conjuntos de datos, seleccionando el más representativo que contiene 1.196 paquetes. De ellos, 597 paquetes corresponden a ataques y 599 a paquetes normales. Adicionalmente, se implementó una aplicación para formatear los datos de los paquetes según el patrón TCP/IP seleccionado, obteniéndose de esta manera los *Datos TCP/IP* que se señala en el método.

Una vez obtenido un conjunto de datos de prueba adecuado, los 419 rasgos que se tienen en cuenta dentro de los *Datos TCP/IP* (siguiendo el patrón seleccionado) serán sometidos al algoritmo de redimensionamiento a través de la aplicación de *PCA* en busca de eficiencia y escalabilidad en la clasificación.

En este caso se cuenta con un conjunto de datos, *Datos TCP/IP,* expresados en una matriz de 419 filas que se corresponden con las variables del *Patrón TCP/IP seleccionado* y 1.196 columnas que son la cantidad de paquetes capturados:

$$X = \begin{bmatrix} X_{11} & \cdots & X_{1j} \\ \vdots & \ddots & \vdots \\ X_{i1} & \cdots & X_{ij} \end{bmatrix}_{419\ X\ 1196} \qquad (6)$$

donde i = 419

j = 1.196

El objetivo del algoritmo es encontrar los PC que conformen la matriz de transformación capaz de expresar los datos en los nuevos ejes coordenados (los ejes de los PC). Siendo el número de PC menor que 419 para lograr la reducción del vector de entrada original.

Con el objetivo de aplicar el algoritmo PCA sobre el conjunto de datos seleccionado se ha utilizado MatLab por las facilidades que proporciona para el trabajo con matrices y vectores. En este caso se ha empleado un grupo de herramientas específico para el análisis exploratorio de los datos, *PLS ToolBox*, puesto que proporciona una buena colección de herramientas para el análisis multivariado de datos.

Para el conjunto de datos de 1.196 muestras y 419 variables por muestra que se utilizó, se obtuvieron modelos de hasta 20 componentes principales (PC) donde cada uno de ellos era capaz de expresar una parte de la información total del conjunto original. En la tabla 3 se muestran los valores de variabilidad expresados por los modelos que superaron el 80 % en el poder de expresión de los datos originales.

Tabla 3. Variabilidad expresada por modelo.

Cantidad de componentes principales	**Porcentaje de varianza explicada**
10	82.26
11	84.34
12	86.36
13	88.26
14	89.70
15	90.97
16	92.19
17	93.22
18	94.01
19	94.69
20	**95.25**

Dentro de los modelos de la tabla 3 se seleccionó el correspondiente a 20 PC, ya que reduce de manera significativa el vector inicial (de 419 variables) y logra expresar un alto porcentaje de la información original. Con esta nueva dimensión se logra disminuir un 95.2 % del tamaño del vector original, repercutiendo directamente en el buen funcionamiento del clasificador neuronal. Al mismo tiempo el modelo de 20 PC es capaz de expresar más del 95% de la información original. Estas características hacen del modelo de 20 PC un modelo de valor para representar los datos utilizados correspondientes al tráfico de red.

Adicionalmente a las capacidades del modelo para expresar la variabilidad de los datos, es importante evaluar su poder de discriminación con respecto a las diferentes clases que contienen éstos. En nuestro caso existen dos clases dentro de los datos: *ataques* y *no ataques*. Luego, es conveniente que en la nueva estructuración de los datos existan diferencias entre estas clases.

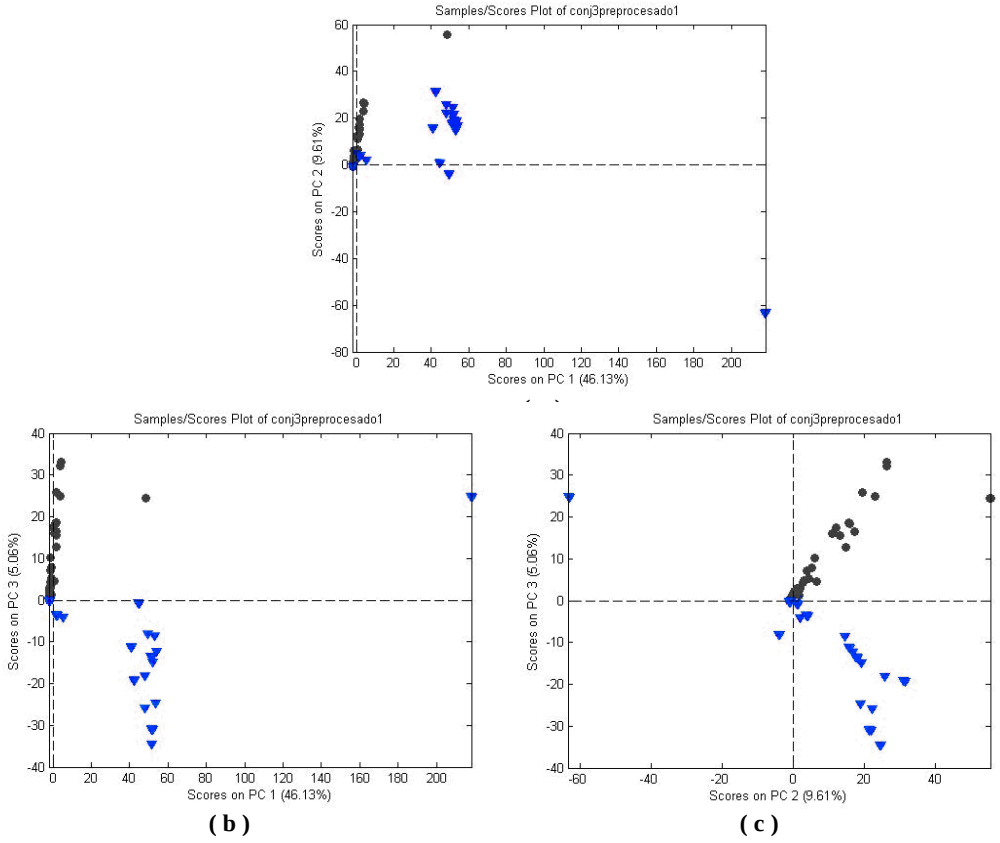

Fig. 6. (a) Scores de PC1 vs PC2, (b) Scores de PC1 vs PC3 y (c) Scores de PC2 vs PC3

El poder de discriminación del modelo se observa a través de las gráficas de los marcadores (*scores)*. Los *scores* son las muestras expresadas en los nuevos ejes coordenados (los PC). Para una mejor visualización, estos datos serán expuestos en 2

dimensiones, ubicando las muestras en gráficos de una PC frente a otra. En el modelo de 20 PC existen 171 posibles variaciones en las que se combinan las diferentes PC para formar un gráfico. Por la cantidad de variaciones posibles serán analizadas solamente las 3 primeras PC (una contra otra) que son las que expresan la mayor variabilidad de los datos.

La figura 6 refleja la distribución de las muestras en los gráficos (a) de PC1 frente a PC2, (b) PC1 frente a PC3 y (c) PC2 frente a PC3 respectivamente. Las clases están diferenciadas en el gráfico, siendo los puntos los ataques y los triángulos los paquetes normales.

Los gráficos muestran que los datos expresados en función de los componentes principales tienen tendencia a agruparse por clases, dando muestra del poder de discriminación del modelo. Se evidencia que PC1 tiene alto poder discriminatorio ya que es posible obtener un valor en este eje (el eje de PC1) que logre dividir con bastante éxito las dos clases. Sin embargo, con PC2 no es posible discriminar las clases. PC3 también posee poder de discriminación entre clases, siendo posible encontrar un valor en ese eje (el eje PC3) que logre separar las clases.

El hecho de que el modelo tenga alta capacidad de expresar la variabilidad de los datos (por encima del 95%) y que contenga componentes principales con poder de discriminación (PC1, PC3) permite concluir que es un buen modelo de los datos originales.

4.3 Aplicación del Filtro PCA

El modelo seleccionado de 20 PC proporciona una matriz de transformación de los datos —lo que se recoge en la representación gráfica del método (figura 5) como *Modelo TCP/IP*—. Dicha matriz constituye el elemento principal del *filtro PCA* sobre el *Tráfico TCP/IP* debido a que su aplicación a los datos capturados proporciona los datos expresados a partir de la nueva base (los PC). La matriz tiene tantas columnas como variables contenía el conjunto de datos (419 en este caso) y tantas filas como la cantidad de PC que se seleccionaron en el modelo (20).

$$P = \begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{m1} & \cdots & P_{mn} \end{bmatrix}_{20\ X\ 419} \qquad (7)$$

donde m = 20

n = 419

Cada fila de la matriz **P** es un PC expresado como combinación lineal de las 419 variables originales. Luego, cada elemento P_{ij} representa el coeficiente de la variable *i* (de las 419) en la combinación lineal que expresa el PC *j*.

Los datos del *Tráfico TCP/IP* deben expresarse en una matriz X que contenga 419 filas correspondientes a las variables de cada paquete y tantas columnas como paquetes se capturen (*CantPaq*).

$$PX = Y$$

$$\begin{bmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{m1} & \cdots & P_{mn} \end{bmatrix}_{20\,X\,419} \begin{bmatrix} X_{11} & \cdots & X_{1j} \\ \vdots & \ddots & \vdots \\ X_{i1} & \cdots & X_{ij} \end{bmatrix}_{419\,X\,CantPaq} = Y_{20\,X\,CantPaq} \quad (8)$$

$$Y = \begin{bmatrix} Y_{11} & \cdots & Y_{1q} \\ \vdots & \ddots & \vdots \\ Y_{p1} & \cdots & Y_{pq} \end{bmatrix}_{20\ X\ CantPaq} \quad (9)$$

donde p = *20*

q = *CantPaq*

Con esta transformación se obtiene la matriz **Y** que expresa cada uno de los paquetes TCP/IP en función de los PC seleccionados. Este resultado se corresponde con el *Tráfico TCP/IP reducido* que va a servir de entrada al clasificador neuronal.

4.4 Desarrollo del IDS basado en un Clasificador Neuronal

Una vez reducidos los datos del tráfico, deben entrar al *IDS basado en ANN* para la obtención de los *informes* de clasificación. La ANN utilizada en esta investigación es un *perceptrón multicapa*. La elección se fundamenta en la sencillez de su uso, que permite centrar la atención en los resultados brindados por la reducción de características aplicada a este tipo de problemas, lo que constituye el objetivo principal del trabajo. Esta arquitectura de red utiliza entrenamiento supervisado, por lo que necesita en esta etapa el conjunto de entradas así como las salidas esperadas para dichas entradas. Para la actualización de los pesos durante el entrenamiento se utilizó la propagación hacia atrás del error (algoritmo de *Backpropagation)*.

Una vez seleccionado el tipo de ANN a utilizar, es preciso entrenarla, para definir la arquitectura que ofrece mejor rendimiento. El primer paso consiste en determinar el número de entradas y salidas de la ANN. Las entradas quedaron definidas por las 20 PC del modelo de datos y como salidas se tiene en cuenta únicamente una salida binaria, donde *0* se corresponde con paquetes clasificados como normales y *1* como peligrosos.

Para el entrenamiento de la ANN se utilizó MatLab por las facilidades que brinda para estos temas. En las pruebas realizadas, se utilizaron como funciones de transferencia de las neuronas las funciones *logsig* y *tansig*.

En el entrenamiento se utilizó solamente las tres cuartas partes de los datos totales (897 paquetes con una cantidad proporcional de paquetes de ambas clases), reservando el resto para las pruebas. El rendimiento de las redes entrenadas se midió calculando la media cuadrática del error (función *mse*) que se obtenía tras evaluar la red con los datos de prueba y los valores deseados para estos datos.

Se entrenaron alrededor de 350 redes a las cuales se le cambiaban ligeramente las características para comparar los resultados. Las 20 redes con mejores comportamientos se muestran en la tabla 4 organizadas ascendentemente según el valor del error de validación obtenido, siendo por tanto la primera la que mejor resultados brindó. Para el entrenamiento fueron definidas 300 épocas como máximo y un error límite de 0,00001. Todas las redes entrenadas llegaron al valor de error esperado, completándose un número de épocas considerablemente menor que el tope máximo.

También se realizaron pruebas añadiendo una nueva capa oculta a la red. Los resultados correspondientes a las 20 mejores redes encontradas se muestran en la tabla 5, ordenadas por el error de validación. En la tabla 5, la primera red se corresponde con la ANN de 4 capas de mejor resultado dentro de las entrenadas, siendo éste inferior a los obtenidos con ANN de 3 capas (ver tabla 4).

Tabla 4. Redes Neuronales entrenadas de 3 capas.

Cantidad de Neuronas			**Función de Transferencia**			**Épocas de entrenamiento**	**Errorde validación**
Capa Entrada	**Capa Oculta**	**Capa Salida**	**Capa Entrada**	**Capa Oculta**	**Capa Salida**		
20	**36**	**1**	**tansig**	**tansig**	**logsig**	**12**	**6,47-16**
20	62	1	logsig	logsig	logsig	8	8,42E-08
20	42	1	logsig	logsig	logsig	7	1,25E-07
20	12	1	logsig	logsig	logsig	5	1,64E-07
20	9	1	logsig	logsig	logsig	20	1,80E-07
20	17	1	logsig	logsig	logsig	7	4,88E-07
20	52	1	tansig	tansig	logsig	10	5,68E-07
20	85	1	tansig	logsig	logsig	42	5,95E-07
20	74	1	tansig	logsig	logsig	16	6,54E-07
20	4	1	logsig	logsig	logsig	8	6,61E-07
20	83	1	logsig	logsig	logsig	8	9,30E-07
20	57	1	tansig	logsig	logsig	39	1,00E-06
20	44	1	logsig	logsig	logsig	25	1,12E-06
20	76	1	logsig	logsig	logsig	14	1,19E-06
20	49	1	tansig	logsig	logsig	15	1,27E-06
20	52	1	logsig	logsig	logsig	9	1,30E-06
20	57	1	tansig	tansig	logsig	10	1,33E-06
20	60	1	tansig	tansig	logsig	10	1,38E-06
20	10	1	logsig	logsig	logsig	10	1,47E-06

Las redes de tres capas tienen ventajas sobre las de mayor número de éstas en cuanto a velocidad de funcionamiento. Por tanto, es recomendable utilizar la red de menor cantidad de capas que logre mejor rendimiento. La ANN que muestra mayor eficiencia en la etapa de pruebas es de tres capas con 20 neuronas en la capa de entrada, 36 en la capa oculta y una en la capa de salida, utilizándose las funciones de transferencia *tansig* y *logsig* respectivamente. En consecuencia, la arquitectura de la ANN seleccionada se corresponde con la que se muestra en la figura 7.

Tabla 5. Redes Neuronales entrenadas de 4 capas.

Cantidad de Neuronas				Función de Transferencia				
Capa Entrada	**Capa Oculta 1**	**Capa Oculta 2**	**Capa Salida**	**Capa Entrada**	**Capa Oculta 1**	**Capa Oculta 2**	**Capa Salida**	**Error de validación**
20	20	33	1	tansig	tansig	logsig	logsig	2,40E-07
20	12	35	1	tansig	tansig	logsig	logsig	5,23E-07
20	17	26	1	tansig	tansig	logsig	logsig	6,85E-07
20	16	24	1	tansig	tansig	logsig	logsig	7,18E-07
20	19	37	1	tansig	tansig	logsig	logsig	7,45E-07
20	13	28	1	tansig	tansig	logsig	logsig	7,73E-07
20	16	38	1	tansig	tansig	logsig	logsig	8,67E-07
20	14	27	1	tansig	tansig	logsig	logsig	9,67E-07
20	10	29	1	tansig	tansig	logsig	logsig	9,75E-07
20	15	32	1	tansig	tansig	logsig	logsig	1,13E-06
20	11	25	1	tansig	tansig	logsig	logsig	1,20E-06
20	10	24	1	tansig	tansig	logsig	logsig	1,32E-06
20	18	33	1	tansig	tansig	logsig	logsig	1,41E-06
20	13	40	1	tansig	tansig	logsig	logsig	1,46E-06
20	11	26	1	tansig	tansig	logsig	logsig	1,50E-06
20	14	40	1	tansig	tansig	logsig	logsig	1,52E-06
20	19	20	1	tansig	tansig	logsig	logsig	1,54E-06
20	17	36	1	tansig	tansig	logsig	logsig	1,55E-06
20	17	38	1	tansig	tansig	logsig	logsig	1,58E-06
20	10	37	1	tansig	tansig	logsig	logsig	1,61E-06

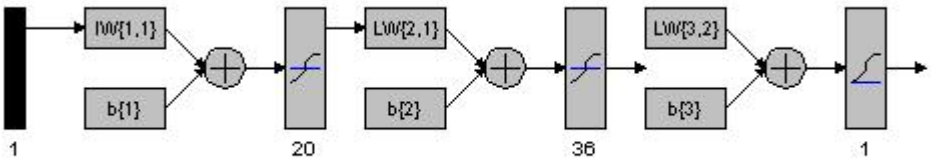

Fig. 7. Arquitectura de ANN seleccionada.

5 Pruebas y resultados

Una vez desarrollado el método propuesto disponemos de: un filtro capaz de compactar el tráfico TCP/IP capturado, un tráfico de red de prueba y un IDS basado en ANN. Con estos ingredientes construimos nuestro escenario de pruebas con la finalidad de comparar la eficacia y el rendimiento de la propuesta frente a otros enfoques análogos en los que no se haya empleado la reducción de características. Para ello se ha dividido el conjunto de datos (tráfico TCP/IP) en cuatro grupos de 299 paquetes. Con esta división se ha realizado una validación cruzada en la cual se utilizó uno de los grupos como conjunto de prueba y los restantes tres como conjunto de entrenamiento. Los cuatro procesos de entrenamiento sobre la ANN seleccionada arrojaron resultados satisfactorios en cuanto al error medido en las pruebas. Estos resultados se muestran en la tabla 6. En estos casos también se tomó 0.00001 como error máximo permitido en el entrenamiento.

Tabla 6. Error de Validación de la ANN seleccionada.

Entrenamiento	Error de Validación
1	6,47E-16
2	1,51E-9
3	8,10E-12
4	2,32E-10

Una vez entrenada la ANN seleccionada se evaluaron los resultados que ésta brinda frente a los datos de prueba para determinar la cantidad de *falsos positivos* y *falsos negativos* que aparecían tras la clasificación. Las pruebas con los primeros 3 grupos arrojaron como resultado un 100% de efectividad tanto para la clasificación de los ataques como para la de los paquetes normales. En la última prueba se detectó un falso negativo, teniendo el clasificador un 100% de efectividad para la clasificación de paquetes normales y 99.93% de efectividad para paquetes intrusivos.

En cuanto al rendimiento, el uso de PCA como algoritmo para redimensionar el vector de entrada proporciona ventajas innegables para la velocidad del ANN. Para las ANN, el tiempo requerido para el entrenamiento se incrementa en forma cuadrática con la cantidad de datos de entrada. De esta manera, se puede afirmar que

la ANN implementada con 20 entradas es, aproximadamente, 438 veces más rápida en tiempo de entrenamiento que la que se hubiera obtenido teniendo en cuenta las 419 entradas. Esto aporta grandes beneficios para los administradores de red, ya que los tiempos de entrenamiento del clasificador dejan de ser un problema para la utilización del detector de intrusos.

Adicionalmente, los resultados muestran que la red de 20:36:1 neuronas, tiene alto poder de generalización, puesto que es capaz de clasificar satisfactoriamente datos para los cuales no fue entrenada. El rendimiento alcanzado por la ANN se muestra en la tabla 7.

Tabla 7. Rendimiento de la ANN seleccionada.

Medidas de Rendimiento	Rendimiento
Efectividad de aciertos en paquetes normales	100%
Efectividad de aciertos en paquetes intrusivos	99.33%
Efectividad de aciertos en todas las pruebas	99.91%

Dentro del conjunto de datos probados, se obtuvo 100% de efectividad para la clasificación de paquetes normales, 99.33% para la clasificación de ataques, para un total de efectividad de 99.91%.

6 Conclusiones

En este trabajo se ha propuesto un método para la detección de intrusos mediante ANN que se basa en compactar los vectores de entrada de forma significativa sin apenas perder información relevante. De esta forma se subsanan algunos de los problemas más importantes asociados con la utilización de ANN a medida que la información crece: el incremento exponencial de los tiempos de aprendizaje y el bajo rendimiento en la detección.

Del desarrollo del método se ha obtenido un modelo PCA capaz de expresar más del 95% de la información original con una reducción también de más del 95% del tamaño original. Tras aplicar el filtro PCA obtenido a partir del anterior modelo PCA a los datos de entrada y entrenar el detector basado en ANN propuesto para las pruebas con dichos datos, los resultados avalan ampliamente la validez del método propuesto, reduciendo el tiempo de entrenamiento más de 400 veces al tiempo que es capaz de mantener los mismos niveles de eficacia del detector original.

Además, gracias a la aplicación del método propuesto, se ha podido incrementar notablemente la información de entrada utilizada para la detección sin que esto suponga un impacto importante en los tiempos de aprendizaje y detección del sistema, logrando unos niveles de eficacia muy superiores con un 0% de falsos positivos y tan solo un 0,08% de falsos negativos.

En la actualidad estamos trabajando en mejorar el método propuesto para conseguir que todo el proceso se pueda realizar de forma desatendida. En concreto, estamos profundizando en un nuevo modelo para la obtención dinámica de los filtros PCA; en el estudio cualitativo de los parámetros relevantes del tráfico TCP/IP sin restricciones de tamaño y; finalmente, en la identificación de modelos de redes neuronales no supervisados que puedan resultar más eficaces para la detección de intrusos.

Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Educación y Ciencia de España bajo el proyecto de investigación TIN2006-04081 y por la Generalitat Valenciana bajo el proyecto de investigación GV/2007175.

Referencias

1. IBM Internet Security Systems, http://www.iss.net
2. CERT, http://www.cert.org
3. A. Mchugh, J., Christie, A., Allen, J.: Defending yourself: the Role of Intrusion Detection Systems. IEEE Software. 17, 42--51 (2000)
4. Debar, H., Viinikka, J.: Introduction to Intrusion Detection and Security Information Management. In: Foundations of Security Analysis and Design III FOSAD 2005. LNCS, vol. 3655, pp. 207--236. Springer, (2005)
5. Kruegel, C., Valeur, F., Vigna, G.: Intrusion Detection and Correlation: Challenges and Solutions. Springer, (2005)
6. Ghosh, A., Schwartzbard.: A Study in using Neural Networks for Anomaly and Misuse Detection. In: 8th USENIX Security Symposium, pp. 23--36. Washington (1999)
7. Ghosh, A., Michael, C., Schatz, M.: A Real-time Intrusion Detection System based on Learning Program Behavior. In: Debar, H., Me, L., Wu, S. (eds.) RAID 2000. LNCS, vol. 1907, pp. 93--109. Springer, (2000)
8. Ghosh, A., Schwartzbard, M., Schatz, M.: Learning program behavior profiles for intrusion detection. In: 1st USENIX Workshop Intrusion Detection and Network Monitoring, pp. 51-- 62. Santa Clara (1999)
9. Cortada, P., Sanroma, G., Garcia, P.: IDS basado en mapas autoorganizados. Technical Report, RedIRIS (2002)
10. Grediaga, A.: Utilización de Redes Neuronales para la Detección de Intrusos. Technical Report, University of Alicante (2000)
11. Mejia, J.: Sistema de Detección de Intrusos en Redes de Comunicaciones utilizando Redes Neurales. Technical Report, University of las Americas (2004)
12. Perez, C., Britto, J., Isaza, G.: Aplicación de Redes Neurales para la Detección de Intrusos en Redes y Sistemas de Información. Scientia et Técnica, 27, 225-230 (2005)
13. Baluja, W.: Acercamiento a los sistemas detectores de intrusos. Telem@tica Revista Digital de las Tecnologías de la Información y las Comunicaciones, ISSN: 1729-3804, 2001

14. Zurutuza, U.: Revisión del estado actual de la investigación en el uso de data mining para la detección de intrusiones. Escuela Politécnica Superior de Mondragón, Departamento. Informática, 2005.
15. Rubin, S., Jha, S., Miller, B.: Language-based Generation and Evaluation of NIDS Signatures. In: IEEE Symposium on Security and Privacy, pp. 3--17. IEEE Computer Society, (2005)
16. Kruegel, C., Kirda, E., Mutz, D., Robertson, W., Vigna, G.: Polymorphic Worm Detection using Structural Information of Executables. In: Valdes, A., Zamboni, D. (eds.) RAID 2005. LNCS, vol. 3858, pp. 207--226. Springer, (2005)
17. Liang, Z., Sekar, R.: Automatic Generation of Buffer Overflow Attack Signatures: An Approach based on Program Behavior Models. In: 21st Annual Computer Security Applications Conference, pp. 215--224. IEEE Computer Society, (2005)
18. Macia, F., Gil, J., Monllor, C.: Intranets de getión de red. Publicaciones de la Universidad de Alicante, Alicante (2002)
19. Denning, D.: An Intrusion Detection Model. IEEE Transactions on Software Engineering. 13, 222--232 (1987)
20. Freeman, J., Skapura, D.: Neural Networks. Algorithms, Applications, and Programming Techniques. Addison-Wesley (1991)
21. Mitchell, T.: Machine Learning. McGraw Hill (1997)
22. Fritzke B.: Growing Cell Structures- A Self-organizing Network for Unsupervised and Supervised Learning. Technical Report TR-93-026, International Computer Science Institute (1993)
23. Fritzke, B.: A Growing Neural Gas Network Learns Topologies. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) NIPS 1994. NIPS, vol. 7, pp. 625--632. MIT Press, (1995)
24. Esbensen, K.: Multivariate Data Análisis – in practice. Camo Press AS (2002)
25. Lippmann, R., Fried, D., Graf, I., Haines, J., Kendall, K., McClung, D., Weber, D., Webster, S., Wyschogrod, D., Cunningham, R., Zissman, M.: Evaluating Intrusion Detection Systems: the 1998 DARPA Off-line Intrusion Detection Evaluation. In: the 2000 DARPA Information Survivability Conference and Expositon, (2000)
26. Rieck, K., Laskov, P.: Detecting Unknown Network Attacks using Language Models. In: Buschkes, R., Laskov, P. (eds.) DIMVA 2006. LNCS, vol. 4064, pp. 74--90. Springer, (2006)
27. UCI KDD archive, Universidad de California, http://kdd.ics.uci.edu, septiembre 2006.
28. Tenable Network Security, http://www.nessus.org
29. Chebrolua, S., Abrahama, A., Thomasa J.: Feature Deduction and Ensemble Design of Intrusion Detection Systems. Computers & Security, 24, 295--307 (2005)
30. Lippmann, R., Cunningham, R.: Improving Intrusion Detection Performance using Keyword Selection and Neural Networks. Computer Networks. 34, 597--603 (2000)
31. Zanero, S., Savaresi, S.: Unsupervised Learning Techniques for an Intrusion Detection System. In: 19th ACM Symposium on Applied Computing, pp. 412--419. ACM Press, (2004)
32. Ramadas, M., Ostermann, S., Tjaden, B.: Detecting Anomalous Network Traffic with Self- Organizing Maps. In: Vigna, G., Jonsson, E., Kruegel, C. (eds.) RAID 2003. LNCS, vol. 2820, pp. 36--54. Springer, (2003)
33. Lichodzijewski, P., Zincir-Heywood, A., Heywood, M.: Dynamic Intrusion Detection using Self-Organizing Maps. In: 14th Annual Canadian Information Technology Security Symposium, (2002)
34. Pinacho, P., Valenzuela, T.: Una propuesta de IDS basada en Redes Neurales Recurrentes. Technical Report, University of Santiago de Chile (2002)

35. Escandon, R.: Empleo de Redes Neuronales en la Detección de Intrusos. Technical Report, Jose Antonio Echevarría Institute of Technology (2005)
36. Mora, F.J., Macia, F., Garcia, J.M., Ramos, H.: Intrusion Detection System Based on Growing Grid Neural Network. In: 13th IEEE Mediterranean Electrotechnical Conference, pp. 839- 842, IEEE Computer Society, (2006)
37. Han, S., Cho, S.: Evolutionary Neural Networks for Anomaly Detection Based on the Behavior of a Program. IEEE Transactions on Systems, Man and Cybernetics, 36, 559—570 (2006)
38. Mukkamala, S., Sung, A.: Feature Ranking and Selection for Intrusion Detection Systems Using Support Vector Machines. Technical Report, Institute of Minería y Tecnología, Nuevo México (2003)
39. Nguyen, D.: A Reconfigurable Architecture for Network Intrusion Detection using Principal Component Analysis. Technical Report, Northwestern University Evanston (2005)
40. Bouzida, Y.: Efficient Intrusion Detection Using Principal Component Analysis. Technical Report Departement RSM GET/ ENST Bretagne (2005)
41. Xu, X., Wang, X.: An Adaptive Network Intrusion Detection Method Based on PCA and Support Vector Machines. In: Li, X., Wang, S., Dong, Z. (eds.) ADMA 2005. LNCS, vol. 3584, pp. 696--703. Springer, (2005)
42. Robertson, W., Vigna, G., Kruegel, C., Kemmerer, R.: Using Generalization and Characterization Techniques in the Anomaly-based Detection of Web Attacks. In: 13th Annual Network and Distributed System Security Symposium. The Internet Society, (2006)
43. Kauzoglu, T.: Determining Optimum Structure for Artificial Neural Networks. Proceedin of then 25th Annual Technical Conference and Exhibition of the Remote Sensing Society, Cardiff, K, pp 675-682, Septiembre 1999
44. Ethereal, http://www.ethereal.com