Utilización de redes neuronales para la detección de intrusos

Mamani Villca Gary Américo
Universidad Mayor De San Andrés
Carrera De Informática
Simulación de Sistemas
hey100@hotmail.com

RESUMEN.

El incremento de intrusiones y malos usos en los sistemas informáticos y redes internas de una gran cantidad de empresas, ha provocado un aumento en la preocupación por la seguridad informática. Desde hace algún tiempo se vienen aplicando medidas basadas en cortafuegos y en sistemas de detección de intrusiones (IDS). En este documento se propone una alternativa al problema de

Los IDS basados en reglas, utilizando dos redes neuronales distintas, una red perceptrón multicapa, se realizan una serie de experimentos y se muestran los resultados que, han sido mucho mejores de lo que en un principio cabría esperar.

Palabras Clave

Multicapa, intrusos, red, Perceptrón.

1. INTRODUCCIÓN

La seguridad en los computadores ha sido estudiada como una disciplina desde 1970. Y se refiere a las medidas y controles que protegen los sistemas de información contra la negación de servicio y la ausencia de autorización (accidental o intencionadamente) para revelar, modificar, o destruir los sistemas de información y datos

Sin embargo, podemos entender como seguridad una característica de cualquier sistema, informático o no, que nos indica que ese sistema está libre de todo peligro, daño o riesgo, y que es, en cierta manera, infalible.

A grandes rasgos se entiende que mantener un sistema seguro, o fiable, consiste básicamente en garantizar tres aspectos : confidencialidad, integridad y disponibilidad.

Sin embargo, la comunidad científica dedicada a la seguridad de los sistemas computadores, se ha dado cuenta que las técnicas de prevención no son suficiente para proteger a los sistemas, tal y como se comprobó en 1988 con el gusano de Internet, y en el año 2000 con el llamado ataque Distributed Denial of Service (DDoS) y que paró la mayoría de los sitios comerciales, incluyendo Yahoo y la CNN que aunque estaban protegidos mediante técnicas de prevención. Estos incidentes demostraron que las técnicas de prevención son inadecuadas, aunque la verdadera razón es que los sistemas de información desarrollados no son infalibles, ya que pueden tener grietas en la implementación, así como fallos en el diseño.

La detección de intrusos fue propuesta como complemento de las técnicas de prevención. Una intrusión se define como una violación de la política de seguridad del sistema; la detección de intrusos de este modo se refiere a los mecanismos que hay desarrollados para detectar la violación de la política de seguridad de los sistemas, se basa pues, en asumir que la actividad intrusa es notablemente diferente de la actividad normal y por lo tanto se puede detectar. La detección de intrusos no se utiliza para remplazar las técnicas de prevención tales como la autenticación y el control de accesos, sino en combinación con las medidas de seguridad existentes par evitar las acciones que intentan puentear los sistemas de control e la seguridad. Por lo tanto, la detección de intrusos, se considera habitualmente como una segunda línea de defe nza de las redes de computadores.

Debido a esta situación, tanto empresas como organismos se están concienciando de la necesidad de aplicar una mayor seguridad a sus sistemas informáticos. Es a partir de aquí donde este documento cobra vida, proponiendo una alternativa a las soluciones clásicas mediante el uso de redes neuronales, centrándonos en la red perceptrón

multicapa y en el Mapa Auto -Organizativo.

2. PROPUESTA DE EXPERIMENTACIÓN

Mediante esta propuesta se pretende probar si mediante una red neuronal es posible realizar la distinción entre paquetes que representan flujo normal en una red de ordenadores, y paquetes que Primer Congreso Iberoamericano de Seguridad 3 representan ataques. Concretamente realizaremos la experimentación con una red perceptrón multicapa, y con un mapa auto -organizativo MAO. En cuanto al tipo de IDS que implementaremos utilizando redes neuronales, será un IDS de host.

2.1. Elección de los datos

Lo primero que hay que plantearse a la hora de entrenar una red neuronal, es qué datos van a contener las muestras que utilice como entradas, tanto para entrenar como para chequear. Esta es la decisión más importante que se puede tomar, ya que de ella dependerá en gran medida la capacidad de distinción que pueda adquirir la red neuronal una vez finalizado el entrenamiento. Para la elección de estos datos, se han tomado como referencia los datos que utiliza un IDS tradicional, concretamente ‘Snort’, el cual, aparte de utilizar numerosos datos propios de los paquetes TCP, UDP, o ICMP, también

considera el contenido del propio paquete. Ésto es algo que plantea un serio problema, ya que en un principio que éste era el dato más importante para distinguir entre un paquete inofensivo y otro peligroso, pero resulta muy complicado introducir el contenido de los datos del paquete en una red neuronal. Finalmente se ha optado por sacar cuatro características del mismo en forma estadística de probabilidad de los cuatro caracteres más frecuentes que aparecen en él. Los datos utilizados han sido los siguientes:

Datos de la cabecera IP del paquete: puerto origen, puerto destino, protocolo utilizado (TCP, UDP, o ICMP), Type Of Service (TOS), tamaño de la cabecera IP (IpLen), tamaño total del paquete (DmgLen), reserved Bit (RB), don't Fragment bit (DF), More Fragments bit (MF), número de opciones IP, etc.

2.2. Obtención de los datos

La obtención de los datos no es tarea fácil, ya que hay que recopilar datos tanto de paquetes inofensivos, como de paquetes peligrosos, separarlos, y asegurarse de que fuesen significativos.

Para obtener los paquetes peligrosos se utilizan dos máquinas, una desde la que se lanzaban los ataques utilizando el scanner 'Nessus' y otra desde la que se capturaban esos paquetes utilizando el IDS tradicional 'Snort', en total se obtuvieron 433 paquetes peligrosos.

Para obtener los paquetes inofensivos, se utiliza una red de ordenadores real, es decir, donde fluyen los paquetes habituales en cualquier empresa o corporación donde se pretenda instalar un IDS. Finalmente el estudio se ha realizado utilizando una pequeña red departamental. El hecho de que la dirección IP origen y la destino sean la misma en todos los paquetes no repercute en los resultados obtenidos, ya que esa información no se tiene en cuenta. Entre Windows y Linux se obtuvieron 5731 paquetes inofensivos.

A continuación se explican las dos redes neuronales utilizadas para el fin propuesto, la red perceptrón multicapa y el mapa auto-organizativo.

3. PERCEPTRON MULTICAPA UTILIZADO

Para nuestro caso particular de red perceptrón multicapa, orientada a distinguir entre paquetes inofensivos y paquetes que representan ataques a un host en una red informática, la configuración que hemos adoptado es la siguiente:

La función de activación para cada neurona, debe ser continua y derivable, para que la red pueda almacenar información sobre la salida de cada neurona de manera más precisa, no sólo diferenciando entre 0 y 1, es decir, que la salida pueda tomar valores

en el continuo entre 0 y 1. Para este fin, se ha utilizado una función sigmoidea, y para ser más precisos, la función logística:

$$f(x) = 1 / (1 + e^{-x}) \qquad [1]$$

El número de neuronas en la capa de entrada es igual al número de datos que se extrae de cada paquete, en este caso 29, y en la capa de salida, dado que sólo distinguiremos entre dos posibles valores, sólo utilizamos una neurona, cuya salida indicará que el paquete analizado es inofensivo cuando sea menor o igual que 0.5, e indicará

que el paquete analizado es peligroso cuando la salida sea mayor que 0.5, ya que la salida está acotada entre 0 y 1 por la función de activación, y las salidas en el entrenamiento se han introducido como 1 para paquete peligroso, y 0 para paquete inofensivo. Se ha utilizado una única capa oculta. Después de realizar diversas pruebas variando el número de neuronas de esta capa, obtuvimos los mejores resultados con un número de 30.

La tasa de aprendizaje que se ha utilizado es variable. Comienza con un valor de 0.5 , y cuando el error empieza a oscilar (se detectan las oscilaciones cuando el error sube 4 veces tras 8 historias), a disminuye en 0.2 unidades. Si el error sigue oscilando, se vuelve a bajar el valor de a, hasta que llega a 0.1. Si sigue oscilando, a bajara hasta 0.06, y ya no bajara más. Después de probar varios valores para el momento (g), el que mejores resultados a dado ha sido el de 0.7. Como función de aprendizaje se ha utilizado el adiestramiento 'Backpropagation', el cuál se resume en los siguientes pasos:

1. Se inicializan los pesos de todas las neuronas de forma aleatoria.

2. Se introduce un patrón en la entrada de la red.

3. Se calculan las salidas de las capas de la red neuronal a partir del patrón de entrada, es decir, de izquierda a derecha.

4. Se obtiene el error que se produce entre la salida esperada y la realmente obtenida para el patrón de entrada introducido, el cual se calcula como la suma de los cuadrados de los errores de cada neurona de salida, dividido entre dos, donde el error de cada neurona de salida es igual a la salida esperada menos la salida obtenida.

$$E(p) = \frac{1}{2} * \sum_{i=1}^{S} (d_i - y_i)^2 \qquad [2]$$

5. Comenzando desde la capa de salida hasta la capa de entrada, se van actualizan do los pesos en función de la tasa de aprendizaje, del momento, y de la influencia que ha tenido cada peso en el error final, el cual se calcula teniendo en cuenta el gradiente del error en el espacio de pesos. De forma general, la actualización de cada

peso se realizará de la siguiente manera:

$$w(t+1) = w(t) - \alpha * \frac{dE(t)}{d(w)} + \gamma * \Delta w(t-1) \qquad [3]$$

6. Si hay más patrones para entrenar, se escoge el siguiente y se vuelve al paso 2.

Si no hay más patrones, la historia ha acabado, y se decidirá si se termina el entrenamiento, o si se continua entrenando con otra historia.

El error medio de cada historia en el aprendizaje, cosa que nos servirá para comprobar si el error va bajando en cada historia, se calcula sumando el error de todas las muestras de la historia y dividiéndolo entre el número de muestras, donde el error de cada muestra se indica en la ecuación [2].

$$E(hist) = \frac{1}{N} * \sum_{p=1}^{N} E(p) \quad [4]$$

La función de parada que se ha utilizado, es la que el propio usuario quiera darle en tiempo de ejecución, ya que se ha capturado la señal SIGINT (Ctrl -C) para parar la ejecución en cualquier momento y guardar el resultado del entrenamiento (pesos y estadísticas), lo cual se puede decidir bien cuando el error de cada historia pase cierto

umbral, o bien cuando el número de historias haya alcanzado cierto número, ya que durante el entrenamiento se visualiza el número de historia y el error medio que genera.

4. RESULTADOS OBTENIDOS

Hay que decir que los resultados obtenidos han superado las expectativas iniciales, ya que el hecho de poder distinguir entre paquetes inofensivos y paquetes peligrosos sin mirar totalmente el contenido de datos del paquete era algo que se pensaba sólo

p o dría conseguirse en un moderado porcentaje de aciertos, y el porcentaje de aciertos conseguido en los test ha superado el 90%. Lo primero que hay que decidir en esta etapa es cómo medir los resultados. Esto es un problema mayor del que en un principio se pensó, ya que no disponemos de más de 433 paquetes peligrosos con los que entrenar y testear, y tampoco se puede probar el resultado del entrenamiento de la red neuronal instalándola en una red de ordenadores, ya que lo más seguro es que se encuentre con paquetes totalmente diferentes de los paquetes con los que se ha entrenado en la pequeña red de dos máquinas utilizadas en el proyecto. Para realizar esto último, tendríamos que haber adiestrado a la red neuronal con los paquetes significativos de la red de ordenadores donde se fuese a instalar el NNIDS (Neural Net -work Intrusión Detection System), pero eso hubiera llevado mucho más tiempo del que se disponía para realizar el proyecto. Por tanto, la conclusión es que la única manera de probar el resultado del entrenamiento, es separando un conjunto de muestras de las obtenidas para entrenar, y utilizándolas para testear. Concretamente se ha escogido un 80% de paquetes para entrenamiento, y un 20% para testeo, tanto para los paquetes inofensivos como para los paquetes peligrosos obtenidos. Los resultados los mediremos en términos de porcentaje de aciertos obtenidos en los paquetes de testeo, que serán: paquetes inofensivos, peligrosos, y la totalidad de ellos.

4.1 Resultados de la red Perceptrón multicapa

A continuación se muestra un resumen del error tras cada historia en el adiestramiento de la red perceptrón multicapa que ha dado mejores resultados, ya que esto depende en gran medida de la inicialización aleatoria de los pesos. Indicaremos los errores de las 10 primeras historias, y luego iremos avanzando más rápidamente, ya que se dejó entrenar hasta que el error no bajaba más, y fueron 2153 historias:

Evolución del error de aprendizaje:

```
Error tras historia 0: 0.167678
Error tras historia 1: 0.110749
Error tras historia 2: 0.206355
Error tras historia 3: 0.191268
Error tras historia 4: 0.121300
Error tras historia 5: 0.093922
Error tras historia 6: 0.084878
Error tras historia 7: 0.063542
Error tras historia 8: 0.043144
Error tras historia 9: 0.024848
Error tras historia 10: 0.027718
Error tras historia 20: 0.155243
Error tras historia 30: 0.017051
Error tras historia 40: 0.016624
Error tras historia 100: 0.015309
Error tras historia 200: 0.013837
Error tras historia 300: 0.012531
Error tras historia 400: 0.010378
Error tras historia 500: 0.005598
Error tras historia 600: 0.004250
Error tras historia 1000: 0.002871
Error tras historia 1500: 0.002649
Error tras historia 2000: 0.002144
Error tras historia 2153: 0.002132
```

Tras este entrenamiento, los resultados en cuanto a tasa de acierto con las muestras de test son las siguientes:

Tabla 1. Relación de transferencia de paquetes

Tipos de paquete	Número aproximado de paquetes de entrenamiento	Número aproximado de paquetes de test	Tasa de acierto para los paquetes de test
Paquetes inofensivos.	4585	1146	0.985395
Paquetes peligrosos.	346	87	0.977273

La tasa de acierto de todos los paquetes de test, aproximadamente 1233, es de: 0.991235.

Para comprobar que los resultados obtenidos no han sido fruto de una casualidad a la hora de seleccionar los paquetes de entrenamiento y los paquetes de test, se han realizado entrenamientos más, cada uno con una división de paquetes de entrenamiento y de test diferentes, Los resultados, tras 10 historias de entrenamiento, son los

siguientes:

Tabla 2. Clasificación de comprobación

	División 1	División 2	División 3
Tasa de acierto en los paquetes de test inofensivos	0.863755	0.849877	0.947186
Tasa de acierto en los paquetes de test peligrosos	1.000000	0.988372	0.876712
Tasa de acierto en todos los paquetes de test	0.872340	0.859004	0.942997

Realizando una media de los resultados obtenidos en los apartados anteriores para la red perceptrón multicapa y para el mapa auto -organizativo, contando con las tres pruebas hechas más la prueba con la que hemos obtenido mejores resultados, obtenemos la siguiente tabla:

Tabla 3. Resultados de la red Perceptrón multicapa

Tipo. de paquete	N. Aprox de Paq. de entrenamiento	N. Aprox de Paq. de test	Tasa de acierto
Paquetes inofensivos.	4585	1146	0.997425
Paquetes peligrosos.	346	87	0.911111

La conclusión que se puede sacar de esta comparación es que el perceptrón, al reconocer peor los paquetes inofensivos, y mejor los paquetes peligrosos, dará una cantidad mayor de falsos positivos. Lo contrario pasa con el MAO(Mapa auto organizativo no visto en este articulo), ya que reconoce muy bien los paquetes inofensivos, y no tan bien los paquetes peligrosos (aunque estamos hablando de un 90%). Por tanto, si el NNIDS se va a utilizar para detectar un ataque y denegar el servicio, tal vez no sea recomendable que se den muchos falsos positivos y nos convenga utilizar el MAO. Sin embargo, si lo que se pretende es detectar el ataque y dar un aviso para que se investigue mejor la procedencia del mismo y sus intenciones, y no se está proporcionando un servicio, será recomendable utilizar la red perceptrón multicapa.

6. CONCLUSIONES

La detección de ataques es algo complicado de manejar, ya que continuamente surgen nuevos exploits y los atacantes inventan nuevas maneras de penetrar en los sistemas.

El uso de las redes neuronales puede suponer una enorme ventaja en la detección de estos ataques, ya que tienen la capacidad de detectar ataques que no se han aprendido directamente. Como primera conclusión se puede decir que el intento de comprobar si una red neuronal podía aprender a distinguir paquetes inofensivos de

paquetes peligrosos, ha sido todo un éxito, aún sin utilizar toda la información del contenido de cada paquete.

7. REFERENCIAS

[1]M. D. Abrams, S. Jajodia, H.J. Podell. “Information Security: An Integrated Collection of Essays” IEEE Computer Society Press 1997

[2]C. P. Pfleeger. “Security in computing”. PHH, 1997.

[3]CERT. Results of the distributed-systems intruder tools workshop, http://www.cert.org/reports/dsit_workshop-final.html, November 1999

[4]J.P. Anderson. Computer security threat monitoring and surveillance. Technical report, James P. Anderson Co. Fort Washington, PA, 1980

[5]S. Kumar. “Classification and Detection of Computer Intrusion” PhD thesis, Purdue University, August 1995