CAPÍTULO 2. SISTEMAS DE DETECCIÓN DE INTRUSOS.

En este capítulo se presentan los conceptos principales de sistemas de detección de intrusos, redes de computación y ataques en Internet, vislumbrando las tecnologías aplicables a solucionar este problema.

2.1 ACTUALIDAD DE LOS SISTEMAS.

Lejos de haber conseguido un modelo de detección de intrusos que sea capaz de detectar a tiempo los ataques que se hacen presentes en una red de computadoras, los sistemas de detección de intrusos actuales muestran una baja eficiencia de detección en aquellos ataques que se presentan como desconocidos, o nuevos.

Esto es debido, principalmente, a que, en la actualidad, la mayor parte de los sistemas de detección de intrusos basan su funcionamiento en el paradigma de comparación de firmas, el cual, sugiere que un sistema de detección debe contar con una base de datos que contenga todos los tipos de ataques conocidos y entonces cuando un requerimiento se hace al sistema, éste se debe comparar con la lista de ataques conocidos y solo si este ataque aparece en la base de datos activar la alarma de intrusión.

Este enfoque sería correcto si el total de los ataques en redes de cómputo pudieran ser registrados en una supuesta base de datos. Sin embargo, día a día surgen nuevos ataques. Variaciones de ataques anteriores, inserción de nuevos comandos, descubrimientos de nuevas fallas en el sistema, y en general sería imposible tener una base de datos de todos los ataques posibles. Ha sido probado entonces, que un sistema de esta naturaleza sufre de una gran ineficiencia al tratar de detectar ataques nuevos [MAN02].

Pero entonces ¿Cuál es la opción para detectar ataques nuevos?, ¿Cómo se puede discernir entre la naturaleza intrusiva de un ataque o un simple comando de ejecución

normal que se encuentre lejos de ser dañino? El siguiente apartado nos presenta un nuevo enfoque para la solución de este problema.

2.2 DETECCION DE ANOMALIAS.

El siguiente enfoque tiene un nuevo propósito, dejando a un lado la utópica idea de buscar el conocimiento absoluto de las cosas, se tratará simplemente de hallar los patrones en los comandos que indiquen la naturaleza de los mismos. Siendo posible que los datos se comporten de una forma normal, o por el otro lado que se trate de datos con naturaleza anómala.

Para llegar a este tipo de caracterización, será necesario contar con suficientes datos de comportamiento normal y anómalo para que podamos distinguir los patrones que determinan cada uno de estos datos y sus efectos causales.

A simple vista, la percepción humana se hace insuficiente para determinar estos patrones, por lo que se sugiere la necesidad de otro sistema que sea capaz de realizar dicha caracterización. A todo esto, a finales del siglo pasado, surgieron de las cenizas tecnologías adaptivas que habían sido dadas por muertas, o en desuso debido a la insuficiencia para separar patrones que no estaban unidos linealmente, es decir, cuya relación era de tipo no lineal [MAS93].

No obstante, esta relación no lineal fue determinada finalmente por una nueva tecnología de la inteligencia artificial que se conoció con el término de red neuronal. La explotación de las redes neuronales en distintos campos de la investigación ha surtido en inmejorables resultados para la solución de problemas en los que se requiere hallar relaciones entre los datos, en otras palabras, hallar patrones en los datos. Como se puede vislumbrar en este momento, las redes neuronales serán la herramienta que nos ayude a

implementar el enfoque de detección de intrusos basado en la detección de anomalías, debido a la capacidad de las redes neuronales para hallar patrones en los datos, cuyo caso es el que estamos tratando actualmente.

2.3 PROTOCOLO HTTP.

Cuando hablamos de los sistemas de detección de intrusos es importante definir la variable que estos sistemas desean analizar y con la que posteriormente determinaran si existe una anomalía o no. Esta variable puede ser de distinta naturaleza a la que otro sistema pudiera tomar como variable, como podría ser un sistema que analice la cantidad de conexiones, o intentos de conexión que existan en un servidor a determinada hora del día. Considerando anómalo el flujo de datos, cuando se sobrepase un umbral de intentos de conexión predefinido.

Y por otro lado un sistema pudiera analizar el flujo de paquetes que circulan por la red, verificando constantemente que el número de paquetes y el contenido de estos sea considerado normal. No obstante este tipo de sistemas sufren distintas dolencias, al no considerar directamente las acciones del atacante ya que basan el análisis de los ataques en los efectos que causan sobre la red, y no en las causas primarias. Un sistema que considere las causas, estaría en ventaja respecto a otros sistemas debido a que el análisis sería más efectivo ya que estaría analizando directamente los comandos con los que un atacante opera.

Visto de otra forma, es más efectivo analizar los datos a nivel de aplicación que a nivel de red, debido a que en la red los datos pudieran estar encriptados, o pudieran ser manipulados de tal forma que parecieran normales. La forma más transparente de analizar los datos es a nivel de aplicación por lo que en este proyecto nos enfocaremos a detectar

anomalías a nivel de aplicación [TOR03]. Sin embargo aún queda una interrogante respecto a ¿Cuál va a ser la variable que nosotros vamos a analizar?

En diciembre del 2002, la empresa de seguridad informática española S21SEC realizó un estudio durante los últimos cinco meses (junio-noviembre). En dicho estudio se pudo establecer que de 2113 vulnerabilidades publicadas, 1320 vulnerabilidades tienen su origen en las aplicaciones lo cual representa un 62,5% de las vulnerabilidades reportadas [WWW2].

Si se toma en cuenta que la mayor parte de los comandos que se ejecutan en Internet suceden en la aplicación del protocolo *HTTP* en el puerto 80 del ordenador, entonces es loable definir que la variable que nosotros vamos a analizar serán los comandos que se ejecuten en este protocolo de comunicación. Siendo así que las cadenas de caracteres que se reciban en este puerto serán analizadas para decidir si hacen referencia a un acto intrusivo o simplemente forman parte del comportamiento normal de la red. Para esto, será necesario que se cuente con un conjunto de datos, que representen dichas de cadenas de caracteres, para poder encontrar patrones en dichos datos y posteriormente diseñar un sistema que decida la naturaleza de estos datos.

Los datos con los que se cuentan, forman parte del proyecto de detección de intrusos desarrollado por el departamento de Ingeniería en Sistemas de la Universidad Javeriana de Bogotá Colombia [TOR03]. Dichos datos se encuentran agrupados en la siguiente clasificación que se deberá tomar en cuenta al hallar patrones de entrenamiento.

2.4 CLASIFICACIÓN DE DATOS EN HTTP.

En el protocolo *HTTP*, los datos pueden caer en la siguiente clasificación en base a las causas y efectos que tiene cada comando:

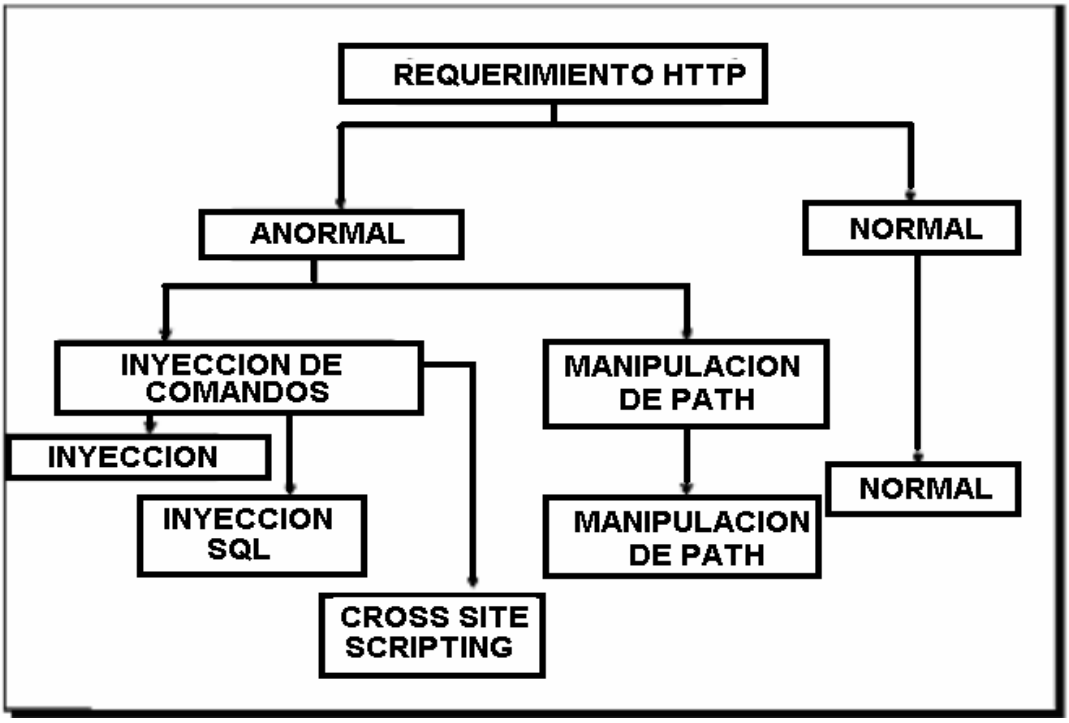

FIGURA 2.1 Clasificación de Datos en *HTTP*.

Esta clasificación fue definida por primera vez, en el proyecto de Detección de Intrusos de la Pontificia Universidad Javeriana, en Colombia, [TOR03].

El sistema de detección de intrusos a diseñar se encargará de clasificar a los datos en alguna de las cinco distinciones posibles mostradas anteriormente de manera que podamos *identificar* la clasificación de los datos obteniendo una indicación del tipo de ataque al que pertenece el dato y por otro lado será capaz de detectar cuando un dato sea normal o simplemente anómalo sin necesidad de identificar el tipo de ataque. De esta manera se pondrá a prueba la capacidad de la red para identificar que tipo de ataque esta sucediendo y en base a esto emitir la respuesta más adecuada.

A continuación presentamos la definición de cada una de las cinco clasificaciones posibles:

DATO NORMAL: Se dice que un dato es normal cuando pertenece al flujo normal de datos de una red, de manera que no afecta el desempeño de la misma.

INYECCION DE COMANDOS: Se dice que un ataque es de este tipo cuando, como su nombre lo indica, se ejecutan directamente comandos del sistema operativo o del lenguaje asociado debido a vulnerabilidades en la validación de entradas a la aplicación, incluyendo vulnerabilidades en el servidor Web. El ataque más común de este tipo es la ejecución de comandos del sistema a través de pequeños programas en lenguaje ensamblador denominados shell generalmente activados por fallas en la implementación de llamados a rutinas que permiten la ejecución de estos comandos.

ATAQUE A BASES SQL: Cuando la inyección de comandos se realiza en un ambiente de interfaz a consultas *SQL* se denomina inyección *SQL* y lo que se intenta es modificar el comportamiento de los resultados de las búsquedas realizadas a estas bases de datos, obteniendo información restringida de estas bases de datos.

CROSS SITE SCRIPT (XSS): Si se inyecta un código como HTML, java, javascript, entonces se dice que es un ataque de script cruzado o XSS. Un ejemplo muy común de este tipo de ataque es la ejecución remota de comandos a través del *CGI* usando *PHF*.

MODIFICACION DE PATH: Este tipo de ataques suceden cuando se modifica la información relativa a la ubicación de uno o varios recursos en un contexto dado, permitiendo manipular el comportamiento de las aplicaciones y otorgando control de ellas al atacante o por otro lado proveer ubicaciones distintas y por ende recursos distintos a los normalmente utilizados. La aplicación más común de este ataque es la creación de un directorio transversal en el cual, una aplicación entrega el contenido de un archivo dentro

de este directorio. Esto puede suceder, si la ruta de entrega de la aplicación es modificable o no existe una correcta validación de esta entrada.

Otro tipo de problemas es la implementación de aplicaciones en PHP la única diferencia en este caso es que el PATH utilizado no es interno al host atacado, por lo general, este es manipulado para hacer referencia a un host externo.

2.5 HALLANDO PATRONES.

Como habíamos mencionado anteriormente, contamos con cadenas de caracteres que representan requerimientos *HTTP* pertenecientes a la clasificación mencionada anteriormente. Un ejemplo de estas cadenas es la siguiente:

nombre.exe?param1=..\..\archivo

En nuestro caso, se tienen un total de 773 cadenas de caracteres como esta, distribuidas en las 5 clasificaciones posibles. Estas cadenas se muestran en el Apéndice B, a simple vista estas cadenas parecen no tener un patrón en común, sin embargo, como se puede notar, parte de la cadena, son caracteres que conforman una cadena alfanumérica que representa el nombre de un archivo. Si tomamos en cuenta, que los nombres de archivos no son significativos ya que éstos cambian de un sistema a otro, entonces se puede concluir que estas cadenas no deben ser consideradas.

En general a lo largo del proyecto, al hablar de cadenas de requerimientos *HTTP* se hace referencia únicamente a la parte significativa de la cadena que incluye los caracteres especiales y las extensiones de archivo por lo que la cadena anterior quedaría de la siguiente forma:

@.exe?@=..\..\@

Sustituyendo las cadenas de caracteres no significativas por un símbolo @. Esta abstracción de la cadena propuesta en [TOR03], nos permite reducir el tamaño total de la cadena, a un tamaño más corto de manera que sea más fácil analizar las cadenas tomando en cuenta únicamente las partes de ésta que son importantes para determinar su naturaleza.

Por otro lado, el análisis que determinará ésta naturaleza, será llevado a cabo por un sistema neuronal, o red neuronal que nos permita extraer los patrones necesarios para determinar si la cadena es intrusiva o normal. En el siguiente capítulo se discutirá como se constituyen las redes neuronales explicando también su funcionamiento y propiedades.