EVADIR: una metodología para la evasión de IDS de red

Sergio Pastrana
Escuela Politécnica Superior
Universidad Carlos III de Madrid
Email: spastran@inf.uc3m.es

Agustín Orfila
Escuela Politécnica Superior
Universidad Carlos III de Madrid
Email: adiaz@inf.uc3m.es

Arturo Ribagorda
Escuela Politécnica Superior
Universidad Carlos III de Madrid
Email: arturo@inf.uc3m.es

***Abstract*—Los sistemas de detección de intrusiones de red o *Network Intrusion Detection Systems* (NIDS) son herramientas software o hardware que monitorizan el tráfico de red en busca de actividad maliciosa de distinta naturaleza. A medida que aparecen nuevas vulnerabilidades y métodos para explotarlas, estos sistemas son convenientemente actualizados. En caso de producirse, los ataques son detectados si un NIDS actualizado está monitorizando el sistema atacado, exceptuando aquellos ataques cuya firma aún no ha sido desarrollada (ataques de día cero). Esta situación ha provocado que los atacantes traten de desarrollar nuevas técnicas para pasar desapercibidos a ojos del NIDS cuando intentan atacar un sistema, o lo que es lo mismo, tratan de evadir su detección. Al igual que ocurre con las nuevas vulnerabilidades, los NIDS deben estar preparados para contrarrestar estas novedosas técnicas evasivas. Actualmente la mayoría de estas técnicas se basan en explotar ambigüedades presentes en los protocolos de red, problema que fue presentado por Ptacek y Newsham [7]. En este artículo presentamos EVADIR, una nueva metodología que facilita la búsqueda de nuevas formas de evadir NIDS. El núcleo principal de la misma consiste en modelar el NIDS mediante Programación Genética (PG), haciendo uso de una sintaxis más sencilla que la del NIDS. De este modo, se facilita la búsqueda de técnicas evasivas sobre el detector al realizar el análisis sobre el modelo generado, cuya semántica es menos compleja que la del NIDS en cuestión.**

I. Introducción

Las tecnologías de la información se han convertido en un componente crítico de la economía actual. La protección de las mismas frente a acciones hostiles determinará, en gran medida, el desarrollo de la sociedad de la información y las comunicaciones. Las medidas de seguridad se suelen clasificar atendiendo a su forma de actuación en: de prevención, detección, corrección y recuperación. Aunque las más aconsejables de las anteriores son las primeras, el hecho de ser más costosas que las restantes, junto con la imposibilidad de anular completamente un riesgo, hace preferible distribuir los recursos entre todas las medidas citadas, dando lugar así a la llamada defensa perimetral, consistente en la construcción de sucesivas barreras de protección: preventivas, detectoras, correctivas y recuperadoras.

Los sistemas de detección de intrusiones son herramientas software o hardware que automatizan el proceso de monitorizar los eventos que acontecen en un ordenador o red en busca de evidencias de intrusiones [20]. Una intrusión se define como un intento de comprometer la confidencialidad, la integridad, o la disponibilidad de la información, o de soslayar los mecanismos de seguridad de un ordenador o red. Los IDS de red o NIDS toman como fuente de información principal el tráfico de red. A este respecto, se define el concepto de evasión sobre un NIDS como aquella técnica que permite transformar un ataque definido y detectable en otra forma del mismo que lo soslaye. Es decir, el paquete o conjunto de paquetes (en caso que haya fragmentación o que el ataque se encapsule en una sesión entera) que contengan el ataque es modificado con el fin de que el NIDS no sea capaz de procesarlo adecuadamente, y por lo tanto, de detectarlo.

Los NIDS son, junto con los cortafuegos, los primeros mecanismos de seguridad usados para proteger a los sistemas de información en red. En consecuencia son objetivos prioritarios de los atacantes, que tratan de deshabilitarlos o forzarlos a producir información errónea. En general, los NIDS no proporcionan una respuesta automatizada a las intrusiones que detectan, y se precisa del análisis por parte de un administrador de seguridad de las alertas emitidas para determinar el alcance de las mismas y establecer qué medidas correctoras o recuperadoras han de llevarse a cabo. Por este motivo, la información proporcionada por los NIDS, en caso de ser falsa, puede dar lugar a la distracción del personal de seguridad, como si de un señuelo se tratara [10].

Las técnicas de evasión propuestas hasta la fecha se basan, fundamentalmente, en el abuso de las ambigüedades existentes en los protocolos de comunicaciones. Dichas ambigüedades pueden provocar que los NIDS y los sistemas finales a proteger interpreten los protocolos de distinta forma, provocando que el procesamiento de las conexiones por parte de los NIDS y de los equipos finales sean diferentes. Esta situación favorece el diseño de ataques que evadan a los NIDS.

La investigación en técnicas de evasión de NIDS basados en usos indebidos es, en conjunción con el descubrimiento y detección de nuevas formas de ataque, el principal modo de mejorar la eficacia de estos. En la actualidad, la rápida actualización de las firmas de los NIDS para la detección de nuevos ataques de red provoca que los atacantes traten de desarrollar técnicas evasivas, más sigilosas y difíciles de detectar. Así, un administrador de seguridad no es consciente de que el NIDS ha sido evadido hasta el posterior análisis forense de los sistemas comprometidos. Este contexto es la principal motivación de este trabajo, cuyo objetivo primordial es establecer una metodología que permita el descubrimiento

de nuevas formas de evadir NIDS y que, en consecuencia, favorezca el desarrollo de mecanismos en los NIDS para su prevención.

Para la consecución de este objetivo, en la metodologia propuesta EVADIR (EVAsión de la Detección de Intrusiones en Red) se generan modelos (mediante PG) que tratan de emular el comportamiento de NIDS existentes con el fin de interpretar su modo de funcionamiento desde una perspectiva alternativa. Estos modelos, además de simplificar la complejidad de los NIDS, permiten modelar el comportamiento de NIDS de código cerrado, de los que se desconoce su modo de operación interno. Su análisis facilita el hallazgo de nuevas formas de evasión, difíciles de encontrar por otros medios. La aplicación práctica de esta metodología puede favorecer el diseño de NIDS más difíciles de evadir.

El resto del artículo se estructura de la siguiente manera. En la sección II se presenta cual es el estado actual de la línea de investigación en las técnicas evasivas y el uso de PG en el ámbito de los NIDS. En la sección III se explica formalmente en qué consiste la metodología propuesta, así como las tareas que la componen. Finalmente, en la sección IV se establecen las conclusiones.

II. Estado de la cuestión

La evasión de NIDS fue tratada por primera vez por Ptacek y Newsham en 1998 [7]. En este artículo, los autores resaltan dos grandes problemas en el procesamiento del tráfico de red por parte de los NIDS. En primer lugar, un NIDS no siempre es capaz de conocer con exactitud la manera en que los paquetes serán procesados en el sistema final, debido a las ambigüedades existentes en los protocolos TCP/IP. Por ejemplo, algunas implementaciones descartan los segmentos TCP cuyo campo *checksum* sea erróneo, pero otras no. Si el NIDS descarta un paquete con dicho campo erróneo, se expone a que el paquete sea procesado en el sistema final, y viceversa, puede procesar el paquete cuando en el sistema final no se procesaría. Un distinto procesamiento de los segmentos TCP puede conllevar que ciertos segmentos de un ataque no sean procesados (y por lo tanto detectados) por el NIDS, o puede darse que se procesen más paquetes de la cuenta, llegándose a un reensamblado distinto en los NIDS y en los sistemas monitorizados. El segundo problema presentado es que es posible realizar ataques de Denegación de Servicio (DoS) sobre los NIDS. Un envío de muchos paquetes mal construidos hacia un NIDS puede provocar que éste genere numerosas alertas, llegando incluso a no procesar algunos paquetes debido a la sobrecarga. Una atacante podría, antes de enviar los paquetes que componen un ataque, provocar esta situación, por lo que ésta sería la antesala del verdadero ataque que pasará inadvertido.

Con el fin de explotar y estudiar las ambigüedades mencionadas anteriormente, se han diseñado algunas herramientas específicas, como Fragroute, la cual intercepta, modifica, y reescribe el tráfico destinado a un sistema [4], o IDSprobe, que genera tráfico modificado a partir de trazas originales [8]. Las modificaciones que estos sistemas proponen están orientadas a la generación de tráfico que provoque evasiones. De esta manera, se pueden realizar auditorías sobre NIDS haciendo uso de técnicas evasivas existentes. Alternativamente a estas herramientas, se pueden utilizar algunas funcionalidades de programas más conocidos como Nikto [13] o Nmap [12], cuyos parámetros de ejecución permiten modificar los paquetes que se envían para intentar evitar la detección por parte de los NIDS.

La literatura existente sobre técnicas que prevengan a los NIDS ante posibles evasiones se centra en el tratamiento del tráfico que le llega al NIDS, con el fin de paliar las ambigüedades en la interpretación de los protocolos de comunicaciones y establecer un procesamiento común. Watson et al. [1] proponen un sistema para depurar paquetes de datos, de manera que se eliminen los posibles intentos de evasión de los NIDS, e implementan un depurador (*scrubber*) sobre el protocolo TCP para generar tráfico bien formado a partir de uno de entrada que pueda contener ambigüedades. Por su parte, Handley et al., proponen el concepto de normalizadores de tráfico [9]. Éstos son elementos que se sitúan como intermediarios en la red y que toman el tráfico antes de que llegue al NIDS con el fin de eliminar las posibles ambigüedades que puedan ocurrir. Dado que algunos de los problemas que pueden llevar a evasiones se dan en el reensamblado de los paquetes, estos normalizadores deben almacenar el estado y los paquetes previos de todas las conexiones entrantes, lo que requiere una gran capacidad de almacenamiento, ya que deben verificar la consistencia de los paquetes subsecuentes. Esta tarea consume una gran cantidad de recursos (tanto de procesamiento como de almacenamiento) y puede convertirse en un cuello de botella cuando se trabaja con redes de alta velocidad [2].

También se han propuesto algunas soluciones alternativas que no precisan de la modificación del tráfico. Varghese et. al. [3] plantean dividir la firma que busca el NIDS para detectar el ataque (NIDS *signature*) en pequeños fragmentos, de manera que cualquier paquete que contenga cualquiera de los fragmentos sería detectado de forma rápida, pasando después el conjunto de paquetes a un análisis más lento pero más profundo y por lo tanto más eficaz. Shankar y Paxson [5] proponen un sistema que notifica al NIDS la topología de red y el comportamiento (entendiéndose como tal la implementación de los protocolos) del sistema final que se monitoriza, de manera que pueda modificarse la configuración del NIDS para ajustarse a la información proporcionada por su sistema. A este respecto Snort [6], uno de los NIDS de código abierto más utilizado en la actualidad, implementa una técnica similar a esta en el preprocesador IP (frag3) de su última versión. Finalmente, Antichi et al. [11] proponen el uso de Bloom Filters, una suerte de filtros distribuidos que consiguen que la firma del NIDS sea detectada sin necesidad de un reensamblado previo de los paquetes. Con este sistema se minimizan los falsos negativos pero se incrementa sobremanera la carga computacional del NIDS.

En lo referente a la búsqueda de evasiones en NIDS, cabe destacar el trabajo de Mutz et al. [27], en el cual

los autores proponen realizar ingeniería inversa (tomando un conjunto de entradas/salidas y analizando el comportamiento de los ficheros binarios de un NIDS comercial) con el fin de obtener información acerca de las firmas que aplican los NIDS cuyo comportamiento no es conocido al ser el código propietario. Una vez analizado su funcionamiento, los autores presentan un ejemplo de evasión con un ataque a servidores Apache. Su trabajo se centra en la evasión de las firmas, las cuales son el último escalón en la arquitectura de un NIDS, sin tener en consideración el preprocesamiento que pueda realizarse anteriormente. Sin embargo, las técnicas evasivas propuestas por Ptacek y Newsham [7] se basan precisamente en ambigüedades de este preprocesamiento.

La idea principal de nuestra propuesta, alternativa e innovadora frente a los antecedentes citados, consiste en desarrollar una metodología que facilite la búsqueda de técnicas de evasión de NIDS. Para la consecución de este objetivo se construirán modelos, que emulen el comportamiento de NIDS existentes. Para ello se utilizará programación genética [24], la cual se ha mostrado como un paradigma eficaz y eficiente de cara al desarrollo de NIDS [14], [15], [16], [17], [18], [19], lo que avala su utilización para el objetivo en cuestión.

III. EVADIR: Metodología para la evasión de NIDS

Este artículo tiene como objetivo principal la propuesta de una nueva metodología para buscar formas de evadir NIDS que permita predecir y prevenir futuros intentos de evasión. Se plantean las siguientes grandes tareas a cumplimentar dentro de la metodología (ver Figura 1):

1) Generar un conjunto de datos que represente adecuadamente la idiosincrasia del tráfico de red actual. Dicho tráfico incluirá tráfico normal y malicioso (incluyendo técnicas evasivas existentes).
2) Definir y diseñar un sistema basado en Programación Genética que aborde el modelado automático de NIDS en producción y que proporcione una representación simplificada pero fiel de los mismos.
3) Desarrollar nuevas técnicas evasivas analizando el modelo generado.

A. Generación del tráfico y de los conjuntos de entrenamiento y test

1) Preparación de la infraestructura virtual: En primer lugar se define la arquitectura lógica de red que se utilizará en el entorno de pruebas. Ésta ha de ser fiel a un entorno real para que el tráfico que se genere sea similar al que podríamos encontrar en los sistemas de información y las organizaciones. Dicha arquitectura (lógica) se compone una serie de equipos para la auditoría, para la instalación de los NIDS y para la máquina que hace las funciones de víctima. La arquitectura dispone además de un canal de comunicación que conecta las máquinas junto con uno o varios routers. Una vez definida la arquitectura lógica, se establece una arquitectura física haciendo uso de la técnica de virtualización [25] ya que ofrece una mayor flexibilidad y no requiere de una gran infraestructura física. El requisito aquí es disponer de equipos

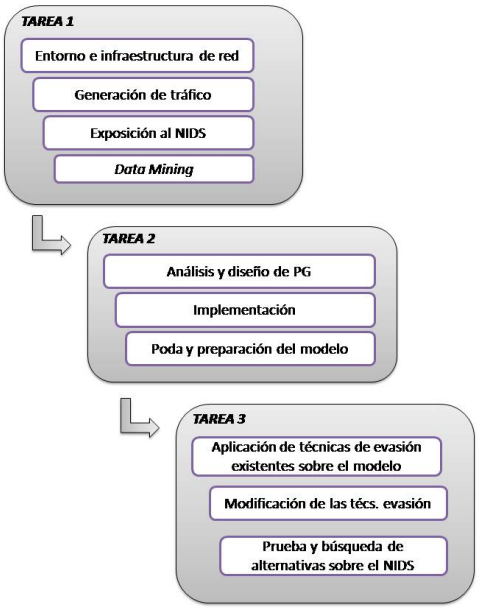

Fig. 1. Arquitectura de EVADIR.

de gran potencia y almacenamiento con varias tarjetas de red que permitan realizar la simulación de la infraestructura. Como software de virtualización, se utiliza alguno de los existentes de código abierto, como VMWare o VirtualBox, ya que permite reducir costes y cumple con los requisitos necesarios en esta tarea.

En esta fase, se instalan y configuran las máquinas virtuales usadas para la adquisición de datos. Han de instalarse tanto los Sistemas Operativos (varios, con el fin de trabajar con NIDS que funcionen sobre distintas plataformas) como el software necesario. La elección del S.O. y el software auxiliar a utilizar es una cuestión que debe definirse en el momento de implementar la metodología.

2) Generación y etiquetado del tráfico: Haciendo uso del entorno virtual configurado, se genera tráfico variado, entendiéndose como tal el tráfico normal (es decir, simples conexiones TCP, UDP e IP sin actividad maliciosa), de ataque (los paquetes llevan encapsulados algún tipo de ataque, sobre todo de tipo HTTP) y evasivo (modificación los paquetes de ataque y normal haciendo uso de las técnicas de evasión existentes en la literatura, las cuales están implementadas y documentadas, [7]). La generación del tráfico se realiza de forma controlada, es decir, estableciendo de manera concisa qué paquetes se envían y filtrando la captura por la máquina auditora, de manera que se garantiza que el tráfico cumple con las características deseadas. Además, los datos se acompañan con una serie de metadatos para etiquetar y documentar las trazas generadas, ya que como se verá más adelante, este tráfico se utiliza en varias etapas de esta metodología.

Para generar el tráfico, se hace uso de herramientas de

código abierto existentes que permiten el envío de segmentos TCP con la suficiente flexibilidad para modificar las cabeceras de los mismos y de los paquetes IP. De esta manera se pueden desarrollar las técnicas evasivas descritas por Ptacek y Newsham y generar así el tráfico evasivo. El contenido del tráfico enviado es algún ataque detectado en principio por cualquier NIDS (como por ejemplo, un ataque XSS sobre un servidor web) cuando se quiera generar tráfico de ataque, o simples peticiones HTTP no maliciosas cuando se quiera generar tráfico normal. Se ha decidido descartar la utilización de conjuntos de datos etiquetados públicamente disponibles en la literatura como LBNL [21] o KDD [22] debido a que el primero sólo incluye tráfico de reconocimiento (y además anonimizado) y el segundo data del año 1999 y su validez despierta controversia [23].

La generación del tráfico es una tarea crucial para la obtención de resultados útiles dentro de esta metodología, ya que va a determinar todos los procesos subsequentes como se verá más adelante. Debe por lo tanto garantizarse, mediante el análisis del tráfico generado, que todos los casos que se pretenden abordar están realmente reflejados en el tráfico, de lo contrario, ha de repetirse esta tarea hasta que los objetivos queden totalmente satisfechos.

3) Exposición del tráfico a los NIDS: El tráfico generado se presenta de manera *offline* a los NIDS bajo estudio con el fin de obtener y registrar la salida que éstos dan. Es posible que parte del tráfico evasivo no sea detectado, por lo que se deben analizar las posibles evasiones que se produzcan. Es de esperar en esta fase que el tráfico normal pase desapercibido en las alertas de los NIDS, y que el tráfico malicioso sea detectado, lleve o no incluídas técnicas de evasión, ya que éstas están documentadas y existen herramientas que las tratan (e.g. Frag3 de Snort).

4) Procesamiento y extracción de características del tráfico: Hasta este punto lo que se ha obtenido es un conjunto de paquetes de tráfico sin tratar junto con la salida que los NIDS estudiados ofrecen cuando dicho tráfico se les presenta de manera *offline*. En este punto, se debe generar un conjunto de datos que relacione los paquetes de tráfico obtenidos con su correspondiente salida dada por cada NIDS. Dado que muchos NIDS trabajan a nivel de conexión y no de paquete, será necesario para esos casos constituir las conexiones presentes en el tráfico junto con la salida dada por el NIDS para esas conexiones.

Debido al gran volumen de datos generado, para modelar NIDS se precisa de una fase de preprocesamiento de los datos, haciendo uso de técnicas de Minería de Datos (*Data Mining*) con el fin de tomar las características del tráfico más relevantes desde la perspectiva de la seguridad, constituyéndose posteriormente sendos conjuntos de entrenamiento y de test. El conjunto de datos está constituido por entradas como por ejemplo las que se muestran en la Figura 2. En esta figura, se pueden ver una serie de campos separados por comas, con 2 etiquetas (los 2 últimos campos). Estas son, un símbolo para denotar si el paquete se corresponde con una actividad benigna (B en el ejemplo) o maliciosa (M en el ejemplo), y otro símbolo que denota si el NIDS ha acertado o no en la predicción (en el ejemplo, - indica que lo etiqueta bien, y + que lo etiqueta mal).

```
22,37273,54637,56676,5,0,0,0,1,1,0,58464,28642,0,M,+
37273,22,56676,54637,5,0,0,0,0,1,0,34800,52234,0,M,-
22,2224,54113,20593,5,0,0,0,1,1,0,32760,44444,0,B,-
1068,139,59831,24387,5,0,0,0,1,1,0,15376,64779,0,B,+
139,1068,24387,59831,5,0,0,0,1,1,0,17520,62547,0,B,-
22,37273,54637,56676,5,0,0,0,1,1,0,58464,28482,0,M,+
37273,22,56676,54637,5,0,0,0,0,1,0,34800,52154,0,M,-
22,37273,54637,56676,5,0,0,0,1,1,0,58464,28402,0,M,+
22,37273,54637,56676,5,0,0,0,1,1,0,58464,28322,0,M,+
22,2224,54113,20593,5,0,0,0,1,1,0,32760,44716,0,B,-
22,2224,54113,20593,5,0,0,0,1,1,0,32760,44580,0,B,-
1417,1103,61855,25411,5,0,0,0,0,1,0,17010,53159,0,B,-
2224,22,20593,54113,5,0,0,0,1,1,0,17460,59804,0,B,+
1103,1417,25411,61855,5,0,0,0,1,1,0,64450,5683,0,B,-
2224,22,20593,54113,5,0,0,0,1,1,0,17400,59728,0,B,-
```

Fig. 2. Ejemplo de trazas que se obtendrán como conjuntos de entrenamiento y test

B. Modelado de los NIDS

1) Análisis y diseño de la arquitectura de la Programación Genética: Primero se determinan cuales son los elementos terminales, operadores, función de fitness, parámetros de ejecución del algoritmo, profundidad máxima del árbol, etc. Los operadores y terminales determinan la semántica de los modelos generados, por lo que se eligen de forma que faciliten la consecución del objetivo principal establecido, esto es, con el fin de obtener un modelo fácil de interpretar que permita la búsqueda de evasiones sobre él. Para la obtención de los valores óptimos de los parámetros del algoritmo se utiliza la técnica de validación cruzada [26]. El funcionamiento es el siguiente:

1) Dividir el conjunto de entrenamiento en k hojas.
2) Establecer aleatoriamente unos valores para los parámetros que se deseen configurar. Se recomienda acotar dichos valores para facilitar el proceso de búsqueda (por ejemplo, no se debería permitir que el valor del número de generaciones fuera menor de 10).
3) Para cada k
   - Generar un fichero que contenga todas las trazas menos la de la hoja k.
   - Entrenar con ese fichero.
   - Testear con el fichero que contenga las trazas de la hoja k.
4) Almacenar las medias de los resultados de test.
5) Volver al paso 2. Este proceso se repetirá al menos $4n$ veces, donde n es el número de parámetros a configurar, para asegurarse una buena variabilidad.

Una vez finalizada la ejecución, se obtiene la configuración de parámetros que mejor resultados haya dado para utilizarla en la búqueda del modelo. La función de fitness en un principio será el error de clasificación, ya que no tiene sentido hablar de tasas de falsos positivos o negativos al tratarse de un modelado

del comportamiento de un NIDS, no de una detección real de ataques. Sin embargo, esta función podrá ser cualquier otra que se estime oportuno.

2) Implementación de la búsqueda del mejor modelo: En esta fase se busca el individuo (mediante PG con los parámetros determinados en B.1) que mejor emula a cada NIDS sobre el conjunto de entrenamiento. Los resultados obtenidos sobre el conjunto de test determinan si es necesario redefinir el diseño para obtener un mejor individuo. Es posible por lo tanto que esta tarea y la anterior estén correlacionadas, es decir, en el momento de analizar los resultados, si se observa que no son los esperados o deseados, es necesario que redefinir la arquitectura de PG, estableciendo nuevos parámetros, generando nuevos operadores o redefinendo la función de fitness.

3) Poda y preparación del modelo: En esta fase, se realiza una optimización manual del modelo para ajustarlo a una semántica de interpretación sencilla por parte de un humano. Esto es, el árbol generado mediante PG puede no ser sencillo de interpretar, puede usar nomenclaturas complejas, o quizás tenga nodos o incluso ramas enteras redundantes.

C. Análisis y estudio de técnicas de evasión

1) Aplicación de técnicas de evasión existentes: Se aplican las técnicas de evasión existentes en los paquetes de ataque para presentárselo al modelo generado y así poder observar su salida. Esto permite saber si el modelo es capaz o no de detectar los ataques cuando éstos han sido modificados para intentar evadir a los NIDS. El comportamiento observado se compara con el que había dado el NIDS original, para ver si realmente este modelo es fiel a su comportamiento. En esta fase se podrán extraer conclusiones acerca de qué técnicas evasivas no funcionan sobre el NIDS original y sí sobre el modelo generado, lo cual no es de mucha ayuda al ser este modelo en principio inexacto (es decir, no se comporta exactamente igual al NIDS). Sin embargo, si se diera alguna evasión en el NIDS original pero no sobre el modelo generado, podrían extraerse conclusión acerca del porqué el NIDS falla, y por lo tanto se podría ofrecer una solución.

2) Modificación de las técnicas de evasión existentes: Una vez estudiado el comportamiento del modelo ante las técnicas evasivas, se buscan otras nuevas para evadirlo. Esto se hace modificando las técnicas evasivas originales con el fin de buscar una manera distinta pero relacionada que, aplicadas sobre el modelo generado, provoquen que éste no sea capaz de detectar el ataque.

Es en esta tarea donde se corrobora la utilidad de modelar los NIDS, ya que los modelos generados nos permiten, mediante una sintaxis más sencilla y definida previamente, analizar el comportamiento que éstos tienen sin tener que entrar en los detalles internos de su implementación. Estos detalles, como se ha comentado previamente, pueden ser demasiado complejos y en muchos casos inaccesibles (cuando el código no sea abierto), por lo que la búsqueda de técnicas evasivas se convertiría en una tarea similar a la fuerza bruta. Sin embargo, utilizando los modelos obtenidos, esta tarea se realiza de una forma más sencilla y eficaz.

3) Prueba de las técnicas modificadas sobre el NIDS original y búsqueda de alternativas: Aquellas técnicas de evasión modificadas que realmente evadan el modelo generado son aplicadas al NIDS original con el fin de estudiar si las modificaciones han provocado también la evasión en éste. Es de esperar que, si el comportamiento del modelo es suficientemente parecido al del NIDS, aquellas evasiones que sean exitosas sobre el modelo lo sean también sobre el NIDS.

Hasta este punto de la metodología las evasiones probadas se basan en técnicas existentes o modificaciones de ellas. En este punto se van a probar nuevas técnicas, de distinta naturaleza, que, basándose en la semántica menos compleja del modelo evadan tanto al modelo como al NIDS (lo cual es el objetivo prioritario). Esta parte de la metodología es la menos automatizada, y la más estocástica, ya que, en función de la calidad del modelo, y aplicando la filosofía de prueba y error, encontrar nuevas evasiones puede no ser una tarea sencilla. Para poder encontrar nuevas técnicas, es necesario un conocimiento exhaustivo tanto del comportamiento del modelo (el cual lo ofrece el propio modelo) como del comportamiento de los protocolos de red (el cual está documentado y estudiado), por lo que es en esta parte en la cual el investigador ha de poner una mayor atención, ya que el resto de la metodología es un proceso automatizado que, una vez definido, es fácilmente replicable.

En la Figura 3 se puede observar un diagrama general del flujo de información en EVADIR. Como se puede observar, el proceso de búsqueda de evasiones se realiza primero sobre el modelo generado, y una vez que se encuentran técnicas para evadirlo, éstas se aplican sobre el NIDS original para evaluar su comportamiento.

IV. Conclusión

En los últimos años la complejidad de las Tecnologías de la Información se ha visto severamente incrementada. Esta complejidad hace que los sistemas requieran cada vez más de técnicas que garanticen su seguridad. Los Sistemas de Detección de Intrusiones de Red (NIDS) basados en usos indebidos son herramientas que detectan posibles intrusiones en un sistema, basándose en ataques conocidos y para los que están preparados. Cada vez que se publica un nuevo ataque, los administradores actualizan las firmas de sus NIDS para que sean capaces de detectar ese ataque. Esta situación provoca que los atacantes busquen nuevas técnicas para evitar ser detectados cuando pretenden comprometer un sistema. Los NIDS basados en la detección de usos indebidos detectan cualquier ataque para el que estén configurados siempre y cuando sean capaces de procesar los paquetes que encapsulan dicho ataque. Sin embargo, existe la posibilidad de evadir este procesamiento [7]. En este artículo se ha presentado una metodología que automatiza el proceso de buscar nuevas técnicas evasivas mediante el modelado del comportamiento de los NIDS a través de técnicas de Programación Genética. Los modelos generados tienen una semántica más sencilla

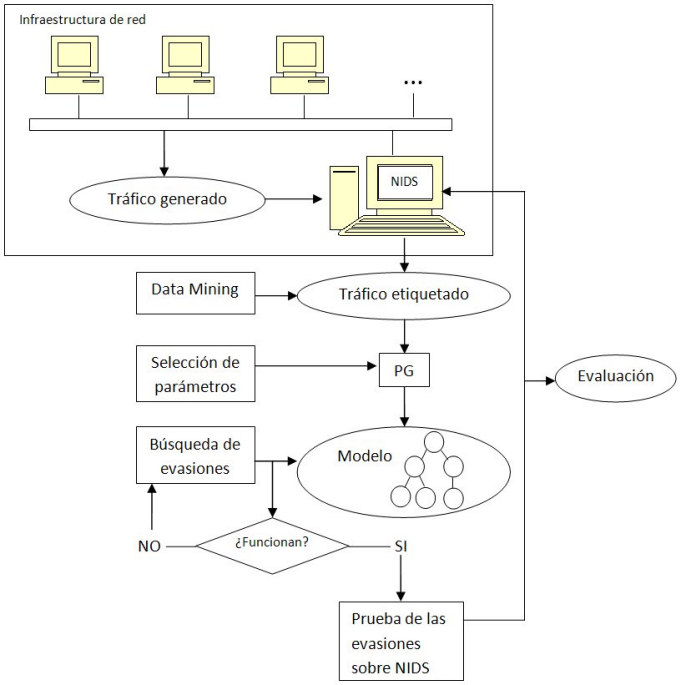

Fig. 3. Diagrama de flujo de información en EVADIR

que los NIDS, por lo que sobre estos modelos es más fácil buscar técnicas evasivas. A este respecto, cabe destacar que los NIDS propietarios no publican su código y por lo tanto no es sencillo conocer su funcionamiento interno [27], por lo que un modelado del mismo es adecuado para el estudio de su comportamiento. Es de suponer que, siempre y cuando el proceso de modelado se haya realizado de una manera correcta, las técnicas evasivas que funcionen sobre el modelo también lo hacen sobre el NIDS original.

Hasta la fecha hemos realizado pruebas de concepto sobre un NIDS basado en un árbol de decisión, haciendo uso de un escenario en el que hay presente tanto tráfico normal como tráfico malicioso (concretamente, escaneos de puertos). Los resultados de la prueba son fructíferos, ya que se han logrado encontrar algunas técnicas evasivas novedosas. Actualmente estamos aplicando esta metodología sobre NIDS reales, tanto de código abierto como propietarios. Como trabajo futuro se automatizará el proceso de buscar evasiones sobre los modelos (actualmente las evasiones son buscadas analizando la semántica del modelo producido manualmente), para lo cual se hace necesario establecer un formato común de la sintaxis que utilizan éstos modelos.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente realizado en el marco del proyecto SEGUR@, subvencionado por CDTI, Ministerio de Industria, Turismo y Comercio de España, dentro del programa CENIT, con referencia CENIT-2007 2004

REFERENCES

[1] D. Watson, M. Smart, R. G. Malan, and F. Jahanian, "Protocol scrubbing: network security through transparent flow modification", en *IEEE/ACM Transactions on Networking*, vol. 12, pp. 261–273, 2004.

[2] M. Vutukuru, H. Balakrishnan, and V. Paxson, "Efficient and Robust TCP Stream Normalization",en *SP '08: Proceedings of the 2008 IEEE Symposium on Security and Privacy*, Washington, DC, USA, 2008, pp. 96–110.

[3] G. Varghese, J. A. Fingerhut, and F. Bonomi, "Detecting evasion attacks at high speeds without reassembly",en *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, Pisa, Italy, 2006, pp. 327–338.

[4] D. Son. (2002) Fragroute. [Online]. *HYPERLINK "http://www.monkey.org/ dugsong/fragroute/"*

[5] U. Shankar and V. Paxson, "Active Mapping: Resisting NIDS Evasion without Altering Traffic",en *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, Washington, DC, USA, 2003, p. 44.

[6] M. Roesch, "Snort - Lightweight Intrusion Detection for Networks", en *LISA '99: Proceedings of the 13th USENIX conference on System administration*, Seattle, Washington, 1999, pp. 229–238.

[7] T. H. Ptacek and T. N. Newsham, "Insertion, evasion and denial of service: Eluding network intrusion detection", Technical report, 1998.

[8] L. Juan, C. Kreibich, C.-H. Lin, and V. Paxson, "A Tool for Offline and Live Testing of Evasion Resilience in Network Intrusion Detection Systems",en *DIMVA '08: Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, Paris, France, 2008, pp. 267-278.

[9] M. Handley, C. Kreibich, and V. Paxson, "Network intrusion detection: Evasion, traffic normalization and end-to-end protocol semantics", en *Proceedings of the 10th Conference on USENIX Security Symposium*, Volume 10, 2001, p. 9.

[10] D. J. Chaboya, R. A. Raines, R. O. Baldwin, and B. E. Mullins, "Network instrusion detection", vol. 8, pp. 36-42, 2006.

[11] G. Antichi, D. Ficara, S. Giordano, G. Procissi, and F. Vitucci, "Counting Bloom Filters for Pattern Matching and Anti-Evasion at the Wire Speed", en *IEEE Network Magazine of Global Internetworking*, vol. 23, no. 1, pp. 30-35, Feb. 2009.

[12] Nmap. [Online]. *HYPERLINK "http://nmap.org/"*

[13] Nikto. [Online]. *HYPERLINK " http://www.cirt.net/code/nikto.shtml"*

[14] A. Orfila, J. M. Estevez-Tapiador, and A. Ribagorda, "Evolving High-Speed, Easy-to-Understand Network Intrusion Detection Rules with Genetic Programming", en *EvoWorkshops '09: Proceedings of the EvoWorkshops 2009 on Applications of Evolutionary Computing*, Tübingen, Germany, 2009, pp. 93–98.

[15] J. Blasco, A. Orfila, and A. Ribagorda, "Improving Network Intrusion Detection by Means of Domain-Aware Genetic Programming",en *Proceedings of the 5th International Conference on Availability, Reliability and Security ARES 2010*, Krakow, Poland, 2010.

[16] G. Folino, C. Pizzuti, and G. Spezzano, "GP Ensemble for Distributed Intrusion Detection Systems",en *ICAPR, 2005*, pp. 54-62.

[17] S. Mukkamala, A. Sung, and A. Abrham, "Modeling intrusion detection systems using linear genetic programming approach", en *IEA/AIE'2004: Proceedings of the 17th international conference on Innovations in applied artificial intelligence*, Ottawa, 2004, pp. 633–642.

[18] S. Peddabachigari, A. Ajith, C. Grosan, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems", en *Journal in Network Computer Applications*, vol. 30, no. 1, pp. 114-132, 2007.

[19] M. Crosbie and E. Spafford, "Applying Genetic Programming to Intrusion Detection", en *Working Notes for the AAAI Symposium on Genetic Programming*, 1995, pp. 1-8.

[20] R. Bace and P. Mell, "NIST Special Publication on Intrusion Detection Systems," 800-31, 2001.

[21] Lawrence Berkley National Laboratory and ICSI. (2005) LBNL/ICSI Enterprise Tracibng Project. [Online]. www.icir.org/entreprise-tracing/

[22] S. Hettich and S. Bay. (1999) The UCI KDD Archive. [Online]. http://kdd.ics.uci.edu

[23] N. Athanasiades, J. G. Levine, H. L. Owen, and G. F. Riley, "Intrusion Detection Testing and Benchmarking Methodologies",en *Proceedings of the International Information Assurance Workshop*, IWIA 03, Maryland, 2003, pp. 63-72.

[24] J. R. Koza, "Genetic Programming: On the Programming of Computers", M. Press, Ed.Cambridge, MA, USA, 1992.

[25] P. Barham , B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization", en *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, October 19 - 22, 2003, ACM, New York, NY, 164-177.

[26] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection", in *IJCAI: International Joint Conference on Artificial Intelligence*, Volume 2, Issue 1, 1137–1143, 1995

[27] D. Mutz, C. Kruegel, W. Robertson, G. Vigna, and R. A. Kemmerer "Reverse Engineering of Network Signatures", in *Proceedings of the AusCERT Asia Pacific Information Technology Security Conference, Gold*, 2005