Sistemas de detección de intrusos en red

Joaquín García Alfaro

PID_00191698

Universitat Oberta de Catalunya
www.uoc.edu

Índice

Introducción

El objetivo de este módulo es profundizar nuestros conocimientos en el área de los sistemas de detección de intrusos (IDS). Como veremos a lo largo de este módulo, dichos sistemas engloban en realidad un vasto número de conceptos, técnicas y herramientas que son a menudo difíciles de definir de manera aislada. De hecho, incluso la utilización del término *intruso* es a veces mal interpretada en la literatura. El concepto de intruso es utilizado por lo general para caracterizar acciones cometidas por usuarios no autorizados que violan la política de seguridad y logran introducirse de manera ilícita en un sistema. Aun así, los IDS no se limitan a descubrir únicamente la fase final de una entrada no autorizada en un sistema, sino que por lo general abarcan también mecanismos capaces de tratar las fases previas a una intrusión, así como muchas otras acciones hostiles contra un sistema, llegando incluso a permitir el tratamiento de cualquier otra tarea que pueda ser considerada como potencialmente peligrosa o con un comportamiento fuera de lo común (respecto al funcionamiento normal del sistema).

Aunque existen, en general, dos grandes categorías de IDS en el mercado, según si su proceso de detección se centra directamente en el tratamiento del flujo de datos de una red (conocidos como NIDS, o sobre las tareas ejecutadas a nivel de equipo (conocidos como HIDS), en este módulo nos centraremos únicamente en los IDS a nivel de red. Veremos también que el proceso de detección puede resumirse en dos modalidades principales. Por un lado, diferenciaremos los mecanismos de detección que realizan un proceso de análisis de datos con el fin de detectar usos indebidos (del inglés, *misuse-based intrusion detection*), conocidos a priori. Esta primera técnica (conocida, en inglés, como *signature-based intrusion detection*) se basa en obtener un conjunto de patrones o firmas definidas explícitamente en forma de reglas de detección. La segunda categoría (definida, en inglés, como *anomaly-based intrusion detection*) se basa en detección de acciones consideradas como anormales, o fuera de lo común. Los IDS que utilizan este segundo tipo de detección se basan en la utilización de heurísticas y modelos probabilísticos.

A continuación, veremos un ejemplo concreto de NIDS a través de la herramienta de software libre Snort. Snort es utilizado de forma directa por operarios de red, del mismo modo que terceras herramientas que basan su captura y registro de paquetes en redes TCP/IP a través del motor de recogida de Snort. De hecho, Snort puede ser utilizado tanto para garantizar una vigilancia continuada en busca de intentos de intrusión o de usos indebidos en una simple red local, como para implementar sistemas de detección distribuidos. En este módulo repasaremos brevemente los orígenes de la herramienta y estudiaremos

de modo general su arquitectura y su modo de trabajar, así como las posibilidades de extender sus características de generación de informes por medio de herramientas gráficas adicionales.

Por último, trataremos la necesidad de funcionalidades adicionales para completar el proceso de detección de un NIDS como Snort. Repasaremos el uso de nuevos elementos encargados de gestionar sistemas y técnicas de detección heterogéneas, con el objetivo final de poder controlar y reaccionar apropiadamente ante escenarios de intrusión más complejos. Estos nuevos elementos, generalizados con el nombre de SIEM, tienen por objetivo facilitar la gestión de grandes volúmenes de información generados por herramientas de detección, no únicamente IDS, sino también eventos generados a partir de sistemas de cortafuegos, escáneres de vulnerabilidades o incluso sistemas antivirus. Veremos cómo dichos elementos proporcionarán técnicas necesarias para normalizar los eventos recibidos y, finalmente, realizar tareas de fusión de información y correlación de alertas.

Objetivos

Los objetivos básicos que se pretenden alcanzar son los siguientes:

1. Comprender las distintas técnicas de detección que pueden ser utilizadas por los sistemas de detección de intrusos, así como aprender a clasificar dichos sistemas según criterios tales como el lugar donde se produce el proceso de análisis de los datos o el mecanismo de detección concreto.
2. Ver un ejemplo concreto de IDS en red a través de la herramienta de software libre Snort.
3. Entender las limitaciones ligadas a los procesos tradicionales de un sistema de detección de intrusos, incluyendo como problemática la posibilidad de falsos positivos y falsos negativos, y la necesidad de complementos adicionales para consolidar el tratamiento de incidentes detectados.
4. Conocer la existencia de herramientas finales de gestión, tales como gestores de eventos capaces de normalizar, fusionar y poner en correspondencia alertas recogidas de manera distribuida por sondas de detección heterogéneas.

1. Detección de intrusos en red

La instalación de un sistema para la detección de intrusos* pretende mejorar la seguridad de un sistema por medio de la incorporación de herramientas o dispositivos capaces de avisar a los operadores en el momento en el que se produzcan ataques conocidos contra la seguridad del sistema; o desviaciones del comportamiento habitual de sus usuarios o equipos.

*** En inglés, *intrusion detection system* (IDS).**

En realidad, la detección de intrusos parte de la idea implícita de violación de la política de seguridad de un sistema, lo que supone un ataque parcial o total cuyo objetivo final es el de obtener un acceso con privilegios de administrador al sistema.

Los mecanismos para la detección de intrusos tratan de encontrar y reportar actividad maliciosa en tráfico de red o a nivel de sistema y aplicaciones, pudiendo llegar a reaccionar adecuadamente antes de que el objetivo final del ataque se produzca.

En la mayoría de los casos es deseable poder identificar el ataque exacto que se está produciendo, de manera que sea posible detener el ataque y recuperarse de él. En otras situaciones, solo será posible detectar e informar de la actividad sospechosa que se ha encontrado, ante la imposibilidad de conocer lo que ha sucedido realmente.

Generalmente, el proceso de detección trabajará con la premisa de que nos encontramos en la peor de las situaciones, es decir, que el atacante ha obtenido un acceso al sistema y que es capaz de utilizar o modificar sus recursos.

A continuación introduciremos dos definiciones básicas en el campo de la detección de intrusos con el objetivo de clarificar términos comunes que se utilizarán más adelante.

Una **intrusión** es una secuencia de acciones realizadas por un usuario o proceso deshonesto con el objetivo final de provocar un acceso no autorizado sobre un equipo o un sistema al completo.

La intrusión consistirá en la secuencia de pasos realizados por el atacante que viola una determinada política de seguridad. La existencia de una política de

seguridad, en la que se contemplan una serie de acciones deshonestas que hay que prevenir, es un requisito clave para la intrusión. Es decir, la violación solo se podrá detectar cuando las acciones observadas puedan ser comparadas con el conjunto de las reglas definidas en la política de seguridad.

La **detección de intrusos** es el proceso de identificación y respuesta ante las actividades ilícitas observadas contra uno o varios recursos de una red.

Esta última definición introduce la noción de proceso de detección de intrusos, que involucra toda una serie de tecnologías, usuarios y herramientas necesarias para llegar a buen término.

A la hora de clasificar este tipo de sistemas encontramos básicamente dos categorías principales, según la localización desde la que se obtienen datos. Así, hablaremos de IDS a nivel de red o NIDS*, y de IDS a nivel de equipo o HIDS**. Los NIDS basan su proceso de detección a partir de, por ejemplo, el flujo de datos capturado a través de la interfaz de red asociada al IDS. Los HIDS analizarán información de eventos a nivel de sistema operativo (como, por ejemplo, intentos de conexión y llamadas al sistema).

*** Del inglés,** ***network-based intrusion detection systems.***
**** Del inglés,** ***host-based intrusion detection systems.***

Necesidad de instalar un NIDS

La mejor manera de entender la necesidad de incorporar estos elementos podría ser la comparación entre la seguridad de una red informática y la seguridad de un edificio: las puertas de entrada ejercen un primer nivel de control de acceso, pero normalmente no nos quedamos aquí; instalaremos detectores de movimiento o cámaras de vigilancia en puntos clave del edificio para detectar la existencia de personas no autorizadas, o que hacen un mal uso de los recursos, poniendo en peligro la seguridad. Además, existirán vigilantes de seguridad, libros de registro en los que se apuntará a todo el personal que accede a un determinado departamento que consideramos crítico, etcétera. Toda esta información se procesa desde una oficina de control de seguridad donde se supervisa el registro de las cámaras y se llevan los libros de registro. Todos estos elementos, proyectados en el mundo digital, configuran lo que se conoce en el ámbito de la seguridad de redes informáticas como mecanismos de detección.

A nivel de red, un NIDS debe ser capaz de poder detectar los ataques siguientes:

- Escáneres de vulnerabilidades maliciosos. Un NIDS deberá ser capaz de detectar actividades maliciosas realizadas por herramientas asociadas a una rootkit, una botnet, etc., y que supongan, por ejemplo, un escaneo ilícito de puertos, búsqueda de versiones antiguas de servicios o interrogaciones que conlleven a una denegación de servicio.

- Propagación de virus, gusanos, o tentativa de instalación de los componentes de una rootkit, o de aplicaciones troyanas.

- Explotación (interna o externa) de vulnerabilidades conocidas en protocolos, tales como DNS, FTP, HTTP, ICMP, SMTP, POP3, RPC.

- Utilización abusiva de servicios de red, tanto por parte de usuarios internos como externos a la red.

- Ataques de ingeniería social contra aplicaciones de mensajería instantánea, chats, P2P, etcétera. La detección de ataques basados en técnicas de *phishing* podría ser un ejemplo dentro de esta categoría.

Ejemplo

Un ejemplo de utilización abusiva de servicios de red es la instalación ilícita de servidores de música, o de servidores de transferencia de ficheros mediante técnicas de P2P.

De manera indirecta, un NIDS podría igualmente detectar y reportar incidentes asociados al malfuncionamiento de la red no solo a causa de ataques o acciones malintencionadas, sino simplemente debido a errores de configuración o fallos en los equipos informáticos del sistema. Algunos NIDS especializados en este último tipo de detección han evolucionado en sistemas completos para la detección de vulnerabilidades en red, conocidos como VDS*. Los VDS suelen incorporar funciones avanzadas para la ejecución de procesos internos de auditoría de una red con el objetivo de identificar y aislar aquellos componentes locales (internos) de la red que provocaron el incidente.

*** Del inglés, *vulnerability detection system.***

El tipo de detección proporcionado por un NIDS se puede clasificar en dos categorías principales: detección basada en usos indebidos y detección basada en anomalías. Presentaremos, a continuación, algunos de los detalles más relevantes de cada una de las categorías.

Detección de intrusos en redes inalámbricas

A diferencia de los sistemas de detección de intrusos que supervisan el tráfico de redes tradicionales por medio de firmas de intrusiones, los sistemas de detección de intrusos en redes inalámbricas suelen ofrecer técnicas adicionales para supervisar el espectro radioeléctrico de estas redes en busca de accesos no autorizados o acciones anómalas. La utilización de estos sistemas es importante, ya que puede prevenir a los administradores del sistema acerca de agujeros de seguridad debidos a errores de configuración, o de malinterpretación de las políticas de seguridad de la organización. Los eventos recogidos por estos detectores pueden prevenir accesos indebidos en la red mucho antes de que los incidentes se produzcan. También se busca detectar con estos componentes ataques de tipo *man-in-the-middle*, así como suplantación de direcciones MAC físicas o indicios de posibles ataques de denegación de servicio (DoS) debido a la saturación del medio inalámbrico utilizado.

1.1. Detección basada en usos indebidos

La detección basada en el modelo de usos indebidos cuenta con el conocimiento a priori de secuencias y actividades deshonestas. Los IDS que implementan este esquema analizan los eventos en busca de patrones de ataque conocidos o actividad que ataque vulnerabilidades típicas de los sistemas supervisados.

Estas secuencias o patrones se conocen bajo el nombre de *firmas de ataques* y podrían ser comparadas con las firmas víricas que utilizan los productos actuales de detección de virus.

Así pues, los IDS basados en el modelo de usos indebidos compararán los eventos recogidos con las firmas de ataque que mantienen almacenadas en sus bases de conocimiento.

En el momento de detectar concordancia de algún acontecimiento o secuencia de eventos con alguna firma de ataque, el componente lanzará una alarma.

La mayoría de los NIDS comerciales basan su detección en el modelo de usos indebidos. Por ello, su configuración suele realizarse mediante el uso de firmas que caracterizan ataques conocidos y puestas al alcance de los operarios de estos sistemas de manera automática mediante una suscripción en línea a los proveedores de los productos. Cada firma define un conjunto de eventos y condiciones que representan el ataque en cuestión. El reconocimiento se basa, así pues, en algoritmos de tipo *pattern-matching* con el objetivo de identificar la huella del ataque en el tráfico de red que está siendo supervisado por el NIDS en cuestión. A continuación, y según la configuración del NIDS, un fichero de registro (en inglés, *log*) o una alarma con mayor nivel semántico asociado al ataque podrán ser generados y almacenados para su posterior análisis por parte del operario de la red.

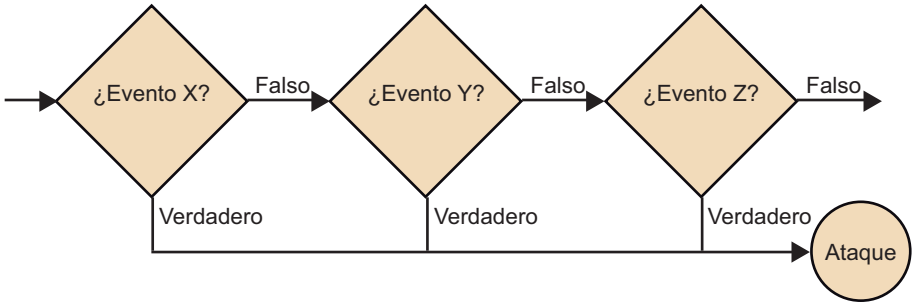
A la hora de implementar un esquema de detección basado en usos indebidos, dos de las técnicas más utilizadas son el reconocimiento de patrones y la transiciones de estados:

Lectura recomendada

S. Northcutt; J. Novak (2002). *Network Intrusion Detection* (3.ª ed.). New Riders Publishing.

- **Detección basada en reconocimiento de patrones.** Mediante la utilización de reglas de tipo *if-then-else* para examinar los datos, un IDS basado en reconocimiento de patrones procesará la información por medio de funciones internas en el sistema, de manera completamente transparente al usuario. La figura 1 muestra el esquema de este tipo de reglas.

Figura 1. Ejemplo de una regla *if-then-else*

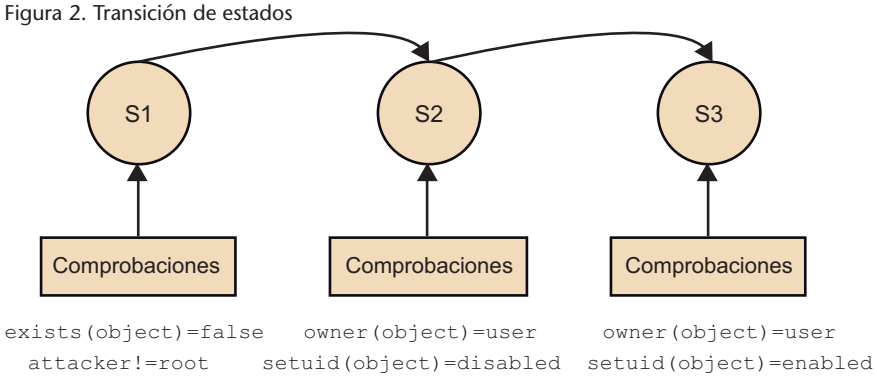


Aunque este modelo permite detectar una intrusión a partir de patrones conocidos a priori, su desventaja principal es que los patrones no definen un orden secuencial de las acciones.

Detectar mediante este modelo ataques compuestos por una secuencia de eventos puede llegar a implicar grandes dificultades. Por otra parte, el mantenimiento y la actualización de la base de datos de patrones son otros puntos críticos del modelo.

- **Detección basada en transición de estados.** Este modelo hace uso de autómatas finitos para representar los ataques, donde los nodos representan los estados y las flechas (arcos), las transiciones. La figura 2 muestra un ejemplo de ello.

Figura 2. Transición de estados



La utilización de diagramas de transición facilita la asociación entre los estados y los distintos pasos que realiza un atacante desde que entra en un sistema, con privilegios limitados, hasta que se hace con su control.

Como principales ventajas de este modelo se puede destacar que los diagramas de transición permiten obtener una representación a alto nivel de escenarios de intrusión, y ofrecen así un modo de identificar una serie de secuencias que conforman el ataque.

Por otra parte, estos diagramas definen de manera muy sencilla los ataques que se deben detectar. El motor de detección de un IDS basado en este modelo podría llegar a utilizar diferentes variantes del mismo diagrama para identificar ataques similares.

Por el contrario, los diagramas de transición y, por lo tanto, los distintos pasos de la secuencia se deben crear mediante lenguajes específicos que, en muchas ocasiones, suelen ser muy limitados e insuficientes para recrear ataques complejos.

Esta limitación provoca que este modelo no pueda detectar algunos de los ataques más comunes, por lo que resulta necesario el uso de técnicas mixtas que complementen reglas de tipo *if-then-else* y técnicas de transición de estados.

1.2. Detección basada en anomalías

Los procesadores de eventos que basan su detección en un esquema de anomalías tratarán de identificar actividades sospechosas comparando el comportamiento de un usuario, proceso o servicio, con el comportamiento de perfil clasificado como normal.

Un perfil sirve como métrica (medida de un conjunto de variables) de comportamientos normales. Cualquier desviación que supere un cierto umbral respecto al perfil almacenado será tratado como una evidencia de ataque o intrusión.

Uno de los requisitos de este modelo es la necesidad de inicialización de un perfil por defecto que se irá adaptando progresivamente al comportamiento de un usuario, proceso o servicio no sospechoso. Es necesario, por lo tanto, el uso de heurísticas y descriptores estadísticos que ayuden a modelar correctamente cambios en el comportamiento tan pronto como suceda. Otras propuestas tratan de incorporar técnicas de inteligencia artificial para ejecutar estas tareas (como, por ejemplo, el uso de redes neuronales o de algoritmos genéticos).

La detección basada en anomalías ofrece claras ventajas respecto a la detección basada en usos indebidos. La ventaja más destacable es la posibilidad de detectar ataques desconocidos. Esto es posible porque, independientemente de cómo el atacante haya conseguido la intrusión en un sistema, tan pronto como sus actividades comiencen a desviarse del comportamiento de un usuario normal, el procesador de eventos lanzará una alarma avisando de una posible intrusión.

Aun así, el esquema de detección basado en anomalías presenta bastantes inconvenientes. El primero que debemos destacar es la falta de garantía en el proceso de detección: un intruso podría ejecutar sus acciones poco a poco para provocar cambios en el perfil de usuario del procesador de eventos con la finalidad de que su presencia en el sistema pasara desapercibida.

Inconvenientes

Los inconvenientes del esquema de detección basado en anomalías provocan que la mayoría de los sistemas de detección comerciales disponibles en la actualidad implementen, por lo general, esquemas basados en el modelo de usos indebidos.

Como segundo inconveniente podemos destacar la dificultad que aparece a la hora de clasificar y describir con precisión los ataques detectados mediante analizadores basados en anomalías. Generalmente, un IDS no solo tiene que lanzar una alarma, sino que deberá especificar de dónde procede el ataque, qué cambios ha sufrido el sistema, etcétera.

Además, la tasa de falsos positivos y negativos que puede darse utilizando este esquema de detección es un gran inconveniente, ya que no siempre una desviación respecto al perfil esperado coincidirá con un ataque o intento de intrusión. En el caso de un NIDS, es posible que el número de alarmas lanza-

das (en una red de tamaño medio) supere fácilmente el centenar. Esto provoca que, con frecuencia, los administradores de la red acaben ignorando las alarmas lanzadas por el sistema de detección, o incluso desactivando el sistema al completo.

Falsos positivos y falsos negativos

Un falso positivo sucede en aquellas situaciones en las que el NIDS caracteriza como malicioso tráfico legítimo, que no forma parte de ningún ataque. Es, por lo tanto, un evento detectado por equivocación. Por el contrario, un falso negativo se produce en aquellas situaciones en las que tráfico malicioso es descartado, considerándolo tráfico legítimo por equivocación. Se trata, por lo tanto, de un evento que debería ser detectado, pero que escapa al proceso de detección.

2. Detección de intrusos en red con Snort

Snort es una completa herramienta de seguridad basada en código abierto para la creación de sistemas de detección de intrusos en entornos de red. Cuenta con una gran popularidad entre la comunidad de administradores de redes y servicios. Gracias a su capacidad para la captura y el registro de paquetes en redes TCP/IP, Snort puede ser utilizado para implementar desde un simple *sniffer* de paquetes para la monitorización del tráfico de una pequeña red hasta un completo sistema de detección de intrusos en tiempo real.

Ved también

En el apartado 3 de este módulo podéis ver algunos ejemplos prácticos sobre Snort.

Como monitor de red, Snort se comporta como una auténtica aspiradora (de ahí su nombre) de datagramas IP, ofreciendo diferentes posibilidad en cuanto a su tratamiento. Desde actuar como un simple monitor de red pasivo que se encarga de detectar el tráfico maligno que circula por la red, hasta la posibilidad de enviar a servidores de ficheros de registro o servidores de base de datos todo el tráfico capturado.

Pero, aparte de unas estupendas características como *sniffer* de paquetes y generador de alertas, Snort tiene muchas otras características que le han permitido convertirse en una de las soluciones software más completas para la construcción de sistemas de detección en entornos de red basados en reconocimiento de patrones. Snort se autodefine como un NIDS ligero*. Este calificativo de *ligero* significa que, como IDS, su diseño e implementación le permite poder funcionar bajo diferentes sistemas operativos y que sus funciones como mecanismo de detección podrán formar parte en distintos productos de seguridad (incluso comerciales).

* Del inglés, *lightweight network intrusion detection system.*

La popularidad de Snort se ha incrementado estos últimos años en paralelo al incremento de popularidad de sistemas operativos de código abierto, como puede ser la familia de sistemas GNU/Linux y BSD (NetBSD, OpenBSD, FreeBSD y Mac OS X).

Pero su naturaleza como producto de código abierto no lo limita a estar disponible únicamente bajo este tipo de sistemas operativos. Snort puede funcionar bajo soluciones comerciales, como, por ejemplo, Microsoft Windows.

Desde el punto de vista de su motor de detección, Snort estaría incluido en la categoría de detección basada en usos indebidos. Mediante un reconocimiento de firmas, Snort contrastará todo el tráfico capturado en sus reglas de detección.

Una regla de detección de Snort no es más que un conjunto de requisitos que le permitirán activar una alarma, en caso de cumplirse. Por ejemplo, una regla de Snort que permitiría verificar el uso de aplicaciones P2P para el intercambio de ficheros a través de Internet verificando el uso de la cadena `GET` en servicios diferentes al puerto tradicional del protocolo HTTP. Si un paquete capturado por Snort coincide con esta sencilla regla, su sistema de notificación lanzará una alerta indicando lo sucedido. Una vez que la alerta sea lanzada, puede ser almacenada de distintas maneras y con distintos formatos, como, por ejemplo, un simple fichero de registro del sistema, una entrada en una base de datos de alertas, un evento SNMP, etcétera.

A continuación veremos los orígenes de Snort, así como un análisis de su arquitectura y algunas de sus características más destacables.

Versión comercial de Snort

Aunque Snort está disponible bajo licencia GPL (GNU Public License), existen productos comerciales basados directamente en Snort y distribuidos por la empresa *Sourcefire*, fundada por el creador de Snort, Marty Roesch. El análisis de estas versiones comerciales quedan fuera del propósito de este módulo didáctico. Para más información, podéis visitar http://sourcefire.com.

2.1. Origen de Snort

De manera muy resumida, podemos definir Snort como un decodificador (o *sniffer*) de paquetes con funcionalidades adicionales para el registro de paquetes, generación de alertas y un motor de detección basado en usos indebidos.

Snort fue desarrollado en 1998 bajo el nombre de APE. Su desarrollador, Marty Roesch, trataba de implementar un decodificador de paquetes multiplataforma (aunque su desarrollo inicial se hizo para el sistema operativo GNU/Linux) que contara con diferentes opciones de clasificación y visualización de los paquetes capturados. Marty Roesch implementó Snort como una aplicación basada en la librería `libcap` (para el desarrollo de la captura de paquetes), lo cual garantizaba una gran portabilidad tanto en la captura como en el formato del tráfico recogido.

Algo más que un *sniffer*

El autor de Snort trataba de indicar con el nombre de Snort que su aplicación era algo más que un *sniffer*. La palabra *snort* significa en inglés una acción de inhalar o esnifar de manera más obsesiva y violenta. Además, Marty dijo en su momento que ya tenía demasiadas aplicaciones que se llamaran `a.out` y que todos los nombres populares para *sniffers* llamados *TCP-something* ya estaban cogidos.

Snort empezó a distribuirse a través del sitio web *Packet Storm** el veintidós de diciembre de 1998, contando únicamente con mil seiscientas líneas de código y un total de dos ficheros fuente. Por aquella época, el uso principal que le dio su autor a Snort era como analizador de sus conexiones de red a través de un cable modem y como *debugger* de las aplicaciones de red que estaba implementado.

***http://www.packetstormsecurity.com**

El primer analizador de firmas desarrollado para Snort (también conocido como analizador de reglas por la comunidad de desarrollo de Snort) se añadió como nueva funcionalidad de la aplicación en enero de 1999. Esta nueva funcionalidad permitió que Snort comenzase a ser utilizado como detector de intrusiones.

En diciembre de 1999 apareció la versión 1.5 de Snort. En esta versión, su autor decidió una nueva arquitectura basada en *plug-ins* que aún se conserva en las versiones actuales. A partir de esta versión, Marty Roesch abandonó la compañía donde trabajaba y se empezó a dedicar a tiempo completo a Snort con el fin de añadir nuevas funcionalidades para mejorar las capacidades de configuración y facilitar su uso en entornos más profesionales. Gracias a la gran aceptación que su NIDS obtenía entre la comunidad de administradores, Marty pensó que era un buen momento para ofrecer su producto con un soporte para empresas y obtuvo la financiación necesaria para fundar *Sourcefire*.

Enlace de interés

En el sitio web http://sourcefire.com encontraréis más información sobre *Sourcefire*.

Sin embargo, Snort continúa siendo código libre y promete seguir siéndolo para siempre. La última versión disponible* de Snort se presenta con más de setenta y cinco mil líneas de código y una restructuración total en cuanto al diseño original de su arquitectura inicial.

*** Versión 2.9 en el momento de escribir este módulo didáctico.**

Aunque el soporte y desarrollo actual de Snort se hace desde *Sourcefire* de forma comercial, existe la versión libre bajo licencia GNU. Esta versión puede ser descargada libremente desde la web de Snort*, lo que permite que cualquier usuario pueda disponer de soporte para las últimas versiones disponibles y las últimas actualizaciones de los ficheros de reglas para dichas versiones.

*** http://www.snort.org**

Actualmente, Snort cuenta un gran repertorio de accesorios que permiten reportar notificaciones en diferentes gestores de base de datos (como MySQL y PostgreSQL) y un gran número de preprocesadores de tráfico que permite poder analizar llamadas RPC y escaneo de puertos antes de que estos sean contrastados con el conjunto de las reglas asociado en busca de nuevos incidentes.

Los conjuntos de reglas de Snort también han ido evolucionando a medida que la aplicación crecía. El tamaño de los conjuntos de reglas para la última versión Snort disponibles para descargar se incrementa de forma similar a la velocidad de aparición de nuevos exploits. Estos ficheros de reglas se encuentran actualmente clasificados en distintas categorías, como, por ejemplo, P2P, ataques de denegación de servicio, ataques contra servicios web, virus, tráfico pornográfico, etcétera.

Cada una de estas reglas está asociada a un identificador único (sensor ID, SID) que permite reconocer y encontrar información acerca del ataque o mal uso detectado. Por ejemplo, el SID para el ataque `SSH banner attack` es

el `1838`. Además, gracias al uso mayoritario de Snort entre la comunidad de administradores de red, otros NIDS han adoptado el formato de las reglas de Snort, así como la codificación utilizada para los volcados de los paquetes capturados (basada el `libcap`).

El soporte de estos ficheros de reglas aumenta a diario. De este modo, cualquier usuario de Snort, o de cualquier otro NIDS con un formato de reglas compatible, podría crear sus propias reglas a medida que nuevos ataques vayan apareciendo y colaborar con la comunidad de desarrollo de Snort para mantener perfectamente actualizada su base de firmas.

2.2. Arquitectura de Snort

Snort proporciona un conjunto de características que lo hacen una herramienta de seguridad muy potente, entre las que destacan la captura del tráfico de red, el análisis y registro de los paquetes capturados y la detección de tráfico malicioso o deshonesto. Antes de ver en mayor detalle las características de Snort es importante conocer y comprender su arquitectura.

Snort se compone de un conjunto de componentes, la mayoría desarrollados como *plug-ins* que permiten la personalización de Snort. Entre estos componentes destacan los preprocesadores, que permiten que Snort manipule de manera más eficiente el contenido de los paquetes antes de pasarlo al elemento de detección, y su sistema de notificaciones, que permite que la información reportada pueda ser enviada y almacenada en distintos formatos y siguiendo distintos métodos.

Plug-ins de Snort

El término *plug-in* (del inglés, *enchufable*) hace referencia a módulos software de una aplicación desarrollados de manera independiente al núcleo general de la aplicación. El objetivo es añadir funcionalidades adicionales, sin necesidad de afectar al código fuente del núcleo o al resto de los componentes. Para ello, la aplicación debe proporcionar una interfaz de programación o API (del inglés, *application programming interface*) que permita el desarrollo y la compilación de tales complementos. Así pues, un *plug-in* Snort es un componente desarrollado conforme la API de *plug-ins* de Snort, que será utilizado junto al núcleo del código de Snort, pero separado de él, de forma que un cambio en el código del componente no afecte al núcleo o a los otros componentes.

La arquitectura central de Snort se basa en los siguientes cuatro componentes:

- Decodificador de paquetes
- Preprocesador
- Motor de detección
- Sistema de notificaciones

Siguiendo dicha estructura, Snort permitirá la captura y el preprocesado del tráfico de la red a través de los dos primeros componentes (decodificador de

paquetes y preprocesador), realizando posteriormente un chequeo contra ellos mediante el motor de detección (según el conjunto de reglas activadas) y generando, por parte del último de los componentes, las notificaciones pertinentes.

La figura 3 muestra la arquitectura básica de Snort que acabamos de comentar. Observando la figura podemos hacer un símil entre Snort y una máquina mecánica para la ordenación automática de monedas:

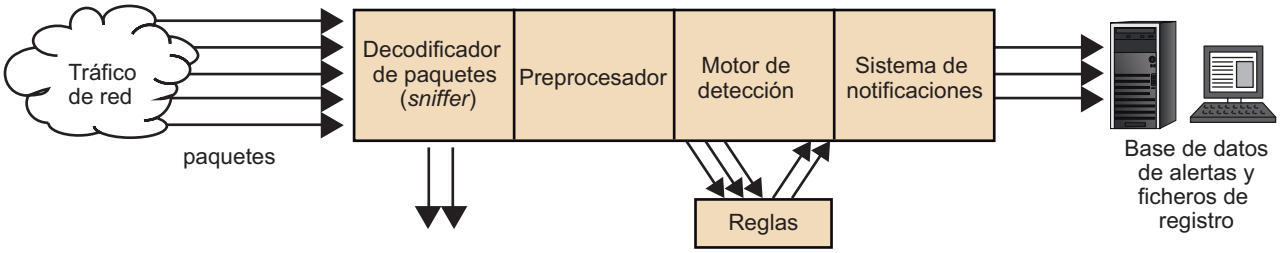
1) Toma todas las monedas (paquetes de la red recogidos por el decodificador de paquetes).

2) Cada moneda se dejará caer por una rampa para determinar a qué grupo de monedas pertenece (preprocesador de paquetes).

3) Ordena las monedas según cada tipo de moneda y las enrolla en forma de canutos según la categoría (motor de detección).

4) Finalmente, el administrador decidirá qué hacer con cada uno de los canutos de monedas ordenadas (sistema de notificaciones).

Figura 3. Arquitectura básica de Snort



Tanto el preprocesador como el motor de detección y el sistema de notificaciones de Snort son también implementados en forma de componentes independientes. A continuación examinaremos con mayor detalle cada uno de los componentes básicos de Snort que acabamos de ver.

2.3. Decodificador de paquetes

Un *sniffer* (o decodificador de paquetes) es un dispositivo (software o hardware) que se utiliza para poder capturar los paquetes que viajan por la red a la que es asociado.

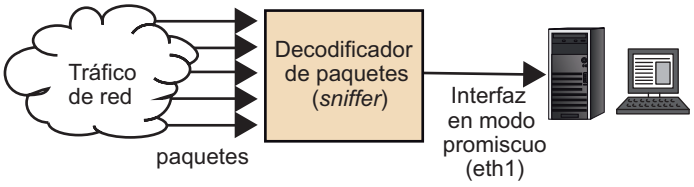
En el caso de redes TCP/IP, este tráfico suele ser tráfico de datagramas IP, aunque también es posible la existencia de tráfico de distinto tipo, como, por

ejemplo, tráfico IPX o tráfico AppleTalk. Además, puesto que el tráfico IP consiste también en distintos tipos de protocolos, como TCP, UDP, ICMP, protocolos de encaminamiento, IPSec, etc., muchos *sniffers* necesitarán conocer a priori el tipo de tráfico para poder interpretar más adelante los paquetes que van siendo recogidos y poder mostrarlos en un lenguaje comprensible por un administrador de red.

Como muchas otras herramientas relacionadas con la seguridad en redes, los *sniffers* pueden ser utilizados con objetivos más o menos deshonestos. Entre los distintos usos que se le puede dar a un *sniffer* podemos pensar en análisis de tráfico para la solución de congestiones y problemas de red, mejora y estudio del rendimiento de los recursos, captura pasiva de información sensible (contraseñas, nombres de usuario, etc.).

Así, como el resto de los *sniffers* tradicionales, el decodificador de paquetes de Snort será el elemento encargado de recoger los paquetes que más adelante serán examinados y clasificados por el resto de los componentes. Para ello, el decodificador de paquetes deberá ser capaz de capturar todo el tráfico que le sea posible, para más adelante pasarlo al siguiente componente (el preprocesador), que se encargará de detectar qué tipo de tráfico se ha recogido. La figura 4 muestra un esquema del funcionamiento del decodificador de paquetes de Snort.

Figura 4. Funcionamiento del decodificador de paquetes de Snort



Interfaz de red en modo promiscuo

El modo promiscuo es el modo por el que la tarjeta de red de un equipo conectado a una red, tanto red cableada como inalámbrica, permitirá la captura de todo el tráfico que circule a través de dicha interfaz de red.

2.4. Preprocesador

A medida que el decodificador de paquetes de Snort va recogiendo el tráfico que pasa por la red, se lo irá entregando al elemento de preprocesado. Este elemento irá adaptando los paquetes capturados y se los entregará al motor de detección.

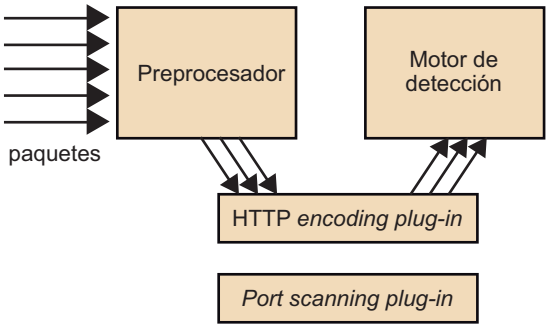
Así pues, el preprocesador irá obteniendo paquetes sin tratar (*raw paquets*) y los verificará mediante un conjunto de *plug-ins*. Por ejemplo, utilizará un *plug-in* para el tratamiento de paquetes relacionados con tráfico de tipo RPC* o un *plug-in* para el tratamiento de paquetes relacionados con escaneos de puertos. Estos *plug-ins* verificarán los paquetes en busca de ciertos comportamientos que permitan a Snort determinar su tipo. Una vez determinado el tipo, el paquete será enviado hacia el motor de detección.

*** Del inglés, *remote procedure calls.***

Esta característica de preprocesamiento es realmente importante para una herramienta de detección, ya que es posible la utilización de terceras aplicaciones que pueden ser activadas y desactivadas según las necesidades del nivel de preprocesado. Por ejemplo, si a un administrador de red no le preocupa el tráfico RPC que entra y sale de su red (y no necesita, por tanto, analizarlo) por cualquier motivo, no tendrá más que desactivar el *plug-in* de RPC y seguir utilizando el resto.

La figura 5 muestra un esquema donde el preprocesador de Snort utiliza dos de sus *plug-ins* para verificar el tipo del paquetes que recibe y decidir si deben ser pasados o no al motor de detección.

Figura 5. Preprocesador de Snort combinando dos *plug-ins* de tratamiento de datos



Los preprocesadores de Snort ofrecen gran flexibilidad para la implementación de distintos algoritmos de tratamiento de tráfico en Snort.

2.5. Reglas de detección

Como ya adelantábamos antes, Snort basa su detección en el modelo de usos indebidos. Por ello, requiere ser configurado mediante un conjunto de reglas que serán utilizadas por el módulo de detección para proceder al reconocimiento de ataques y firmas de intrusión. Las reglas de Snort son agrupadas, por lo general, en conjuntos de firmas que categorizan los incidentes. Así, encontraremos conjuntos de reglas asociadas a la detección de troyanos, a la detección de ataques de tipo *buffer overflows*, etcétera.

Cada regla puede ser dividida en dos partes. En primer lugar, tenemos la cabecera de la regla, en la que indicamos la acción asociada a esta regla en caso de cumplirse (generación de un fichero de registro o generación de una alerta), el tipo de paquete (TCP, UDP, ICMP, etcétera), la dirección de origen y destino del paquete, etc. En segundo lugar, tenemos el campo `option` de la regla, donde encontraremos la información que debe contener el paquete (en la parte de datos, por ejemplo) para que se active la acción asociada a la regla.

Para indicar el contenido de estas dos partes, Snort posee una sintaxis propia que permite especificar hasta el más mínimo detalle las condiciones que han de cumplirse para que un paquete sea asociado a las acciones indicadas por cada una de la reglas. A modo de ejemplo, el formato general de una regla Snort es el siguiente (el uso de los símbolos '[' y ']' indica que dichos atributos en la regla son opcionales):

```
<Acción><Protocolo>
<IP_origen><Puerto_origen>-><IP_destino><Puerto_destino>
[(<opcion_1>; ...; <opcion_k>)]
```

Ejemplo

La siguiente regla indicaría a Snort generar alertas de tipo "*FTP root access attempt*" en cuanto vea tráfico de tipo TCP que contenga la cadena "USER root", procedente de una dirección IP con máscara de red 10.0.0.1/24, con cualquier puerto de origen y que vaya destinado a alguna dirección IP dentro de la red 192.168.1.0/24:

```
alert tcp
any any ->192.168.1.0/24 21
(content: 'USER root'; msg: 'FTP root user access attempt';)
```

Un ejemplo más completo de regla Snort es el siguiente:

```
alert tcp $EXTERNAL_NET any ->$HOME_NET 21 (msg:"FTP EXPLOIT STAT *
dos attempt"; flow:to_server,established; content:"STAT "; nocase;
content:"*"; reference:bugtraq,4482; classtype:attempted-dos;
sid:1777; rev:1;)
```

La acción definida en una regla de Snort puede ser elegida de entre cinco acciones básicas:

- **Alert**. Generación de una alerta, conteniendo además la información de registro (*log*) correspondiente al paquete que active la regla.
- **Log**. Generación únicamente de la información de registro asociada con el contenido del paquete asociado a la activación de la regla.
- **Pass**. Deja pasar el paquete asociado a la regla, sin necesidad de reportar (*log*) o alertar al respecto.
- **Activate**. Generación de una alerta, junto con la activación de una regla dinámica (podéis ver la acción *Dynamic*).
- **Dynamic**. Regla que permanece inactiva, y que es activada a través de la acción *activate* para, por ejemplo, activar la generación de registros asociados a los paquetes asociados a la regla inicial, con el objetivo de obtener información adicional respecto al tráfico posterior al paquete que activó la regla (y con una duración de tiempo determinada).

Acciones básicas

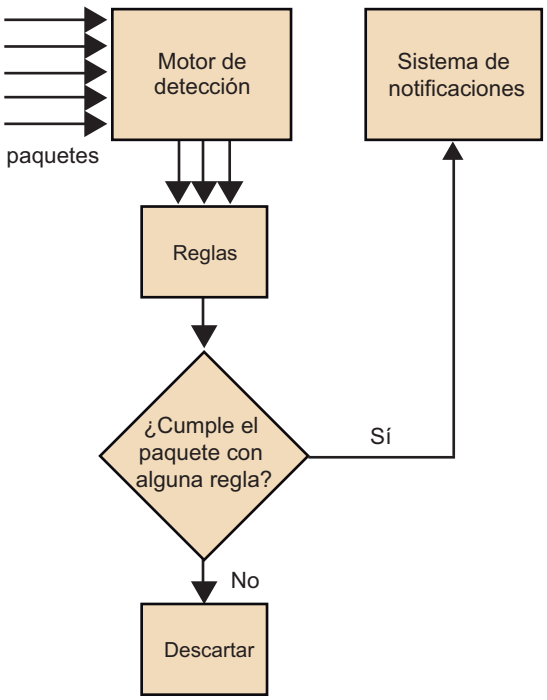
Versiones extendidas de Snort pueden proporcionar acciones adicionales, por ejemplo, acciones reactivas para bloquear el tráfico detectado. Podéis ver, por ejemplo, las acciones adicionales de Snort Inline, una versión reactiva de Snort disponible en la página web

http://snortinline.sourceforge.net. El proyecto no es mantenido actualmente, pero es posible acceder a las últimas versiones del proyecto, que fueron liberadas hasta mediados de 2008 y que ofrecían las acciones adicionales (*drop*, *sdrop* y *reject*). Estas acciones permiten transformar Snort en un IDS reactivo (o IPS, del inglés *intrusion prevention system*).

2.6. Motor de detección

El motor de detección contiene los algoritmos de tratamiento necesarios para concluir el proceso de detección. A partir de la información proporcionada por el preprocesador y sus *plug-ins* asociados, el motor de detección contrastará estos datos contra la base de reglas definida por el operador. Si alguna de las reglas coincide con la información obtenida, el motor de detección se encargará de avisar al sistema de notificaciones indicando la regla que ha saltado. La figura 6 muestra un sencillo esquema sobre el comportamiento general del motor de detección de Snort.

Figura 6. Motor de detección de Snort



De todos los elementos que hemos visto, el motor de detección y la sintaxis utilizada por las reglas de detección son las partes más complicadas de comprender a la hora de estudiar el comportamiento de Snort. Aun así, una vez que nos pongamos a trabajar con Snort y hayamos aprendido mínimamente la sintaxis utilizada, es bastante sencillo llegar a personalizar y ajustar el comportamiento de la funcionalidad de detección de Snort. Además, los conjuntos de reglas pueden ser activados o desactivados con facilidad para poder así definir el comportamiento de detección deseado según el tipo de red donde Snort va a ser configurado.

Nota

Al final de este módulo encontraréis un conjunto de actividades para profundizar vuestros conocimientos sobre la configuración de Snort.

2.7. Sistema de notificaciones

Una vez que la información capturada por el decodificador de paquetes de Snort es analizada por el motor de detección, los resultados deben ser reportados de alguna manera. Mediante este componente será posible realizar esta función, pudiendo generar los resultados en distintos formatos y hacia distintos equipos.

Cuando una alerta es lanzada por el motor de detección, el resultado puede suponer la generación de un fichero de registro (*log*), el envío de la alerta a través de la red mediante un mensaje SNMP o incluso el almacenamiento de las informaciones asociadas a la alarma de forma estructurada por algún sistema gestor de base de datos, como, por ejemplo, MySQL o PostgreSQL.

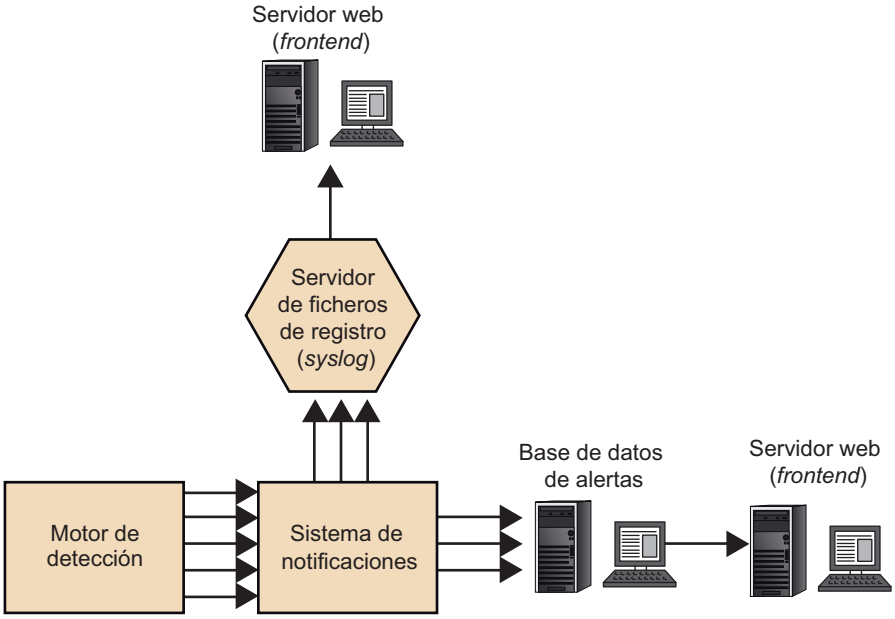
Además, es posible la utilización de herramientas alternativas (desarrolladas por terceras partes) que faciliten la visualización y el tratamiento de la información reportada por Snort. Muchas de estas herramientas están disponibles para poder ser descargadas libremente de Internet.

Como en el caso del motor de detección y el preprocesador, el sistema de notificaciones de Snort también utiliza un esquema de *plug-ins* para el tratamiento de la información. La figura 7 muestra un esquema sobre cómo funciona este sistema de notificaciones a través de *plug-ins*.

Alertas de Snort

Si tratáis de poner en marcha Snort en una red congestionada y con un número razonable de firmas de detección activadas, la cantidad de alertas lanzadas puede alcanzar la centena en poco tiempo. Este enorme volumen de información no será fácil de tratar con la utilización de un simple navegador de ficheros de registro.

Figura 7. Sistema de notificaciones de Snort, a base de *plug-ins*



El propósito de instalar Snort en una red no es únicamente obtener información (intentos de intrusión), sino analizar tal información y poder tomar las acciones necesarias en función de los datos obtenidos. Si el número de reglas activadas es elevado, y el tráfico de la red aumenta con facilidad, no será del

todo sencillo analizar la información reportada por Snort a no ser que utilicemos herramientas de visualización adecuadas.

Por otro lado, la faceta interesante de un sistema de detección como Snort no es simplemente registrar eventos, sino ser capaz de reaccionar a los intentos de intrusión en un periodo de tiempo razonablemente corto. Así pues, será necesario el uso de segundas aplicaciones que ayuden a consolidar y analizar la información reportada por Snort con el objetivo de alertar a los administradores de la red de los intentos de intrusión analizados.

Actualmente existe un gran número de utilidades para trabajar con las informaciones generadas por Snort. Algunas de estas herramientas son mantenidas por la propia comunidad de desarrolladores de Snort, aunque también podemos encontrar aplicaciones desarrolladas por terceras partes o aplicaciones comerciales.

Ved también

Algunas de las soluciones existentes para reportar la información recogida por Snort de manera gráfica se estudian en el subapartado 3.6 de este módulo.

3. Gestión de eventos, alertas e incidentes

Los sistemas de detección de intrusos tradicionales, como, por ejemplo, Snort, permiten la detección de acciones elementales que suelen corresponder a etapas previas a una intrusión o acciones aisladas que forman parte de ataques más complejos. En realidad, para poder anticiparse y detener una intrusión, será necesario la utilización de elementos adicionales para completar y concluir el proceso de detección realizado por los IDS tradicionales.

En este apartado trataremos sobre dichos complementos, resumidos a partir del concepto de SIEM*. Al igual que los IDS, los SIEM son componentes imprescindibles para garantizar la seguridad de una red informática. Aunque existen desde hace más de dos décadas, ha sido la expansión de Internet en corporaciones e instituciones lo que ha propiciado las recientes evoluciones y mejoras que resumimos a continuación:

* Del inglés, *security information and event management system.*

Ataques distribuidos

Ataques que no pueden ser identificados buscando patrones de manera aislada, sino que han de ser detectados a partir de la combinación de múltiples indicios encontrados en diferentes equipos de la red.

- **Recogida y normalización de eventos**. Un SIEM debe ser capaz de gestionar la recogida de los eventos provenientes de fuentes extremadamente heterogéneas. Es, por lo tanto, necesario realizar un proceso de normalización de datos, no solo a nivel sintáctico, sino también a nivel semántico. A nivel sintáctico, la existencia de formatos y estándares, como el formato IDMEF** o el formato IODEF***, puede ayudar en las tareas de normalización sintáctica de alertas provenientes de sondas de detección heterogéneas. En relación con la normalización semántica, la representación ontológica de las alertas lidera la mayoría de los trabajos actuales en esta materia.

** Del inglés, *intrusion detection message exchange format.*
*** Del inglés, *incident object description and exchange format.*

- **Consolidación de las funciones de supervisión de herramientas de detección**. La siguiente función vital atribuida a un SIEM es la consolidación de alertas de bajo nivel producidas por los componentes de seguridad de una red, entre los que destacan sistemas de cortafuegos, IDS, antivirus y sistemas de detección de vulnerabilidades. Como hemos visto para el caso de los IDS, las herramientas de seguridad en red son vulnerables a problemas ligados con falsos positivos y falsos negativos. Por ello, lo que se espera de la utilización de un SIEM es proceder a procesos de fusión, agregación y correlación de alertas procedentes de los equipos anteriores y reducir la tasa de falsos positivos, así como mejorar el diagnóstico para reducir también los falsos negativos.

- **Activación de la reacción**. Por lo general, la mayoría de los IDS suelen ser configurados como mecanismos puramente pasivos. Sin embargo, la

mayoría de las soluciones existentes hoy en día proporcionan la tecnología necesaria para convertirlos en soluciones activas o semiactivas (a la espera de confirmación de un operador antes de activar el proceso de reacción), con el objetivo de poder reaccionar y neutralizar aquellas actividades o acciones detectadas. Esta funcionalidad, a veces desconocida, debe ser analizada y activada con precaución, ya que el proceso de detección aun mejorado gracias a las tareas de normalización y correlación de un SIEM no es infalible. Por lo tanto, una configuración automática para reaccionar ante el proceso de detección, en el caso de falsos positivos, podría llevar a situaciones indeseadas, tales como bloqueo de usuarios legítimos de un sistema o desactivación de servicios por error. La tendencia actual es dejar la activación de esta funcionalidad a partir de la interfaz de un SIEM, dado la problemática de falsos positivos durante el proceso de detección.

Detallaremos a continuación algunas de las tareas necesarias para poder proceder a las funcionalidades que se esperan de un SIEM.

3.1. Configuración de las sondas de detección

Aparte de información proveniente de un IDS, los datos recogidos por un SIEM pueden provenir de cualquier otro componente con capacidad de creación de ficheros de registro, tales como:

- **Sistemas de cortafuegos** (o cualquier otro tipo de componente para el filtrado de paquetes a través de listas de control de acceso a nivel de red). Aunque a menudo hemos visto estos componentes como sistemas de protección, cuyo objetivo es bloquear tráfico considerado como peligroso para el sistema, estos componentes pueden ser configurados para reportar la información observada a través de sus interfaces de red. Sus ficheros de auditoría son, por lo tanto, de gran interés para complementar el proceso de agregación y correlación de eventos del SIEM.

- **Servidores de correo electrónico** (basados en protocolos tales como SMTP, POP o IMAP). De nuevo, los ficheros de auditoría generados por estos servidores ofrecerán una gran cantidad de información para caracterizar y descubrir actividades maliciosas, tales como la propagación de gusanos, ataques de tipo *zero-day*, o tráfico de control asociado a actividades de botnets.

- **Servidores de gestión de tráfico** (basados, por ejemplo, en el protocolo SNMP). La información reportada por estos componentes contendrá igualmente información asociada a acciones relacionadas con la violación de políticas de seguridad interna de la organización donde estén instalados.

Otra información que se debe tener presente durante la configuración de las sondas de detección de un SIEM serán los datos, los eventos y las alarmas proporcionadas por un IPS*. Un IPS se basa principalmente en la investigación de vulnerabilidades de los equipos y sistemas. La mayoría de las soluciones de tipo IPS combinan en realidad técnicas de detección de intrusos junto con mecanismos de control de acceso tradicionales. La frontera entre IDS e IPS es actualmente difícil de definir y puede ser complementada con muchas otras soluciones preventivas, tales como sistemas de detección de vulnerabilidades** y sistemas antivirus.

*** Del inglés, *intrusion prevention system*.**

Ved también

Los mecanismos de control de acceso tradicionales se estudian en el módulo "Sistemas de cortafuegos" de esta asignatura.

****En inglés, *vulnerability detection systems* o VDS.**

Los VDS tratan de analizar la configuración de sistemas desplegados en red con el objetivo de descubrir partes malconfiguradas, que potencialmente presentan vulnerabilidades que podrían ser atacadas. Dichos sistemas comprenden también detección de defectos software (es decir, errores de programación o *bugs*), errores de concepción en la configuración topológica de una red, errores hardware, etcétera. Por otro lado, los sistemas antivirus están diseñado para proteger estaciones de trabajo y servidores contra malware conocido. La mayoría de estos sistemas utilizarán una base de datos de firmas de antivirus que identifica al malware conocido. Desde el punto de vista de prevención, es de esperar que tanto un VDS como un antivirus sea capaz de corregir las vulnerabilidades, así como desinfectar los equipos víctimas del malware. Así pues, la instalación y combinación de IDS con IPS, VDS y sistemas antivirus tiene por objetivo final detectar y reaccionar ante el concepto general de intrusión.

3.2. Políticas de recogida de información

Puesto que el objetivo de un SIEM es poder ofrecer a los operarios del sistema capacidades de gestión centralizada, es habitual que la configuración de los componentes de recogida asociados al SIEM (tanto IDS, como los componentes de ejemplo anteriores) sea realizada a través de la propia interfaz de usuario del SIEM. Para ello, se suele utilizar una política de recogida de información global, administrada por el SIEM, y que podrá ser posteriormente refinada para la configuración local de cada uno de los equipos asociados al SIEM (IDS, cortafuegos, servidores de correo, IPS, VDS, etcétera).

Esta política, o sus subpolíticas asociadas, suele ser definida a partir del uso de las reglas basadas en el uso de expresiones regulares, búsqueda de patrones en tráfico, señalización de protocolos, etc. También es posible que los componentes incorporen la posibilidad de tratar una recogida de eventos a través de configuraciones basada en el reconocimiento de actividades anómalas. La política deberá, por lo tanto, ofrecer la sintaxis y semántica necesaria para poder tratar el uso de modelos estadísticos, recogida a través de minería de datos, sistemas expertos, redes bayesianas, etcétera. Por último, la política de recogida de eventos e información de registro del SIEM deberá permitir también un conjunto de acciones individuales (por ejemplo, una regla para el recono-

cimiento de un evento por parte de un componente específico) o generales (por ejemplo, una regla para el reconocimiento de acciones que pueden afectar a múltiples componentes de detección, sin especificar el componente en concreto que deberá realizar la tarea de recogida).

Otro de los criterios técnicos que deberá ser considerado en la política de recogida de eventos de un SIEM será determinar con precisión la localización de los componentes de recogida que controlar en el sistema. Como veremos más adelante, este aspecto determinará la cobertura y los módulos de visión global que el sistema incorporará para garantizar que el posterior proceso de agregación y correlación de información garantice una gestión de calidad. Por ello, será también necesario especificar dentro de las políticas de recogida tanto la topología física como la descripción lógica del sistema que supervisar por el SIEM. Es importante poder tratar y estructurar dicho sistema en términos de subredes de cara a poder identificar aquellas partes del sistema que presenten mayor necesidad de vigilancia, de modo que las funciones de correlación puedan reconocer tarea asociadas a incidentes específicos por cada subred del sistema. Por ejemplo, si el sistema contiene zonas de tipo DMZ (zonas desmilitarizadas), deberá ser posible crear reglas específicas para vigilar más de cerca las acciones que se desarrollen en esa parte de la red. El uso de máscaras de subred (especificado en términos de CIDR*, por ejemplo) suele ser habitual en las políticas de recogida de ficheros de registro de la mayoría de los SIEM.

*** Sigla de *classless inter-domain routing*.**

3.3. Normalización de la información recogida

Una vez recogida la información, y antes de pasar a los subsiguientes procesos para la detección de escenarios de ataque, el SIEM debe garantizar que es posible poner en correspondencia todos aquellos datos que provengan de la detección de un mismo evento (a priori, eventos maliciosos, sospechosos o anómalos). Así pues, deberá proveer de un conjunto de procesos de normalización para preprocesar los datos recogidos y anticiparse a aquellos problemas que podrían entorpecer la puesta en correspondencia de datos relativos a eventos derivados de un mismo flujo de tráfico de red (tales como origen del tráfico, destino, puertos, etc.) y que corresponden a las tareas de supervisión de distintas sondas de monitorización que fueron instaladas en distintos lugares del sistema (ya sean sondas desplegadas en un mismo dominio o en distintos).

El proceso de normalización ha de garantizar, por ejemplo, que datos asociados con el tráfico de red presenten el mismo formato respecto a cómo clasificar el origen del tráfico, así como metadatos asociados al instante de tiempo en el que el evento fue detectado, protocolos y servicios asociados, direcciones de origen y de destino, contenido de los paquetes asociados al tráfico, etcétera. Puesto que la naturaleza de las sondas es potencialmente heterogénea,

el proceso de normalización deberá asegurar, bien con formatos propietarios del SIEM o bien con esfuerzos provenientes de estándares existentes, que no existan problemas de interoperabilidad que limiten la expresividad de las informaciones que deberán ser agregadas por el módulo de correlación del SIEM.

Ejemplos de esfuerzos

Algunos ejemplos de esfuerzos al respecto del problema de normalización son los siguientes:

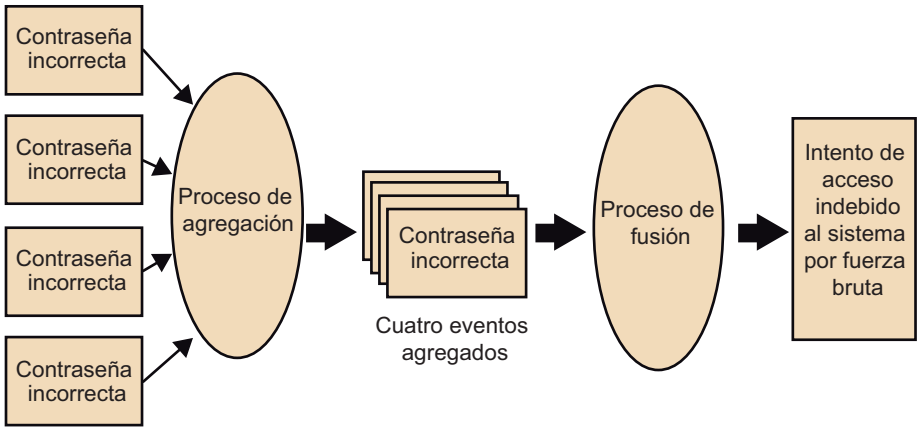
- CIDF (*common intrusion detection framework*)
- IDWG (*intrusion detection working group*)
- IDMEF (*intrusion detection message exchange format*)
- INCH (*extended incident handling*)
- FINE (*format for incident report exchange*)
- IDIP (*intrusion detection and isolation protocol*)
- OSEC (*open security evaluation criteria*)
- IODEF (*incident object description and exchange format*)

Cada uno de los ejemplos anteriores ha tratado de definir lenguajes comunes para especificar la descripción de los eventos y las actividades asociadas a los eventos que intercambiar por componentes de seguridad. Aparte de lenguajes (tanto sintácticos como semánticos) para garantizar homogeneidad en el intercambio de datos, es también prioritario permitir la descripción de un proceso común que especifique el protocolo preciso para el intercambio de datos entre las distintas sondas configuradas en un SIEM. De los ejemplos enumerados anteriormente, dos de los formatos y procedimientos que cuentan con el apoyo de grupos de trabajo del Internet Engineering Task Force (IETF) son IDMEF e IODEF. IODEF es el formato más reciente y proporciona compatibilidad hacia formatos anteriores (por ejemplo, compatibilidad con informaciones normalizadas en el formato IDMEF), y se espera que en el futuro sea el formato usado por SIEM y componentes de seguridad en general para el intercambio de información de incidencias.

3.4. Agregación y fusión de la información

Las funciones de agregación y fusión de datos son utilizadas para reducir de modo inteligente grandes volúmenes de datos que, probablemente, pueden contener eventos redundantes (repeticiones e informaciones congruentes). Ambas funciones deben ser aplicadas antes de pasar a poner en correspondencia eventos detectados por distintos componentes del sistema. En primer lugar, el proceso de agregación deberá encargarse de agrupar aquellos datos que se hayan producido a partir de la detección de un mismo evento, y reportado por una misma sonda o por sondas diferentes. Una vez producida la agrupación de dichas informaciones, se producirá el proceso de fusión de la información con el objetivo de resumir y ofrecer un único dato que caracterice al evento detectado. La figura 8 muestra con un sencillo ejemplo la diferencia entre las funciones de agregación y fusión.

Figura 8. Diferencia entre agregación y fusión



Durante el proceso de agregación, las informaciones correspondientes a eventos detectados serán agrupadas en sesiones, tratando de unificar aquellos datos de un mismo evento que puedan ser utilizados más adelante durante el proceso de fusión, tales como dirección de origen, dirección de destino, puertos, protocolos, etcétera. De este modo, los diferentes datos asociados con un ataque hacia un elemento específico del sistema se agruparán en una sola sesión y un único identificador. El resto de los datos reportados por otras sondas del sistema se enlazarán con la misma referencia, de modo que durante el proceso final de fusión sea posible la generación de una alerta por identificador. Dicha alerta contendrá toda la información observada por las distintas sondas configuradas por el SIEM. Las alertas generadas a partir del proceso de fusión contendrán, así pues, una síntesis de todo el conocimiento del SIEM sobre cada uno de los ataques básicos que fueron observados por el sistema de supervisión. Como resultado, se reduce la cantidad de datos necesaria para almacenar en el sistema sin que se produzca pérdida información. Una vez fusionada toda la información reportada por el sistema, las alertas serán comunicadas al último componente del SIEM, encargado de gestionar y poner en correspondencia (correlacionar) el flujo de alertas.

3.5. Correlación de alertas

De manera general, podemos definir el proceso de correlación de alertas como la interpretación conceptual de múltiples alertas con el objetivo de proporcionar una mejora semántica y de reducir la cantidad global de alarmas en un sistema de detección de intrusos.

Lectura recomendada

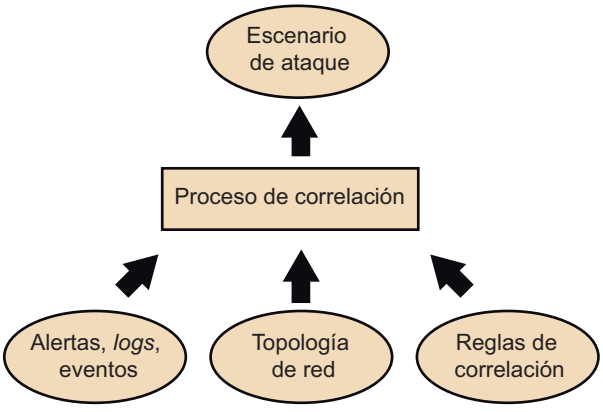
C. Kruegel; F. Valeur; G. Vigna (2004). *Intrusion Detection and Correlation: Challenges and Solutions*. 1.ª edición. Springer.

La correlación de alertas es considerada, así pues, como una de las claves en la evolución de los sistemas de detección de intrusos, ya que trata de solucionar los inconvenientes más destacados de estos sistemas, es decir, el exceso de alertas, su mejora semántica y la disminución de los falsos positivos y negati-

vos. Los trabajos relacionados con la correlación de alertas en el campo de la detección de intrusos son relativamente recientes. La mayor parte de ellos tratan sobre observaciones y experimentación realizados sobre sistemas actuales. La parte teórica en este campo está, así pues, aún en proceso de consolidación.

La mayoría de los SIEM comerciales carecen aún de auténticas funcionalidades para la realización de una correlación de alertas completa. Aunque la mayor parte de las soluciones existentes hablan de servicios de correlación, la gran mayoría se limita a almacenar enlaces lógicos entre alertas almacenadas en una misma base de datos relacional para posteriormente poder ser consultada por el administrador responsable a través de una consola de control. Por el contrario, dentro del campo de investigación académica sobre sistemas de detección de intrusos existe un gran número de propuestas para la inclusión de técnicas de correlación de alertas en sistemas de nueva generación. La mayor parte de estas propuestas explotan esencialmente la información contenida en el interior de las alertas tratando, adicionalmente, de hacer referencias explícitas a conocimientos anexos, necesarios para el proceso final de la puesta en correspondencia de todos los eventos observados en un mismo sistema. Por lo tanto, la correlación en estos sistemas no se limita únicamente a la información contenida en el interior de alertas generadas por las sondas del SIEM, sino que hacen uso de un conocimiento a priori del estado de vigilancia de los sistemas, sobre los ataques que se pueden realizar e incluso sobre la topología del sistema, y reglas de correlación generadas por los operarios e interpretadas por sistemas expertos. La figura 9 resume, de modo general, este tipo de sistemas de correlación.

Figura 9. Función de correlación



Como vemos en la figura anterior, la primera entrada en el proceso de correlación corresponde a los eventos ya agregados y fusionados por los módulos inferiores del SIEM. El objetivo del proceso de correlación es precisamente poner en correspondencia estas agrupaciones de eventos y lograr reconstruir escenarios de ataque a los que podrían pertenecer las acciones observadas. Para ello, será necesario combinar las alertas recibidas con informaciones físicas y lógicas del sistema (por ejemplo, la topología del sistema y la estructuración

del direccionamiento IP de los equipos) y el conocimiento predefinido de actividades maliciosas. Este último, representado en la figura 9 en forma de reglas de correlación.

La propiedades físicas y lógicas de la red deberán ser especificadas en una base de datos topológica, bien de manera manual (por los administradores del sistema) o bien mediante el uso de servicios automáticos de descubrimiento y creación asistida de topologías de red. Estos últimos deberán encargarse de la creación de mapas topológicos con las configuraciones de los dispositivos y las políticas de seguridad asociadas al sistema de información protegido.

La puesta en conocimiento de las actividades maliciosas se debe configurar en el sistema mediante reglas de correlación. Cada regla de correlación se definirá para caracterizar conjuntos de acciones que corresponden a un mismo escenario de intrusión, y especificará las condiciones necesarias para llevar a cabo una acción, y las consecuencias en el sistema tras la ejecución de cada acción. Al igual que ocurre con las reglas de detección de un IDS basado en usos indebidos, estas reglas de correlación serán el elemento básico para garantizar la detección de escenarios de ataque compuestos por las acciones detectadas en el sistema. Así pues, para garantizar la eficacia del proceso de correlación, será de vital importancia garantizar una correcta configuración de las reglas de correlación pertinentes para cada sistema, al igual que su correcta actualización y puesta en funcionamiento. La eliminación o corrupción de una sola regla que contenga información sobre múltiples incidentes podrá ocasionar la pérdida de detección de multitud de escenarios. La configuración de esta parte del SIEM será, por lo tanto, muy propensa a errores y requiere un alto conocimiento por parte de los operadores encargados de configurar el sistema.

La detección de un incidente o escenario de intrusión se deriva durante el proceso de correlación a partir de las series de eventos indicados a través del conjunto de reglas de correlación preconfiguradas en el sistema. A partir de un conjunto pequeño de reglas de correlación, es posible la definición de multitud de escenarios de intrusión. Así pues, el objetivo del proceso de correlación es reducir de este modo el exceso de información que deberá ser gestionada por el operador del sistema. En lugar de solicitar al administrador el análisis de miles de eventos, el proceso de correlación proporciona la generación de informes de incidentes. Cada informe contendrá la representación de aquellos escenarios que, con alta probabilidad, podrían haberse desarrollado en el sistema a partir de los eventos (es decir, acciones primitivas) detectados por las sondas del SIEM.

Con el mismo objetivo, el proceso de correlación puede ser configurado para reducir también el número de falsas alarmas que se analizan (falsos positivos). Con este fin, el sistema puede ser configurado para ejecutar acciones de verificación internas desencadenadas tras la generación de cada escenario de

intrusión y verificar la certeza de que dicho incidente haya o no tenido lugar en el sistema. De esa manera, aquellos incidentes que con alta probabilidad se puedan descartar serán eliminados del informe final que el operador deberá analizar. Esta verificación interna puede limitarse, por ejemplo, a la ejecución de un análisis de vulnerabilidades del sistema, el cual sirva para decidir si un incidente concreto puede ser descartado, si las vulnerabilidades asociadas no están presentes en el sistema representado en la base de datos topológica del sistema. Es decir, un incidente que suponga la explotación de vulnerabilidades o servicios no desplegados en el sistema podrá ser descartado con alta probabilidad, y evitar así sobrecargar las capacidades de análisis del operador del sistema. Adicionalmente, dichos incidentes podrán ser reportados como falsos positivos en lugar de alarmas o escenarios de intrusión. Todo esto facilitará la ordenación de los mecanismos de respuesta en el sistema, así como la optimización de las contramedidas que desplegar en el sistema sobre los incidentes reales que hayan sido detectados.

3.6. Generación de informes e interfaz gráfica de control

El elemento más característico de un SIEM es el componente gráfico ofrecido para gestionar la generación de informes y ofrecer el control de mandos al usuario final. En este aspecto, multitud de soluciones existen en la actualidad, tanto gratuitas como comerciales. Comentaremos en este subapartado algunas herramientas de ejemplo, la mayoría de ellas encargadas de ofrecer una interfaz gráfica para soluciones de seguridad tradicionales, como es el caso del NIDS Snort.

Ved también

Snort se estudia en el apartado 2 de este módulo.

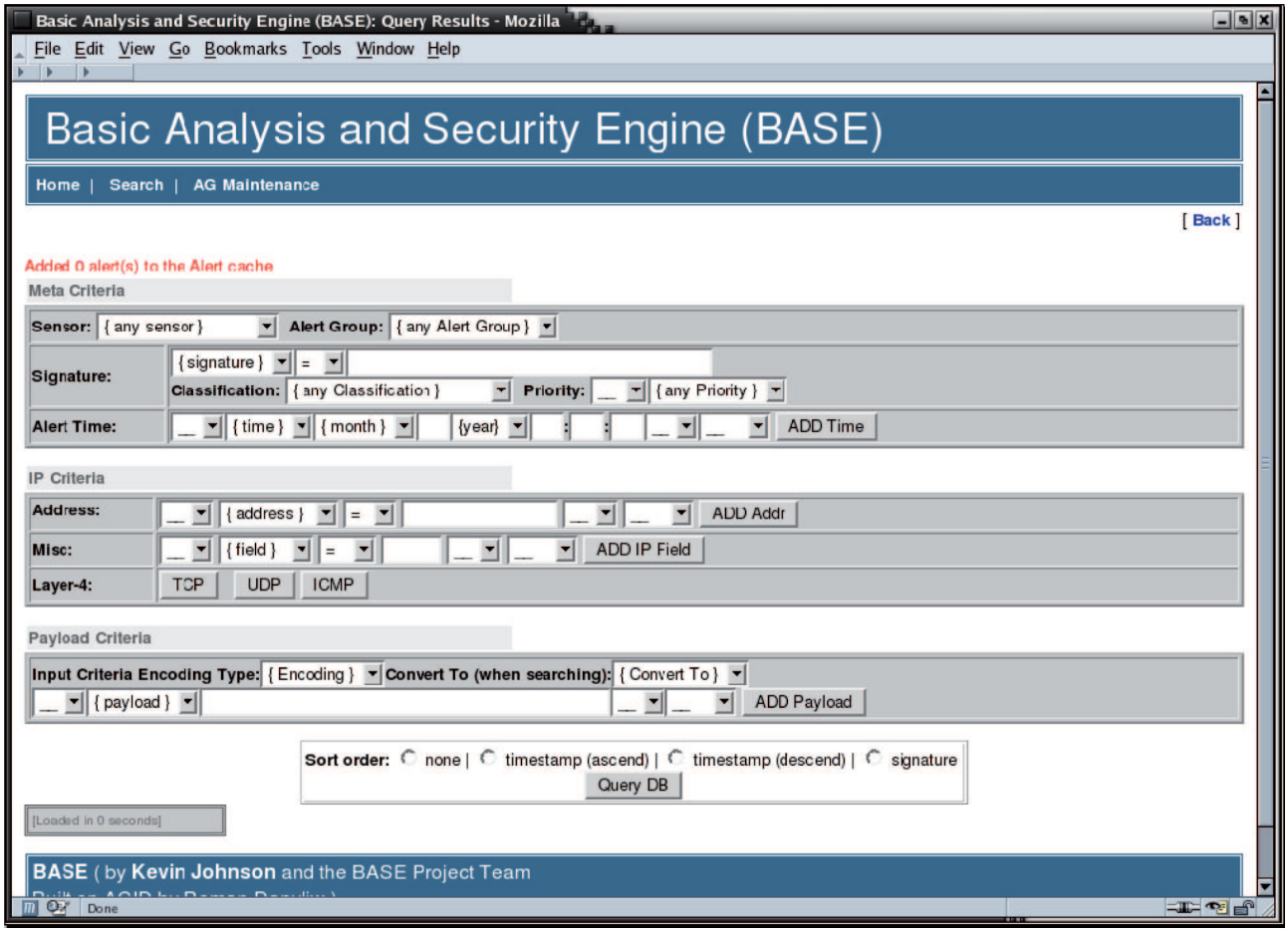
3.6.1. BASE

Basic Analysis and Security Engine (BASE)* proporciona una completa interfaz web para gestionar alertas y reportes de actividad maliciosa basado en sondas Snort. Se trata de un proyecto gratuito y de código abierto, basado a su vez en un proyecto anterior (ACID, *Analysis Console for Intrusion Database*) desarrollado dentro del proyecto *AIRCERT* del centro de coordinación CERT de *Carnegie Mellon*. BASE puede ser utilizado también para configurar de forma gráfica distintas sondas de detección basadas en Snort, sobre una única red, o distintas redes. La figura 10 muestra una captura de pantalla de la interfaz principal de BASE para la búsqueda de alertas generadas a través de las distintas sondas.

*** http://base.secureideas.net**

El conjunto de herramientas asociadas con BASE proporciona también todo un conjunto de *scripts* escritos en lenguaje PHP para gestionar la base de datos donde las sondas almacenarán las alertas. La interfaz y la base de datos de BASE pueden ser también utilizadas para almacenar información independiente a los datos reportados por sondas Snort. Por ejemplo, es posible combinar en la misma base de datos información reportada a través de cortafuegos basa-

Figura 10. Interfaz para la búsqueda de alertas a través de BASE



dos en netfilter, o mensajes de control de acceso generados por productos de seguridad Cisco. Otras de las características destacables son las siguientes:

- Decodificación y visualización de paquetes TCP/IP asociados (reportados) a las alertas o informes reportados por las sondas de detección.
- Creación de diagramas y estadísticas basadas en fechas, horas, firmas, protocolo, etcétera.
- Interfaz para la consolidación de búsquedas y creación de vistas con múltiples consultas. Los resultados de estas acciones se devolverán de manera estructurada con información para facilitar la comprensión de las alertas lanzadas por las sondas configuradas a través de BASE. En esta información se resaltarán, entre otras informaciones, las direcciones de origen/destino, los puertos de origen/destino, el estado de las banderas TCP/IP (*TCP/IP flags*), etc., asociados con los paquetes detectados por las sondas.
- Gestión de incidentes. Se proporciona la posibilidad de crear grupos de alertas lógicas donde almacenar la información de los incidentes que sean necesarios destacar. También existen opciones de manejo de múltiples incidentes, permitiendo descartar aquellas alertas consideradas como falsos positivos y la exportación final a través de correo electrónico y/o almacenamiento de las alertas encontradas en la base de datos.

Otros productos equivalentes a BASE que proporcionan una manera similar de representar gráficamente las alertas y los incidentes reportados por herramientas tipo Snort son *RazorBack**, *Snorby*** y *Engage Security IDScenter****.

* http://www.intersectalliance.com/projects/RazorBack

** http://www.snorby.org

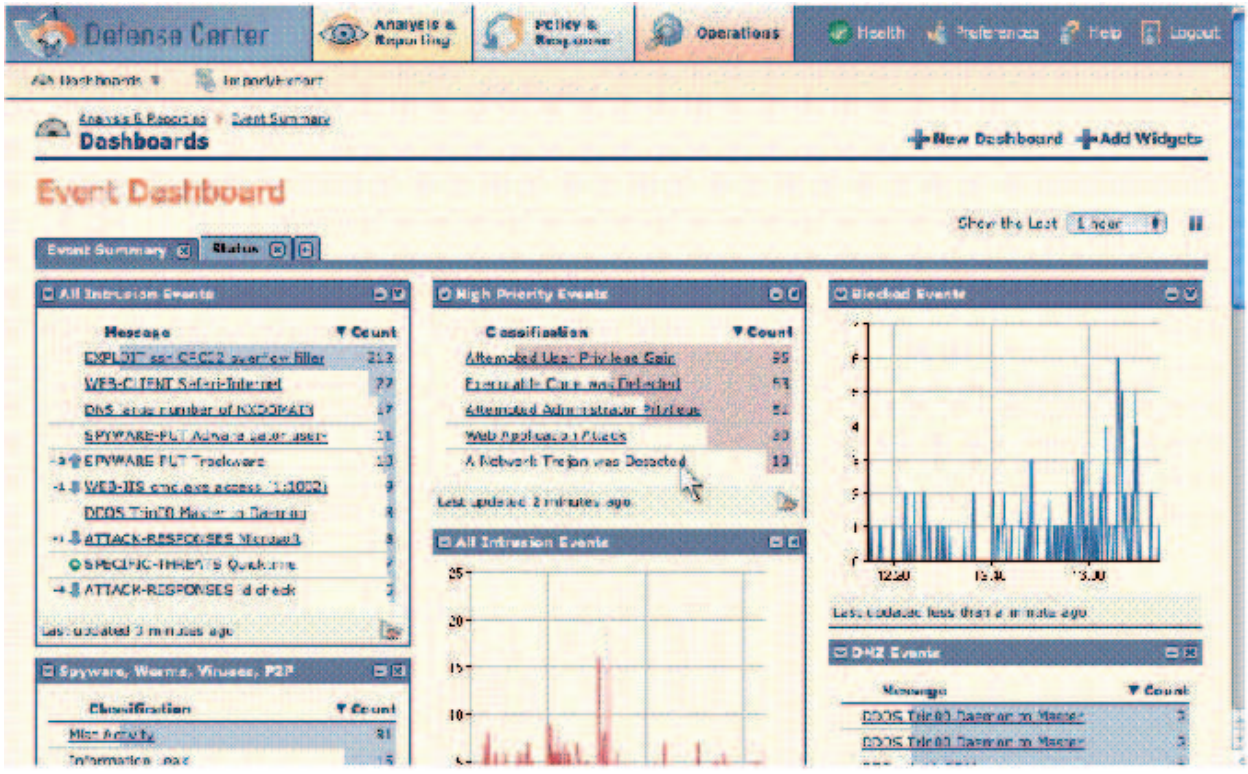
*** http://www.engagesecurity.com/products/idscenter

3.6.2. Sourcefire 3D System

El sistema Sourcefire 3D es una familia de productos comerciales que pueden ser combinados para configurar, gestionar y visualizar el sistema de notificaciones de la versión comercial del IDS Snort (Sourcefire). Otras funcionalidades consisten en la gestión de firmas criptográficas, necesarias para la construcción y monitorización de redes privadas virtuales (VPN) a través de Sourcefire. Entre los elementos de Sourcefire 3D, destacamos los siguientes:

- Sensores 3D IPS, que dotan de capacidades reactivas (tipo IPS o *intrusion prevention system*) para configurar reglas de configuración que permitan contrarrestar los ataques detectados a partir de las sondas de Sourcefire.
- Centro de defensa (del inglés, *defense center*), que proporciona una utilidad gráfica para gestionar las funciones de agregación de alertas, gestión de informes y tratamiento de políticas de seguridad distribuidas. La figura 11 muestra una captura de pantalla del tablero de controles asociado con el DC de Sourcefire.

Figura 11. Tablero de control de la familia de productos Sourcefire 3D



- *Real-time Network Awareness* (RNA), que ofrece un control de la supervisión pasiva de Sourcefire, así como gestión de inventario en tiempo real de los componentes asociados.

- *Real-time User Awareness* (RUA), que ofrece gestión para las funciones de correlación de Sourcefire, tales como clasificación a través de direcciones IP, tipo de tráfico, protocolos, etcétera.

El sistema Sourcefire 3D DC proporciona medidas de reacción, de manera que un proceso activo o semiactivo (a la espera de confirmación de un operador) de reacción puede ser inicializado como respuesta a las actividades detectadas por las sondas de detección del sistema. El sistema de respuesta puede ser inicializado a partir de componentes de tipo IPS configurados en el sistema, así como a partir de eventos del subsistema RNA. Las respuestas pueden ser configuradas para ir desde una simple generación de sesiones SNMP hasta la activación de scripts y tareas programadas a partir de la API proporcionada por los fabricantes de Sourcefire 3D. Dicha API contiene una serie de módulos reactivos que se pueden combinar mediante terceras herramientas, incluyendo la posibilidad de reacción basada en cambio de tablas de encaminamiento a través de encaminadores Cisco, así como la creación de reglas de sistemas de cortafuegos (Cisco, Check Point, etc.) o escáneres de vulnerabilidades tipo NMAP y Nessus.

3.6.3. OSSIM

Por último, concluimos este subapartado con OSSIM*, un completo SIEM de código abierto. Además de poder combinarse con Snort, OSSIM ofrece multitud de funcionalidades para almacenar y proporcionar información gráfica para una gran familia de aplicaciones de seguridad en red, tales como:

* **Del inglés,** ***Open Source Security Information Management Tool.***

Enlace de interés

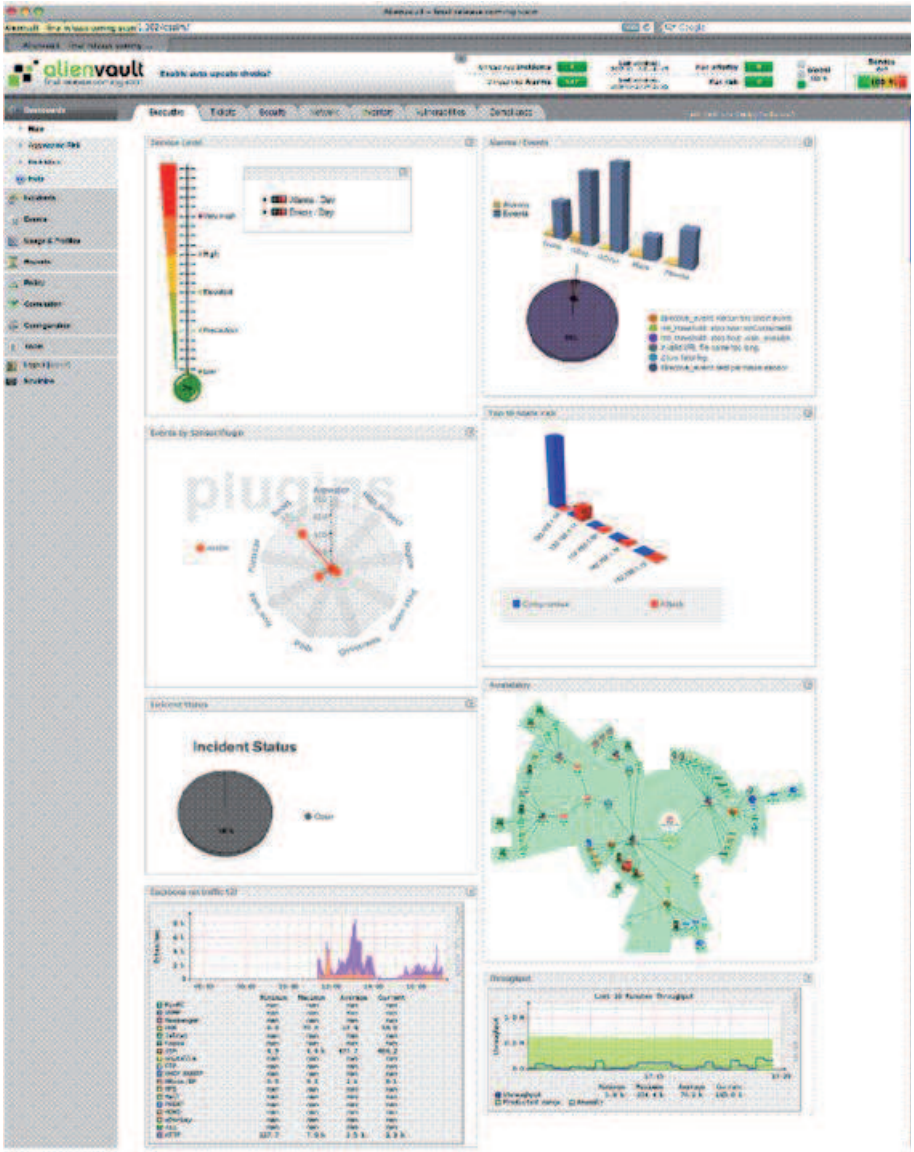
OSSIM está disponible en el sitio web http://www.alienvault.com/.

- **Nessus:** escaneo de vulnerabilidades.
- **P0f:** detección remota de sistemas operativos.
- **Nagios:** monitorización de redes.
- **Pads:** sistema de detección basado en anomalías.
- **Osiris:** sistema de detección de intrusos basado en equipo.
- **OSSEC:** sistema para la detección de ataques de integridad a nivel de sistema operativo, sistema de ficheros, instalación de rootkits, etcétera.

La mayor parte de las aplicaciones anteriores se integran en OSSIM en forma de sondas de detección o gestores de información independientes (combinando, por ejemplo, sus bases de datos con las de OSSIM). Aunque en algunos casos es posible adaptar sus interfaces de usuario en la consola de administración general de OSSIM, la mayoría se integran a OSSIM a partir de sus funciones de exportación de datos o a través de su API. En algunos casos, puede ser necesario modificar o adaptar parte del código fuente para incorporar fun-

cionalidades adicionales y permitir las tareas de agregación o correlación de alertas. Una vez realizada las modificaciones, OSSIM puede ser utilizado como interfaz global y todas las aplicaciones anteriores serán gestionadas como subaplicaciones de OSSIM (figura 12).

Figura 12. Interfaz gráfica de OSSIM



Como ocurre con Snort y Sourcefire, la versión básica de OSSIM puede ser también complementada con una versión comercial más completa y con soporte técnico llamada *AlienVault Professional SIEM**. Ambas versiones contienen funcionalidad suficiente como para supervisar y gestionar la seguridad de grandes redes a través del amplio conjunto de herramientas de detección y prevención integrables en el producto final.

*** Disponible en http://alienvault.com.**

Resumen

Las redes de ordenadores se encuentran expuestas a ataques informáticos con tanta frecuencia que es necesario imponer una gran cantidad de requisitos de seguridad para la protección de sus recursos.

Aunque las deficiencias de estos sistemas se pueden comprobar mediante herramientas convencionales, no siempre son corregidas. En general, estas debilidades pueden provocar un agujero en la seguridad de la red y facilitar entradas ilegales en el sistema.

Del mismo modo que para garantizar la seguridad física de un edificio se suelen instalar detectores de movimiento, cámaras de vigilancia, libros de registro, etc., una red informática requiere componentes equivalentes en el mundo digital para recoger y generar escenarios de intrusión, procesar alertas y prevenir actividades intrusivas.

En este módulo didáctico se han presentado los sistemas de detección de intrusos (IDS), cuyo objetivo es precisamente proporcionar dichos elementos complementarios a los mecanismos de seguridad tradicionales y poder ofrecer capacidades adicionales para avisar y guiar a los administradores de la red en el momento en el que se produzcan ataques e intrusiones informáticas.

Se ha realizado también una primera aproximación a Snort, un sistema de detección de intrusos en red, basado en código abierto y ofrecido a la comunidad de administradores de red como herramienta de software libre. El objetivo principal de Snort es ayudar a los administradores de una red a realizar una vigilancia continuada del tráfico de la red en busca de intentos de intrusión o usos indebidos en ella.

Por último, hemos concluido el módulo con una presentación general de funcionalidades adicionales para facilitar la normalización, consolidación y puesta en correspondencia de los eventos recogidos por sondas de detección heterogéneas. Se ha presentado en detalle alguna de estas funcionalidades y se han repasado ejemplos de productos que proporcionan una interfaz gráfica adicional para configurar y ampliar las capacidades ofrecidas por sistemas de detección de intrusos en red, como Snort, combinándolos con sistemas preventivos, antivirus y sistemas de detección de vulnerabilidades.

Actividades

1. Tratad de instalar la última versión disponible de Snort y listad las distintas categorías en las que se encuentran recogidas las firmas de detección en la distribución general de Snort.
Pista: Consultad los enlaces http://www.snort.org y http://www.snort.org/snort-rules/.

2. Buscad cinco de los preprocesadores utilizados por Snort y estudiad cuál es el flujo de ejecución de tales componentes.
Pista: Consultad las informaciones relacionadas con Frag3, Stream5, RPC Decode, SSL/TLS, DNS, etcétera.

3. Buscad y estudiad cuál es el algoritmo utilizado por Snort para realizar la operación de *multiple-string matching*. Tratad de aislar y analizar en qué parte del código de Snort dicho algoritmo está implementado.
Pista: Iniciad la búsqueda por Aho-Corasick.

4. Tratad de explicar con ejemplos situaciones en las que un NIDS como Snort será propenso a la generación de (1) falsos positivos; (2) falsos negativos.
Pistas:

- Falso positivo: Situación en la que las firmas de configuración de Snort son demasiado generales (poco explícitas), y tráfico normal, que no corresponde a ningún ataque, es alertado por Snort.
- Falso negativo: Reglas que son tan precisas (poco generales) que una modificación mínima en el tráfico asociado a un ataque logra evadir la detección de Snort.

5. Enumerad algunos ejemplos de ataque que no pueden ser caracterizados (o descritos) mediante reglas de Snort. Explicad por qué no es posible dicha caracterización.
Pista: Pensad en ataques que requieran detección basada en caracterización de anomalías.

6. Describid dos o tres técnicas de evasión para escapar a la detección de Snort.
Pista: Consultad los artículos de Handley-Paxson y Blunden y estudiad técnicas tales como la fragmentación de tráfico y el polimorfismo.

Referencias bibliográficas

M. Handley; V. Paxson (2001). *Network Intrusion Detection: Evasion, Traffic Normalization and End-to-End Protocol Semantics*. USENIX Security Symposium.
B. Blunden (2009). *The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System*. Jones & Barlett.

Glosario

amenaza *f* Violación de la seguridad en potencia, que existe en función de unas circunstancias, capacidad, acción o evento que pueda llegar a causar una infracción de la seguridad y/o causar algún daño en el sistema.

ataque *m* Agresión a la seguridad de un sistema fruto de un acto intencionado y deliberado que viola la política de seguridad de un sistema.

bug *m* Error de programación que puede desencadenar una deficiencia de seguridad.

caballo de Troya *m* Programa, aparentemente inofensivo, que contiene en su interior un ataque contra una vulnerabilidad no corregida.

CERT (*Computer Emergency Response Team*) *m* Equipo de respuestas a emergencias informáticas. Una de sus principales tareas consiste en la gestión de vulnerabilidades.

CSIRT (*Computer Security Incident Response Team*) *m* Equipo de respuesta a incidentes de seguridad informática. Una de sus principales tareas consiste en la gestión de vulnerabilidades.

CVE (*Common Vulnerabilities and Exposures*) *m* Estándar publico para la identificación de vulnerabilidades. Asocia un identificador único a cada vulnerabilidad diferente.

CVSS (*Common Vulnerability Scoring System*) *m* Marco común para la evaluación de la criticidad de vulnerabilidades.

denegación de servicio *f* Ataque que tratará de saturar recursos de la víctima, tales como memoria o capacidad de cálculo y procesamiento.

DDoS *f* Denegación de servicio distribuida (en inglés, *distributed denial of service*).

DoS Véase *denegación de servicio* (en inglés, *denial of service*).

exploit *m* Programa o *script* que permite explotar una o varias vulnerabilidades, es decir, programa que permite realizar un ataque aprovechando la vulnerabilidad.

exploración de puertos *f* Técnica utilizada para identificar los servicios que ofrece un sistema o un equipo en particular.

escáner de vulnerabilidades *m* Aplicación que permite comprobar si un sistema es vulnerable a un conjunto de deficiencias de seguridad.

malware *m* Programa con fines malintencionados.

política de seguridad *f* Conjunto de reglas y prácticas que definen y regulan los servicios de seguridad de una organización o sistema con el propósito de proteger sus recursos críticos y sensibles. En otras palabras, es la declaración de lo que está permitido y lo que no está permitido hacer.

riesgo *m* Expectativa de pérdida expresada como la probabilidad de que una amenaza particular explote una vulnerabilidad concreta con resultados especialmente perjudiciales.

rootkit *f* Programa que permite acceso privilegiado a un ordenador y consigue ocultar su presencia al administrador. Suele hacer uso de diversas vulnerabilidades para instalarse y conseguir su propósito.

sniffer *m* Aplicación que intercepta toda la información que pase por la interfaz de red a la que esté asociado.

troyano *m* Véase *caballo de Troya*.

vulnerabilidad de día-cero (*zero-day vulnerability*) *f* Vulnerabilidad de cuya existencia no se tiene conocimiento en el momento de ser explotada.

vulnerabilidad de seguridad *f* Fallo o debilidad en el diseño, la implementación, la operación o la gestión de un sistema, que puede ser explotado con el fin de violar la política de seguridad del sistema.

Bibliografía

Beale, J.; Foster, J. C.; Posluns J.; Caswell, B. (2003). *Snort 2.0 Intrusion Detection*. Syngress Publishing

Garcia-Alfaro, J. (2004). "Mecanismos para la detección de ataques e intrusiones". En: J. Herrera; J. Garcia-Alfaro; X. Perramon. *Seguridad en redes de computadores*. Fundació Universitat Oberta de Catalunya

Garcia-Alfaro, J. (2007). "Detección de ataques en red con Snort". En: J. Herrera; J. Garcia-Alfaro; X. Perramon. *Aspectos avanzados de seguridad en redes*. Fundació Universitat Oberta de Catalunya

Koziol, J. (2003). *Intrusion Detection with Snort*. Sams Publishing

Kruegel, C.; Valeur, F.; Vigna, G (2004). *Intrusion Detection and Correlation: Challenges and Solutions*. Springer-Verlag

Northcutt, S.; Novak, J. (2002). *Network Intrusion Detection, 3.ª edición*. New Riders

Miller, D. R; Harris, S.; Harper, A. A.; Vandyke, S.;Blask C. (2011). *Security Information and Event Management (SIEM) Implementation*. Mc Graw Hill

Proctor, P. E. (2001). *The practical intrusion detection handbook*. Prentice-Hall

Rehman, R. (2003). *Intrusion Detection Systems with Snort. Advanced IDS Techniques Using Snort, Apache, MySQL, PHP, and ACID*. Prentice Hall PTR