Universidad de Murcia

Facultad de Informática

Departamento de Ingeniería de la Información y las Comunicaciones.

PROYECTO FIN DE CARRERA

SIRIUS: Sistemas de Detección en la Red de Intrusos. Aplicación en la Universidad de Murcia

Septiembre 2001

Carlos Coll Almela

Tutores:

Antonio F. Gómez Skarmeta

Francisco Jesús Monserrat Coll

Alumno:

Carlos Coll Almela

A mi madre, y a todos mis hermanos.

Índice de contenidos

Índice de figuras

0. Abstract

La idea con la que surge este proyecto, es el de realizar una serie de experiencias en el seno de la Universidad de Murcia, para diseñar un sistema que sea capaz de detectar los intentos de acceso no autorizados a los ordenadores de esta Institución, así como la detección de tráfico anormal, o anómalo en estos equipos.

Para la consecución de estos objetivos, intentaremos centrarnos en la utilización de herramientas y Sistemas Operativos de libre distribución, así como en la obtención de un sistema que no sea demasiado exigente en recursos de máquina (tanto a nivel de microprocesador como a nivel de memoria).

Por ello utilizaremos un Sistema Operativo Linux, junto con unas herramientas de detección de intrusos en red.

Las características generales del sistema propuesto son:

- Desarrollado para un ambiente abierto, donde no es posible llevar un control centralizado de los ordenadores.
- Remoto: la detección no se hace en cada una de las máquinas.
- Fácilmente extensible: se pueden insertar más sensores al sistema de una manera sencilla
- Altamente configurable.
- Bajo costo: el sistema debe de ser barato.

Su finalidad, por tanto, es un sistema que permita, la detección tanto de los actuales procedimientos de intrusión en sistemas, como la inserción de nuevos procedimientos y nuevas trazas de tráfico anómalas.

Actualmente existe gran variedad de sistemas de monitorización de redes, así como una gran variedad de herramientas para impedir estos accesos no autorizados. Numerosas empresas dedicadas tanto a dispositivos software como hardware, están

lanzando al mercado una serie de productos dedicados a garantizar la seguridad de los ordenadores bajo los cuales se está utilizando o ejecutando sus productos.

Esta variedad de herramientas y dispositivos físicos para la monitorización tanto del tráfico existente en las redes, como para monitorizar eventos, posibles accesos no autorizados, etc. tiene la finalidad de garantizar la seguridad de los sistemas que tienen por debajo, o como mínimo advertir a los administradores de estos sistemas de posibles vulnerabilidades o posibles accesos no autorizados.

Así mismo también existen distintos sistemas para detección de vulnerabilidades en redes, sistemas de auditoria, sistemas de chequeo de los ficheros de configuración de las máquinas, etc.

Todas estas herramientas proveen al administrador de la red de una gran cantidad de información sobre los incidentes que se produzcan o se puedan producir en la red que están administrando. Pero esta cantidad de información suele estar distribuida por los diferentes ordenadores que componen la red, dificultando su análisis. Además suele ser tan grande esta información, que puede llegar a ser totalmente inútil.

Este proyecto, pretenden solucionar estos inconvenientes, dotando al administrador de la red de un mecanismo de control lo más centralizado posible, y que además no requiera de ninguna modificación de la configuración de los ordenadores de los usuarios finales.

1. Introducción

Día a día estamos viendo que el número de usuarios que acceden a Internet desde sus hogares, trabajos, o centros de formación (bien sean Escuelas, Institutos o Universidades), está experimentando un gran aumento.

Esto se debe al gran avance que se ha producido en los últimos años en cuanto a las tecnologías de transmisión de datos, y en el abaratamiento de las mismas.

Así mismo se aprecia que desde los medios de comunicación, se está dando un gran apoyo a estas tecnologías. Cada día son más los periódicos y organizaciones que abren su portal en Internet.

Este aumento del uso de Internet no acaba aquí, sino también gran cantidad de empresas están abriendo sus páginas en la Red, puesto que han encontrado en ella un modo económico y rápido de hacer llegar información sobre sus productos a los usuarios.

No solo el aumento del uso de la Red de Redes se da en lo referente a usuarios domésticos o en lo que atañe a publicidad. La introducción de las tecnologías de la información en las empresas está provocando que también sean cada vez más las empresas que comienzan a abrir sus redes internas para que sus empleados puedan trabajar desde sus propias casas, dándoles acceso a través de Internet a sus aplicaciones y a sus datos confidenciales.

A su vez cada día es más frecuente encontrase en los medios de comunicación noticias referentes a la Red, y los incidentes que en ella se producen. A nadie le extraña que en cualquier noticiario se le diga que tal o cual compañía a tenido problemas con su sistema informático, o que está circulando tal virus que se propaga de esta manera.

Además, también es más común que en la misma red se puedan encontrar informaciones sobre las vulnerabilidades de un sistema concreto. Habitualmente, el fin de la divulgación de estas informaciones es preventivo e informativo. Pero no siempre

se hace un uso correcto de esta información, y en muchas ocasiones, más de las que se debería, esta información es utilizada para poder acceder a sistemas a los cuales no se tiene acceso.

Por tanto se está produciendo un aumento de los intentos de accesos a sistemas por parte de personas que no tienen autorización para hacerlo. Muchos de estos accesos se producen por personas que quieren "poner a prueba" sus conocimientos o el resultado de sus indagaciones, sin ningún otro propósito que la mera satisfacción personal. En estos casos no se suelen producir daños a la información que los ordenadores a los que se accede contienen.

Pero también es cierto que el número de accesos a sistemas con el fin de explotar la información que contienen de alguna forma, está también creciendo.

Muchos de estos accesos se producen desde dentro de la misma organización que tienen los datos, y estos son posiblemente los más difíciles de detectar y de impedir.

A lo largo de esta memoria, nos centraremos en intentar exponer un sistema de monitorización de redes para intentar descubrir estos accesos no autorizados, bien se produzcan desde dentro de la red de la organización donde el sistema se ha puesto en marcha, o bien sea desde fuera de ella.

Hay multitud de formas de intentar acceder a ordenadores, y muchas de ellas dependen del sistema operativo instalado en la máquina o de los programas que en ellos estén instados, así como de los servicios que estén activos en ella, dado que en muchas ocasiones, no se pretende atacar a un ordenador en concreto, si no que se buscan aquellos ordenadores que tengan una serie de vulnerabilidades ante las cuales se sabe como actuar para poder acceder al equipo.

La forma más común de proceder en determinados sistemas (aquellos que tienen sistemas operativos tipo Unix) es la siguiente:

- El atacante, realiza un escaneo con el fin de detectar algún servidor con un fallo de seguridad, normalmente ampliamente conocido y difundido por las listas de

distribución de seguridad, en las cuales pueden aparecer incluso, algunos programas que demuestran estas vulnerabilidades.

- Se utiliza este programa o exploit para lograr una puerta de acceso al sistema. En muchas ocasiones, es el mismo programa el que genera un interprete de comandos con privilegios de root.
- El atacante instala un conjunto de programas de nombre y comportamiento similares a los del sistema que sin embargo muestran información sobre determinados estados del sistema.
- El atacante instalará / compilará algunas herramientas para escanear o atacar a otros equipos utilizando esta máquina recién atacada como puente.

Esta situación se produce hasta que alguien, habitualmente desde fuera de la organización a la que pertenece este ordenador atacado, nota un comportamiento anómalo (por ejemplo que se están produciendo ataques desde la organización a la que este ordenador pertenece a la suya propia), y al contactar con el administrador de esta red, es cuando se advierte que el ordenador ha sido atacado.

Por escaneo, nos referimos a la utilización de determinados programas, muchos de ellos disponibles de manera sencilla y gratuita a través de multitud de páginas WEB, que realizan una consulta a los puertos de comunicación de un ordenador, en busca de una serie de servicios que estén a la escucha (por ejemplo un servidor http, el cual habitualmente está a la escucha de peticiones en los puertos 80 o 8080).

Examinando esta forma habitual de actuación, llegaremos a la conclusión de que en nuestro sistema para la detección de intrusos en red, no solamente deberemos centrarnos en los accesos no autorizados que se produzcan, sino también en los escaneos de los puertos de aquellos ordenadores que queramos controlar, puesto que de esta forma, lograremos llegar un paso por delante al intento de ataque, y podremos ser capaces de estar alerta en determinados equipos o de tomar medidas correctoras en ellos para lograr evitar el ataque.

Centrándonos un poco más en el entorno en el que se ha desarrollado el presente proyecto en detección de intrusos, en la Universidad de Murcia, y más en concreto la

Facultad de Informática de dicha Universidad, nos encontramos con los siguientes problemas en cuanto al control de los ordenadores.

- Hay gran dificultad de llevar un control centralizado de los equipos y los usuarios. Respecto a los equipos, hay gran cantidad de laboratorios de investigación y de desarrollo de prácticas, cada uno de ellos con unos programas distintos, y con unas necesidades que impiden que se pueda llevar una política de configuración centralizada e incluso impiden la instalación de ciertas medidas de protección de las comunicaciones de los mismos, como podría ser la instalación de un cortafuegos. En cuanto a los usuarios, la gran cantidad de los mismos, los diferentes usos que se requieren, y en algunos casos los niveles de accesos para poder llevar a cabo estas prácticas, provocan que tampoco se pueda llevar a cabo de manera sencilla una política de gestión y control de las cuentas de los mismos.
- Hay una multitud de redes que controlar, y en muchos casos no son homogéneas. Cada laboratorio suele disponer de su propia red de comunicación para los equipos que se encuentran en ella, y además estas redes no son homogéneas, puesto que se da la coexistencia de redes basadas en concentradores o hub's y redes basadas en conmutadores o switch dentro de la propia Facultad.
- Aparición de equipos sin control, vulnerables a ser atacados. Puesto que no existe un control centralizado de los equipos, pueden aparecer ordenadores en los cuales para cubrir alguna necesidad concreta se ha instalado una aplicación, o sistema operativo, el cual no está debidamente configurado, dejando abiertos, o no actualizados, determinados servicios, dejando una puerta en la red.

Las formas más comunes de acceder a equipos a los cuales no se tiene acceso dentro de la Universidad de Murcia, se asemeja bastante al procedimiento más general descrito anteriormente. Pero aquí, y debido a la existencia de redes no conmutadas dentro de la Universidad de Murcia, aparece otro método, como es la introducción de un sniffer en uno de los ordenadores de esta red. Un sniffer no es más que un programa residente en un ordenador que no hace otra cosa que interceptar todos los paquetes que circulan a través de la red. Por tanto entre los paquetes que intercepta, se encuentran aquellos que

contienes passwords no encriptadas. Por tanto se logran las claves para acceder a otros equipos.

Así pues, el objetivo del presente proyecto será una serie de experiencias mediante las cuales se puedan lograr los siguientes resultados:

- Lograr detectar ataques sin tener que configurar nada en los equipos finales y de usuarios
- Generar alertas de los ataques que se produzcan para poder ser enviados a los administradores de la red
- Evaluar herramientas para poder llevar esto a cabo.
- Desarrollar este proyecto, con el máximo de recursos disponible, sin que se requiera invertir en el desarrollo de las mismas.

Durante esta memoria, expondremos el desarrollo de este proyecto. Para ello, en el siguiente punto, metodología, expondremos la estructura de la red con la que vamos a trabajar, así como realizaremos una comparación entre las distintas herramientas disponible, para acabar hablando de las herramientas concretas que se van a utilizar.

Una vez visto lo anterior, pasaremos en el siguiente punto, diseño de una solución concreta, a ver la opción por la que nos hemos decantado, tanto a nivel hardware como software, haciendo un análisis de las resultados obtenidos, y describiendo en detalle los ataques mayoritarios que se han detectado, así como la respuesta del sistema desarrollado ante estos ataques.

Finalmente, en el apartado de conclusiones detallaremos como se han logrado obtener los objetivos propuestos, así como las vías que creemos que se abren con este proyecto para futuros desarrollos.

2. Metodología

Antes de entrar en materia y realizar un explicación de las tecnologías y herramientas empleadas para la consecución de los objetivos anteriormente planteados, mostraremos la actual Red de Datos de la Facultad de Informática, para mostrar el entorno en que desarrollaremos este proyecto.

2.1. Estructura de la Red de Datos de la Facultad de Informática

A finales de 2000, principios de 2001 se llevó a cabo un cambio en la Red de Transmisión de datos de la Facultad de Informática, con el objetivo de pasar a una red de conmutación, de tal manera que idealmente, todo ordenador estaría conectado a una toma de datos que partiría de un conmutador o switch.

De esta manera sería muy difícil lograr llevar a cabo con éxito la puesta en funcionamiento de un sniffer para capturar todos los datos que circulen por el segmento, puesto que desde la toma del switch al que el ordenador estaría conectado, únicamente partirían los datos que van hacia un ordenador concreto.

De todos modos, el motivo de la instalación de estos switch, no está directamente relaciona con la obtención de una mayor seguridad, sino más bien para lograr un ancho de banda dedicado (10/100 MHz) entre cada equipo.

Por desgracia, la implementación de este sistema en todos los ordenadores de la Facultad de Informática es un poco compleja, puesto que hay un gran número de equipos, y tanto la electrónica como el cableado necesario para que absolutamente todos los ordenadores estuvieran conectados de esta manera resultaría excesivamente costoso.

Así pues, si bien una buena parte de los ordenadores están directamente conectados a una toma del *switch*, no todos ellos lo están.

La figura 1 describe la actual red de la Facultad de Informática de la Universidad de Murcia. En ella, podemos distinguir lo siguiente:

- La Red se distribuye por los conmutadores localizados en las distintas plantas de la Facultad de Informática ,con dos conmutadores distintos para cada una de las Aulas de Libre Acceso (A.L.A.)
- Cada uno de estos conmutadores se conecta a un *router* a través de un enlace de fibra.
- A su vez, este *router* se conecta a la red ATM que la Universidad de Murcia tiene para la interconexión de los distintos edificios del Campus de Espinardo.

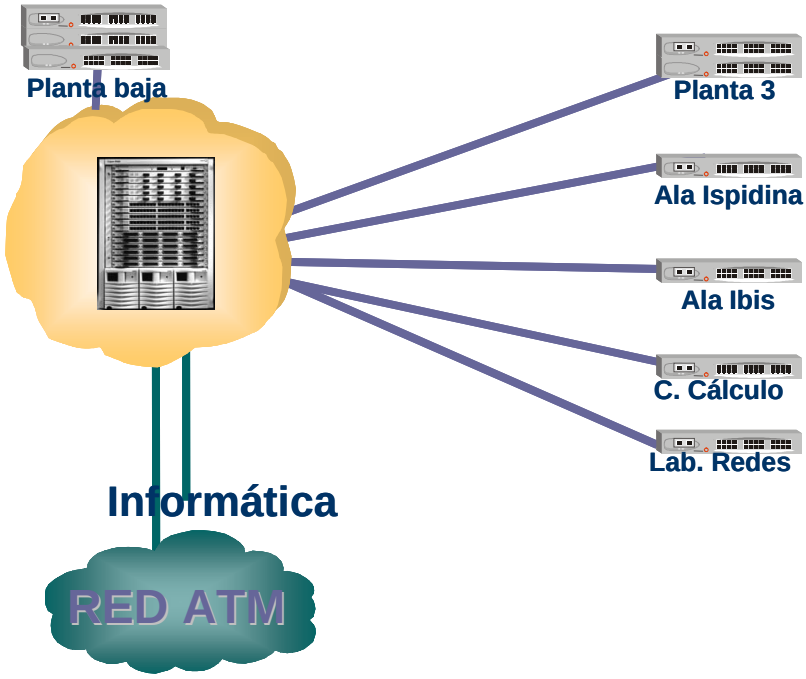


Figura 1. *Topología de la red ATM de la Universidad de Murcia*

La principal ventaja de este tipo de red, es que se logra evitar el problema del *sniffing* en los ordenadores que estén conectados directamente a tomas del conmutador, salvo que el atacante logre acceder a máquinas de propósito general, sobre las cuales existe un mayor control.

El inconveniente, se ha comentado anteriormente. Posiblemente en los laboratorios, se continúe con la utilización de concentradores, en cuyo caso, el problema del sniffing no estará resuelto en aquellos segmentos que estén conectados a estos concentradores.

2.2. Sistemas de detección de intrusos

Un sistema de detección de intrusos busca los rastros que los atacantes dejan, y que están definidos mediante patrones que indican un comportamiento malicioso o cuanto menos sospechoso. Cuando un sistema trata de detectar estos patrones mediante la utilización de un dispositivo de red, el cual ha sido configurado para un funcionamiento en modo promiscuo (que lea todos los paquetes que circulan por su segmento de red, aunque estos no vayan dirigidos a este equipo) entonces se considera un sistema de detección de intrusos basado en red (NIDS). Además de estos NIDS, también aparecen los llamados sistemas de detección de intrusos basados en equipos (HIDS). Hay tres tipos de estos sistemas:

- El primero de ellos, busca patrones en los logs de la propia máquina.
- El segundo examina patrones dentro del tráfico de la red (pero sin tener un interface en modo promiscuo, por tanto únicamente analiza los paquetes que van dirigidos a esta máquina)
- Por último, una combinación de ambos sistemas, son los denominados sistemas de detección de intrusos basados en pilas (del inglés Stack – Based IDS).

La industria de los IDS, ha experimentado un rápido crecimiento en los últimos tiempo. Durante el último año, las casas comerciales que han comenzado con el desarrollo de productos destinados a la detección de intrusos, tanto productos software como hardware, han crecido espectacularmente.

La mayoría de los productos que hemos consultado, han comenzado con la detección de los ataque más populares, y han ido incluyendo otros tipos de ataques en sus librerías poco a poco. Estos nuevos ataques suelen ser publicados de una forma rápida en listas que han llegado a hacerse bastante populares, como son las Bugtraq, por tanto, la frecuencia de las actualizaciones en las librerías de estos productos es también un aspecto importante, así como la posibilidad de incluir nuevos ataques en estas librerías.

En esta sección, realizaremos un estudio de los distintos tipos de sistemas de detección de intrusos así como la realización de una comparativa entre los diferentes

sistemas de detección de intrusos que podemos encontrar, para facilitar la decisión de cual o cuales de ellos vamos a utilizar para el desarrollo de las experiencias que se exponen en esta memoria.

2.2.1. Sistemas de detección de intrusos basados en red

Estos sistemas toman el trafico de la red como entrada de datos. Habitualmente utilizan un interface de red que esta configurado para su funcionamiento en modo promiscuo, de modo que escuchan y analizan todo el tráfico que circula por la red en tiempo real. Normalmente, aplican un filtro a este tráfico, para determinar cuales de los paquetes serán descartados, y cuales serán dejados para ser analizados. Esto ayuda a que el rendimiento de estas aplicaciones sea mejorado, puesto que aquel tráfico que se sabe que no es malicioso, es pasado por alto. Un ejemplo de esto, lo encontramos en los paquetes SNMP. Si sabemos qué máquinas son las que generan este tipo de tráfico, podemos pasar por alto los paquetes SNMP que provengan de las mismas. Esto ha de ser hecho con mucho cuidado, dado que una configuración errónea nos puede llevar a que demasiados paquetes sean pasados por alto. Estos sistemas suelen poseer un módulo de reconocimiento de paquetes en los cuales se suelen aplicar tres metodologías distintas.

- Marcas típicas de ataques
- Reconocimiento de patrones en el contenido de los paquees
- Detección se situaciones anómalas, tales como una gran cantidad de tráfico en la red

2.2.2. Sistemas de detección de intrusos basados en host

Este tipo de sistemas son anteriores a los NIDS. En la década de los 80 era habitual revisar los ficheros de log de los sistemas para intentar encontrar actividades sospechosas o relevantes en lo que respecta a la seguridad del sistema.

Hoy en día estos sistemas aún se continúan utilizando, pero son mucho más sofisticados de lo que eran en su comienzo, y son capaces de realizar el análisis en tiempo real de estos ficheros de log, para detectar comportamientos anómalos.

Actualmente, estos sistemas tan pronto como detectan un cambio en estos ficheros de log o en los ficheros de configuración del sistema, los comparan con sus ficheros de configuración en los cuales tienen establecidas las políticas de seguridad del sistema, y la respuesta que deben dar ante determinados cambios. Hay dos formas de realizar esta comprobación de los ficheros, o bien se realiza en tiempo real cada vez que se realiza un cambio en ellos, o bien se realiza periódicamente por algún proceso que esté esperando en background. Algunos de estos sistemas tienen además la capacidad de escuchar en determinados puertos del sistema, y lanzar una alarma cuando estos puertos son accedidos.

2.2.3. Sistemas de detección de intrusos basados en pilas (*Stack based* IDS)

Esta es la técnica de detección de intrusos más reciente, y varía enormemente de fabricante a fabricante. Estos sistemas trabajan muy cerca de la pila de protocolos IP, permitiendo un seguimiento del paquete conforme va ascendiendo por la pila, de esta forma son capaces de analizar el paquete y retirarlo antes de que el nivel de aplicación o el sistema operativo tenga la oportunidad de procesarlo. Para que esta detección sea completa, estos sistemas chequean tanto el tráfico entrante como el saliente. La razón de monitorizar únicamente un equipo, es lograr que este proceso no sobrecargue en exceso el equipo en el que se está ejecutando de manera que se pueda realizar en todos los equipos de la organización.

2.2.4. Beneficios de los sistemas de detección de intrusos basados en red

- Coste: NIDS permiten la aplicación de estrategias de instalación únicamente en aquellos puntos de la red que son considerados como puntos de acceso críticos. Por tanto no es necesario instalar, configurar y mantener tantos equipos como en el caso de los HIDS. De este modo el coste de mantenimiento decrece.
- Análisis de paquetes: NIDS examinan todas las cabeceras de los paquetes para la detección de actividades sospechosas en la red. Muchos de los ataques por denegación de servicio que se producen actualmente, pueden ser detectados únicamente examinando las cabeceras de los paquetes que circulan a través de la

red. Por ejemplo, uno de los ataques por denegación de servicio, consiste en inyectar en la red, paquetes que tienen la misma dirección origen y destino, de manera que el equipo abrirá conexiones a si mismo, y puede causar la pérdida de rendimiento, o la caída del mismo. Este tipo de ataques puede ser fácilmente detectado y corregido por los NIDS mientras que los HIDS no serían capaces de detectarlo. Además la mayoría de los NIDS son capaces de mirar dentro del contenido del campo de datos, por tanto, son capaces de detectar determinados patrones que pueden indicar un comportamiento malintencionado.

- Eliminación de las evidencias. NIDS utilizan el trafico de red para detectar los ataques, por tanto, los atacantes no podrán eliminar los rastros que sus acciones dejan. Este es uno de los problemas que se encuentran en los HIDS, puesto que el atacante puede ser capaz de eliminar cualquier rastro, borrando determinadas líneas de los ficheros de log.
- Detección y respuesta en tiempo real.
- Detección de intenciones maliciosas. Si el NIDS es puesto delante de la detección de un cortafuegos, entonces, puede detectar ataques que se estén produciendo, y que en otro caso nunca serían detectados, puesto que el cortafuegos los eliminaría del trafico que deja entrar en la red.
- Complemento y verificación de las políticas de seguridad. En caso de haber alguna política definida respecto a encriptación, por ejemplo, un NIDS será capaz de averiguar si realmente esa política se está aplicando, viendo la cantidad de tráfico encriptado que circula a través de la red. Así mismo, también podrá ayudar a verificar el funcionamiento de los cortafuegos, verificando si ciertas direcciones o tipos de tráfico están filtrados o no.
- Independencia del sistema operativo. Los NIDS no son dependientes del sistema operativo que se esté ejecutando en las máquinas, mientras que los HIDS requieren distintas instalaciones dependiendo del sistema operativo que esté instalado en cada uno de estos equipos.

2.2.5. Beneficios de los sistemas de detección de intrusos basados en equipos

- Verificación de los ataques. Dado que estos sistemas se basan en el análisis de los ficheros de log de los equipos en los que están instalados, tienen la ventaja

de que es posible saber si cierto exploit o intento de ataque ha dado resultado o no, siempre que el rastro que estos ataques puedan dejar, no sea eliminado de los sistemas.

- Actividad específica del sistema. Los HIDS pueden monitorizar y detectar fácilmente actividades directamente relacionadas con el sistema en sí, como pueden ser la modificación de ficheros o los accesos a la máquina.
- Monitorización de los componentes claves del sistema. Son capaces de monitorizar importantes componentes del sistema, tales como la ejecución de determinados programas, o librerías específicas, así como los cambios en determinados ficheros o componentes, como pueden ser los ficheros de password, o el registro de sistemas en las máquinas basadas en sistemas Windows.
- No es necesario la instalación de hardware adicional. Los HIDS no requieren la instalación de ningún componente hardware, puesto que utilizan los recursos hardware ya existente. Esto puede hacer que en determinados entornos sea más adecuado la utilización de estos sistemas.

2.2.6. Necesidad de ambos tipos de detección

Ambos sistemas de detección de intrusos, tienen ventajas que no las tiene el otro sistema, por tanto podía ser conveniente la utilización de ambos sistemas de detección. Esto es lo que intentan recoger los sistemas de detección de intrusos más novedosos. De todos modos no hay en lo que respecta a la seguridad en los sistemas de información, ningún sistema que nos garantice la seguridad total y absoluta de las acciones que en nuestra red se realicen.

En la figura2, se muestran algunas situaciones en las que primero por separado y después juntos, estos sistemas pueden ser de utilidad.

En esta memoria nos centraremos en los NIDS, puesto que uno de los objetivos de este proyecto es lograr un control lo más centralizado posible, por tanto creemos que la mejor forma de lograr estos objetivos, es basándonos en aquellos elementos que son comunes a todos los equipos que queremos controlar, como es el caso de la red en sí misma.

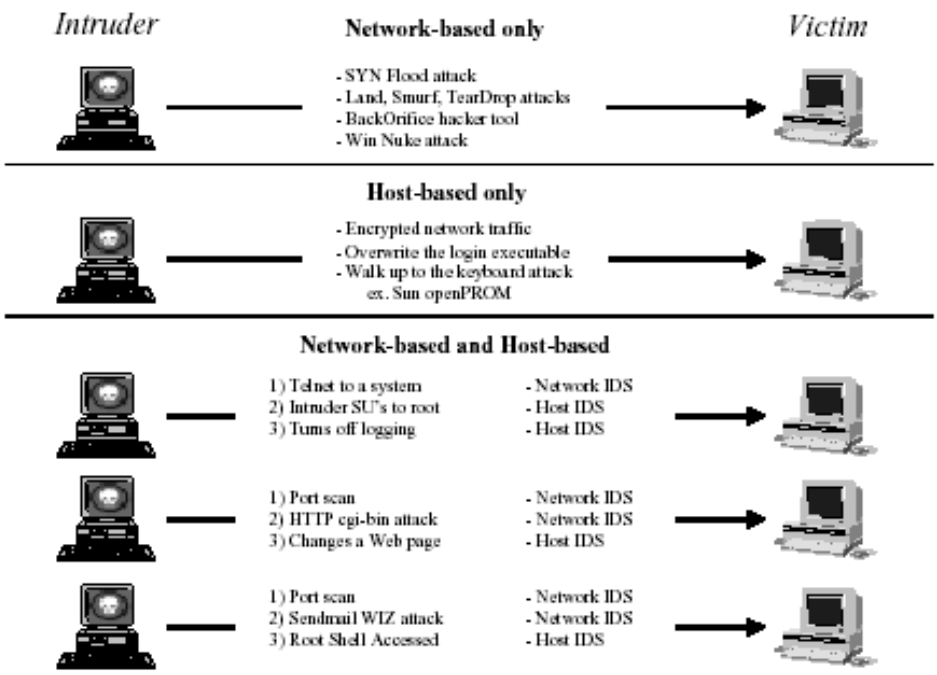


***Figura 2**. Combinación de distintos IDS*

2.3. Algunos Sistemas de Detección de Intrusos en Red

Veamos alguno de los sistemas que hemos evaluado para el desarrollo de este proyecto.

2.3.1. NFR (NFR systems)

2.3.1.1. Descripción

NFR Network Intrusion Detection (NFR NID) es un sistema de detección de intrusos que realiza una monitorización en tiempo real de actividades tales como posibles ataques conocidos, comportamiento anormal, intentos de accesos no autorizados así como infracciones a las políticas de integridad definidas.

La información asociada a la actividad que parece ser sospechosa, es guardada y se disparan alertas en caso de ser necesario. NFR NID proporciona dos tipos de detección, mal uso y detección de situaciones anómalas.

Las situaciones de mal uso se dan cuando se producen ataques conocidos, como por ejemplo un ataque destinado a tirar abajo un servidor de correo electrónico. Una situación anómala, se da cuando el sistema detecta algo fuera de lo ordinario, como puede ser, un aumento del tráfico en la red que el sistema está monitorizando.

Las detecciones de estos malos usos funcionan comparando el tráfico que pasa por la red, con patrones de ataques conocidos. Las situaciones anómalas, son "aprendidas", o se le dice al sistema que es lo que constituye un funcionamiento normal, desarrollando conjuntos de modelos, que son continuamente actualizados, de modo que la actividad que se está monitorizando, es comparada con estos modelos.

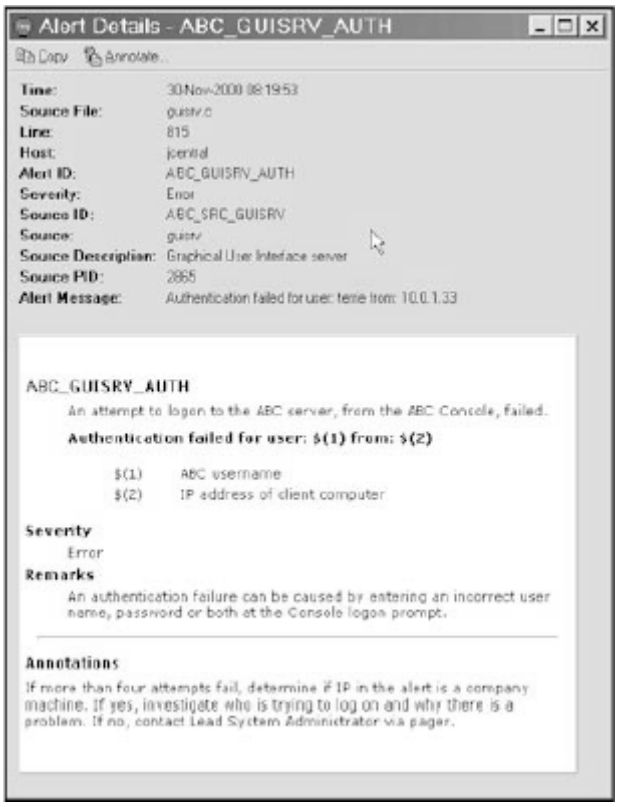


***Figura 3**. Ejemplo de alarma de NFR*

NFR NID está disponible en dos versiones, una de ellas el proveedor nos da el ordenador basado en una máquina multi-procesador con el software ya instalado y listo para personalizar y comenzar a ejecutarlo. En la otra, se nos vende únicamente el software, siendo el usuario el que ha de instalarlo y ponerlo en marcha.

2.3.1.2. Requerimientos del sistema

- Procesador: Pentium III ® o superior
- Memoria: 512 MB
- Espacio en disco: 20 GB
- Dos tarjetas de red (se recomienda una combinación de tarjetas de determinados fabricantes, no dos tarjetas iguales)

2.3.2. SecureNetPro (de Intrusión.com)

2.3.2.1. Descripción

Este NIDS, también disponible en versión hardware (con distintas posibilidades en cuanto a la potencia de la máquina) y software, reconoce 400 patrones de ataque, los cuales son scripts de análisis para intentar minimizar las falsas alarmas.

Este NIDS consta de un total de 26 paquetes, los cuales están dispuestos de forma que cada paquete sea capaz de analizar el tráfico de un protocolo, así pues únicamente tendremos que instalar el módulo que nos sea necesario, para monitorizar el tráfico en el que estamos interesados.

Contiene una herramienta para poder conocer el estado de la red en tiempo real, mediante la realización de conexiones a una aplicación que nos muestra el estado del sistema.

También tiene la posibilidad de poder desarrollar nuevos scripts de búsqueda de patrones mediante la realización de scrpits en un lenguaje propio, así como puede enviar alertas mediante correos electrónicos, y alertas SNMP.

2.3.2.2. Requerimientos del sistema

- Pentium II ® 400 MHz o superior
- 128 MB Ram
- 205 MB de espacio libre de disco

- Sistema Operativo Linux Red Hat versión 6.X

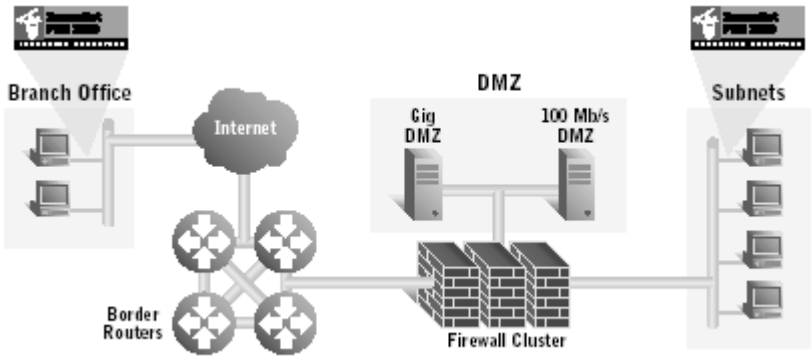


***Figura 4**. Topología propuesta por Intrusion.com*

2.3.3. BlackICE Sentry (Network ICE Coorporation)

2.3.3.1. Descripción

BlackICE Sentry monitoriza segmentos completos de red en tiempo real para localizar actividades sospechosas. Estas actividades incluyen intentos de ataque a mainframes, servidores de impresión u otro tipo de pruebas. Cuando se detecta un ataque, inmediatamente comienza a guardar la información sobre el ataque, dispara una alerta y la envía al servidor ICEcap (otro software de la misma compañía que ha de ser utilizado en otra máquina del segmento, y que se utiliza para guardar la información de los ataques producidos).

Esta herramienta es capaz de funcionar con éxito en redes FastEthermet con una gran carga, sin que por ello se produzca una degradación del rendimiento de la red.

En lugar de utilizar una tecnología basada en el reconocimiento de patrones, esta herramienta implementa un protocolo de análisis avanzado para analizar la estructura y decodificar cada paquete de manera individual.

Además BlackICE Sentry, no solo detecta actividad sospecha y posiblemente hostil en cualquiera de los ordenadores que monitoriza, sino que además guarda información acerca del ataque, tal como la dirección IP del ordenador atacante, el nombre del ordenador (el nombre NetBIOS), y la dirección MAC del interface de comunicaciones que se está utilizando.

2.3.3.2. Requerimientos del sistema

- Procesador: Mínimo Intel clase Pentium ® o superior aunque se recomienda un ordenador dual con dos Intel Pentium Celeron 550 MHz o superior
- Memoria 64 MB RAM mínimo, recomendada 128 MB
- Espacio en disco duro: mínimo 10 MB recomendado 100 MB
- Sistema operativo: Windows NT 4.0
- Solamente monitoriza TCP/IP
- Tarjeta de red Ethernet 10/100 Mbps.

2.3.4. Snort

2.3.4.1. Descripción

Snort es un sistema de detección de intrusos ligero, capaz de realizar un análisis en tiempo real del tráfico presente en la red y de los paquetes en redes IP. Puede realizar análisis de los protocolos, búsqueda de patrones en el contenido de los paquetes, y puede ser utilizado para detectar una gran cantidad de ataques y pruebas tales como desbordamientos de buffers, escaneo de puertos , ataques CGI, pruebas SMB, y muchos otros. Posee un lenguaje de reglas sencillo para describir el tráfico que ha de analizar, así como un motor de búsqueda con una arquitectura modular. A su vez, es capaz de lanzar alertas de distintas maneras, para ajustarse a las necesidades del usuario.

Hay tres formas principales de utilizar este software. Puede ser utilizado como sniffer de paquetes únicamente, así como para guardar el tráfico de la red (útil para hacer un posterior análisis del mismo), o como un completo NIDS

2.3.4.2. Requisitos del sistema

Snort basa sus requerimientos en el sistema operativo que esté instalado en la máquina, sin realizar ningún tipo de recomendación en cuanto al hardware de la misma. Con esto, se logra marcar las necesidades del sistema en el que se va a instalar Snort, en función de la cantidad de tráfico, y el tamaño de la red, en lugar del fijar unas necesidades por el software que se está ejecutando. Puede ser instalado en cualquiera de los siguientes sistemas:

X86	Sparc	M68k/ppc	Alpha	Otros	Sistema operativo
X	X	X	X	X	Linux
X	X	X			OpenBSD
X			X		FreeBSD
X		X			NetBSD
X	X				Solaris
	X				Sun OS 4.1.X
				X	HP-UX
				X	AIX
				X	IRIX
			X		Tru64
		X			MacOS X Server
X					Win32(Win9x/NT72000)

Sistemas operativos en los que Snort se puede instalar.

2.3.4. Comparación entre los NIDS anteriormente expuestos

A continuación tenemos los resultados de la comparación de los distintos productos anteriormente expuestos, para poder tomar una decisión en cuanto a cuales de ellos probaremos en nuestras experiencias. Esta comparación la realizaremos en función de los siguientes parámetros: documentación, cantidad de ataques reconocidos, claridad de los informes, falsas alarmas y rendimiento.

- Documentación y fácil instalación en un entorno seguro. Esto es particularmente importante cuando el producto va a ser instalado. Muchos de estos productos tienen la posibilidad de configurar las reglas por las que van a realizar la búsqueda de patrones, esto puede ser realmente útil, siempre que las personas que se hagan cargo de esta "puesta a punto", sean personas con un conocimiento técnico suficiente, para que puedan realizarlo. Snort permite esta característica. Otros productos, como BlackICE, NFR o SecureNetPro, no dan muchas facilidades para que esto se pueda realizar.
- Cantidad de ataques reconocidos. Este es el factor más importante a tener en cuenta para poder controlar la efectividad de estos dispositivos. La mayoría de

productos consultados, cubren distintos tipos de ataque, por tanto, para lograr una mayor seguridad, sería necesario la utilización de distintos sensores con distintos productos instalados. Dado que esto no es posible, Snort, proporciona la mayor cantidad de reglas por las que analizar, pero tiene el inconveniente de mostrar las alarmas que genera de una forma poco inteligible.

- Claridad de los informes. ¿Cómo de entendibles son las alertas que almacena, y que cantidad de información proporcionan?. NFR tiene la mejor herramienta de notificación de alertas, pero proporciona poca información acerca de las alertas que se han disparado. BlackICE muestra unos gráficos fácilmente interpretables, proporcionando una gran cantidad de información, pero dejando en el tintero información tan importante como de donde viene el ataque . SecureNetPro es la herramienta que da una información en tiempo real mostrando el nivel de tráfico analizado, y dando detalles de este tráfico conforme es analizado. Snort da una información en tiempo real del tráfico y las alertas que genera, siendo el sistema que proporciona una información a más bajo nivel, aunque por desgracia, no integra ningún sistema que permita un análisis rápido y sencillo de las alarmas que se disparan.

  En lo que se refiere a la claridad de los informes, hay que citar que han surgido una cantidad considerable de herramientas, de las que posteriormente daremos más detalle, las cuales tomando como entrada los ficheros de texto en los que el NIDS almacena sus alarmas es capaz de analizar y realizar informes de los ataques que se han realizado.

- Falsas alarmas. Un problema que puede presentarse cuando el número de patrones por el que busca es alto, es la cantidad de falsas alarmas que se disparan con el sistema. En el caso de que el número de estas falsas alarmas (paquetes correctos pero que el sistema reconoce como un posible ataque) sea muy grande, la utilidad de estos sistemas empieza a decrecer. En estas falsas alarmas, influye positivamente la posibilidad de que el usuario pueda poner a punto la base de reglas por las que el sistema analizará el tráfico. NFR, y Snort ofrecen la posibilidad de filtrar las falsas alarmas (en el caso de NFR no ofrece muchas facilidades para la personalización de las reglas por las que analiza, pero incluye una potente forma de discriminar las falsas alarmas). BlackICE no proporciona esta posibilidad, y SecuredNetPro, ofrece posibilidades muy limitadas en lo que

respecta a la modificación / personalización de las reglas o filtrado de falsas alarmas.

- Rendimiento. En lo relativo a los interfaces de red, para pequeñas organizaciones, esto no puede ser un problema, así como con los ordenadores actuales, tampoco deben de plantearse problemas de rendimiento, puesto que las CPUs actuales permiten grandes velocidades de computación, nos debemos de ceñir al material del que disponemos, dado que uno de los objetivos de este proyecto es la utilización de los recursos disponibles. Por tanto debemos señalar respecto al rendimiento que Snort es el sistema más apropiado para el desarrollo de este proyecto.

En general, si es posible sufragar la instalación de diversos NIDS, la arquitectura de seguridad obtenida proporcionará mayores niveles de seguridad que en caso contrario. Aún así, al igual que en el caso de los programas anti virus, el mantener las reglas por las cuales se filtra actualizadas, permitirá mayores niveles de seguridad.

En este sistema, hemos optado por la utilización de Snort como sistema de Detección de Intrusos en Red a utilizar.

2.4. Herramientas

Veamos a continuación las herramientas que hemos utilizado a lo largo del desarrollo de este proyecto.

2.4.1. Snort

Snort es un sistema de detección de intrusos ligero, una herramienta para redes pequeñas y poco utilizadas. Es útil cuando el precio de instalar sensores de sistemas de detección de intrusos en red, es demasiado alto. Los sistemas de detección de intrusos en red modernos, cuestan como mínimos 1000$ y en ocasiones hasta miles de dólares. Snort está disponible bajo las condiciones de la licencia publica general GNU, y es gratuito para ser usado en cualquier entorno.

2.4.1.1. ¿Que es un sistema de detección de intrusos ligero?

Un sistema de detección de intrusos ligero puede ser instalado en la mayoría de los nodos de una red. Los IDS ligeros suelen ser sistemas multiplataforma, con poco impacto en el sistema, y fácilmente configurable por los administradores de estos, que necesiten una solución específica en lo que respecta a la seguridad de los mismos en un pequeño plazo de tiempo. Estos, pueden disponer de un conjunto de herramientas que pueden ser ensambladas y puestas en funcionamiento de forma conjunta para lograr una solución global en cuanto a la seguridad de sus sistemas. Los IDS ligeros, son pequeños, flexibles y suficientemente potentes como para ser utilizados como elementos permanentes de la infraestructura de seguridad de una red.

Snort es un IDS ligero, que proporciona estos conjuntos de herramientas, ocupa entorno a 100 KB en su paquete de distribución comprimido. En la mayoría de arquitecturas modernas, tarda únicamente unos cuantos minutos en ser compilado e instalado, y quizás unos 10 minutos en ser configurado y activado. Comparemos esto con la mayoría de sistemas de detección de intrusos en red comerciales, los cuales requieren de una plataforma dedicada a ellos y un aprendizaje para ser instalados de una formar útil. Snort puede ser configurado, y puesto en marcha por periodos largos de tiempo sin que requiera monitorización o mantenimiento.

2.4.1.2. Descripción

Snort es un sniffer que se basa en las librerías libpcap, estándar en las distribuciones UNIX, y que puede ser utilizado como un sistema de detección de intrusos en red ligero. Es un sistema basado en reglas que analiza el contenido de los paquetes para comprobar y detectar si este contenido coincide con las reglas definidas. De esta forma es capaz de detectar una gran cantidad de ataques y pruebas, como desbordamiento de buffers, escaneos de puertos, ataques basados en CGI, pruebas SMB (Server Message Block) y muchos otros tipos de ataque, los cuales son detectados y alertados en tiempo real, mediante el envió al sistema de logs del ordenador, mediante mensajes por “WinPopup”, o almacenados en un fichero de alertas.

El motor de detección esta programado usando un sencillo lenguaje que describe el contenido de los paquetes y las acciones a tomar mediante una serie de reglas las cuales

trataremos más adelante en profundidad, y que facilitan la escritura de nuevas reglas, en cuanto un nuevo tipo de ataque aparece en alguna lista de bugtrack.

Por ejemplo, a principios de este año, comenzó a distribuirse un gusano que atacaba a los ordenadores que tenían determinado servicio activo, aprovechando un agujero de seguridad en el puerto 27374. Pues bien, las reglas pertinentes para detectar si este gusano había entrado en el sistema y se estaba distribuyendo (en este caso dos reglas), no llevaron más de una hora en ser desarrolladas (incluyendo en este tiempo la investigación necesaria para ver cual era el funcionamiento del gusano, y como detectarlo).

2.4.1.3. Funcionamiento interno de Snort

La arquitectura de Snort está basada en el rendimiento, la simplicidad y la flexibilidad. Esta dividida en tres subsistemas, los cuales son el decodificador de paquetes, el motor de detección y el subsistema de alertas y logs. Estos subsistemas se sitúan por encima de la librería de captura de paquetes libpcap, la cual provee a Snort de las funciones necesarias para un sniffer de paquetes, utilizando para ello un interface de red configurado en modo promiscuo. La configuración del programa, el parser de las reglas, y la generación de las estructuras de datos, tienen lugar antes de ser inicializada la sección del sniffer, manteniendo el aumento de procesamiento por paquete al mínimo para ejecutar las funcionalidades base del programa.

2.4.1.3.1. Decodificador de paquetes

El decodificador de paquetes se organiza alrededor de los niveles de la pila de protocolos presentes, el interface y la definición de los protocolos TCP/IP. Cada subrutina en el decodificador impone orden en el paquete de datos mediante la colocación de las estructuras de datos en el paquete que llega. Estas rutinas de decodificación son llamadas en orden a través de la pila de protocolos. La velocidad en cuanto a la decodificación de los paquetes es muy importante en esta sección, por ello la mayor parte de las funcionalidades del decodificador de paquetes, consisten únicamente en la inserción de punteros entre los paquetes de datos para su posterior análisis por el motor de detección. Snort es capaz de decodificar paquetes de los protocolos Ethernet, SLIP y PPP.

2.4.1.3.2. El motor de detección

Snort mantiene sus reglas de detección en una lista enlazada de dos dimensiones que son llamadas “Chain Headers” y “Chain Options”. Estas son listas de reglas que han sido condensadas en una lista de atributos comunes en “Chain Headers” con los modificadores que permiten insertar opciones en la detección en la parte de “Chain Options”. Por ejemplo, en las 45 reglas de detección de pruebas de ataques por CGI-BIN, que están especificadas en la librería común de reglas de Snort, generalmente la dirección IP origen y destino así como el puerto origen y el puerto destino es el mismo, por tanto, todas estas reglas detección están condensadas en una única “Chain Header” y las características particulares de cada regla están insertadas en las estructuras de “Chain Options”. Esto se hace así, para lograr que el motor de detección funcione de una forma más rápida.

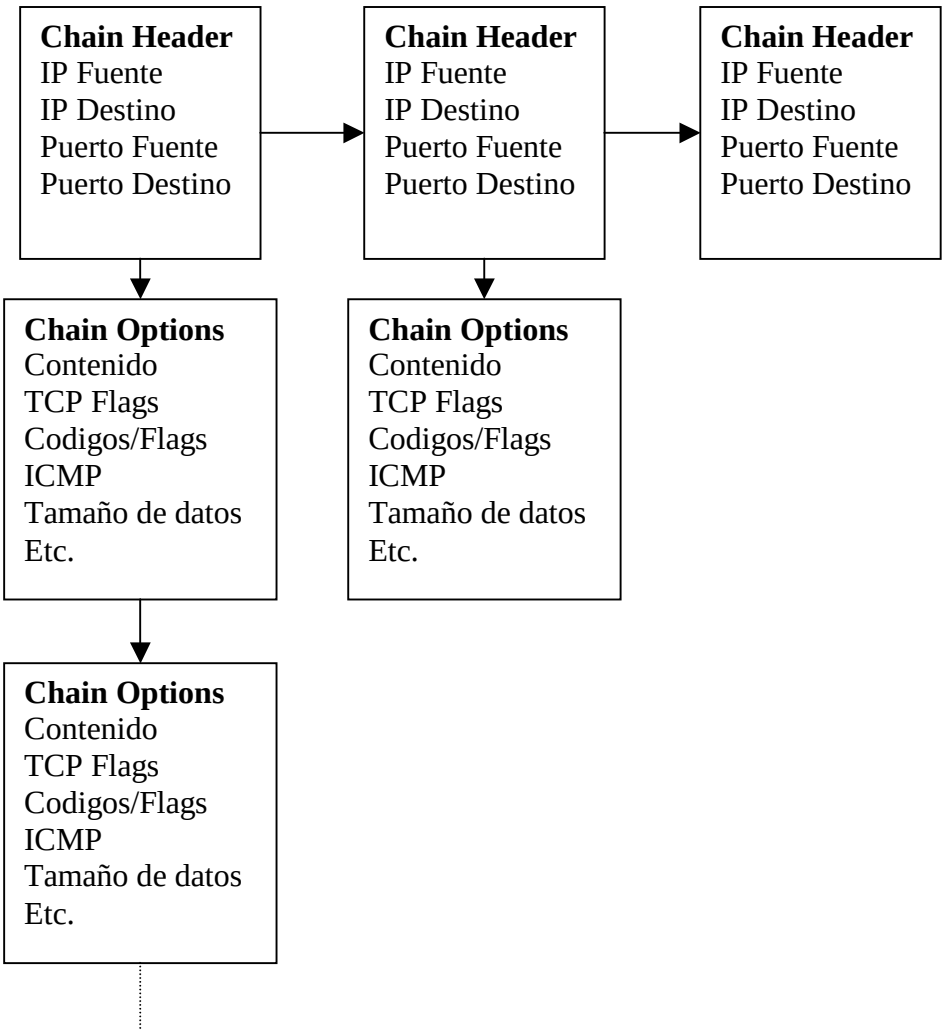


***Figura 5**. Estructuras de datos de Snort*

Estas cadenas de reglas son buscadas recursivamente para cada paquete en ambas direcciones. El motor de detección solamente chequea aquellas cadenas que hayan sido inicializadas al lanzarse el programa, por tanto para que cualquier modificación de los ficheros en los que están definidas las reglas se incluyan, ha de pararse el programa y volverse a lanzar. La primera regla que coincida con el paquete decodificado lanza la acción que lleva asociada en la definición de la regla, y retorna.

Además Snort permita a los usuarios la creación de paquetes con nuevas funcionalidades de manera que se logre hacer que la aplicación pueda realizar trabajos más específicos, y lograr por tanto que las funcionalidades que este NIDS proporciona, se adapten más a los trabajos concretos que se desean realizar.

2.4.1.3.3. Subsistema de logs y alertas

El subsistema de logs y de alertas es seleccionado al lanzar el programa, mediante modificadores que se incluyen en la línea de comandos. Hay tres opciones de log y cinco de alertas. Las tres formas de log son las siguientes, almacenar los paquetes de manera decodificada, almacenar los paquetes de una manera legible, y almacenar los paquetes en el formato que utiliza la aplicación Unix tcpdump. La forma decodificada de almacenamiento, permite un análisis rápido de los datos que se han recogido. Sin embargo, la tercera forma de almacenamiento, es mucho más rápida de guardar en disco y debe de ser usada en aquellos sistemas en los que se requiera un alto rendimiento del programa, o en aquellos sistemas que no tengan demasiados recursos. El log de las alertas, puede además ser eliminado por completo. Esta el la manera en la que este subsistema requiere menos recursos.

Las alertas, pueden ser enviadas al sistema de log de la maquina (syslog, sólo en máquinas UNIX) almacenadas en ficheros de log en dos formatos diferentes, enviadas mediante “WinPopup” (utilizando el cliente SMB proporcionado por Samba). Hay dos opciones para enviar las alertas a un fichero de texto plano, mandar las alertas por completo, o una manera rápida de almacenar esta alertas en el fichero de texto. Mediante el envío completo de la alerta, se escribe el mensaje asociado a ella y la información contenida en la cabecera del paquete . La forma rápida de almacenamiento escribe un conjunto condensado de información, por último también se puede deshabilitar por completo esta opción de almacenamiento de las alertas. Esto puede ser

útil en aquellos casos en los que el sistema vaya excesivamente cargado, o si se están desarrollando tests de intrusión en el sistemas.

2.4.2. Como escribir reglas en Snort

Como ya hemos mencionado en las secciones anteriores, Snort es un NIDS basado en reglas, las cuales marcan el funcionamiento del sistema, mediante los paquetes que han de ser tenidos en cuanta, y las acciones asociadas a estos paquetes.

En esta sección trataremos de describir como se escriben estás reglas para que sean interpretadas correctamente por el motor de detección de Snort.

Snort utiliza un lenguaje sencillo y ligero de descripción, el cual es flexible y suficientemente potente. Hay unas cuantas normas bastante simples que nos pueden servir como guía para el desarrollo de nuevas reglas.

La primera de ellas es que la descripción de cada regla ha de estar contenida en una única línea, dado que el parseo que Snort realiza para transformar los ficheros de reglas a su estructura de datos descrita anteriormente, se realiza línea a línea. Además como se ha mencionado anteriormente, para que cualquier modificación de las reglas tenga efecto, ha de relanzarse el programa.

Cada una de las reglas está dividida en dos secciones lógicas, la cabecera de la regla y sus opciones. La cabecera de la regla contiene la acción asociada a la regla, el protocolo la dirección I.P. fuente y destino así como las máscaras de red y los puertos fuente y destino del paquete. La parte de las opciones de la regla contiene los mensajes de alerta e información adicional en las que se le dice en que partes del paquete se ha de fijar, para determinar si la acción definida en la cabecera de la regla ha de ser tomada o no.

Aparte de las reglas, otra de las funcionalidades importantes que Snort nos permite, es la utilización de lo que se conoce como preprocesadores. Al final de esta sección veremos que son estos preprocesadores y cuales hay disponibles.

2.4.2.1. Cabecera de la regla

La cabecera de la regla ha de contener información relativa al paquete en lo referente a:

- Acciones asociadas.
- Protocolos.
- Direcciones IP.
- Números de puerto.
- Dirección de la operación.

2.4.2.1.1. Acciones

Le indican al motor de detección que ha de hacer cuando encuentre un paquete que coincida con el criterio de la regla. Snort proporciona tres posibles acciones:

- Alert. Genera una alerta utilizando el método de alerta seleccionado al invocar al programa desde la línea de comandos.
- Log. Archiva el paquete.
- Pass. Ignora el paquete

2.4.2.1.2. Protocolos

El siguiente campo que nos vamos a encontrar en una regla es el protocolo. Hay tres protocolos IP que Snort analiza para detectar un comportamiento sospechoso. Estos tres protocolos son TCP, UDP e ICMP.

2.4.2.1.4. Direcciones IP

La siguiente porción de la regla corresponde con los campos de la dirección IP e información relativa a los puertos origen y destino. Dentro del lenguaje de definición de las reglas, la palabra reservada **any** puede ser utilizada para referirse a cualquier dirección, sin especificar ninguna en concreto. Por desgracia, Snort no proporciona ningún mecanismo para traducir de dirección IP a nombre de equipo.

Las direcciones están formadas por una dirección IP junto con su bloque CIDR. Este bloque CIDR indica la mascara de red que debe de ser aplicada a la dirección IP de la regla, así como a cada paquete entrante que es comparado con este. Un bloque CIDR de

/24 indica que la dirección IP de la regla, pertenece a una red Clase C, /16 indica que pertenece a una red Clase B así como /32 indica una dirección IP de una máquina específica. Por ejemplo, la pareja dirección IP / bloque CIDR 155.54.1.0/24 indicará el rango de direcciones que van desde la dirección 155.54.1.1 hasta la dirección 155.54.1.255.

Además el lenguaje de definición de reglas de Snort, nos provee del operador de negación. Este operador indica al programa que aplica determinada regla para todas las direcciones que no coincidan con el rango que se le especifica. El operador de negación es **!**.

2.4.2.1.4. Números de puerto

Al igual que se le especifica la dirección fuente así como la dirección destino, también es necesario indicar el puerto fuente y el puerto destino. Este número de puerto puede ser definido de distintas formas, incluyendo **any**, definición de puertos estáticos, rangos y utilizando el operador de negación. Los puertos estáticos están indicados por un puerto en concreto, es decir un único valor numérico, y por último los rangos de puertos de definen con el operador **:**. Los rangos de puertos pueden ser utilizados de diversos modos teniendo significados diferentes:

- Cualquier puerto fuente y un rango en los puertos destino:

Log udp any any -> 155.54.12.0/24 1:1024

- Cualquier puerto fuente, y un puerto destino menor o igual a un puerto:

Log tcp any any ->155.54.12.0/24 :6000

- Puerto fuente mayor que un determinado número, y puerto destino menor que un determinado puerto:

Log tcp any :1024 -> 155.54.12.0/24 500:

Cualquier combinación de las formas de especificar los puertos anteriormente mencionadas, junto con la especificación estática de puertos, es válida. La negación de

un puerto se especifica mediante el operados **!**. Este operador puede ser aplicado contra cualquiera de las formas de especificar puertos anteriormente mencionadas, salvo aplicando en operador de negación a la palabra reservada **any**, puesto que en la definición de los puertos en la regla, ha de haber al menos un puerto al que la regla pueda ser aplicada.

2.4.2.1.5. Dirección de la operación

Este operador -> indica la orientación o dirección del tráfico al que la regla se aplica. La dirección IP y puerto de ambos lados de este operador indican que la regla se aplicara a cualquier paquete que tenga como origen el par IP puerto de la parte izquierda de la regla y como destino el par IP puerto de la parte derecha de la regla. Además de este operador unidireccional, también existe un operador bidireccional. Este operador bidireccional es <>. Este operador es útil para grabar o analizar ambas partes de la conversación. Un ejemplo en el que el operador bidireccional puede resultar útil, es para registrar todo el tráfico que se produzca por el protocolo *TELNET* desde fuera de la red que se pretende monitorizar. La regla para esto sería tal que:

Log tcp ¡155.54.12.0/24 any <> 155.54.12.0/24 23

2.4.2.2. Opciones de la regla

Estas opciones son la parte central del motor de detección que Snort proporciona. Cada una de las opciones que Snort proporciona han de estar separadas por un **;**, así como el identificador de la opción ha de estar separado de la opción en si por **:**. De esta forma, tenemos veinte opciones distintas que pueden ser aplicadas a una regla:

- **Msg**. Imprime un mensaje tanto el fichero de alertas como en el logs.
- **Logto**. Almacena el paquete en el fichero que el usuario le defina, en lugar de el fichero estándar.
- **Ttl**. Testa el valor del campo TTL de la cabecera del paquete IP.
- **Id**. Testa el valor del campo ID de la cabecera del paquete IP para ver si coincide con un valor determinado.
- **Dsize**. Testa el tamaño del campo de datos contra el de la cabecera IP.
- **Content**. Busca un determinado patrón dentro del campo del contenido.

- **Offset**. .Modificador de la opción anterior, establece el desplazamiento dentro del campo de contenido del paquete IP a partir del cual ha de buscar el patrón.
- **Depth**. Modificador de la opción contenido, establece la profundidad máxima a la que busca, dentro de una conversación entre varios ordenadores.
- **Nocase**. Establece la busca de patrones para que no sea sensible a las mayúsculas.
- **Flags**. Testa el valor de los flags TCP para ver si corresponden con cierto valor.
- **Seq**. Testa el número de secuencia del paquete TCP para ver si coincide con determinado valor.
- **Ack**. Testa el campo ACK del paquete TCP para ver si coincide con determinado valor.
- **Itype**. Testa el valor de este campo dentro del paquete ICMP para ver si coincide con un determinado valor.
- **Icode**. Testa el valor de este campo dentro del paquete ICMP para ver si coincide con un determinado valor.
- **Session**. Descarga la información del nivel de aplicación para una determinada sesión.
- **Icmp_id**. Testa el valor del campo ICMP ECHO ID para ver si coincide con un determinado valor.
- **Icmp_seq**. Testa el valor del número de secuencia de un paquete ICMP ECHO para ver si corresponde con un determinado valor.
- **Ipoption**. Busca dentro del campo de opciones de paquete IP.
- **Rpc**. Busca servicios RPC para una determinada llamada a una aplicación / procedimiento.
- **Resp**. Activa respuesta (caída de conexiones, etc).

Veamos cada una de las opciones mencionadas anteriormente con mayor detenimiento.

2.4.2.2.1. Msg

La opción msg dice al sistema de log y de alertas que un determinado mensaje ha de ser escrito con el volcado del paquete o con la alerta que se genera. Este mensaje, es un simple texto en el cual se utiliza el carácter \ delante de cualquier carácter que pueda provocar que Snort se confunda, por ejemplo **;**.

Formato

Msg: "<mensaje de texto>";

2.4.2.2.2. Logto

Esta opción le dice al programa que guarde en el fichero de log todos los paquetes que coincidan con esta regla a un fichero de salida especial. Esta opción puede ser útil en el caso que no se quieran mezclar cosas como la actividad NMAP, http, escaneos Cgi,... Hay que tener en cuenta que esta opción no funcionará si se ha seleccionado en la línea de comandos la opción de generar el fichero de log en formato binario.

Formato

Logto: "<nombre de fichero>";

2.4.2.2.3. TTL

Esta opción es usada para establecer valor específico del campo del paquete IP. Esta opción solamente se cumple si el valor que se le define en la opción es exactamente el mismo que el valor del campo TTL del paquete IP. Este opción es útil para la detección de internos de traceroute.

Formato

Ttl: "<número>";

2.4.2.2.4 ID

Esta opción es utilizada para buscar un número concreto de identificador de fragmento dentro del campo de la cabecera IP. Algunas herramientas para acceder a sistemas ponen este campo a un valor concreto para lograr diversos objetivos, por ejemplo, el valor 31337 es muy utilizado por algunos hackers.

Formato

Id: "<número>";

2.4.2.2.5. Dsize

Esta opción es utilizada para comprobar el tamaño del campo de datos. Puede ser utilizada para comprobar contra un valor concreto, o contra un valor mayor o menor que un número. Esta opción puede ser útil en caso de que dispongamos de un servicio con un buffer de un determinado tamaño. Además esta opción tiene la ventaja de ser mucho más rápida que el chequeo del contenido del campo de datos.

Formato

Dsize [>|<] <número>;

Los caracteres >< son opcionales

2.4.2.2.6. Content

Esta opción es una de las más importantes de Snort. Permite al usuario establecer reglas para la búsqueda de determinados contenidos dentro del campo de datos del paquete IP, y asociarle a la regla una acción en función de este contenido. La comprobación se realiza a lo largo de todo el campo de contenido del paquete. Si se encuentra este patrón, el resto de las opciones, en caso de haber alguna, se realiza. Esta comparación del contenido del paquete con el argumento de la opción es sensible a las mayúsculas.

Esta opción es un tanto complicada; puede contener una mezcla entre textos y datos en binario. Estos datos binarios, han de ser encerrados entre el carácter | y se representan como bytes en formato hexadecimal.

Formato

Content: "<cadena con el contenido>"

Ejemplo

Alert tcp any any -> 155.54.12.0/24 143 (content: "|90C8 C0FF FFFF|
/bin/sh"; msg: "Desbordamiento buffer IMAP";)

2.4.2.2.7. Offset

La opción de offset es usada para modificar la acción de la opción content. Esta opción cambia el comienzo de la posición a partir de la cual comienza la búsqueda de la

cadena dentro del campo de datos del paquete IP. Es muy útil para detectar cosas como detección de escaneos CGI, donde el contenido que se busca nunca es encontrado dentro de los primeros cuatro bytes del campo de datos. Esta opción no puede ser utilizada sin utilizar en la misma regla la opción content.

Formato

Offset: "<número>";

2.4.2.2.8. Depth

Depth es otra opción para modificar el comportamiento de la opción content. Esta opción establece la profundidad máxima hasta la que se busca el contenido del parámetro de la opción content, dentro del área de búsqueda. Es útil para especificar un máximo dentro del área de búsqueda, y evitar de este modo mayor ineficiencia en la búsqueda del patrón.

Formato

Depth: "<número>";

Ejemplo

Alert tcp any any -> 155.54.12.0/24 80 (content:"cgi-bin/phf"; offset: 3;
Depth: 22; msg: "Acceso a CGI-PHF";)

2.4.2.2.9. Nocase

La opción nocase es utilizada para desactivar la sensibilidad a las mayúsculas en la opción content. Esta se especifica sola dentro de la regla, y cualquier cadena ASCII que es comparada con el contenido del campo de datos es tratada sin tener en cuanta las mayúsculas o minúsculas.

Formato

Nocase;

2.4.2.2.10. Flags

Esta opción chequea el contenido de los flags en un paquete TCP para ver si coinciden con un determinado patrón. Actualmente el programa nos proporciona consultar los ocho flags de la cabecera TCP. Estos son:

- F. FIN (bit menos significativo del byte de flags)

- S. SYN
- R. RST
- P. PSH
- A. ACK
- U. URG
- 2. Segundo bit reservado
- 1. Primer bit reservado (bit más significativo del byte de flags)

Los bits reservados pueden ser utilizados para detectar un funcionamiento anormal. Para que la acción tenga lugar, todos los flags por los que se pregunta han de estar activos.

Formato

Flags: <flags activos>;

Ejemplo

Alert tcp any any -> 155.54.12.0/24 any (flags: SF; msg: "Posible escaneo SYN FIN";)

2.4.2.2.11. Seq

Esta opción se refiere a los números de secuencia TCP. Básicamente, detecta si el paquete tiene un número de secuencia concreto. De hecho esta opción no es demasiado útil, y únicamente se ha incluido en esta memoria por cuestiones de completitud.

Formato

Seq: <número>

2.4.2.2.12. Ack

La opción Ack se refiere al campo de confirmación de la cabecera TCP. Esta regla tiene únicamente un propósito práctico, como es la detección de pings NMAP TCP. Un ping NMAP TCP pone este campo a cero y manda un paquete con el flag TCP ACK activo, para determinar si un equipo esta activado.

Formato

Ack: <número>;

Ejemplo: detección de un ping NMAP TCP

Alert tcp any any -> 155.54.12.0/24 any (flags: A; ack: 0; msg: "ping NMAP TCP";)

2.4.2.2.13. Itype

Esta opción chequea el valor del campo type del paquete ICMP. El valor del parámetro que se le pasa puede no coincidir con ningún valor de un tipo válido, esto puede ocurrir en algunos tipos de ataque por denegación de servicio.

Formato

Itype: <número>

2.4.2.2.14. Icode

Esta opción es muy parecida a la comentada anteriormente, solamente establecer un valor numérico, y el programa detectará todo el tráfico ICMP que utilice ese valor como código. Igual que en la opción anterior se permite que se pase por parámetro un valor no válido del campo de código, con el fin de detectar tráfico sospechoso.

Formato

Icod: <número>

2.4.2.2.15. Session

Esta opción es utilizada para extraer los datos de usuario de una sesión TCP. Este campo es sumamente útil, para ver que usuarios están realizando un telnet, rlogin, ftp o incluso sesiones web. Hay dos argumentos disponibles para esta opción. Estos argumentos son **printable** o **all**. Si se pone el argumento *printable* en esta opción, únicamente se verán aquellos argumentos que el usuario se capaz de escribir o de ver. El atributo *all* sustituirá aquellos valores que el usuario no sea capaz de ver, por su valor hexadecimal. Esta opción puede provocar que el programa vaya demasiado lento, por tanto ha de ser utilizada con mucho cuidado.

Formato

Session: [printable | all];

2.4.2.2.16. Icmp_id

La opción icmp_id examina dentro de un paquete ICMP ECHO el contenido del campo ICMP ID para ver si coincide con un valor concreto. Es útil para la detección de bastantes ataques potenciales.

Formato

Icmp_ip: <número>;

2.4.2.2.17. Icmp_seq

Esta opción examina dentro de los paquetes ICMP Echo el contenido del número de secuencia ICMP. La utilidad de esta opción es prácticamente la misma que en la opción anterior. Es útil para detectar ciertas tramas de tráfico que pueden ser potencialmente un ataque.

Formato

Icmp_seq: <número>;

2.4.2.2.18. Ipoption

Si el campo de opciones IP está presente en el paquete, esta opción buscara si determinadas opciones, como *enrutaminento fuente* están activadas. Los valores válidos son:

- rr. Guardar ruta.
- eol. Fin de lista.
- nop. No operación.
- ts. Marca de tiempo.
- sec. Opción de seguridad IP.
- lsrr. Liberar enrutamiento fuente.
- ssrr. Enrutamiento fuente estricto.
- satid. Identificador de flujo.

Las opciones por las que habitualmente más se consultan, son las relacionadas con el enrutamiento fuente. Además solamente se puede consultar por una opción de la trama IP por cada regla.

Formato

Ipoption: <opciones de la lista anterior>;

2.4.2.2.19. RPC

Esta opción examina las peticiones RPC (Remote Procedure Call) y automáticamente decodifica la aplicación procedimiento y versión del programa, indicando un éxito cuando las tres variables han sido comparadas con éxito. El formato de esta opción es aplicación, procedimiento, versión. Se puede utilizar el carácter ***** para indicar cualquier procedimiento o versión.

Formato

*Rpc: <número, [número | *], [número | *]>;*

2.4.2.2.20. Resp

La palabra clave resp implementa la respuesta flexible a determinado tráfico. La respuesta flexible permite al programa cerrar determinadas conexiones consideradas peligrosas. Los siguientes argumentos son admitidos como parámetros:

- rst_snd. Envía un paquete TCP_RST al socket de envío.
- rst_rcv. Envía un paquete TCP_RST al socket de recepción.
- rst_all. Envía un paquete TCP_RST en ambas direcciones.
- icmp_net. Envía un paquete ICMP_NET_UNREACH al equipo emisor.
- icmp_host. Envía un paquete ICMP_HOST_UNREACH al equipo emisor.
- icmp_port. Envía un paquete ICMP_PORT_UNREACH al equipo emisor.
- icmp_all. Envía todos los paquetes anteriores al equipo emisor.

Estas opciones pueden ser combinadas para enviar múltiples respuestas a un determinado equipo. Cuando esto se hace estos argumentos han de estar separados por comas, y pueden ser de utilidad para que el NIDS corte el tráfico entre el atacante y la red. Por ejemplo, se podría insertar una regla que automáticamente cortase los accesos de un equipo externo de nuestra red a cualquier puerto 23 (el de telnet).

Formato

Resp: <argumento[, argumento...]

2.4.3. Prepocesadores

Los preprocesadores permiten a Snort, extender su funcionalidad mediante la utilización de plugins. Estos plugins o añadidos permiten a los usuarios y programadores extender la funcionalidad de Snort insertando módulos de una manera sencilla. El código de estos preprocesadores es llamado antes que el motor de detección sea iniciado, pero después de que el paquete haya sido codificado.

Para la carga y configuración de estos preprocesadores se utiliza la palabra clave **preprocessor** dentro de los ficheros que contiene las reglas. Esta directiva tiene el siguiente formato:

preprocessor <nombre>:<opciones>

2.4.3.1. Preprocesadores disponibles

2.4.3.1.1. minfrag

Este preprocesador examina los paquetes fragmentados hasta un determinado tamaño. Los paquetes son fragmentados, debido principalmente a la acción de los routers que el paquete se encuentra a través de su camino desde el quipo origen hasta el destino. Por norma general, no hay ningún equipo comercial que fragmente los paquetes en piezas más pequeñas de 512 bytes. Por tanto podemos utilizar este preprocesador para ver si hay tráfico de un tamaño menor a este número, puesto que esto puede indicar que hay alguien que está intentando ocultar su tráfico mediante la utilización de paquetes pequeños.

Formato

minfrag: <tamaño del paquete>

2.4.3.1.2. http decode

El decodificador http es utilizado para procesar la cadenas URI (Universal Resource Identifiquier) http, y convertir estas cadenas en una única cadena ASCII. El fragmentar estas URI's puede ser realizado para inutilizar la acción de escaneos de url y atacantes que de otro modo pueden eludir el análisis del contenido de la URL, fragmentando esta. Este preprocesador toma los números de puerto de los cuales ha de juntar las cadenas en una.

Formato

http_decode: <lista de puertos separados por espacios>

2.4.3.1.3. Detector de escaneos de puertos

Un escaneo de puertos se define como una conexión TCP que intenta acceder a más de P puertos en S segundos, o paquetes UDP que son enviados a más de P puertos en S segundos. Estos escaneos pueden ser realizados a numerosas direcciones IP destino así como a numerosos puertos a través de distintas direcciones IP. Por desgracia, este añadido a Snort únicamente permite la detección de escaneos de puertos desde una única máquina origen.

Los argumentos que se le pueden pasar a este módulo son los siguiente:

- Redes a monitorizar. La pareja dirección IP / bloque CIDR a monitorizar.
- Numero de puertos. Número de puertos accedidos en periodo de detección.
- Periodo de detección. Número de segundos.
- Directorio / fichero de almacenamiento. Donde se van a almacenar los resultados de estas detecciones.

Formato

Portscan: <red a monitorizar><numero de puertos><periodo de detección><directorio / fichero de almacenamiento>

2.4.3.1.4. Ignorar determinados host en los escaneos de puertos:

Otro de los módulos disponibles es útil para evitar que se disparen alarmas de escaneo de puertos que se suponen que se han lanzado desde determinados ordenadores. En el caso de que haya servidores que no se quiera que sean controlados por el preprocesador de escaneo de puertos (como pueden ser servidores NTP, NFS, y servidores DNS). Los argumentos que se le pasan a este preprocesador consisten en una lista de los pares dirección IP / bloque CIDR que han de ignorarse.

Formato

portscan-ignorehosts: <lista de ordenadores a ignorar>

2.4.3.1.5. Otros

Por último, comentar que al igual que hay preprocesadores para modificar o añadir funcionalidades a Snort, También existe módulos para modificar el funcionamiento del subsistema de log /alertas que hemos visto al principio de este módulo. Algunos de estos módulos permiten lo siguiente:

- Enviar las alertas a través de las interrupciones de log de los sistemas basados en UNIX, lo cual da mucha más flexibilidad que la opción, comentada cuando se trató el subsistema de almacenamiento, de el modificador por la línea de comandos cuando se llama al programa.
- Logear la salida de Snort en formato tcpdump, en el fichero que nosotros decidamos
- Logear la salida de Snort a una base de datos. Esto puede ser de utilidad para la generación de informes y estadísticas acerca de los ataques que se han producido en nuestra red, así como para el almacenamiento de datos históricos.
- Almacenamiento de los datos en formato XML, de manera que posteriormente se puedan tratar estos datos por distintas aplicaciones.

2.4.2. NMAP

Nmap es un escáner de puertos. Un escáner es útil para el administrador para supervisar los servicios activos en las máquinas a su cargo, y así saber limitarlos a los imprescindibles.

Se ha utilizado esta herramienta en los estadios iniciales de este proyecto, para entre otras cosas, comprobar si el preprocesador utilizado para detectar los escaneos de puertos funcionaba de manera correcta, así como para ver los puertos que dejábamos abiertos en el equipo que es utilizado como sensor.

Esta herramienta, es capaz de realizar también la detección del sistema operativo que se está ejecutando en el equipo al que se realiza el escaneo de los puertos, así como realizar distintos tipos de escaneos para detectar vulnerabilidades en el equipo. Basándose en estos parámetros, es capaz a su vez de dar una estimación de la protección del equipo escaneado.

Un ejemplo de la ejecución de esta herramienta es el siguiente:

Nmap <nombre o dirección del equipo>

Starting nmap V. 2.54BETA22 (www.insecure.org/nmap/)
Interesting ports on (*dirección del equipo*):
(The 1530 ports scanned but not shown below are in state: closed)

Port	State	Service
21/tcp	open	ftp
23/tcp	open	telnet
25/tcp	open	smtp
79/tcp	open	finger
98/tcp	open	linuxconf
111/tcp	open	sunrpc
113/tcp	open	auth
513/tcp	open	login
514/tcp	open	shell
515/tcp	open	printer
959/tcp	open	unknown
1024/tcp	open	kdm

Figura 6 *.Ejemplo máquina desprotegida, con puertos vulnerables.*

2.4.3. Ethereal

Ethereal, es un analizador de protocolos, que permite examinar los datos que son capturados por la tarjeta de red, bien en tiempo real, bien desde un fichero que se ha guardado a disco. Esto nos permite analizar los paquetes que Snort ha capturado, puesto

que si se almacenan los paquetes capturados por Snort en formato tcpdump, entonces esta herramienta nos permite un análisis a bajo nivel de estos paquetes.

Esta herramienta dispone además de un interface gráfico, el cual facilita el análisis de los paquetes capturados.

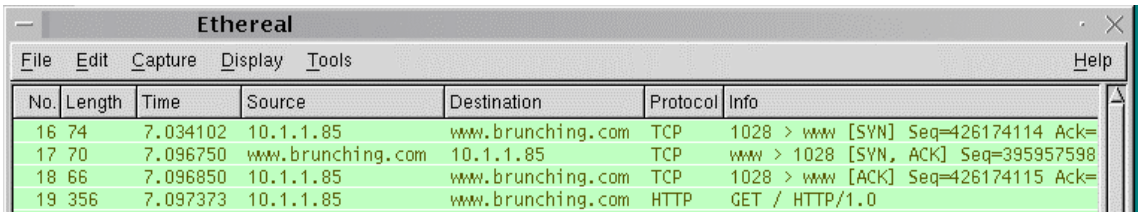


***Figura 7**. Ejemplo de la ventana de Ethereal*

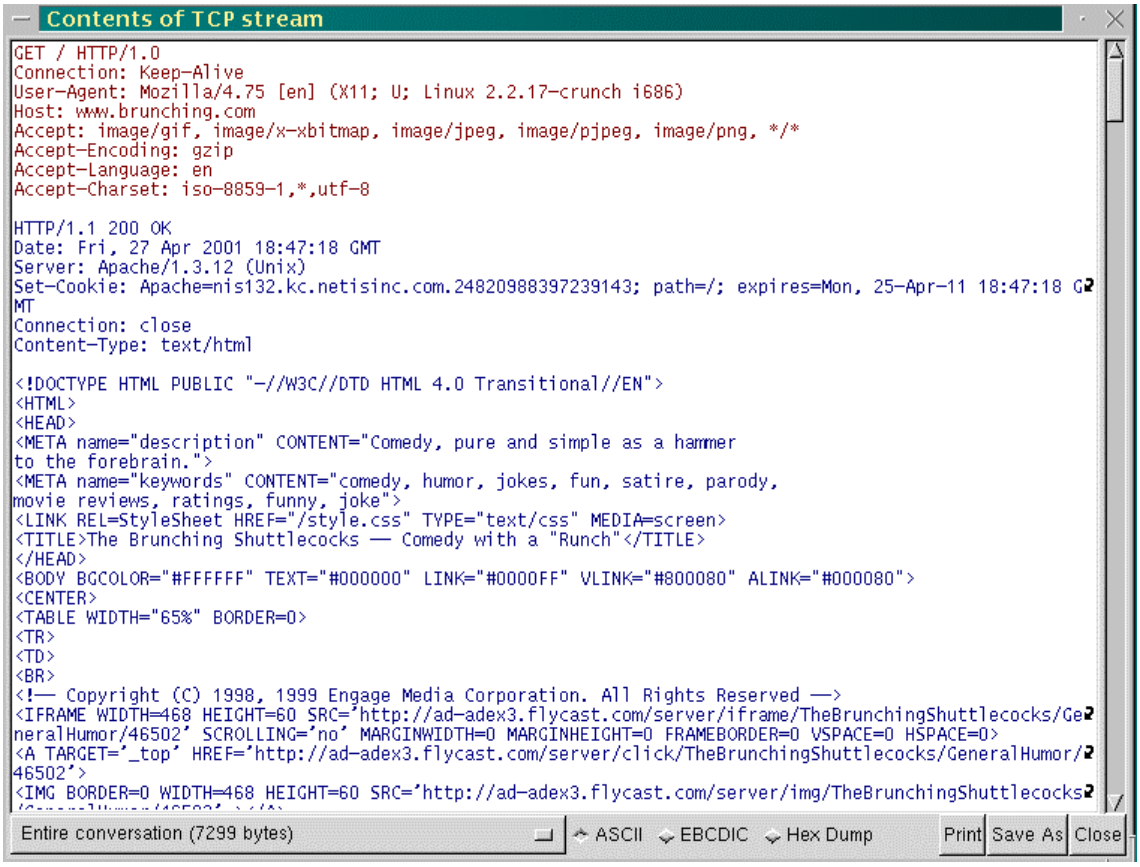


***Figura 8**. Detalle de uno de los paquetes capturados.*

2.4.4. Aris

Aris es un analizador de logs para tratar los ficheros que han sido generados por el sistema de detección de intrusos en red. Suple las deficiencias que se comentaron anteriormente en cuanto a los reportes que los NIDS ofrecen sobre las incidencias que se han registrado en las redes monitorizadas.

Aris (Attack Registry and Intelligence Service), es capaz de realizar análisis en los ficheros generados por la mayoría de los NIDS que actualmente se encuentran disponibles en el mercado.

Esta herramienta, consta de dos sistemas distintos, uno para el envío de los ficheros generados, y otra para el análisis de los mismos.

En cuanto a la primera de ellas, es un programa que se ejecuta en el equipo local, el cual, transforma la salida de los ficheros que se van a transmitir para su posterior análisis en un formato intermedio, que será el que realmente se transmite. Este formato, no es más que un formato en XML, que posteriormente será utilizado para el análisis de las incidencias notificadas.

Respecto a esta parte, es de notar que la aplicación permite ser configurada para que se realice este envío de los ficheros de manera periódica, de forma que el sistema automáticamente enviaría los ficheros para su posterior análisis.

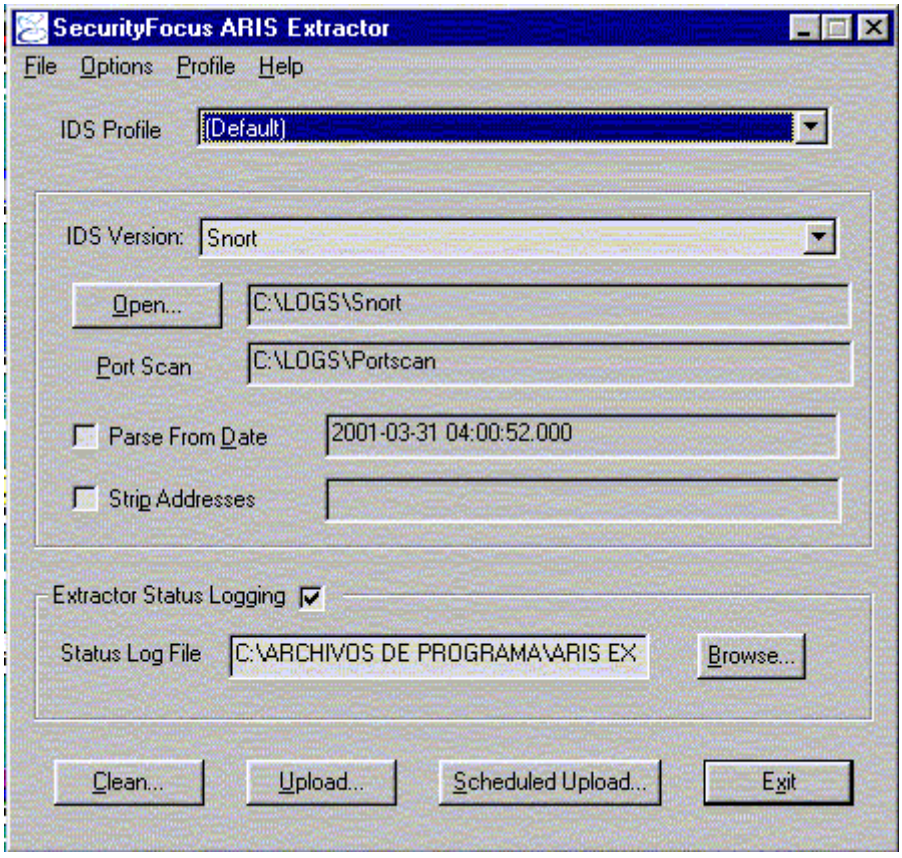


***Figura 9**. Aplicación cliente de Aris.*

Antes de realizar el envío de datos, es necesaria la creación de una cuenta en el sistema remoto. Al crear la cuenta, se le dan al sistema una serie de datos, entre los que figuran un identificador de usuario y una clave de acceso. En ese momento ya se puede realizar el envío de información para su análisis.

En lo que respecta el sistema remoto, al cual nos conectamos a través de un navegador, y una vez enviados los datos, se nos da la posibilidad de ver información sobre los incidentes que se han notificado, así como la posibilidad de realizar ciertos gráficos a partir de los mismos.

3. Diseño de una solución concreta

En esta sección vamos a tratar el diseño de una solución que cumpla con los objetivos del presente proyecto. En primer lugar, veremos como es la traza de un ataque típico a cualquiera de los ordenadores o servidores que utilizan la Red de Transmisión de datos de la Facultad de Informática.

Este ataque suele ser muy similar al ataque típico descrito en la introducción de la presente memoria:

- El atacante lanza un escaneo de puertos a una máquina concreto, o a un rango de máquinas (las direcciones de todos los ordenadores de la Universidad de Murcia, están comprendidos entre la dirección I.P. 155.54.0.0 y la 155.54.255.255, que se corresponde con una clase B). También es posible que el atacante pretenda hacer un escaneo a un solo puerto de un rango de máquinas para averiguar en cuales de ellas ese servicio está activo.

- Una vez que ya se ha visto contra que máquina proceder, el atacante utiliza un programa para ver si mediante ese servicio que se ha buscado, o sobre ese rango de servicios activos en una máquina en concreto, es posible aprovechar alguna vulnerabilidad conocida, utilizando para ello un programa o exploit. Mediante el cual lograría acceder a la máquina con ciertos privilegios.

- Una vez hecho esto, el atacante extraerá información de esta máquina (si este era su propósito) o utilizará esta máquina para atacar a otras.

El efecto que este ataque provocaría sobre un sistema típico, sería muy difícil de detectar, puesto que lo que únicamente provocarían, serían una vez que el atacante ha logrado acceder, la inserción de algunas líneas en los ficheros de configuración del sistema, para poder facilitar futuros accesos por parte del atacante, o la inserción de otros tantas líneas en los ficheros de log de la máquina. Estos ficheros de log, contienen determinada información sobre lo que ha ocurrido o está ocurriendo en el sistema, como

puede ser los últimos comandos ejecutados, las últimas personas que accedieron a determinados recursos, etc.

Por tanto, y dado que estos efectos son muy difíciles de detectar, y mucho menos de lograr llevar un control centralizado sobre dichos ficheros, ya que estos están ubicados en cada uno de los ordenadores, y si bien es posible el envío de estos ficheros a los administradores del sistema, también es cierto que estos ficheros contienen tanta información que realmente su análisis es inviable. Por tanto la reacción ante un ataque puede llegar a ser demasiado lenta para que se logre evitar efectos indeseados y dista mucho de ser inmediata o cuanto menos rápida.

Lo que se propone pues, es implantar un sistema automatizado que permita a los administradores reaccionar en un tiempo admisible (idealmente en tiempo real) ante las alertas que se produzcan en los equipos de la Facultad de Informática. De modo que los administradores del sistema podrán ser capaces de comprobar si el servicio que está a la escucha en el / los puertos sobre los cuales se ha producido un escaneo o un ataque son seguros.

Cabe la posibilidad de que si se generan alarmas en cada escaneo de puertos que se ha producido, la cantidad de alarmas sea tan grande que al final se acabe por ignorarlas. Por ello, las alarmas serán enviadas a los administradores de la red, una vez que se haya detectado un intento de acceso no autorizado. A pesar de ello, la reacción no será mucho más tardía y se podrán tomar las acciones correctoras a tiempo. Además un fichero con todos los escaneos de puertos que se hayan producido en ese día seguirá estando accesible para su consulta e investigación.

Así pues, con el sistema propuesto en este proyecto conseguimos una mayor rapidez y agilidad de respuesta por parte de los administradores.

La reacción ante un posible ataque siguiendo el esquema más usual, puede ser la siguiente,

- El atacante lanza un escaneo de puertos a una de las máquinas que se están monitorizando (dicho escaneo queda registrado en un fichero que contiene todos los escaneos registrados durante ese día)

- El atacante ejecuta un exploit para comprobar si la vulnerabilidad existe. En este momento se genera una alerta que puede ser consultada por el administrador del sistema, el cual puede comprobar si esa vulnerabilidad afecta al equipo atacado.

- En caso que esa vulnerabilidad afecte al equipo atacado, el administrador del sistema, tomará las acciones adecuadas para contrarrestar los efectos de dicho ataque.

Como se puede comprobar el tiempo de respuesta ante un posible ataque se reduce drásticamente, así como se puede evitar rápidamente los efectos de los mismo. A pesar de lo anterior el tiempo de respuesta ante un ataque queda supeditado a la disponibilidad de los administradores del sistema para tomar las medidas correctoras, puesto que no se da un servicio *24/7* a los equipos de la Facultad de Informática.

3.1. Solución elegida

Finalmente expondremos la solución que hemos tomado para la realización de este proyecto.

Uno de los requisitos para la realización de este proyecto, es que la solución que se tome no requiera un gasto en cuanto a licencias software y en cuanto a la adquisición de equipos necesarios para la realización de las mismas. Por tanto, se ha decidido la utilización de software de libre distribución, y que este software no necesite un equipo con muchos recursos, dado que era conveniente la reutilización de alguno de los equipos que teníamos a nuestra disposición.

Por tanto la solución elegida se basa en la utilización de un sensor (equipo que lleve a cabo la Detección de Intrusos en Red) en el cual se está ejecutando un software de libre distribución. En este caso el software que se está ejecutando en Snort en su versión 1.6. Existe una versión superior de Snort, la versión 1.8, esta versión ofrece mas funcionalidades que la versión 1.6, pero ha sido desarrollada y ofrecida en Agosto de 2001, por tanto no hemos realizado experiencias con ella. Además, el formato de las

reglas, cambia totalmente, por tanto, sería necesario el desarrollo de una base de reglas nueva, o la adaptación de la que actualmente está en uso.

3.1.1. Disposición del sensor dentro de la red

Veamos ahora como se ha dispuesto este sensor dentro de la topología de la Red de Transmisión de datos de la Facultad de Informática descrita en el punto 2 de la presente memoria.

Esta disposición se muestra en la figura 10. En esta figura se muestra lo siguiente:

- La topología de Red de Transmisión de datos de la Facultad de Informática
- La disposición de nuestro sensor dentro de la misma red.
- Como se comunica nuestro sensor con el exterior, así como los mirroring necesarios en el router para que el sensor sea capaz de recibir los datos a analizar para generar las alarmas correspondientes a los ataques que se produzcan.

Dentro de la figura 10 las líneas de color lila, representan los enlaces de fibra para la comunicación entre los conmutadores y el router. Las líneas de color verde, no se corresponden con líneas de comunicación en si, si no que representan un mirroring que se ha tenido que hacer a nivel de router para que nuestro sensor tenga datos que analizar.

Este mirroring únicamente consiste en la replicación de todo el tráfico que tiene como origen o destino uno de los puntos del router que corresponde con alguno de los conmutadores donde se conecta algún equipo cuyo tráfico estamos interesados en analizar.

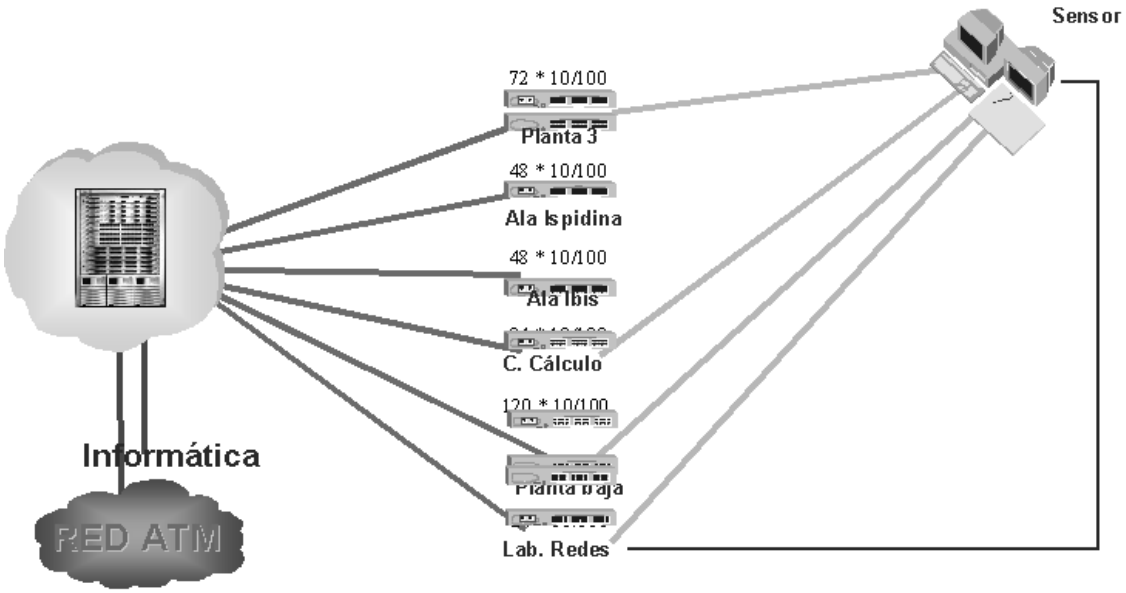


Figura 10 *Disposición del sensor dentro de la Red de la Facultad de Informática*

3.1.2. Sensor

En este apartado nos centraremos en la configuración del equipo en el que se ha realizado este proyecto, viendo como el equipo cumple con uno de los objetivos del presente proyecto como era el bajo costo.

3.1.2.1. Software

En cuanto al software que se encuentra instalado en el equipo, todas las aplicaciones que se han instalado son de libre distribución y ya han sido mencionadas en el segundo apartado de la presente memoria. Veamos cada una de ellas

- Sistema operativo. El sistema operativo que hay instalado en la máquina es Linux, con la versión del núcleo 2.2.15-2.0smp. Esta versión de núcleo, permite el aprovechamiento de los dos microprocesadores que esta máquina tiene instalados. Además ha sido configurado para dejar la menor cantidad de puertos posibles abiertos, de manera que se puedan utilizar la menor cantidad de exploits y otros tipos de ataque contra esta máquina.
- Snort. Como ya hemos comentado ampliamente a lo largo de la presente memoria, ha sido el NIDS que se ha utilizado para la realización de este proyecto, en su versión 1.6.

- SSH2. SSH2 ha sido instalado para permitir el acceso remoto de esta máquina. Además al ser una aplicación que codifica la comunicación, permite que no se puedan realizar ataques a ella, mediante la instalación de un sniffer en el mismo segmento de red (puesto que el punto de red en el cual está conectada la tarjeta de red que se utiliza para las comunicaciones de este equipo, está conectado a un concentrador)
- Ethereal. Como ya se comento anteriormente, esta aplicación está instalada para poder realizar un análisis de los paquetes que Snort guarda como resultado de su funcionamiento.
- Nmap. Utilizado para realizar escaneos de puertos y, comprobar si el módulo de detección de escaneos de puertos en Snort funciona correctamente, así como para ver que puertos están a la escucha en algunas máquinas.
- ArisExtractor. Es la herramienta que se utiliza para enviar los logs obtenidos a la herramienta de análisis anteriormente citada.

Como se ha comentado anteriormente, todo el software que se ha instalado es de libre distribución, bajo licencia G.N.U.

3.1.2.2 Hardware

En cuanto al hardware que hay disponible, este es el siguiente:

- Placa base Pentium Dual
- Dos Procesadores Pentium 100 MHz.
- Dos tarjetas de red Fast Ethernet 100 Mbps.

El motivo de tener dos tarjetas de red, es para intentar dificultar que la ejecución del Snort sea detectable, así como para evitar que el tráfico de gestión del equipo, interfiera con la captura de datos.

Estas tarjetas de red están configuradas de manera que una de ellas es utilizada para la comunicación normal del sistema, de manera que es la que se utiliza cuando se accede a la máquina remotamente utilizando SSH2, o cuando se está trabajando normalmente en ella. En cambio, la configuración del otro interface de red es distinta, puesto que esta tarjeta no tiene dirección IP, en cambio, está en modo de operación

promiscuo, y es la tarjeta que recibe los datos a analizar que le envía el router. Esta tarjeta, si que está conectada directamente a un conmutador.

3.2. Resultados obtenidos

A continuación mostraremos los resultados obtenidos en estas experiencias, mostrando una serie de gráficos, que surgen del análisis de los logs generados.

Dado que Snort no está dotado de una herramienta de generación de gráficos, los gráficos que a continuación mostraremos, han sido obtenidos mediante ARIS.

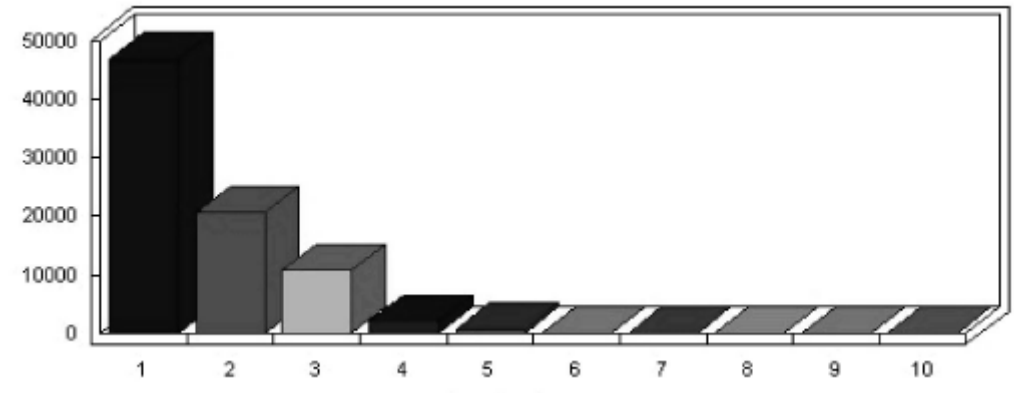
Para la interpretación de estos logs, es necesario hacer una distinción entre los dos tipos de incidentes que podemos notificar. Estos dos tipos de incidentes, se dividen en ataques y pruebas.

Los ataques consisten en la utilización de determinados exploits, u otros métodos, mediante los cuales se pretende entrar en los equipos que se están monitorizando. Por desgracia, no somos capaces saber si este ataque ha tenido éxito o no, pero si que podemos ver cuales son las vulnerabilidades

Por el contrario, las pruebas consisten en los escaneos de puertos que se han realizado a equipos de nuestra red, así como la utilización de otro tipo de programas y procedimientos que tienen por objetivo ver si una determinada vulnerabilidad se presenta en determinada máquina.

3.2.1. Número de ataques por categorías

En la figura 11, podemos apreciar que desde que el sistema fue puesto en funcionamiento (Enero de 2001) hasta el mes de Agosto de este año, el sistema ha detectado un total de 81.085 posibles ataques. En este número hay que tener en cuenta que no se han incluido las pruebas, puesto que en ese caso, el número sería mucho mayor, al incluir también los escaneos de puerto y otras pruebas detectadas. En concreto este número subiría hasta 293.113 ataques y pruebas detectadas.



1	Backdoor/Trojan Attacks	46880
2	Remote Services Attacks	20615
3	WWW Attacks	10733
4	SMB/NetBIOS Attacks	2092
5	FTP Attacks	626
6	Network File System Attacks	75
7	Manipulation/Spoofing	45
8	DoS	19
Total		81085

Figura 11. *Número total de ataques por categorías.*

Analizando este gráfico, nos damos cuenta que la mayoría de lo que el sistema detecta como ataques son puertas traseras dejadas abiertas por el atacante o troyanos. Estos programas están diseñados para dejar nuevos servicios a la escucha, y mediante un programa cliente, acceder al equipo donde se ha empleado el programa servidor, que suele llegar mediante ficheros anexados a correos electrónicos.

Este tipo de ataques tiene un alto índice de falsas alarmas, puesto que se detecta mediante el puerto destino, por tanto, puede haber gran cantidad de tráfico que no se corresponda con un ataque, pero que al ir dirigido al puerto donde se cree que está ese programa a la escucha, se dispara la alarma.

Otra categoría de la que se han detectado una gran cantidad de ataques, es la de ataques remotos a servicios. Los ataques incluidos en esta categoría, son en su mayoría aquellos que detectan en el contenido del paquete, cierto contenido que se corresponde con un determinado código ensamblador de la máquina a la que vayan dirigidos. La cantidad de falsos positivos que se generan en este tipo de alarmas es bastante menos.

La tercera categoría más numerosa, son los ataques utilizando el protocolo WWW. En esta categoría se incluyen aquellos ataques que intentan explotar los errores de

instalación de los servidores http, como puede ser incluir en la URL algunos **..** para ver si se logra acceder a otros directorios, o como es el intentar explotar ciertos fallos bien conocidos en algunos de los CGI's que estos servidores traen por defecto, o la inserción de una línea concreta pasada como parámetro.

3.2.2. Ataques más frecuentes

Veamos a continuación aquellos ataques que se han cometido con más frecuencia, viendo como las categorías que anteriormente han sido detectadas como las utilizadas más frecuentemente, corresponde con este análisis.

En la figura 12, se aprecia como el ataque mas habitual se corresponde con intentos de desbordamiento de buffer en ordenadores de la familia x86. Estos ataques son incluidos en la anterior categoría de ataque remoto a servicios.

Esta circunstancia, si vemos con detenimiento la figura anterior, es bastante previsible, puesto que bajo la categoría de troyanos y puertas traseras se incluyen gran cantidad de distintos tipos de ataque, bajo el ataque más frecuente, se ve que la inmensa mayoría de los ataques remotos a servicios se corresponde con un único tipo de ataque.

Otra conclusión que se desprenden de la comparación entre las figuras anteriores, es que, algo más del 76% del total de los ataques se corresponden con 10 tipos concretos, por tanto, y a pesar de que muchos de ellos, son susceptibles de la generación de falsas alarmas, podríamos reducir en gran número los intentos de ataque.

Veamos a continuación de una manera más detallada, alguno de estos ataques más frecuente.

El primero de ellos, se dispara, cuando dentro del contenido del paquete se incluye el código hexadecimal 0x90. Este código se corresponde con la instrucción NOP de el ensamblador de los microprocesadores basados en tecnología x86 y es lanzada por el sistema cuando se detectan dentro del contenido de un mismo paquete la aparición de varios de estos códigos. Es fácil detectar si esta alarma es falsa o no, puesto que viendo las trazas que el sistema ha guardado, se puede ver si puede tratarse de una ataque,

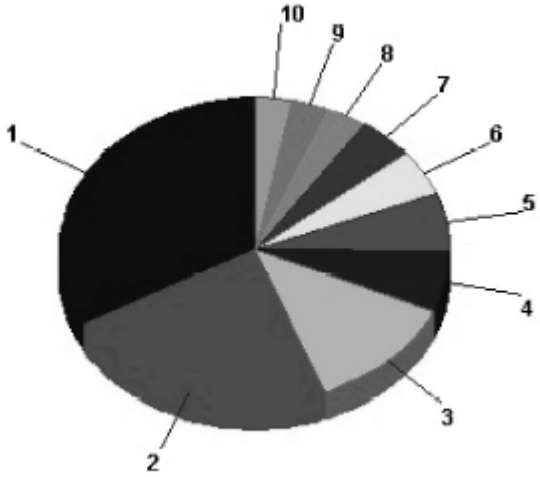
puesto que cuando este código se incluye, suele ir junto con una gran cantidad de este mismo tipo de instrucciones ensamblador.

La detección por parte del sistema de este tipo de ataques se produce mediante el análisis del contenido de los paquetes IP, dado que hay una regla en la cual se le dice que dispare una alarma cuando esto ocurre. La regla que dispara esta acción es:

alert udp any any -> $INTERNAL any (msg: "x86-nops-udp"; content: "|90 9090 90 90 90 90 90 90 90 90 90 90 90 90|";)

Donde por $INTERNAL considera cualquier paquete con destino a la red que se está monitorizando.

Como este, hay diversos ataques de este tipo, cuya diferencia de uno a otro radica en el contenido del paquete y el puerto al que van dirigidos.



	1	Generic X86 Buffer Overflow Attack	20482
	2	Shivka-Burka Backdoor Server Attack	14068
	3	Muhammad A. Muquit Count.cgi Attack	7688
	4	VooDoo Doll Remote Connection Attack	4116
	5	PsyberStream Backdoor Server Attack	3768
	6	GateCrasher Backdoor Server Attack	3010
	7	Ultor Server Backdoor Attack	2929
	8	SGI IRIX cgi-bin wrap Attack	2020
	9	FTP99cmp Backdoor Server Attack	1953
	10	Wincrash Trojan Remote Communication Attack	1868
		Total	61902

***Figura 12**. Ataques más frecuentes*

Los siguientes 6 ataques más frecuentes, corresponden en realidad a la misma categoría, es decir, todos ellos se basan en lo mismo, diferenciándose del puerto al que hacen la conexión. Estos ataques provocan un gran número de falsas alarma.

Para ver esto, veamos uno de ellos, el más numeroso. Este ataque se basa en un programa servidor que se instala en la máquina atacada. Este programa, instala un servicio que está a la escucha en el puerto 1.600.

El sistema disparará una alarma, cuando detecte que por el puerto 1.600 se está estableciendo una conexión mediante TCP. La regla que dispara la alarma sería:

Alert tcp any any -> $INTERNAL 1600 (msg: "*Shivka-Burka backdoor")*

La forma de detectar si es una falsa alarma, es ver que equipo es y si tiene algún servicio legítimo a la escucha en ese puerto. Todos los ataques de este tipo, actúan de forma similar, y habitualmente en lo único que se diferencian es en las posibilidades del servicio que está instalado, así como el puerto en el que están a la escucha. De igual forma su detección puede ser un poco más compleja, si se llega a tener acceso al código fuente de este programa servidor, puesto que de esta manera se puede cambiar el puerto en el cual el servicio se queda a la escucha.

Por último veamos un ataque por explotación de CGI´s, de los cuales se han registrado 2.020 incidencias. Este ataque consiste en hacer un telnet al servidor web, intentando ejecutar lo siguiente:

GET /cgi-bin/handler/<lo que sea>;cat /etc/passwd| ?data=Download HTTP/1.0

O incluyendo esta línea directamente en el navegador. Esta línea no tiene la intención de acceder a ningún fichero CGI, se intenta que se ejecute la instrucción cat /etc/passwd. Este ataque afecta especialmente a los servidores que tienen instalado un sistema operativo IRIX en determinadas versiones. Pero el principal interés de estos ataques, en lo que respecta a la respuesta del sistema, es que aquí se ve un ejemplo claro de la utilidad de uno de los preprocesadores explicados en el punto 2.4. de esta memoria,

como es el http-decode. Sin el uso de este preprocesador, simplemente haciendo que el tamaño del paquete sea muy pequeño, sería imposible que este ataque fuera detectado.

La reacción del sistema ante este tipo de ataque es la siguiente, primero ensambla localmente todo lo que llega por los puertos que se le indica, que han de ser aquellos en los que hay un servidor web, y una vez que ha detectado el final de la URL, pasa a comparar el contenido de esta cadena que ha ensamblado con la que queramos que salte la alarma. En este caso concreto, la regla sería:

alarm tcp any any -> $INTERNAL 80 (msg: "Ataque por IRIX CGI"; content : "/cgi-bin/handler/";)

El número de falsos positivos que esta alarma dispara, es bastante bajo. Pero ese tipo de ataque es fácilmente evitable. Para ello, solo es necesario instalar un parche al servidor web de este sistema operativo.

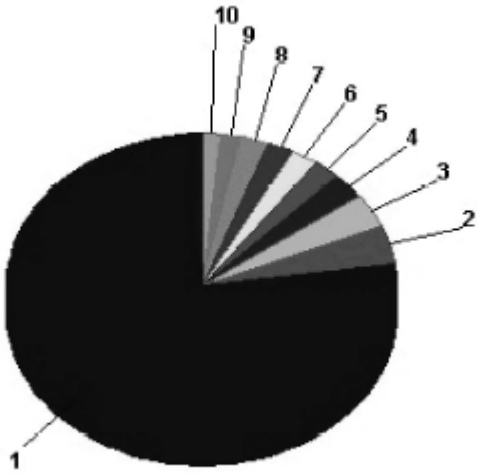
3.2.3 Origen de los ataques

Una vez que hemos visto el número de ataques que encontramos en nuestra red, así como cuales de ellos son los más frecuentes, veamos ahora de donde provienen estos ataques.

En la figura 13, vemos los diez proveedores de servicio a Internet desde los que se han producido un mayor número de ataques, así como el numero total de ataques que se han realizado desde cada uno de ellos.

Como se puede apreciar en la figura 13, el mayor número de ataques provienen de la propia Universidad de Murcia.

No es raro que el mayor número de ataque vengan de dentro de la red que estamos monitorizando, a pesar de las falsas alarmas que se puedan disparar, pero lo que si que puede causar un poco de extrañeza es que haya un número tan grande de accesos desde Corea. Esta circunstancia, es atribuible a que sean utilizados sistemas de este país como puente para realizar ataques a otros equipos.



■	1	UNIVERSIDAD DE MURCIA	52704
■	2	Telefonica Data Espana (NCC#2000013794)	2782
■	3	KRNIC	2505
■	4	KOREA TELECOM	1814
■	5	Texas Unwired Networ	1573
■	6	Finsys, Inc	1514
■	7	Chunghwa Telecom Data communication Business Group	1478
■	8	KRNIC	1458
■	9	Tatung Company	1308
■	10	Demon Internet Inc.	1006
		Total	68142

***Figura 13**. ISP más frecuentemente utilizados*

4. Conclusiones y vías futuras

En los apartados anteriores, hemos expuesto la utilidad de los Sistemas de Detección de Intrusos, y en particular de los Sistemas de Detección de Intrusos en Red, para lograr una mayor seguridad en las redes que estos sistemas están monitorizando.

En este apartado veremos como realmente sirven para detectar tanto las vulnerabilidades de nuestra red como los ataque producidos.

A su vez también expondremos los pasos que en nuestra opinión, se deberían de dar a partir de este proyecto.

4.1. Conclusiones

En primer lugar veremos como se han cubierto los objetivos de los que partimos en el presente proyecto.

Estos objetivos eran:

- Lograr detectar ataques sin tener que configurar nada en los equipos finales y de usuarios. Este objetivo se ha cubierto plenamente como se ha expuesto en el apartado tercero de esta memoria. Este punto ha sido cubierto mediante la instalación y configuración de un sensor, que recibía los datos que debían de ser analizados.
- Generar alertas de los ataques que se produzcan para poder ser enviados a los administradores de los sistemas. Este punto ha sido cubierto de dos formas distintas. La primera de ellas es la generación de una serie de ficheros producidos por el sistema en el cual se almacenan las alertas generadas. A su vez, también ha sido cubierto mediante la utilización de ARIS, una herramienta muy útil para la administración y análisis de estas alertas.
- Evaluar herramientas para poder llevar esto a cabo. Mediante la comparación de los posibles sistemas disponible, realizada en el segundo apartado de esta memoria, y en función de los recursos disponibles, se han evaluado y elegido las herramientas necesarias para desarrollar este proyecto.
- Llevar a cabo el desarrollo de este proyecto, con el máximo de recursos disponible, si que se requiera invertir en el desarrollo de ellas. Este punto se

ha logrado mediante la reutilización de hardware anteriormente utilizado, y en algunas ocasiones, incluso no en uso en el momento en el que este proyecto comenzó, así como mediante la utilización de herramientas de libre distribución, lo cual ha permitido la elaboración de este proyecto sin haber ocasionado ningún coste.

Una vez que ya hemos visto que se ha logrado cumplir con los objetivos del presente proyecto, veamos ahora los resultados obtenidos.

En primer lugar, y como vimos en el apartado 3.2 de esta memoria, se ha logrado detectar un gran número de ataques. A pesar de que este número puede verse reducido por las falsas alertas que el sistema haya disparado, la necesidad de implementar de una forma más profunda algún sistema de protección de la Red de Transmisión de Datos de la Facultad de Informática puede apreciarse claramente.

Además también se aprecia que la mayor parte los ataques que se han registrado, se concentran en algunos procedimientos muy definidos, en concreto, el 76% de los ataques registrados, pertenecen a diez tipos concretos de ataque. Esto no quiere decir que únicamente haya que concentrarse en lograr que los equipos sean inmunes a estos tipos de ataque, aunque esto sea cierto, sino que esto ayudará también a que se detecten otros tipos menos frecuentes, pero no por ello menos dañinos, que también se dan en la Red.

Así por ejemplo algunos tipos de ataques que no aparecen reflejados en las estadísticas, como conexiones al puerto 111/PORTMAP de los servidores, intentos de envíos de correo SPAM, no aparecen en estas estadísticas, debido a que el router de la Universidad de Murcia está filtrando este tipo de tráfico.

Otro dato que se ha logrado ver con muchísima claridad, es que el mayor número de ataques, poco más del 64% de los mismos, se producen desde dentro de la misma red que se está monitorizando. Esto no ha de causar extrañeza, puesto que la gran mayoría de los ataques en cualquier entorno, suelen provenir de personas cercanas al mismo. Incluso si volvemos a mirar los pasos en los que consiste un ataque típico, veremos que lo más normal, es que la persona intente lograr accesos a otros sistemas, es para tratar de

atacar a un tercer sistema, sin que sea personalmente identificable, así como es detectar que nuestros equipos han sido atacados, cuando recibimos avisos de los administradores de otros sistemas.

Además de esto, en la Universidad confluye un aspecto más, como es la juventud del personal que tiene acceso a los ordenadores, la mayoría de ellos estudiantes, acostumbrados a este tipo de tecnología y con ganas de probar cosas que posiblemente hayan encontrado en alguna página web, puesto que Internet es una gran fuente de información a la hora de tratar de proteger los equipos de una red, pero por desgracia esta información no siempre es leída con un buen propósito.

4.2 Vías Futuras

Las vías futuras para la continuación de este proyecto se enmarcan en el ámbito de la implementación dentro del seno Universitario de sistemas que sean capaces de detectar ataques, utilizando para ello algún sistema similar al aquí expuesto, pero con mayores capacidades de procesamiento de información así como de comunicaciones, de manera que sea posible la implementación de un sistema de Detección de Intrusos centralizado, no ya solo para la Red de Datos de la Facultad de Informática, sino para toda la red de la Universidad de Murcia, en el cual se pueda mejorar un poco la gestión de las alarmas disparadas, puesto que de esta forma, se lograría reducir su número.

Para ello sería necesario una gestión más eficaz de estas alarmas, así como la implementación del sistema en un equipo más potente, puesto que esto permitiría la elaboración de reglas más complejas, que permitan un mejor análisis del tráfico, sin causar una degradación en el rendimiento de la sistema.

Respecto a lo anterior, cabe destacar que RedIRIS, a comenzado el desarrollo de un proyecto destinado a cubrir el párrafo anterior.

También seria interesante la realización de experiencias destinadas a minimizar uno de los aspecto negativos de los Sistemas de Detección de Intrusos en Red, como es la falta de información sobre lo que realmente está ocurriendo en la máquina atacada, puesto que la única fuente de información disponible es la que circula por la red.

El punto anterior ha sido cubierto mediante la realización de un Proyecto de Fin de Carrera en la Facultad de Informática de la Universidad de Murcia, acerca de los Honey Points. Esto es, sistemas que a los ojos de un atacante parezca una máquina normal pero que ofrece alguna facilidad para ser atacada, de manera, que se pueda vigilar en esa máquina lo que el atacante realiza.

5. Bibliografía

"Intrusion Detection"
Rebecca Gurley Bace
Editorial MacMillan Technical Publishing
isbm 1-57870-185-6

“IDS Review”
Autor: Dragos Ruiu
http://www.securityportal.com/articles/idsintroduction20010226.printerfriendly.html

“How To Guide-Implementing a Network Based Intrusion Detection System”
Autor: Brian Laing
http://www.snort.org/docs/iss-placement.pdf

“Linux Intrusion Detection”
Autor: Joe Brockmeier
http://www.samag.com/extra/poster.htm

“BlackICE Sentry”
Autor: NetworkICE
http://www.networkice.com/products/blackice_sentry.html

“About NFR”
Autor: NFR Systems
http://www.nfr.com/about/

“Prodcuts”
Autor: Intrusion.com
http://www.intrusion.com/product/product.asp?lngProdNmId=28

“Snort - Lightweight Intrusion Detection for Networks“
Autor: Martin Roesch
http://www.snort.org/docs/lisapaper.txt

“The Ethereal User Guide”
Autores: Richard Sharpe y Ed Warnicke
http://www.ns.aus.com/ethereal/user-guide/book1.html

“Attack Registry Intelligence Service”
Autor: ARIS Analyzer
http://aris.securityfocus.com

“Writing Snort Rules. How to write Snort rules and keep your sanity”
Autor: Martin Roesch
http://www.clark.net/~roesch/snort_rules.html

“Improving security”
http://www.cert.org/tech_tips/

"CERT Summaries"
http://www.cert.org/summaries/

Lista de Seguridad Bugtrack
listserv@lists.securityfocus.com