# CAPITULO 3

# IDS

En este tercer capítulo realizaremos un estudio sobre los sistemas de detección de intrusos o IDS (*Intrusion Detection System*). Diferenciaremos entre los distintos tipos de sistemas existentes y nos centraremos en los sistemas de detección de intrusos para redes de ordenadores o NIDS (*Network IDS*).

Se analizarán los diferentes componentes, arquitecturas y configuraciones que forman los sistemas NIDS. También se comentarán los diferentes protocolos y paradigmas standard que existen en la actualidad para la comunicación entre los distintos componentes de sistemas IDS.

Describiremos las distintas técnicas de análisis utilizadas sobre datos obtenidos por los distintos componentes del IDS así como los efectos derivados de la detección automática de intrusos en redes de ordenadores. Se introducirán los conceptos de falsos positivos y falsos negativos.

También se enumerarán los principales inconvenientes y limitaciones que presenta la utilización de sistemas IDS/NIDS así como las futuras líneas que pueden ir marcando la evolución de los NIDS, centrándonos en los sistemas de prevención o IPS (*Intrusion Prevention System*).

Finalmente comentaremos un ataque típico y clásico en la bibliografía de los sistemas de detección, el ataque del famoso hacker Kevin Mitnick.

# 3.1 Firewalls

Durante mucho tiempo el mecanismo de seguridad en redes más extendido ha sido únicamente el uso de un *firewall*. Este sistema nos permite de una manera simple y eficaz aplicar filtros tanto para el tráfico de entrada como para el de salida en nuestra red.

Podemos diferenciar entre dos políticas básicas de configuración de firewalls [Nor99]:

- **Permisividad máxima** (*allow everything*) dónde el uso de filtros es mínimo o inexistente. Esta política permite prácticamente la circulación de todo el tráfico y se utiliza principalmente en Intranets/LAN, campus universitarios y organizaciones dónde la libertad de uso de aplicaciones (o la gran cantidad de ellas) es necesaria para el funcionamiento ordinario del sistema.

  Es una política que dificulta enormemente el uso de otros sistemas y deja a la red muy vulnerable a prácticamente cualquier tipo de ataque interno o externo.

En estos casos se recomienda segmentar la red en dominios y acotar cada uno de estos dominios, ya que raramente todos los ordenadores tienen que acceder a todos los recursos disponibles de la red.

- **Permisividad mínima** (*deny everything*) aplica la política contraria a la anterior. En este caso se deniega acceso a todos los servicios de la red y se van permitiendo accesos a estos a medida que se necesiten.

  De esta forma es bastante improbable que recibamos un ataque a un servicio que desconocíamos que teníamos en la red. Por otro lado, el trabajo de otros sistemas se facilita enormemente ya que pueden configurarse para que detecten fácilmente cualquier comportamiento anómalo en la red (simplemente se debe monitorizar los accesos a los servicios y comprobar si esos accesos están permitido expresamente o no).

  Cabe notar que este tipo de política requiere un gran esfuerzo ya que es poco flexible y en organizaciones con gran cantidad de usuarios con diferentes requerimientos puede llevar a tener que permitir tantos accesos cruzados que deje de ser práctico.

Destacar que el simple uso de un firewall puede crear una falsa sensación de seguridad que de nada sirve si no son configurados y "mantenidos al día" (aplicación de los *parches*/*patches* del fabricante, supervisión y adaptación al tráfico de la red...).

Muchas organizaciones con cientos de ordenadores y decenas de firewalls no disponen de una sola persona cualificada asignada exclusivamente a su mantenimiento!!

Por otro lado, este tipo de sistemas son incapaces de detectar tipos de ataques más sofisticados (DOS por ejemplo), lo que hace necesario la adopción de otros sistemas de control para completar la seguridad en nuestra red.

## 3.2 Historia de los IDS

La historia sobre los sistemas de detección de intrusos en ordenadores empieza en 1972 cuando James P. Anderson de las fuerzas aéreas norteamericanas (USAF) publica un texto sobre la seguridad en los ordenadores [Bru01].

Este tema empieza a cobrar fuerza paralelamente al desarrollo de la informática, puesto que cada vez hay mas procesos "críticos" controlados por ordenadores y los militares temen cualquier cosa que no controlan.

Algunos estudios se realizan durante los años siguientes hasta que en 1980 James P. Anderson escribe **"Computer Security Threat Monitoring and Surveillance"** [And80], dónde se inician las bases de la detección de intrusos en sistemas de computadores principalmente mediante la consultas de ficheros de log.

Entre 1984 y 1996, Dening y Neummann desarrollan el primer modelo de IDS denominado IDES (Intrusion *Detection Expert System*) basado en reglas. A partir de este momento, se han ido proponiendo y creando nuevos sistemas de detección de intrusos [MC01] hasta obtener una separación clara entre los sistemas que efectúan la detección dentro de los ordenadores y aquellos que la efectúan en el tráfico que circula por la red.

## 3.3 IDS

Podemos definir la detección de intrusos (*Intrusion Detection* o ID) como *"un modelo de seguridad aplicable tanto a ordenadores como a redes. Un sistema* ***IDS*** *recolecta y analiza información procedente de distintas áreas de un ordenador o red de ordenadores con el objetivo de identificar posibles fallos de seguridad. Este análisis en busca de intrusiones incluye tanto los posibles ataques externos (desde fuera de nuestra organización) como los internos (debidos a un uso abusivo o fraudulento de los recursos).*

*Los sistemas IDS suelen utilizar técnicas de análisis de vulnerabilidades (a veces referenciado en bibliografía como scanning), es decir examinan todos nuestros sistemas en búsqueda de alguna vulnerabilidad."* [WWW59]

Un sistema de detección de intrusos o **IDS** (*Intrusion Detection System*) es un paradigma que por su naturaleza intrínseca de supervisión de los recursos es aplicado tanto a ordenadores como a redes de computadores [And80]:

- En el caso de los ordenadores se realiza a nivel de sistema operativo para controlar los accesos de los usuarios, modificación de ficheros del sistema, uso de recursos (CPU, memoria, red, disco...) con el objetivo de detectar cualquier comportamiento anómalo que pueda ser indicativo de un abuso del sistema.

  En la bibliografía [Fran02] a veces pueden encontrarse como **HIDS** (*Host Intrusion Detection System*).

  Un ejemplo de aplicación de este tipo de herramientas en sistemas operativos de ordenadores puede ser el conocido software TRIPWIRE. [WWW60]

- En el caso de redes de ordenadores pueden monitorizarse usos de anchos de banda, accesos a/desde direcciones no permitidas, uso de direcciones falsas... con el objetivo de encontrar un comportamiento anómalo o atípico en el tráfico de la red supervisada que nos pueda indicar una posible anomalía.

  También pueden referenciarse en la bibliografía como **NIDS** (*Network Intrusion Detection System*) [Gra00][HFC02][WWW94].

En este capítulo nos centraremos en los sistemas de detección de intrusos (IDS) aplicados a redes de ordenadores (NIDS).

Todo y que el concepto de IDS engloba el de NIDS, no toda la bibliografía acepta esta segunda diferenciación dentro de los sistemas de detección de intrusos, con lo que nosotros utilizaremos IDS o NIDS indistintamente,

## 3.3.1 Monitorización de redes en tiempo real

Realizar la supervisión de una red de comunicaciones implica necesariamente realizar un estudio detallado y pormenorizado de todo el tráfico que circula por esta. Si hacemos unos simples cálculos podemos observar que resulta del todo inviable (por no escribir imposible) filtrar toda la información que circula por nuestra red en tiempo real.

Para demostrar esta afirmación, realizaremos un sencillo pero gráfico estudio sobre los estándares de las redes mas populares actualmente (tanto LAN como WAN) y el tráfico que generan en un día completo[ℜ].

Para poder realizar este cálculo de forma completa, necesitamos también conocer el tamaño de los datagramas IP que circularán por la red. Como este tamaño es imposible de calcular debido a que un datagrama IP tiene un tamaño máximo de 64K, realizamos una estimación real a partir del tráfico de Internet. Según el estudio de Kc Claffy y Sean McCreary [CMC00] podemos asumir un tamaño medio típico para los datagramas IP que circulan por Internet es de 1500 Bytes (ver figura 3-1).

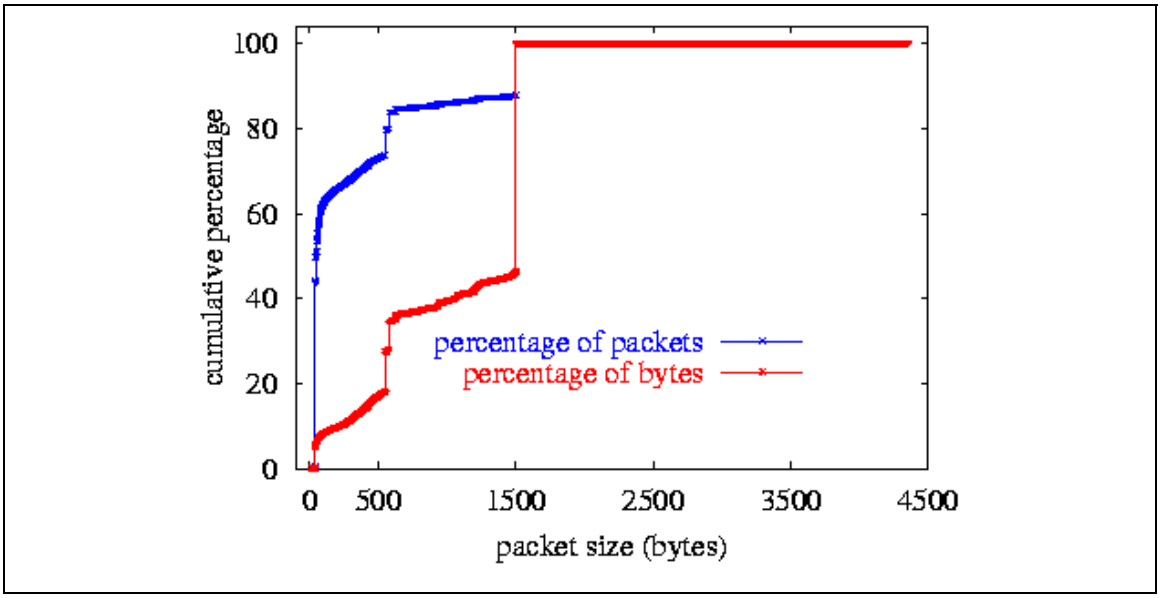


FIG 3-1: Porcentaje acumulativo de datagramas IP en INTERNET en el año 2000.

[ℜ] 1 Día = 24 Horas = 24 * 60 minutos = 24 * 60 * 60 segundos = **86.400** Segundos.

En este análisis consideraremos que todo el tráfico generado en estas redes es tráfico IP a monitorizar, cosa que no es cierta puesto que los datagramas IP van encapsulados en *frames* (capa 2 del modelo OSI) que dependen de la tecnología que use la red.

Sin embargo, esta asunción nos dará una cota superior de la capacidad necesaria para procesar todos los paquetes IP, ya que también asumimos como inexistente el tiempo de análisis de cada paquete (nuestro objetivo es saber que potencia necesitamos para el proceso en tiempo real).

De esta forma la asunción de que todo el tráfico es IP queda nivelada con la asunción de que el tiempo de proceso de cada paquete que circule por la red es cero (ver la figura 3-2).

| Tecnología<br><br>**T** | Espacio de disco<br>Cantidad de información transmitida por día.<br><br>EDD = (T * 86400 s) / 1 GByte | Potencia de procesador<br>Número de paquetes por día.<br><br>**PP** = EDD / 1500 Bytes |
|---|---|---|
| ADSL (256Kbits/s) | (32768 Bytes/s * 86400 s) / 1Gbyte = **2,6 Gbytes** | **1.887.436** |
| ADSL (2Mbit/s) | **21,09 Gbytes** | **15.099.494** |
| LAN (10 Mbit/s) | **105,4 Gbytes** | **75.497.472** |
| LAN (100 Mbit/s) | **1054,6 Gbytes** | **754.974.720** |
| WAN ATM (155 Mbit/s) | **1634,7 Gbytes** | **1.170.210.816** |
| LAN (1 Gbit/s) | **10546,8 Gbytes** | **7.549.747.200** |

FIG. 3-2: Anchos de banda y tamaño diario de las redes más comunes.

Por otro lado, los procesadores más potentes en el momento de realizar este trabajo funcionan a 3GHz. Con lo que necesitaríamos procesar cada paquete **en un ciclo de reloj** (cosa totalmente imposible) ya que necesitaríamos un ciclo de reloj para recibir el paquete, uno para procesarlo y otro para retransmitirlo si fuera necesario y no quisiéramos eliminarlo.

Los ordenadores actuales acceden a memoria externa y disco cuyas velocidades distan mucho de los 3GHz del procesador. Por otro lado, con las tecnologías superiores al Gigabit (que ya existen actualmente) como la fibra OC48 (2.48Gbits/s), esto queda aún mas lejos.

Como se ha demostrado anteriormente, el filtrado de todo el tráfico generado por una red es imposible en tiempo real (tanto en potencia de proceso CPU como en espacio de almacenamiento), lo que implica que en caso de realizarse este proceso debería ser a posteriori (*off-line*) con lo que su utilidad baja mucho.

Si bien es cierto que es muy importante conocer que se ha recibido un intento de ataque, conocerlo días o semanas después no sirve como argumento ante los consejos de dirección o superiores.

Cabe notar que el almacenamiento de esta información es **básico** para realizar análisis forense (*computer forense*), sin embargo nuestro objetivo (y por lo tanto el de este estudio) no debe ser el de examinar los registros y logs cuando nuestro sistema ya ha sido comprometido, seducido y abandonado por el hacker de turno.

## 3.3.2 Signatures (Firmas)

Sin embargo, en cualquiera de los paquetes que circulan por la red puede encontrarse un intento de ataque, un ataque en toda regla o incluso el paseo triunfal de un hacker que ya ha conseguido entrar en algún sistema. ¿Qué podemos hacer?

Definiremos una **firma** (*signature*) como *"aquello que define o describe un patrón de interés en el tráfico de nuestra red."* [Nor99]

Análogamente, diremos que un **filtro** (*filter*) *"es la trascripción de una firma (signature) a un lenguaje compresible por los sensores que monitorizan nuestra red."* [Nor99]

Las firmas (*signatures*) nos permiten diferenciar entre todo el tráfico generado por la redes y obtener un subconjunto de este lo suficientemente pequeño como para que sea tratable computacionalmente y lo suficientemente amplio como para poder detectar comportamientos anómalos en tiempo real (o casi).

Las firmas utilizadas en IDS usualmente son simples patrones que permiten detectar ataques ya conocidos (*Smurf*, *Land*...). Su funcionamiento es el mismo que el de los antivirus, se basan en encontrar coincidencias de ataques ya conocidos en el tráfico actual de la red.

Algunos productos IDS permiten la creación de filtros por el usuario (ver figura 3-3), lo que facilita la adaptación del sistema a nuestras necesidades. Sin embargo, realizar filtros útiles necesita al menos de:

- Que el administrador de seguridad esté cualificado (conozca perfectamente el protocolo IP y su propia red).

- Que el IDS proporcione un lenguaje suficientemente potente como para poder expresar nuestras necesidades.

- Que los filtros (tanto los creados por defecto como los personalizados) sean revisados/modificados/ampliados frecuentemente.

```
Filter pptp ip()
  {
      # El ataque "Land" se basa en dar como dirección de origen y destino la misma IP

       if (ip.src == ip.dest)
          {

              # Guardamos la hora y direcciones MAC e IP de origen y destino

              record system.time, eth.src, ip.src, eth.dst, ip.dst to land_recdr;
          }
  }
```

FIG 3-3: Ejemplo de un filtro para el ataque *Land*.

La capacidad de crear filtros permite una gran flexibilidad y potencia en la detección de tráfico anómalo. Un ejemplo de esto podrían ser los filtros que se realizan sobre un protocolo en concreto (por ejemplo en HTTP) y que permiten incluso guardar log de las conexiones que envían/reciben determinada información (como por ejemplo los filtros contra pornografía).

**Hay que tener en cuenta que este tipo de filtros pueden atentar contra la intimidad de las personas y que deben ser conformes a la ley vigente (LORTAD en España).**

Obviamente, como pasa en los antivirus, la actualización de las firmas debe ser constante ya que usualmente un nuevo tipo de ataque no será detectado por el sistema IDS que simplemente se dedique a buscar patrones en el tráfico de la red.

El uso de un firewall bien configurado (nos limitará la cantidad de tráfico a procesar) así como la personalización adecuada de los filtros aplicados en la red, nos dotarán de un sistema seguro y útil en el que podemos confiar [WWW66].

## 3.3.3 Eventos de interés (EOI)

Una vez demostrado que no es práctico realizar el filtrado de todo el tráfico producido en una red, debemos centrarnos en ir refinando los resultados obtenidos para conseguir un conjunto razonablemente pequeño de muestras que nos permita el proceso en tiempo real.

Definiremos los **eventos de interés** (*Events Of Onterest, EOI*) como el subconjunto mínimo de muestras que debemos analizar para considerar nuestra red "segura" o como el subconjunto de los genuinos positivos verdaderos (*genuine non-false positives*) [Nor99].

Este subconjunto se obtiene a partir del resultado de aplicar los filtros, firmas y reglas personalizadas del sistema de detección de intrusos al tráfico total (ver figura 3-4).

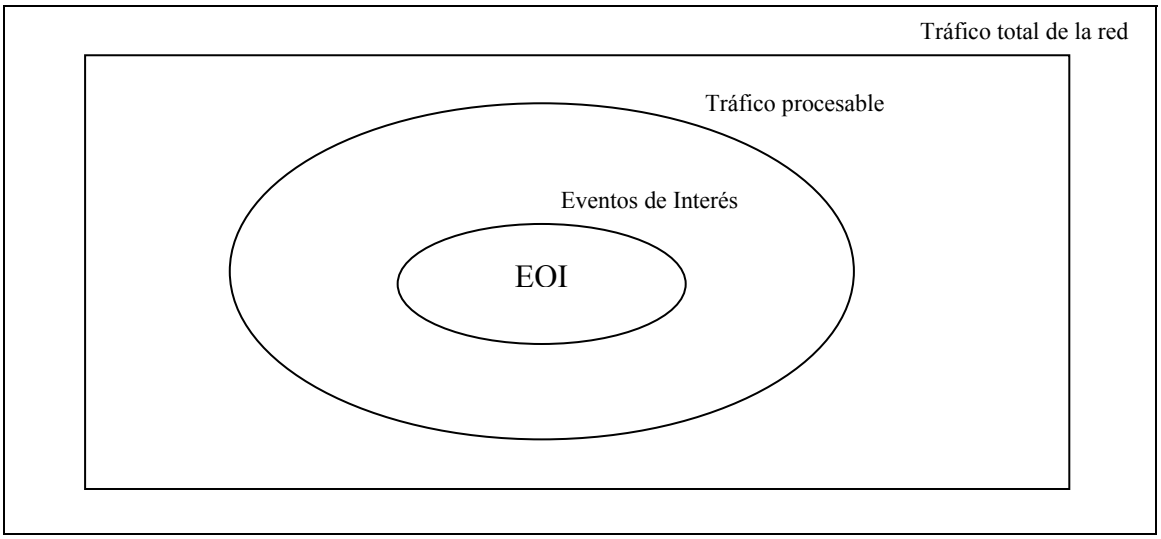


FIG 3-4: Eventos de interés (EOI).

Muchos estudios proponen una fórmula de evaluación de riesgos para los eventos de interés detectados (EOI) y que nos puede ayudar a cuantificar numéricamente la gravedad del evento detectado [Car00][Car01][Nor99] [WWW69]. Los aspectos a tener en cuenta en esta fórmula son (ver figura 3-5):

1. **Criticidad** (*Criticality*). No todos los ordenadores tienen la misma función. De esta forma un ataque a un servidor DNS, a un firewall o a un PC Cliente se valoran de distinta forma.

2. **Letalidad** (*Lethality*). Los diferentes ataques (*exploits*) no tienen siempre el mismo objetivo. Dependiendo de si simplemente tiene posibilidades de funcionar en algún sistema no parcheado a si permite bloquear la máquina (un ataque de tipo DOS por ejemplo) o ganar acceso como usuario simple o root, evaluaremos el ataque.

3. **Contramedidas** (*countermeasures*). Una vez detectado un ataque (o un inicio de posible ataque), nuestra capacidad de respuesta es otro factor a tener en cuenta y que pondera en la fórmula (de aquí la importancia de IDS en tiempo real). Las contramedidas a aplicar pueden ser de sistema (bloquear una cuenta de usuario, parar un servicio temporalmente o incluso apagar la máquina) o de red

(interceptar las comunicaciones desde una dirección determinada, ajustar anchos de banda de entrada y salida de la red...)

**Severity** = (Criticality + Lethality) – (System countermeasures + network countermeasures)

FIG 3-5: Fórmula de evaluación de la gravedad de un EOI.

De esta forma, de todos los eventos de interés detectados por son sensores podemos obtener una representación numérica en la consola que nos facilite de una forma rápida un orden de prioridad.

## 3.4 Arquitecturas de los NIDS

Según la aproximación utilizada para la monitorización del tráfico de la red, en la bibliografía [Tim01] se agrupa los NIDS en tres grupos distintos:

a) **Signature based NIDS**: Al igual que muchos antivirus, estos IDS se basan en la búsqueda de patrones conocidos (firmas) en el tráfico de la red.

b) **Anomaly based NIDS**: Se basan en analizar el tráfico de la red creando estadísticas y asignándoles pesos. Cuando se detecta tráficos sospechoso, se confronta con las estadísticas anteriores y en función del peso asignado y la cantidad de ocurrencias del evento se dispara una alarma o no.

c) **Protocol modeling NIDS**: Estos sistemas de detección de intrusos buscan paquetes que contengan anomalías o configuraciones poco comunes de los protocolos de red (a fin de cuentas, los datos van encapsulados en distintos datagramas de distintos protocolos).

Existen dos componentes básicos para cualquier sistema de detección de intrusos (IDS) que son los sensores y la consola (ver figura 3-6):

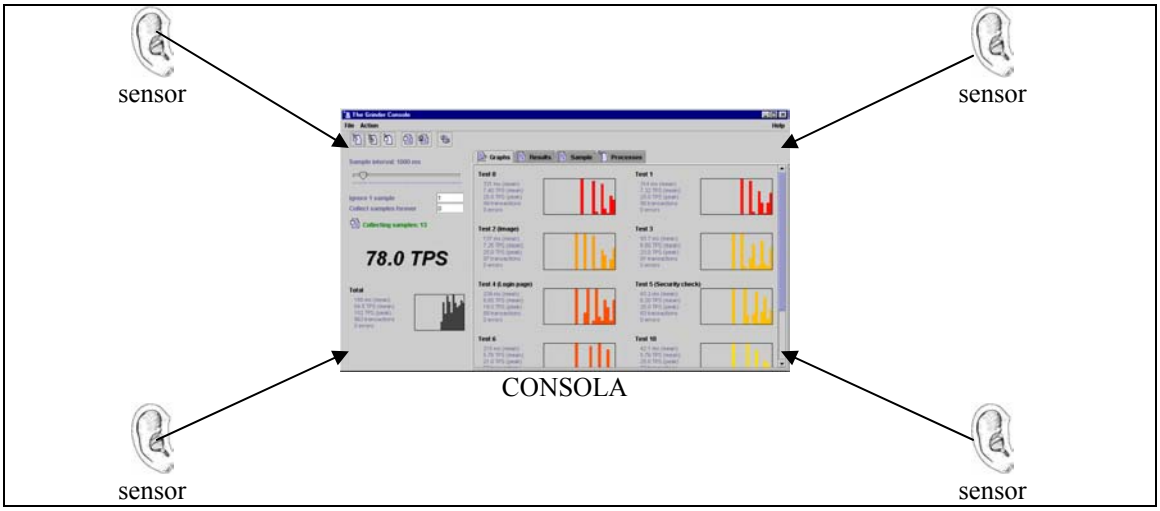


FIG 3-6: Elementos básicos de un IDS.

1. Los **sensores** (*sensors*) de un IDS son elementos pasivos que examinan todo el tráfico de su segmento de red en búsqueda de eventos de interés. Dependiendo del paradigma que utilicen para comunicarse con la consola del sistema detector de intrusos pueden clasificarse en dos grupos [Nor99][WWW72][WWW73]:

   *push*: Cuando se detecta un evento de interés el sensor crea un paquete de datos que envía a la consola. Un protocolo muy utilizado con este tipo de sensores es el SNMP que permite la definición de *traps* para el envío de información a un receptor (la consola en este caso).

   Su principal inconveniente es que pueden ser observador por el atacante para descubrir cómo ha sido configurado y ante que tipo de patrones reacciona, lo que permite establecer que patrones ignora el sensor y por tanto cuales emplear en un ataque.

*pull*: Almacenan los eventos de interés hasta que la consola pregunta por ellos al sensor. Si se establece un protocolo de intercambio de mensajes cifrado y se pregunta regularmente a los sensores puede ofrecer un mecanismo eficaz de comunicación con la consola.

Para evitar susceptibilidades, en caso de que el sensor no detecte eventos incluirá una cadena de caracteres aleatoria para evitar enviar siempre el mismo paquete de cuando no existen eventos detectados.

2. La **consola** (*console*) de un sistema de detección de intrusos se encarga de recibir toda la información de los sensores (ya sea mediante "*pull*" o "*push*") y presentarla de forma entendedora al operador. Desde la consola se pueden configurar los distintos sensores así como emprender acciones en respuesta a los datos recibidos de los sensores.

## 3.4.1 CIDF

El **CIDF** (*Common Intrusion Detection Framework*) [WWW79] es un proyecto iniciado a finales de la década de los noventa por la oficina de información y tecnología (*Information technology Office*) del DARPA [WWW80].

Inicialmente se enmarcó como parte de programa de vigilancia de la información de redes de ordenadores, pero progresivamente, conforme los sistemas de detección de intrusos han alcanzado suficiente importancia, se convirtió en una rama independiente apoyada por otros organismos internacionales como el grupo IDWG (*Intrussion Detection Working Group / Exchange Format*) de IETF [WWW81].

Su objetivo es el de crear interfaces de aplicaciones (API) y protocolos que permitan la comunicación entre los diferentes IDS con el objetivo de reaprovecharlos para otros sistemas [Lee00][Ruo00][Wan00].

CIDF (ver figura 3-7) nos propone una serie de estructuras que denomina cajas (*boxes*) que encarnan cada una de las distintas funciones que deberían realizar los sistemas IDS.

De esta forma, simplemente define las funcionalidades genéricas deseadas y sus interconexiones, dejando abierto a cada fabricante la implementación final de estas. Los componentes principales son:

- **Event generator** *(E boxes):* Estas cajas describen la función de los sensores del sistema de detección de intrusos. Su función es recoger información de la red y generar informes/alertas para la consola de monitorización.

- **Analysis** *(A boxes):* Son las encargadas de procesar la información obtenida de los sensores y realizar su análisis. Se admite como ampliación de su funcionalidad [Nor99] la capacidad de realizar recomendaciones al operador e incluso de actuar proactivamente.

- **Database** *(D boxes):* Esta entidad simboliza la base de datos (o base de conocimiento) dónde se almacenan los informes, firmas y patrones detectados en la red por el IDS. El objetivo de mantener esta información es doble:

    1. Obtener la posibilidad de generar informes históricos y permitir el uso de técnicas de *Business Intelligence* y *Data Warehouse* (BI/DW) para la extracción de información y obtención de nuevo conocimiento.

    2. Proactividad del sistema. Conseguir una respuesta rápida del IDS ante un ataque nuevo consultando otros ataques similares registrados.

- **Response** *(R boxes):* Este elemento es el encargado de proporcionar una respuesta ante los eventos obtenidos de las otras cajas (*E/A/D Boxes*) o sugerir acciones al operador de consola (bloquear el acceso desde una dirección IP, limitar el numero de conexiones nuevas aceptadas por segundo....).

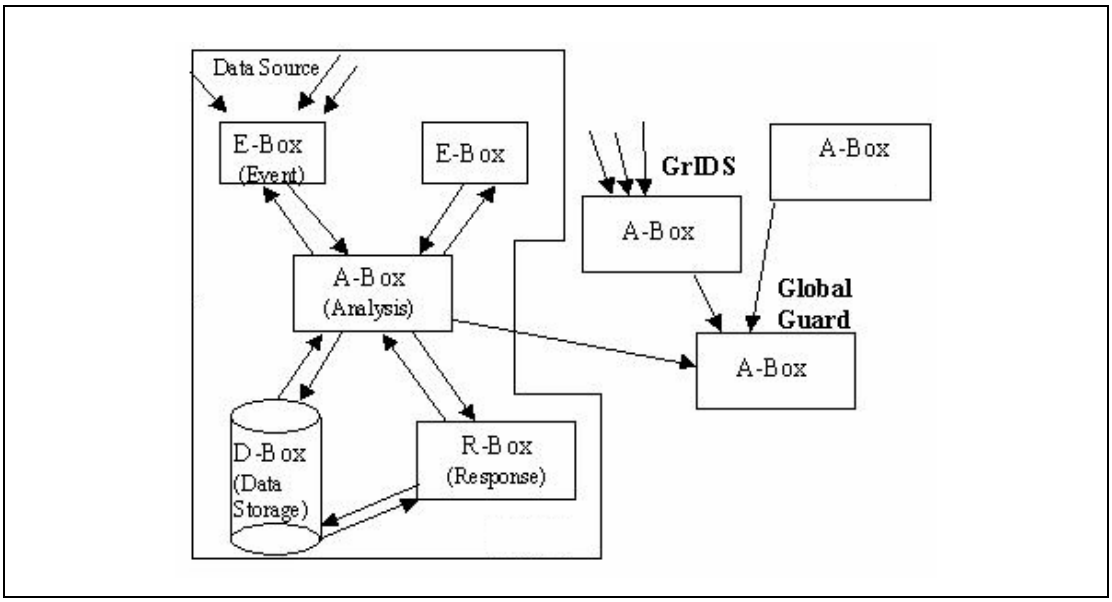


FIG 3-7: Ejemplo de uso del modelo CIDF.

Muchas veces existe la tendencia de economizar esfuerzos y recursos, instalando el sensor y la consola en la misma máquina. Esta arquitectura se desaconseja totalmente, ya que como hemos comentado en puntos anteriores, los sensores deben filtrar el tráfico (cosa que requiere de mucha potencia) y añadir la consola u otros programas (firewalls...) no hace mas que cargar el sistema y restarle potencia al sensor.

## 3.4.2 DIDS

En grandes organizaciones (multinacionales) o universidades (dónde hay diferentes facultades, departamentos, laboratorios...) un único sistema IDS no proporciona la flexibilidad necesaria para la heterogeneidad de los elementos de que disponemos.

Los sistemas **DIDS** (*Distributed Intrusion Detection System*) [Ein01][Vil02] proporcionan este servicio de detección de intrusos para grandes redes.

El análisis de los DIDS es tan o mas complejo que el realizado con los NIDS, con lo que queda fuera de este trabajo aunque se cita la bibliografía correspondiente.

Su característica diferenciadora respecto a los sistemas NIDS tradiciones, es la presencia de dos elementos nuevos en su arquitectura (ver figura 3-8):

- **Central Analysis Server**: Es el centro del sistema DIDS y es el encargado de recibir toda la información procedente de los agentes y realizar un repositorio común de conocimiento. También realiza las funciones de control y sincronización de los diferentes nodos que forman parte del sistema.

- **Co-operative Agent network**: Es un sistema autónomo encargado de la monitorización de una red. Detecta posibles incidentes e informa al servidor central para que comunique a todos los nodos el ataque detectado así como las contra-medidas a realizar. Dependiendo de la implementación, el agente puede llegar a tomar contra-medidas de forma autónoma (auque siempre informando y supeditándose al servidor central).

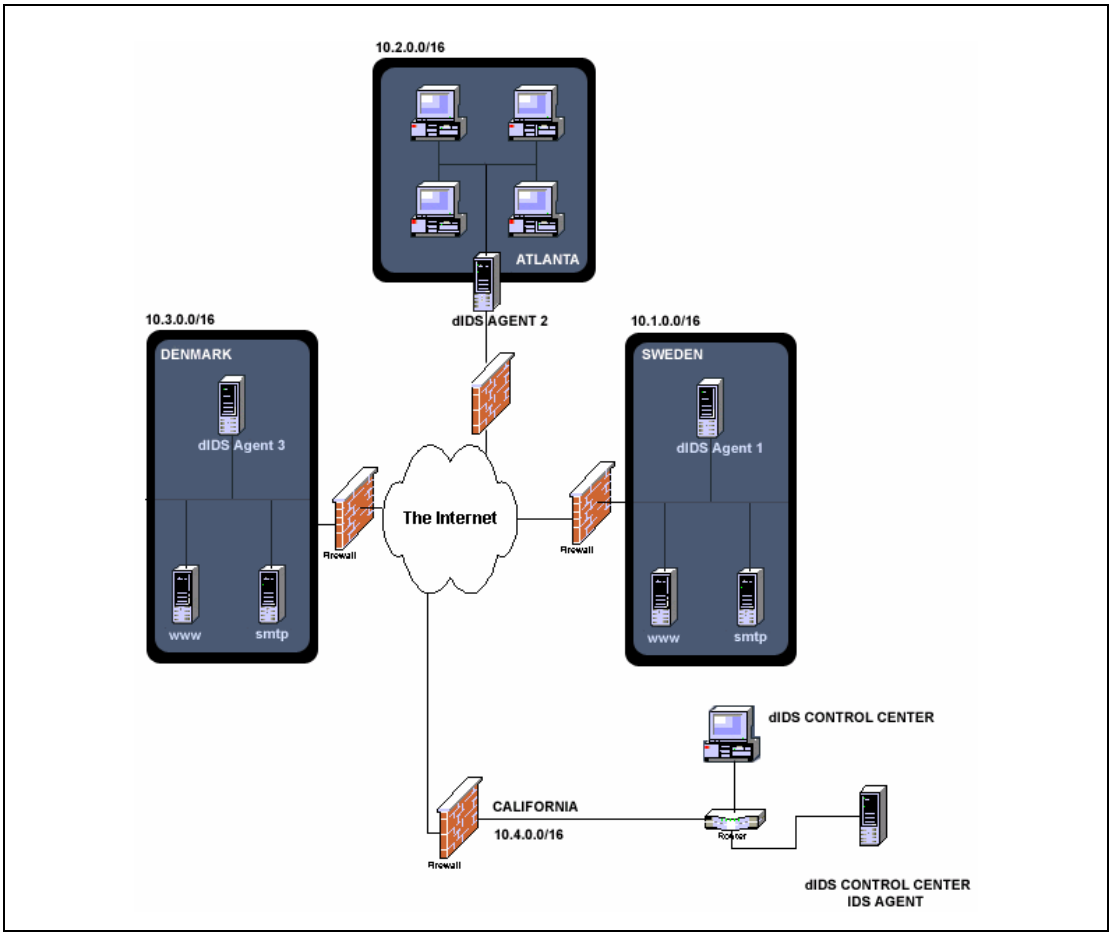


FIG 3-8: Ejemplo de sistema DIDS.

# 3.5 Ubicación de los NIDS

Una vez explicados los componentes básicos de un sistema IDS, se ha de decidir de qué tipo han de ser los distintos sensores que utilizará el NIDS (*pull* o *push*), y finalmente se ha de concretar en que lugar o lugares de la red deben colocarse:

A) **Antes del firewall** (ver figura 3-9): Esta arquitectura se basa en detectar todos los paquetes que llegan a nuestra red antes de ser filtradas por el firewall. De esta forma, realizamos una búsqueda de eventos de interés en todo el tráfico recibido sin interferencias de filtros del firewall.

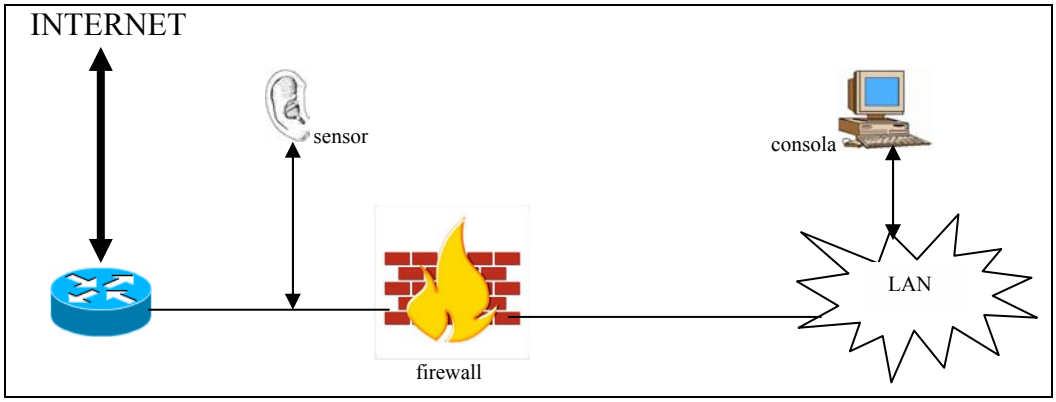


FIG 3-9: Sensores antes del firewall.

B) **En el firewall o adyacente** (ver figura 3-10): Consiste en situar el sensor en el propio firewall. De esta forma se evitan ataques de intrusos a los sensores externos y se eliminan muchos falsos positivos, ya que procesamos únicamente el tráfico que el firewall deja pasar.

C) **Antes del firewall y en el firewall o adyacente** (ver figura 3-11): Es la opción es más costosa pero que ofrece mayor seguridad, ya que permite obtener lecturas del tráfico total y del tráfico filtrado por el firewall. Permite examinar la configuración del firewall y comprobar si filtra correctamente o no.

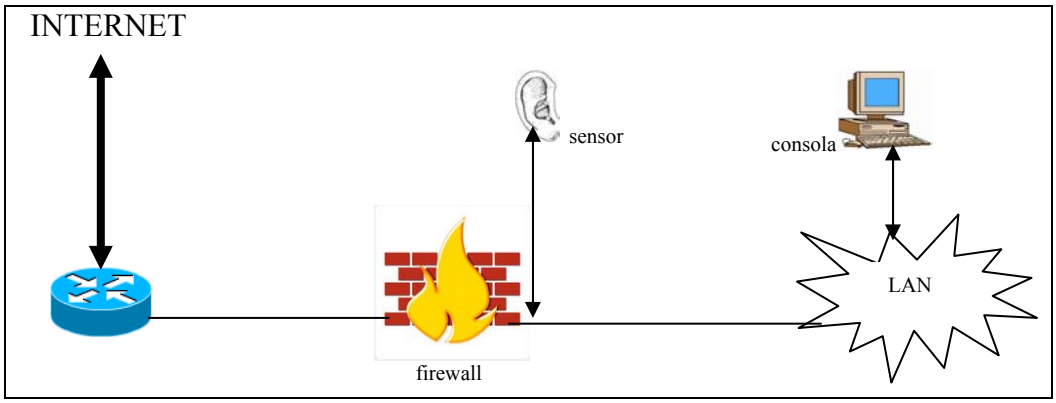


FIG 3-10: Sensores en el firewall.

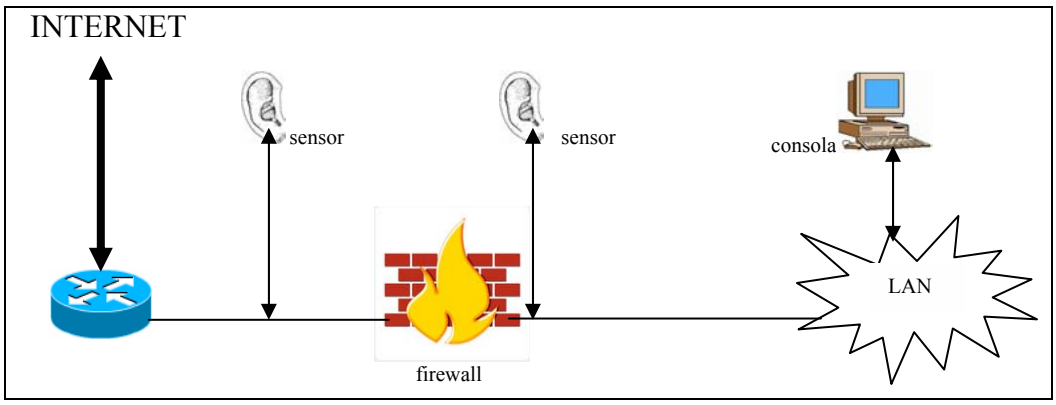


FIG 3-11: Sensores antes y en el firewall.

## 3.6 Protocolos de comunicación Sensor-Consola

Como siempre suele pasar en informática, cuando una compañía desarrolla un nuevo producto suele acompañarlo de un protocolo propietario encargado de gobernar las comunicaciones entre las diferentes partes del sistema. En los primeros productos (o primeras versiones) de sistemas de detección de intrusos esto ha sido una constante, lo que obligaba al usuario a depender de un único fabricante/producto.

Conforme se han ido extendiendo los productos IDS y el mercado ha ido creciendo, los usuarios y expertos demandaban la búsqueda de un lenguaje/protocolo común, ya que si simplemente quiero compartir mis datos con mi propio ISP para mejorar la seguridad o

detectar nuevas firmas de ataques desconocidos, estábamos obligados ambos a poseer el mismo producto (a veces incluso era necesaria la misma versión!!) o esto es del todo imposible.

Además, muchas veces los ataques a redes de ordenadores se concentran en determinadas máquinas que sostienen los servicios vitales de la empresa o universidad (servidores de ficheros o impresoras, servidores de nombres,...). Esto es debido a que muchos atacantes se nutren de listas de ordenadores denominadas *shopping lists* (listas de la compra) que contienen las direcciones IP de los servicios ofrecidos en cada red.

Estas listas circulan por Internet en servidores WWW *undergrounds* y listas de discusiones de temática hacker.

Poder compartir los datos obtenidos por nuestro IDS con el de otras organizaciones u empresas puede evitar problemas en un futuro, ya que usualmente los atacantes utilizan las mismas técnicas en las diferentes organizaciones (obtenidas usualmente de listas de la compra) con la esperanza de encontrar una que sea vulnerable al ataque perpetrado.

Si los diferentes IDS tienen un lenguaje común, la publicación de nuevas firmas/trazas de ataques detectados será más rápida, de forma que el primero que lo detecte puede compartir sus nuevos filtros/firmas con los demás que podrán utilizarlas directamente en su IDS sin tener que adaptarlas.

Para conseguir esta interrelación entre distintos productos IDS se han propuesto diversos protocolos, a continuación enumeramos los más importantes:

**CISL** (*Common Intrusion Specification Language*) [Nor99][WWW79]

Junto con la especificación de CIDF, DARPA e IETF [WWW80][WWW81] propusieron un standard de representación de la información asociada a sistemas de detección de intrusos basado en un lenguaje de expresiones regulares de tipo S dónde se insertan marcas (*tags*) y datos (ver figura 3-12).

Estas expresiones están delimitadas por paréntesis imitando el lenguaje LISP y permiten expresar tres tipos de información:

1. *Raw Event Information*: hace referencia a información básica obtenida directamente sin ningún tipo de post-proceso. Principalmente tráfico de red y resultados de auditorias (*audit trail).*

2. *Analysis Results*: Descripciones de anomalías o ataques detectados.

3. *Response prescriptions*: Conjunto de acciones de respuesta predefinidas para algunos comportamientos observados (ej. Bloquear acceso desde una dirección IP si detectamos más de 1.000 conexiones por segundo procedente de ella).

```
(Login
        (Location   (Time '07:07:07 09 Julio 2003')  )
        (Initiator    (Hostname 'gabriel.lsi.upc.es')   )
)
```

FIG 3-12: Ejemplos de una expresión formal en CISL.

**OPSEC** (*Open Platform for Secure Enterprise Connectivity*) [Nor99]

Esta propuesta parte de la empresa privada de seguridad Checkpoint Software [WWW78], que como suele ocurrir cuando una compañía lidera un sector intenta imponer su standard mediante una gran cuota de mercado.

El paradigma propuesto a finales de 1997 permite agrupar en un único modelo las distintas necesidades y capacidades de un modelo de seguridad. OPSEC se fundamenta en el uso de interfaces de aplicaciones (API) que permiten la interconexión de los distintos módulos, protocolos standards y un pseudo-lenguaje de alto nivel tipo "*shell script*" que permite gobernar todos los elementos y funciones (por ejemplo permite la interconexión con sistemas de firewalls para la adopción de contra-medidas ante comportamientos sospechosos).

OPSEC es el standard "*de facto*" para la industria en la comunicación entre sistemas IDS y Firewalls ya que ha sido adoptado por más de 250 compañías.

**CCI** (*Common Content Inspection*) [Nor99]

Análogamente al OPSEC, las diferentes compañías lideradas por Chekpoint [WWW82][WWW83] propusieron otro standard para el análisis de la información contenida en los paquetes que recorren la red. En lugar de las cajas (*boxes*) propone:

- *Content redirector:* Servicio que redirecciona el contenido a los motores de inspección y análisis.

- *Content Inspector:* Es el servicio que realiza la inspección y el análisis de los contenidos.

EL API propuesto está muy interrelacionado con OPSEC (nacen de la misma empresa de seguridad) y ambos se modifican teniendo en cuenta esta dependencia funcional.

**ANSA** (*Adaptive Network Security Alliance*) [Nor99]

La empresa de seguridad ISS también propuso su standard para la interoperatividad de distintos sistemas de seguridad [WWW84][WWW85]. Sus principales características son:

- *Automated Response*: Permite reconfigurar dinámicamente a los distintos equipos de red (firewalls, switches…) para responder a las amenazas detectadas en "tiempo real".

- *Lockdown*: Esta característica hace referencia a "encerrar" o "enjaular" las distintas direcciones IP de los atacantes para evitar que puedan acceder o reconfigurar servicios críticos de la red (DNS, servidores de mail…).

De esta forma, aunque nuestro gestor de correo sea vulnerable, si nuestro sistema detecta que la dirección es hostil proactivamente no le permitirá acceder a él.

- *Decision Support*: La funcionalidad de asistencia para la toma de decisiones permite la utilización de técnicas de data mining (correlación...) y consultas históricas a bases de datos para sugerir acciones ante las posibles amenazas detectadas.

- *Security Management*: Históricamente, los elementos de seguridad existentes en las redes informáticas se encontraban diseminados sin ningún tipo de control centralizado que los agrupara. El control/supervisión centralizado de los diferentes elementos de seguridad de nuestra red nos permitirá mantener una coherencia en nuestras actuaciones. Cada uno de estos elementos en lugar de tomar decisiones aisladamente de forma unilateral se comunican con el resto del sistema.

## 3.7 Análisis de los datos obtenidos por los sistemas NIDS

Una vez comentadas las distintas arquitecturas de sistemas de detección de intrusos y sus protocolos de comunicación así como sus diferentes elementos, pasamos a profundizar en el análisis de los datos obtenidos por nuestros sistemas de seguridad.

Tal como se demostró anteriormente el tamaño de disco necesario para almacenar el tráfico de un solo día es totalmente desorbitado (ver figura 3-2). Con los filtros aplicados al tráfico supervisado y los eventos de interés disminuimos substancialmente esta cantidad, sin embargo, no solamente nos interesa el tráfico registrado en un solo día, sino que nos interesan históricos de la semana, mes o incluso de años.

Toda esta cantidad de información debe ser almacenada de forma óptima para poder ser consultada ágilmente. De otra forma, dejará de ser utilizada y por lo tanto de ser útil.

El problema de almacenar/recuperar información es un tema muy amplio que no trataremos en este documento. Sin embargo sí comentaremos que últimamente muchos productos IDS empiezan a incorporar soporte de bases de datos relacionales como elementos importantes de soporte para sus sistemas de información (Oracle, SQL Server, MySQL, PostgreSQL...).

Sea cual sea el formato usado para el almacenamiento de la información debe cumplir las siguientes características:

1. Debe realizar una reducción importante del volumen de datos pero conservando la información importante.

2. Debe poseer un formato compacto, fácil de consultar, escribir y actualizar.

3. Debe permitir la interrelación (cruce) de todos los datos entre sí.

El formato **TCP QUAD** [Nor99][WWW86] es una reducción compacta de la información contenida en los paquetes IP que se basa en almacenar una cuádruple tupla que contiene los siguientes campos[ℜ]:

**(Fecha, Dirección origen, Dirección de destino, Tipo de protocolo)**

El objetivo de almacenar históricos de tráficos de red es doble, por un lado poder realizar informes y estadísticas sobre incidentes en nuestra red. Por otro lado nos debe permitir conocer cuando un ataque presenta características similares (o iguales) a otro recibido anteriormente, ya que podemos obtener información de cómo reaccionar ante él de forma proactiva ("en tiempo real"), evitando ser sujetos pasivos y lamentarnos posteriormente.

Definiremos la **correlación** como la relación mutua entre dos o más elementos de un conjunto dado. De esta forma, gran parte de las consultas a las bases de datos se basarán únicamente en buscar correlaciones entre los eventos de interés detectados y los datos históricos almacenados.

[ℜ] Dependiendo de la bibliografía estos campos varían ligeramente. Otros autores proponen añadir el campo de opciones (flags o banderas).

- **Correlaciones por dirección de origen**: Se basan en encontrar similitudes entre conexiones provenientes de la misma fuente (por ejemplo un escáner de puertos a nuestra red detectará que desde el mismo origen se realizan miles de peticiones a distintos puertos).

- **Correlaciones por dirección de destino**: Considera las conexiones que tienen como destino la misma dirección IP (por ejemplo un DOS. Tenemos miles de peticiones hacia un mismo destino).

- **Correlaciones de firmas** *(crafted packed signatures)*: Se basan en buscar conexiones desde/hacia un puerto determinado (muchos virus y programas de *backdoor* basan su comunicación en puertos no standards). También se buscan configuraciones no estandards de los distintos campos del paquete IP, como por ejemplo las banderas o *flags* (hay varios ataques de DOS que se basan en activar todas las opciones de los datagramas IP para ver si el sistema operativo no sabe como reaccionar ante ellos y queda inutilizado).

- **Correlaciones de contenidos**: Estas correlaciones hacen referencia al contenido (datos) de los distintos paquetes de información que circulan por la red. Buscar patrones como "*/etc/passwd*" en conexiones TELNET o FTP, inspeccionar conexiones HTTP en busca de "*EXEC C:\WINNT\COMMAND\FORMAT*"...

Otras posibilidades que nos brinda la correlación es la de poder evaluar la importancia (criticidad) de un posible incidente (ver figura 3-13).

Por ejemplo podemos observar la periodicidad de los paquetes recibidos, lo que nos permite por ejemplo saber de que ancho de banda dispone el atacante "a priori". Si es una simple prueba rutinaria de un hacker "a ver que pesca" enviando peticiones muy espaciadas en tiempo a distintos servicios y direcciones IP, o un ataque en toda regla a un servidor concreto.

El sistema IDS debe permitir realizar estas consultas interactivamente al operador de la consola para investigar libremente en las bases de datos. Además deben realizarse de forma automática sólo para los eventos de interés de extrema criticidad, ya que el

proceso de correlación es muy lento y colapsaría el sistema, de forma que cuando detectásemos el ataque ya sería demasiado tarde para actuar.

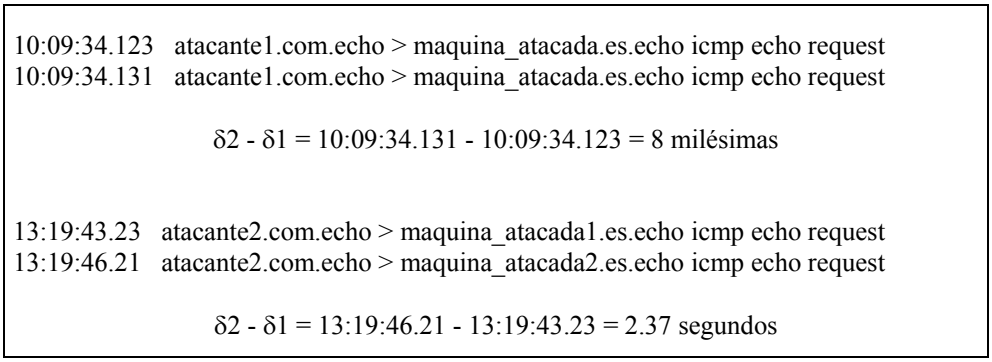


FIG 3-13: Ejemplos de trazas para evaluar la criticidad.

Existe una gran controversia en la bibliografía [WWW88] sobre la cantidad de tiempo (semanas, meses, años?) que debemos tener almacenada en la base de datos del IDS para poder realizar consultas tanto el operador como el propio sistema automáticamente.

Obviamente, lo deseable sería tener todo el histórico de la red, sin embargo debemos tener en cuenta que:

- Más tiempo implica mas espacio de disco (crecimiento exponencial).
- Más tiempo implica una ralentización de las búsquedas (pérdida de eficiencia).
- Más tiempo no implica necesariamente más seguridad.
- En caso de necesitar la informática forense (*computer forense*) nuestro histórico debe permitir la reconstrucción total los actos sucedidos en la red.
- También debemos tener en cuenta las leyes vigentes de protección de datos y almacenamiento de logs (LORTAD...)

Por lo que una solución propuesta [Nor99] aboga por una ventana de al menos tres meses (90 días).

A partir de esta fecha, los datos se deben "reducir" o "compactar" (mediante TCP QUAD por ejemplo) para almacenarlos en otra base de datos de históricos. La normativa de la administración americana PDD63 establece una ventana de 72 a 96 horas dónde se deben poder realizar trabajos de reconstrucción forense.

## 3.8 Falsos positivos y falsos negativos

En los puntos anteriores hemos analizado las diferentes partes que integran un esquema IDS. También hemos comentado los diferentes protocolos utilizados para conseguir la comunicación entre los distintos programas existentes en el mercado así como varias herramientas que se utilizan para la detección de los posibles incidentes (firmas, reglas, correlaciones...).

Un punto básico a tratar tras el análisis de las muestras obtenidas (eventos de interés) de nuestra red es el de la detección de **falsos positivos** y **falsos negativos**.

*"Un **falso positivo** (false positive) es un término aplicado a un fallo de detección en un sistema de alertas (usualmente en sistemas antivirus o de detección de intrusos). Sucede cuando se detecta la presencia de un virus o una intrusión en el sistema que realmente no existe." [WWW87]*

*"Un **falso negativo** (false negative) es un término que hace referencia a un fallo en el sistema de alerta (usualmente en sistemas antivirus o de detección de intrusos). Sucede cuando un virus o una intrusión existe en nuestro sistema y es **'permitida'** (ignorada o no detectada) por el sistema de alerta." [WWW87]*

Los falsos positivos pueden agruparse en cinco grupos dependiendo de la naturaleza de su origen [Tim01]:

- **Reactionary Traffic alarms:** Se detecta un comportamiento sospechoso como consecuencia de tráfico generado anteriormente (generalmente no malicioso). Por ejemplo la detección de muchas respuestas "*ICMP network unrecheable"* procedentes de un router porque el equipo destino no se encuentra operativo o accesible en esos momentos.

- **Equipment-related alarms:** Las alarmas del NIDS detectan paquetes dentro del tráfico de la red que identifica como no "usuales". Esto puede ocurrir por ejemplo con balanceadores de carga, puesto que generan paquetes específicos para el control de todos los nodos.

- **Protocol Violations:** Estos avisos se producen por software mal programado (bugs) o que implementan de forma incorrecta o anticuada algunas partes de los protocolos de Internet.

- **True False Positives:** Todos aquellos falsos positivos que no se encuadren en ninguna de las categorías anteriores.

- **Non Malicious alarms:** Alarmas producidas al detectar rastros de comportamientos maliciosos pero que en ese contexto determinado no lo son. Si publicamos en nuestra página WWW el código de un virus analizándolo, cada vez que una persona descargue la página creará una alerta en el IDS porque detectará el virus en nuestra red.

Obviamente, nuestro sistema de detección de intrusos debe producir los mínimos falsos positivos posibles y **ningún** falso negativo (porque con uno sólo, ya tenemos al intruso en nuestro sistema, y toda la inversión en seguridad se vuelve inútil y de difícil justificación).

Según Kevin Timm [Tim95] el 90% de las alarmas detectadas por sistemas de detección de intrusos son falsos positivos, con lo que tan sólo el 10% son realmente peligrosas. Si personalizamos el NIDS a nuestra red, generalmente con los sistemas convencionales podemos llegar a reducir los falsos positivos a un 40% del total de alarmas.

Adhitya Chittur [Chi01] afirma en su tesis "**Model generation for an intrusion detection system using genetic algorithms**" que con modelos genéticos se puede bajar la tasa de falsos positivos hasta menos del 10%.

Por otro lado, hemos de tener en cuenta que un IDS que reporte cientos de alertas diarias dejará de ser útil y muy probablemente lleve a que alguna alerta verdadera sea ignorada por los operarios entre tantos avisos.

En la figura 3-14 podemos observar 9 peticiones de establecimiento de conexión en un segundo procedentes de "*maquina1.com*" para "*maquina2.es*". Generalmente y a priori el IDS detectaría una masiva llegada de peticiones desde una misma dirección IP en un lapso de tiempo corto: Estamos recibiendo un Ataque!

```
09:09:34.123   maquina1.com.601 > maquina2.es.login S 13961:13961(0) win 4096
09:09:34.153   maquina1.com.601 > maquina2.es.login S 13962:13962(0) win 4096
09:09:34.159   maquina1.com.601 > maquina2.es.login S 13963:13963(0) win 4096
09:09:34.183   maquina1.com.601 > maquina2.es.login S 13964:13964(0) win 4096
09:09:34.323   maquina1.com.601 > maquina2.es.login S 13965:13965(0) win 4096
09:09:34.823   maquina1.com.601 > maquina2.es.login S 13966:13966(0) win 4096
09:09:34.943   maquina1.com.601 > maquina2.es.login S 13967:13967(0) win 4096
09:09:34.980   maquina1.com.601 > maquina2.es.login S 13968:13968(0) win 4096
09:09:34.998   maquina1.com.601 > maquina2.es.login S 13969:13969(0) win 4096
```

FIG 3-14: Ejemplos de falso positivo para SYN FLOOD.

Esta conclusión que para la mayoría de casos es correcta (¿porque nuestra "maquina2.es" debe recibir tantas peticiones de conexión casi simultáneas?), no siempre lo es de forma absoluta.

Si "maquina2.es" es por ejemplo un punto de entrada hacia nuestra red corporativa puede pasar simplemente que a primera hora (09:09 horas) se conecten muchos empleados para realizar su trabajo. Sin embargo, esto no explica que todas las direcciones de origen sea la misma. ¿Por qué alguien se conectará 9 veces en un segundo?

Un ejemplo práctico y real de falso positivo puede ser el uso de **NAT** (*Network Address Translation*) [Has97][WWW90] en redes corporativas. Todas las máquinas de una misma red realizan sus conexiones al exterior desde una única dirección IP.

Esta metodología se utiliza mucho por motivos de seguridad (DMZ por ejemplo), económicos (es más barato una dirección IP que 10) o incluso técnicos (ADSL, MODEM-cable...).

De esta forma, podemos observar que una situación potencialmente peligrosa puede simplemente ser un uso normal de los recursos de red.

En la actualidad y debido a la política de todos los ISP de tecnologías domésticas de conexión a Internet (ADSL, Cable…) se están implementando "*proxys*" [WWW91] automáticos de acceso para contenido HTTP. De esta forma, las peticiones de cualquier usuario de ADSL a un servidor WWW son automáticamente redireccionadas al PROXY de su proveedor de conexión.

Los ejemplos de falsos negativos son mucho mas complicados de encontrar en bibliografía y exponer en este escrito, ya que implicaría que alguien consiguió un acceso no autorizado a una red "segura" y a nadie le hace gracia reconocer esto en "públicamente" (muchas veces no se reconoce ni en privado). Generalmente, los falsos negativos suelen producirse por:

1. **Configuración deficiente de los recursos de la red:** Tener varios elementos de seguridad (IDS, firewalls, VPN...) no es suficiente para considerar nuestra red segura. Estos deben estar convenientemente configurados y adaptados a su medio (el tráfico de las redes no es estático y varía).

2. **Ataques desde dentro**: Muchas veces se obvia un uso pernicioso de los recursos de nuestra red desde dentro. El atacante no siempre es un hacker de un país extraño que se dedica a hacer el mal a quien puede. Tener controles internos también permitirá detectar programas o troyanos encargados de facilitar acceso desde dentro a posibles atacantes.

3. **Equipos no parcheados y víctimas de los últimos "*exploits*"**: Tener servidores con versiones de software anticuadas son un reclamo excesivamente apetitoso y que ningún sistema de seguridad puede proteger.

# 3.9 IPS

En los últimos artículos y congresos, la bibliografía [Des03][WWW109] recoge una de las tendencias futuras de los sistemas de detección de intrusos, los sistemas de prevención de intrusos o **IPS** (*Intrusion Prevention System*).

Definiremos un *"sistema de prevención de intrusos (Intrusion Prevention System, IPS) como un dispositivo (hardware o software) que tiene la habilidad de detectar ataques tanto conocidos como desconocidos y reaccionar a esos para impedir su éxito."* [Des03]

Los sistemas de prevención de intrusos pueden verse como la evolución de dos elementos que han dominados la seguridad en todas las redes informáticas del mundo:

- **Firewall**: Es el elemento que garantiza o bloquea el acceso a los recursos de nuestra red.

- **IDS**: Permite mantener el estado de las conexiones y examinar el contenido de los paquetes que circulan por nuestra red.

Los IPS pueden agruparse en cinco categorías dependiendo de su arquitectura y ubicación dentro de la red a proteger:

1. **Inline IPS**: Estos sistemas de prevención de intrusos se caracterizan por colocarse entre la red y el ordenador (o tramo de red) a proteger. Su arquitectura se basa en dos tarjetas de red (ver figura 3-15):

- La primera conectada a la red exterior y que no dispone de dirección IP. Su función es la de realizar *bridging* transparente entre la red y el IPS. Al no disponer de dirección IP, este interface es totalmente invisible y no puede recibir ningún tipo de tráfico expreso (ni ser atacado).

- La segunda tarjeta esta conectada a la red "segura" o interna y nos permite conectarnos al sistema IPS para su gestión y configuración.

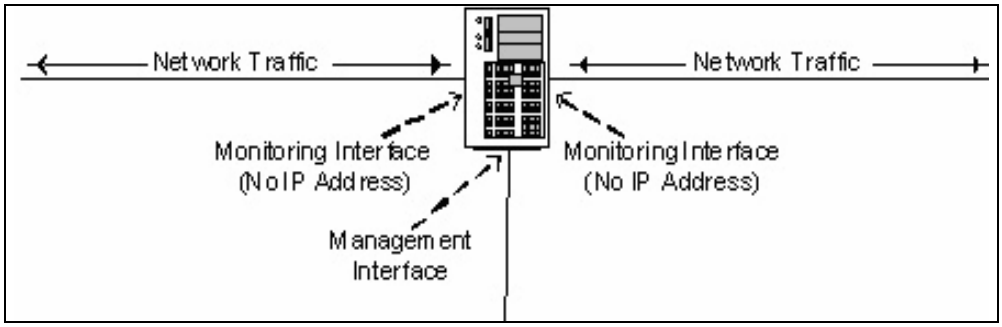


FIG 3-15: Ejemplos de IPS inline.

El sistema IPS procesa todo el tráfico entrante y saliente de manera que cada paquete puede pasar (*forward*), eliminarlo (*drop*), grabarlo en el log (*log*), borrarlo del log (*unlog*) o incluso rescribir el paquete eliminando elementos potencialmente peligrosos (*rewrite*) [HL01][Des03].

Estos sistemas deben ser muy sólidos, ya que si el IPS deja de funcionar (fallo hardware/software), se pierde todo acceso a la red.

2. **Layer seven switches**: Los conmutadores o switches son dispositivos de capa 2 del modelo OSI [Ric98-1]. En el caso de las redes IP, los diferentes protocolos para los distintos servicios (HTTP, FTP, SSH...) se sitúan en la capa 7 del modelo OSI. De esta forma, para poder inspeccionar paquetes IP de diferentes servicios necesitamos un conmutador de nivel 7.

   Estos IPS contienen un componente hardware muy importante que le proporciona una velocidad de proceso en el filtrado de paquetes de red muy superior a los sistemas convencionales basados en un programa que se ejecuta

en un ordenador. Por otro lado, la flexibilidad de reconfiguración y nuevas versiones de los IPS basados en programas queda sacrificada al ser un elemento hardware en su casi totalidad (ver figura 3-16).

Su modo de funcionamiento se basa en un único puerto del *switch* encargado de mantener la conexión de red con el exterior, y el resto de puertos conectados a los distintos servidores a monitorizar.

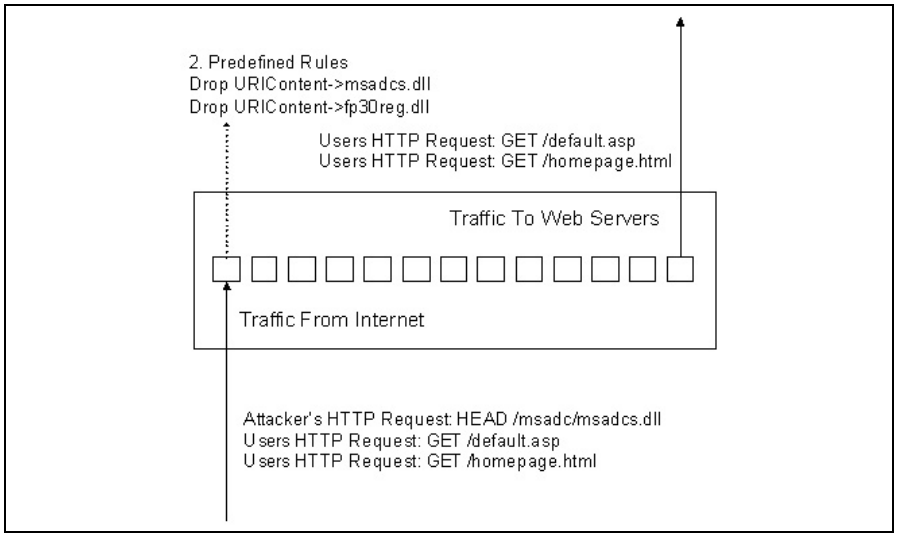


FIG 3-16: Ejemplos de IPS layer 7 switch.

Como los otros sistemas de prevención de intrusos, es capaz de filtrar el tráfico, buscar patrones en la red y controlar las conexiones de entrada y salida a los servidores. A diferencia de los sistemas basados únicamente en hardware, son capaces de controlar eficientemente el ancho de banda utilizado en cada uno de los servidores y variarlo en función de las necesidades de cada momento.

3. **Application firewall/IDS**: Este grupo de IPS son programas autónomos que corren en cada uno de los servidores a proteger. De esta forma, se encargan de proteger únicamente un ordenador o servicio concreto, y no una red o conjunto de ordenadores.

La sobrecarga (*overhead*) de proceso que producen en el ordenador, queda compensada por su gran capacidad de configuración, lo que permite que se centre únicamente en las partes o servicios que nos interese proteger (por ejemplo WWW) en lugar de controlar todo el sistema (ver figura 3-17).

El sistema IPS crea unos perfiles (*profile*) para cada uno de los servicios a monitorizar de forma que podemos ajustar de manera muy específica los parámetros de control minimizando la sobrecarga.

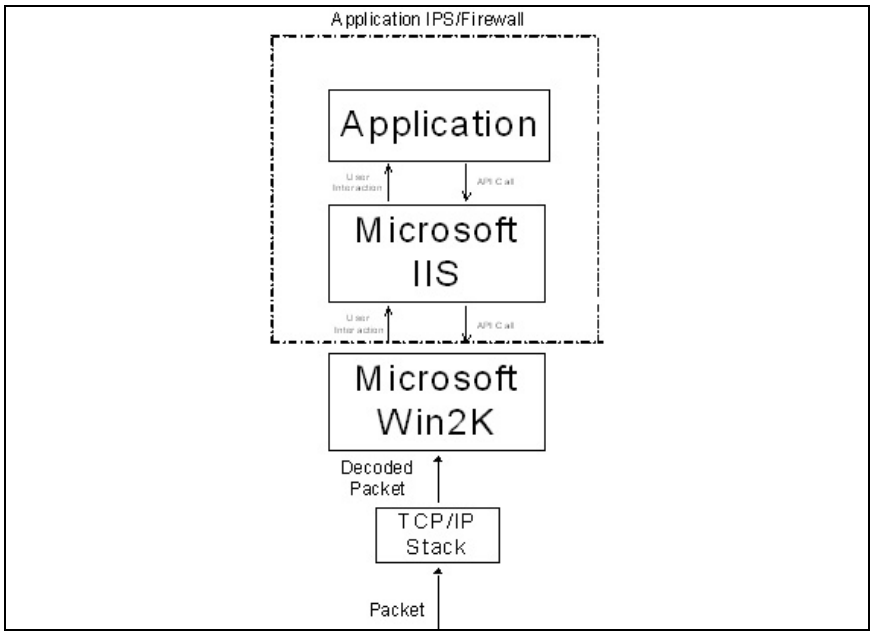


FIG 3-17: Ejemplos de IPS application firewall/IDS.

La aplicación IPS se coloca entre el gestor de comunicaciones del sistema operativo y la aplicación, monitorizando todo el tráfico y efectuando todas las acciones que considere necesarias (*forward, drop, log, unlog, rewrite*) según la configuración creada (perfil).

Hay que tener en cuenta que debemos ir sincronizando el perfil con los distintos cambios o actualizaciones de la aplicación que se vayan produciendo, ya que el IPS puede detectarlo como un intruso y cesar el servicio.

4. **Hybrid switches**: Estos sistemas de prevención de intrusos se basan en combinar una parte de software y otra de hardware.

   La parte hardware es la encargada de filtrar el tráfico en tiempo real (debido a su mayor velocidad de proceso), limitar el numero de peticiones a los servidores y regular los anchos de banda de entrada y salida adecuándolos a las necesidades y demandas en cada momento.

   La parte software se instala en cada uno de los servidores a monitorizar de forma que se crea un perfil específico para cada una de los servicios. Este componente se comunica con la parte hardware para conseguir un comportamiento mas adecuado a las necesidades reales. Por otro lado, también realiza parte del filtrado para descargar el hardware y eliminar los cuellos de botella.

5. **Deceptive applications**: Este tipo de aplicaciones empezaron a utilizarse en el 98 y se basa en una aproximación diferente y más proactiva frente a la detección de intrusos. Su aplicación se divide en tres fases (ver figura 3-18):

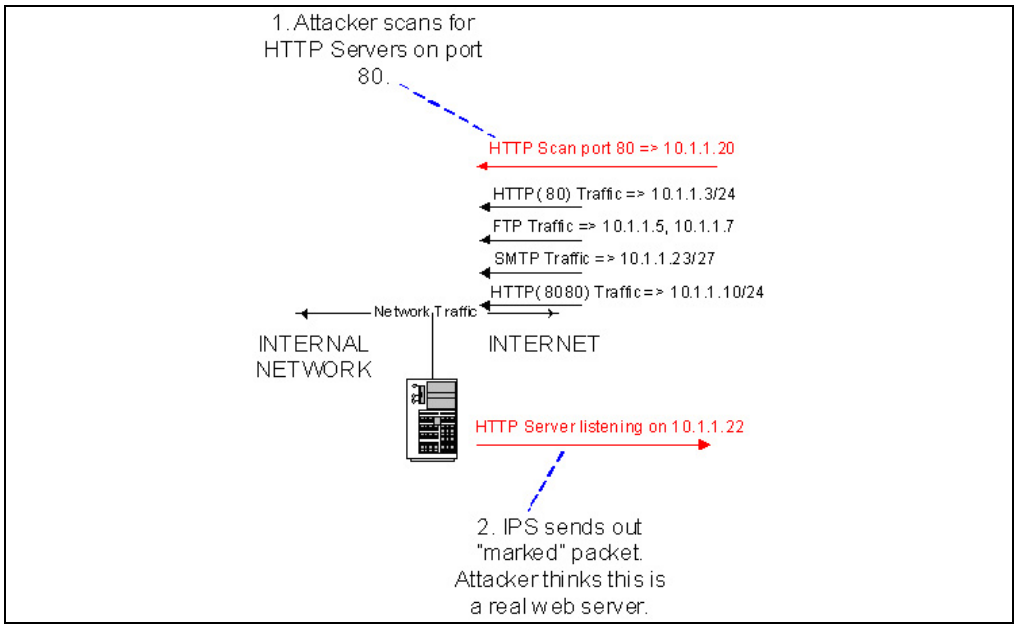


FIG 3-18: Deceptive applications.

En una primera fase se analiza todo el tráfico de la red con el objetivo de crear un modelo que represente todo el tráfico "normal" que circula por la red.

Después (opcionalmente, aunque se recomienda) se retoca el modelo manualmente por parte del administrador del sistema para adecuarlo a la realidad de la red (*profiling*). Finalmente el sistema asignará unos pesos a cada uno de los distintos eventos que ha observado en el análisis.

En una segunda fase, el sistema monitoriza el tráfico y las peticiones que circulan por la red. Cuando observa peticiones a servicios que no existen o que no están disponibles, reacciona enviando como respuesta un paquete "marcado" simulando la existencia del servicio y anotando los parámetros de origen de la petición.

La tercera fase se produce cuando el atacante utiliza los datos obtenidos en incursiones anteriores a la red (fase 2) para lanzar un ataque sobre servidores o servicios. En este momento el sistema reconoce al atacante y bloquea su acceso a la red.

## 3.10 Limitaciones de los IDS

Después de analizar los distintos tipos, arquitecturas y características de los sistemas de detección de intrusos, comentaremos brevemente que aspectos dejan más abiertos o no resueltos totalmente en la actualidad.

Los sistemas NIDS se basan en la observación del tráfico que circula por la red, esta única fuente de información le confiere una serie de limitaciones:

- Si alguien crea una puerta trasera ("*back-door*") o crea una red paralela en la intranet, probablemente los IDS no lo detectarán puesto que están pensados y configurados para el rango IP señalado.

Si alguien se apropia de una cuenta y se autentica con el *login* y *password* correcto, tampoco.

- Si los sensores que alimentan al IDS o el propio IDS no funciona (la máquina, el sistema operativo o el programa no se encuentra operativo) no observaremos nada.

  Es importante realizar un seguimiento periódico de la actividad de estos programas, ya que el hecho de tenerlos instalados o hacer un simple *ping* a la máquina para ver si está operativa no es una política de seguridad aceptable.

- Muchos IDS simplemente soportan protocolos muy extendidos (usualmente basados en IP como FTP, HTTP...) pero no protocolos de usos minoritarios comos el IPX/SPX o SNA. Existen muchos "*exploits*" disponibles para estos protocolos que pasarán inadvertidos en nuestra red. Si utilizamos Novell, NetBIOS, SNA... debemos tener en cuenta este punto.

- Los IDS tienen un máximo de capacidad de proceso, por lo que en momentos de mucho tráfico suelen descartar los paquetes que no pueden procesar.

  Raramente un IDS avisa de este punto, con lo que es conveniente calibrar periódicamente si nuestro IDS es capaz de soportar realmente toda la carga de la red (accesos a discos, swap, filtros muy complicados...). No hay que olvidar que usualmente un IDS no es más que un programa que funciona en un ordenador.

Por otro lado, un IDS basado únicamente en la búsqueda de firmas (al igual que muchos antivirus) presenta dos grandes problemas:

1. El número de firmas va aumentando cada año (actualmente existen más de 20.000 para los antivirus) y tan sólo refleja los ataques que han sido ya identificados y analizados (UC Davis nos da una muestra de firmas para IDS en [WWW67]).

En los informes de ataques recibidos, siempre salen reflejados los mismos ataques que en el resto de Internet. Esto es debido a que los nuevos ataques aún no han sido detectados y documentados, con lo que no existe una firma a la que asociar el ataque recibido, lo que puede crear una falsa sensación de seguridad al ver que detectamos siempre los mismos ataques.

2. En el caso de una red con cientos de ordenadores, todo y detectar ataques ya conocidos con nuestro IDS ¿podemos estar seguros que todos nuestros ordenadores son invulnerables a este ataque?.

   Por otro lado el responsable de los ataques muy conocidos que detecta nuestro IDS puede ser ignorado como un aprendiz de hacker (*script-kiddie*). ¿Quien nos garantiza que no irá "in crescendo" probando otros ataques no documentados que no detectemos?.

El uso de otras técnicas de seguridad que complementen a los sistemas de detección de intrusos (como los Honeypots y Honeynets) es vital para conseguir un sistema de seguridad fiable y eficaz.

## 3.11 El ataque Mitnick

En este último apartado del capítulo realizaremos un breve pero completo análisis a un ataque considerado como el umbral mínimo de detección por un sistema IDS actual para considerarse una herramienta útil y fiable [Nor99].

Kevin Mitnick es el hacker mas famoso de todos los tiempos [WWW103][WWW104], fue capturado por Tsutomu Shimomura y entregado al FBI. Fue condenado a 68 meses de prisión de los que sólo cumplió 60. Está en libertad desde el 21 de enero de 2000.

El ataque de Kevin Mitnik se basa en el conocimiento profundo de los protocolos de Internet (TCP, UDP, OC, ICMP...) y la combinación de dos técnicas de hacking [Nor99]:

- **SYN Flooding**: Esta técnica consiste en realizar miles de peticiones de conexión a un ordenador sin finalizar completamente ninguna de ellas. De esta forma, se intenta bloquear a la máquina que recibe las peticiones consumiendo todos sus recursos de red (para mas información ver el capítulo de ataques DDOS).

- **TCP Hijacking**: Esta técnica consiste en conseguir apropiarse de una conexión ya iniciada, desviándola hacia el ordenador del atacante (para mas información ver el capítulo de Redes IP).

El objetivo de nuestro ataque es el de conseguir acceso a una máquina a la cual legalmente no podemos entrar. La manera de conseguirlo consiste primero en detectar las conexiones existentes en la máquina que queremos atacar. Posteriormente seleccionamos a una víctima y le robamos la conexión que tenía establecida. De esta forma, conseguimos un acceso a la máquina sin necesidad de validarnos en el sistema (login).

Kevin Mitnick utilizó las relaciones de confianza (*trust relationships*) entre el cliente autorizado y la máquina a atacar para robar un acceso una vez que este se hubiera autenticado en el sistema.

Para ello, primero se instaló un programa de tipo "*sniffer*" (**tcpdump** por ejemplo) para examinar el tráfico de la red dónde está situado el ordenador atacado. Se supone que también se usaron otras técnicas standard para la obtención de información del ordenador a atacar (consultas a los *daemons* de servicios de finger, NFS, RPCinfo...)

Observando el tráfico (ver figura 3-19) podemos deducir (predecir mas concretamente) que cada vez que se inicia una petición de conexión a "*x-terminal*" (servicio de *shell* remoto bajo X-Windows), la respuesta del servidor contiene un número de secuencia 128.000 unidades mayor que el enviado en la petición.

**IMPORTANTE:** este número NO es siempre el mismo y depende del programador del software de red de cada sistema.

1. Primer paso del protocolo de tres pasos para establecimiento de conexión TCP (three-way handshake[39])

14:18:25.906002 apollo.it.luc.edu.1000 > x-terminal.shell: S **1382726990**:1382726990 (0) win 4096

2. Segundo paso. El servidor nos da su número de secuencia y el ACK del nuestro + 1.

14:18:26:094731 x-erminal.shell > apollo.it.luc.edu.1000: S **2021824000**:2021824000 (0) ACK **1382726991** win 4096

3. Tercer paso. El cliente envía su número de secuencia y el ACK del número del servidor + 1

14:18:26.172394 apollo.it.luc.edu > x-terminal.shell: S 1382726991: 2021824001 (0) win 0 ACK **2021824001**

**En sucesiones peticiones de x-term observamos (extracto del tráfico):**

14:31:25.906002 apollo.it.luc.edu.1000 > x-terminal.shell: S **1382726992**:1382726992 (0) win 4096
14:31:26:094731 x-erminal.shell > apollo.it.luc.edu.998: S **2021952000**:2021952000 (0) ACK **1382726993** win 4096
14:31:26.172394 apollo.it.luc.edu > x-terminal.shell: S 1382726993:2021952001 (0) win 0 ACK **2021952001**
.....

De lo que podemos deducir (predecir) que el número de secuencia enviado por el servidor es siempre 128.000 unidades superior al anterior.

2021952000 - 2021824000 = **128.000**

FIG 3-19: Establecimientos de conexión observados por Mitnick.

La obtención de los números de secuencia es vital, ya que en el caso de enviar un paquete con un número de secuencia erróneo, automáticamente se genera un RESET y el servidor cierra la conexión.

Una vez deducido el número de secuencia que obtendremos, debemos proceder a "secuestrar" la conexión del usuario correctamente autenticado. Esto fue posible debido a que una vez autenticada la comunicación (dirección IP de origen + puerto) <=> (dirección IP de destino + puerto) esta no vuelve a ser verificada por el servidor.

La dirección IP se encuentra en la cabecera del paquete IP mientras que los números de secuencia se encuentran en la cabecera TCP. Las aplicaciones simplemente mantienen el número de secuencia para asegurarse que el datagrama recibido pertenece a la

[39] En el primer capítulo está ampliamente explicado el protocolo de tres pasos para el establecimiento de conexiones TCP.

secuencia correcta (de aquí la importancia de poder predecir el número de secuencia enviado por el servidor).

De esta forma, una vez iniciada la conexión desde un cliente autorizado (paso 1 de la figura 3-18), Mitnick "silencia" al cliente mediante un ataque SYN FLOOD [Tan03].

Simultáneamente el servidor completa el segundo paso de la conexión (paso 2 de la figura 3-18) que nunca llegará a su destino, puesto que el cliente ya ha sido silenciado.

A continuación, Mitnick "crea" manualmente un paquete IP que contiene el número de secuencia del servidor + 1 (ACK predecible) falseando la dirección de origen por la del cliente autorizado.

En este punto la respuesta del servidor (paso 3 de la figura 3-18) **NO** es vista por Mitnick, ya que al falsear la dirección de origen nunca le llegará la respuesta. Se "confía" en que este paquete del servidor se ha enviado y que el ataque de SYN FLOOD sea exitoso y silencie al cliente autorizado, ya que si no respondería a esta petición con un RESET.

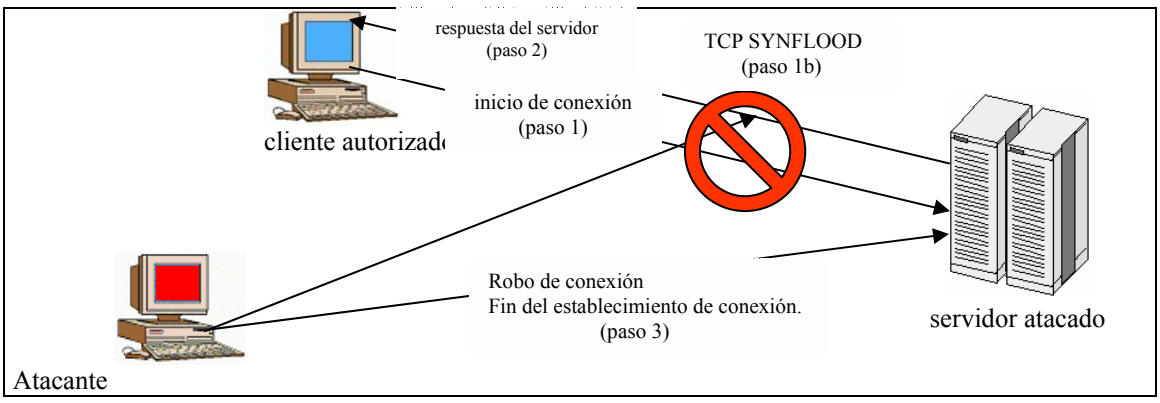


FIG 3-18: Ataque de Kevin Mitnick.

Una vez conocidos y validados los números de secuencia de la conexión TCP, el atacante ejecuta el siguiente comando [WWW106]:

```
rsh x-terminal "echo + + >>/.rhosts
```

Que permite a cualquier usuario iniciar una conexión X-terminal como root desde cualquier máquina. Una vez llegado a este extremo, el atacante tiene acceso total a la máquina.

## 3.12 RESUMEN

En este capítulo dedicado a los sistemas de detección de intrusos o IDS, hemos definido los conceptos de IDS, NIDS y IPS. Se han definido las distintas arquitecturas o configuraciones que forman cada uno de estos sistemas.

Se ha comentado la importancia de introducir filtros y firmas que nos permitan evitar el imposible trabajo de filtrar todo el tráfico de la red en tiempo real reduciéndolo al análisis de los eventos de interés. También se han comentado los distintos standares propuestos para la comunicación de los distintos sistemas NIDS (CIDF, CIDF, OPSEC...).

Además se han descrito las problemáticas asociadas al análisis de los datos obtenidos por los sistemas de detección de intrusos y las consecuencias asociadas (falsos positivos y falsos negativos). También se han comentado los sistemas de prevención de intrusos (IPS), que se vislumbran como una de las posibles líneas de evolución de los sistemas NIDS.

Finalmente se realiza una explicación del famoso ataque de Kevin Mitnick, considerado el umbral mínimo de detección para un sistema IDS.