A lightweight Intrusion Detection System for Networks

www.snort.org

Contenuto della presentazione

- Introduzione a Snort e al suo funzionamento
- Costruzione ed uso delle regole
- Analisi dell'architettura
- Test

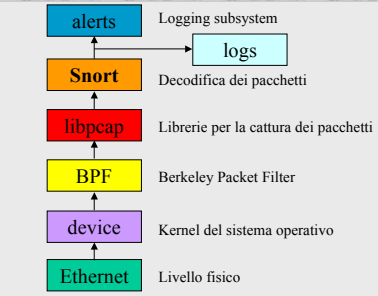
Un po' di storia

- L'autore è Martin Roesch
- L'idea nasce dal programma ***Ip-grab*** di Mike Borella.
- La prima release è datata 21/12/1998
- La svolta c'è l' 8/12/1999 quando viene aggiunta l'architettura modulare. La versione era la 1.5
- Oggi Snort è alla versione 1.7 (stabile), mentre la release beta è alla 1.8.3 ed il team di sviluppo ufficiale comprende una decina di persone sparse in tutto il mondo.

Caratteristiche principali

- **Leggerezza** (eseguibile è circa 250kb)
- **Portabilità** (Unix, Linux, Windows, *BSD, HP-UX, IRIX)
- **Velocità**
- **Configurabilità** (grazie alla semplicità del linguaggio delle regole e alle varie opzioni per il log)
- **Modularità** (possibilità di inserire dei plugin)
- **La licenza GPL/Open Source software**

Introduzione a Snort



Struttura semplificata di Snort

Uso di Snort

Snort può essere usato in diversi modi:

- Come packet sniffer (tipo tcpdump)
- Come packet logger
- Come Network IDS

Tipi di alert

Snort può creare diversi tipi di alert:

- Messaggi diretti al syslog
- File di testo in un percorso specificato dall'utente
- Messaggi inviati ad un socket Unix
- WinPopUp Message per i client Windows che usano il sistema Samba

Tipi di log

I file di log possono avere i seguenti formati:

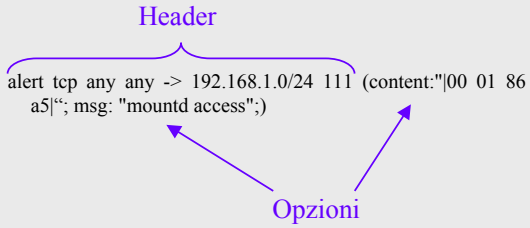
- Formato binario tcpdump (veloce ma difficile da analizzare)
- File divisi per directory in base all'indirizzo IP dell'host attaccato

Queste opzioni potrebbero considerarsi obsolete in quanto sostituite dai plugin output

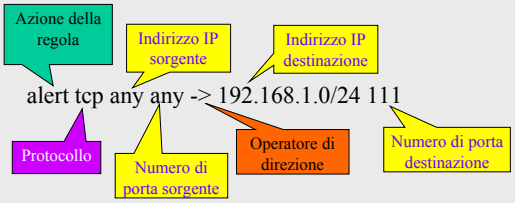
Le regole di Snort

Snort usa un linguaggio semplice e potente per la costruzione delle regole.

Un semplice esempio di regola



Header della regola



Le azioni della regola

Le possibili azioni sono:

- **Alert** – genera un alert usando il metodo di alert selezionato, e poi fa il log del pacchetto
- **Log** – fa il log del pacchetto
- **Pass** – ignora il pacchetto
- **Activate** – genera un alert e poi attiva una regola *dynamic*
- **Dynamic** – rimane inattiva finché non è attivata da una regola *activate*, poi agisce come un *log*

I protocolli

Attualmente ci sono tre protocolli IP che Snort può analizzare:

• TCP

• UDP

• ICMP

Gli indirizzi IP e le porte

- Gli indirizzi IP vengono scritti come numeri decimali seguiti dal blocco CIDR che indica la netmask:

  Es: 192.168.0.0/24

- Per le porte viene usato il numero decimale ad esse corrispondente. Può essere indicato anche un intervallo usando l'operatore " : ".

  Es: 1:1024

In entrambi i casi si può usare la parola **"any",** la quale indica che viene accettato qualsiasi valore

L'operatore di direzione

Indica la direzione del traffico a cui viene applicata la regola:

• -> Dalla sorgente alla destinazione

• <- Dalla destinazione alla sorgente

• < > In entrambe le direzioni

Opzioni della regola

Quali parti del pacchetto devono essere ispezionate per determinare se devono essere intraprese le azioni previste dalla regola.

(content:"|00 01 86 a5|"; msg: "mountd access";)

Il messaggio di alert.

Elenco delle opzioni

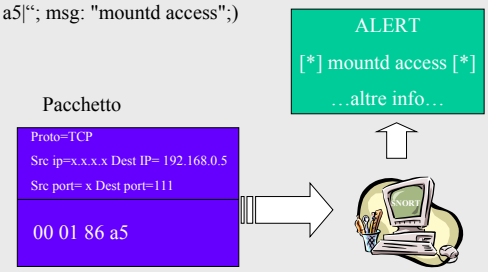
- **msg** – stampa un messaggio negli alert e nei log del pacchetto
- **logto** – Crea il log del pacchetto in un file specificato dall'utente invece che sullo standard output
- **ttl** – testa il valore del campo TTL nell'header IP
- **tos** – testa il valore del campo TOS nell'header IP
- **id** – testa il campo fragment ID nell'header IP per uno specifico valore
- **ipoption** – testa il valore dei campi IP option per cercare un codice specifico
- **fragbits** – testa i bit di frammentazione dell'header IP
- **dsize** – confronta la taglia del carico (payload) di un pacchetto con un determinato valore
- **flags** – testa i flag TCP per alcuni valori
- **seq** – testa il valore del campo sequenze number TCP con un determinato valore

- **ack** – testa il valore del campo acknowledgement TCP con un determinato valore
- **itype** – testa il valore del campo type ICMP con un determinato valore
- **icode** – testa il valore del campo code ICMP con un determinato valore
- **icmp_id** – testa il valore del campo ID ICMP ECHO con un determinato valore
- **icmp_seq** – testa il valore del campo sequence number ICMP ECHO con un determinato valore
- **content** – cerca un determinato pattern nel payload del pacchetto
- **content-list** – cerca un insieme di pattern nel payload del pacchetto
- **offset** – modificatore per l'opzione content, setta l'offset da cui iniziare un pattern match
- **depth** – modificatore per l'opzione content, setta la massima profondità cui effettuare un tentativo di pattern matching

- **nocase** – fa il match della stringa specificata dall'opzione content senza far differenza tra lettere maiuscole e minuscole
- **session** – scarta tutte le informazioni aggiunte dal livello applicazione per un data sessione
- **rpc** – controlla i servizi RPC per specifiche applicazioni/procedure chiamate
- **resp** – causa una risposta attiva (interrompe le connessioni, ecc.)
- **react** – causa una risposta attiva (blocca siti web)

Un semplice esempio di regola

alert tcp any any -> 192.168.0.0/24 111 (content:"|00 01 86 a5|"; msg: "mountd access";)



Definizione di variabili ed inclusioni di file

Nel file delle regole è possibile:

- includere altri file di regole
  es: include backdoor-lib
- dichiarare delle variabili globali
  es: var HOME_NET 192.168.0.0/24

Reperimento delle regole

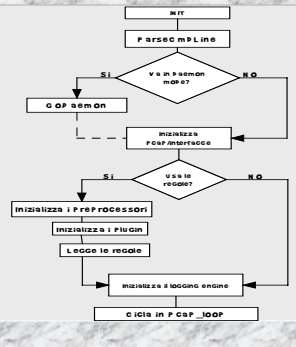
I file delle regole sono periodicamente aggiornati e sono scaricabili dal sito ufficiale di Snort http://www.snort.org/.

L'architettura di Snort

La struttura interna di Snort può essere divisa in quattro moduli principali che sono inizializzati all'avvio di Snort:

- *packet capturing e decoding engine*
- *rules parsing e detection engine*
- *logging engine*
- *plugin e preprocessor engine*

Avvio e inizializzazione

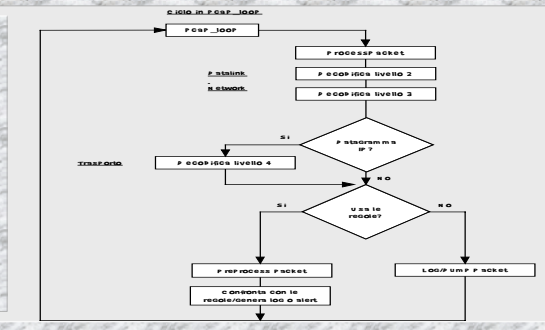


Packet capturing e decoding engine

Modulo per la cattura e la decodifica dei pacchetti.

Il cuore di questo modulo si trova nel ciclo **pcap_loop**

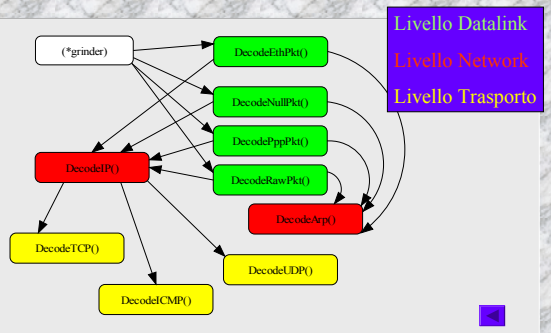
Ciclo pcap_loop



La libreria libpcap

Per la cattura dei pacchetti Snort usa le funzioni offerte dalla libreria **libpcap**.

Decodifica dei pacchetti: diagramma



Rules parsing e detection engine

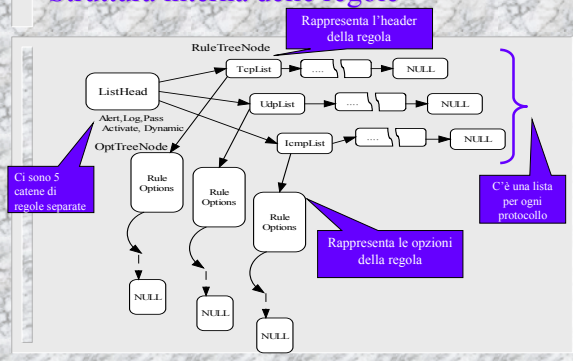
Questo modulo può essere a sua volta diviso in due sottomoduli:

- Il traduttore delle regole
- Il modulo di rilevamento (detection engine)

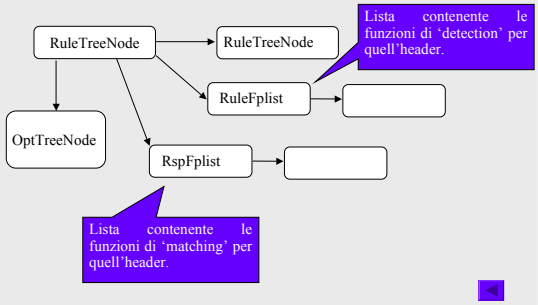
Traduttore delle regole

- Viene attivato nella fase di inizializzazione.
- Traduce le regole presenti in un file di testo in una struttura interna al programma.

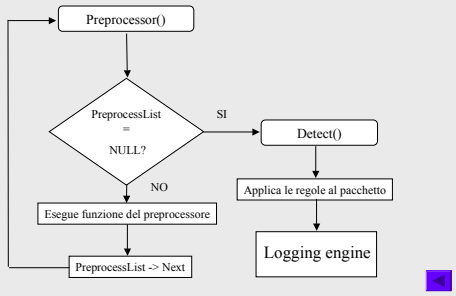
Struttura interna delle regole



Struttura *RuleTreeNode*

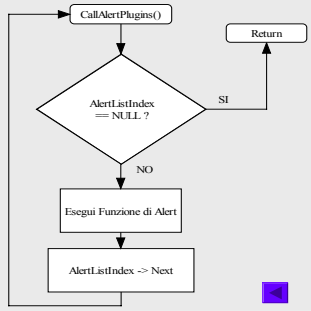


Detection engine



Logging engine

Si occupa della creazione dei log prodotti dai messaggi di alert generati da una regola.



Plugin e preprocessor engine

Questo modulo si occupa della gestione dei diversi tipi di moduli add-on previsti per Snort. Essi sono:

- Preprocessori
- Plugin
- Output plugin

Registrazione e inizializzazione dei plugin

Il modulo offre funzioni per la registrazione :

- *RegisterPlugin(),RegisterPreprocessor() e RegisterOutputPlugin()*

e funzioni per l'inizializzazione dei moduli:

- *InitPlugins(), InitPreprocessors(), InitOutputPlugins()*

Snort Test

Il test è stato eseguito con:

- attacchi CGI
- Tool per la generazione di falsi positivi
- attacchi Denial of Service.

Uso di Snort con un insieme di regole

Vediamo un esempio di regola per rispondere alle richieste ICMP.

```
alert icmp any any -> $HOME_NET any (msg:"IDS166 - PING Seer
Windows";
content:"|8804202020202020202020202020202020|";itype:8;depth:32;)
```

```
alert icmp any any -> $HOME_NET any (msg:"IDS152 - PING BSD";
content: "|08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17|";
itype: 8; depth: 32;)
```

Esempio di esecuzione di Snort

 # snort –i eth0 –h 193.205.162.1/24 –c ping.rules –d –l ./log –s –D

-d dumps the application layer

-h home network

-i log location

-c rule set name

-s log alerts to syslog

-i interface to listen on

-D run snort as a background process (daemon mode)

Questi sotto sono i messaggi che Snort potrebbe produrre in seguito ad un attacco rilevato con le regole precedenti

Apr 19 11:18:22 [**] IDS152 – PING BSD: 193.205.162.191 -> 193.205.162.192 [**]

Apr 19 11:18:23 [**] IDS152 – PING BSD: 193.205.162.191 -> 193.205.162.192 [**]

File di log generato da snort.

```
[**] IDS152 – PING BSD [**]
04/19-11:18:22.211702 193.205.162.191:200 -> 193.205.162.1
ICMP TTL:49 TOS:0x0 ID:17933
ID:48507  Seq:0 ECHO
06 28 6D 39 5A C8 04 00 08 09 0A 0B 0C 0D 0E 0F .(m9Z.....
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F ....
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F  !"#$%&'()*+,-./
30 31 32 33 34 35 36 37                          01234567
```

Per informazioni aggiuntive su un attacco si può visitare il sito http://www.whitehats.com/ ed effettuare una ricerca per il numero IDS visualizzato. In questo caso, una ricerca per IDS152.

Web-scanner test

Lo scopo di qualunque tattica di anti-IDS consiste in:

mutare una richiesta in modo tale che l'IDS venga ingannato, ma il web server risulti in grado di riconoscerla.

Richiesta HTTP

Composizione di una richiesta HTTP (rif . RFC1945):

```
[ GET /cgi-bin/some.cgi HTTP/1.0 ]
  |-1-||------2-----| |---3---|
  |--------------4--------------|
```

1. Il **metodo della richiesta**, tipicamente **GET, HEAD, POST**, etc.Indica che formato dei dati il server deve aspettarsi e fornire.
2. L'**URI.** Può essere relativo ('/some/file') o assoluto ('http://server/some/file').
3. La **versione**, che è sempre nel formato "**HTTP/x.x**".
4. Le tre componenti formano una **"richiesta"**.

Tattiche anti IDS

Analizzeremo le seguenti tattiche anti IDS:

- Null method processing
- URL encoding
- Self reference directory
- Long URL
- Session splicing

Null method processing

Si usa il carattere NULL (%00) di fine stringa

Es : HEAD%00 /cgi-bin/some.cgi HTTP/1.0

Lo funzionamento è:

- **Il Web server riceve la richiesta, separando il metodo dall'URI**
- **Decodifica il metodo e l'URI**
- **Il metodo è valido per il web server anche con il NULL**
- **L'IDS decodifica l'intera richiesta**
- **L'IDS cerca di applicare operazioni di stringa sulla richiesta, fermandosi quando il NULL viene raggiunto**

Null method processing: risposta Snort

Sebbene l'Apache non processa nessuna richiesta che contiene '%00' o '%2f' Snort non ha nessun problema nella rilevazione di questo tipo di attacco:

```
[**] spp_http_decode: CGI Null Byte attack detected [**]
05/08-14:51:10.802958 192.205.162.191:1226 -> 192.205.162.192:80
TCP TTL:64 TOS:0x0 ID:2207 IpLen:20 DgmLen:162 DF
***AP*** Seq: 0x5211238A  Ack: 0x524142AF Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 166576 166575

[**] spp_http_decode: CGI Null Byte attack detected [**]
05/08-14:51:10.802958 192.205.162.191:1226 -> 192.205.162.191:80
TCP TTL:64 TOS:0x0 ID:2207 IpLen:20 DgmLen:162 DF
***AP*** Seq: 0x5211238A  Ack: 0x524142AF Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 166576 166575

[**] IDS235 - CVE-1999-0148 - CGI-HANDLERprobe! [**]
05/08-14:51:11.124244 192.205.162.191:1233 -> 192.205.162.192:80
TCP TTL:64 TOS:0x0 ID:2284 IpLen:20 DgmLen:177 DF
***AP*** Seq: 0x529A86A2  Ack: 0x51FB17F7 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 166608 166608

[**] IDS235 - CVE-1999-0148 - CGI-HANDLERprobe! [**]
05/08-14:51:11.124244 192.205.162.191:1233 -> 192.205.162.192:80
TCP TTL:64 TOS:0x0 ID:2284 IpLen:20 DgmLen:177 DF
***AP*** Seq: 0x529A86A2  Ack: 0x51FB17F7 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 166608 166608
```

URL encoding

Questa tecnica consiste nel codificare l'URI con il suo equivalente escape, usando la notazione `%xx`, dove `'xx'` è il valore esadecimale del carattere.

URL encoding: risposta Snort

Poiché Snort effettua una decodifica dell'URI in maniera del tutto simile ad un web server, prima di effettuare un controllo reale delle segnature, tale attacco viene rilevato senza alcun problema:

```
[**] IDS228 - CVE-1999-0237 - Guestbook CGI access attempt [**]
05/08-14:56:11.289502 193.205.162.191:2524 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:16486 IpLen:20 DgmLen:220 DF
***AP*** Seq: 0x65CC0789  Ack: 0x64E69C74 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 196625 196625

[**] IDS228 - CVE-1999-0237 - Guestbook CGI access attempt [**]
05/08-14:56:11.289502 193.205.162.191:2524 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:16486 IpLen:20 DgmLen:220 DF
***AP*** Seq: 0x65CC0789  Ack: 0x64E69C74 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 196625 196625

[**] IDS272 - Piranha Passwd.php3 [**]
05/08-14:56:11.443289 193.205.162.191:2540 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:16662 IpLen:20 DgmLen:233 DF
***AP*** Seq: 0x650196CF  Ack: 0x65A6791D Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 196640 196640

[**] IDS272 - Piranha Passwd.php3 [**]
05/08-14:56:11.443289 193.205.162.191:2540 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:16662 IpLen:20 DgmLen:233 DF
***AP*** Seq: 0x650196CF  Ack: 0x65A6791D Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 196640 196640
```

Self-reference directories

Una stringa del tipo "/cgi-bin/phf", viene cambiata nell'equivalente "/./cgi-bin/./phf". Questo significa che l'IDS ha tre opzioni:

· Notificare un alert in presenza di un /./

· Disinteressarsi; questo produce perdite di attacchi

· Rimpiazzare logicamente '/./' con **'/'**

Self-reference directories: risposta Snort

Anche in questo caso Snort non presenta nessun problema nella rilevazione di tali attacchi:

```
[**] IDS234 - WEB-CGI-Cgiwrap CGI access attempt [**]
05/08-14:59:01.422470 193.205.162.191:3317 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:25228 IpLen:20 DgmLen:176 DF
***AP*** Seq: 0x7076D873  Ack: 0x70C11A55 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 213638 213638

[**] IDS207 - WEB-MISC - Phorum Code [**]
05/08-14:59:01.557699 193.205.162.191:3330 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:25371 IpLen:20 DgmLen:180 DF
***AP*** Seq: 0x700FA517  Ack: 0x70CA7DF2 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 213652 213649

[**] IDS207 - WEB-MISC - Phorum Code [**]
05/08-14:59:01.557699 193.205.162.191:3330 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:25371 IpLen:20 DgmLen:180 DF
***AP*** Seq: 0x700FA517  Ack: 0x70CA7DF2 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 213652 213649

[**] IDS272 - Piranha Passwd.php3 [**]
05/08-14:59:01.598627 193.205.162.191:3334 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:25415 IpLen:20 DgmLen:191 DF
***AP*** Seq: 0x70E3FEC7  Ack: 0x701FF1D4 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 213656 213656
```

Long URL

Questa tecnica si basa sul fatto che gli IDS di solito controllano i primi xx byte della richiesta, quindi si includono abbastanza caratteri per muovere il resto della richiesta fuori dalla portata dell'IDS.

GET /rfprfp<molti caratteri>rfprfp/../cgi-bin/some.cgi HTTP/1.0

Long URL: risposta Snort

Non presenta alcun problema nel rilevamento.

```
[**] IDS205 - WEB-MISC - Phorum Admin [**]
05/08-15:06:29.695693 193.205.162.191:1339 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:47270 IpLen:20 DgmLen:1597 DF
***AP*** Seq: 0x8C4DD69A  Ack: 0x8C75D9B6 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 258465 258465

[**] IDS205 - WEB-MISC - Phorum Admin [**]
05/08-15:06:29.695693 193.205.162.191:1339 -> 193.205.162.192:80
TCP TTL:64 TOS:0x0 ID:47270 IpLen:20 DgmLen:1597 DF
***AP*** Seq: 0x8C4DD69A  Ack: 0x8C75D9B6 Win: 0x7960 TcpLen: 32
TCP Options (3) => NOP NOP TS: 258465 258465
```

Session splicing

Questa tecnica consiste nel trasmettere parti della richiesta in pacchetti differenti. Per esempio, la richiesta

"GET / HTTP/1.0"

può essere spezzata attraverso pacchetti multipli nel modo seguente:

"GE", "T", "/", "H", "T", "TP", "/1", ".0".

La difesa appropriata per questa tecnica è il riassemblamento della sessione.

Risposta Snort

Snort non rileva questo tipo di attacco.

False Positive Generator

È stato usato un tool per testare Snort in presenza attacchi fasulli.

Risultato:

Snort ha generato un gran numero di falsi positivi.

Falsi positivi generati da Snort

```
[**] IDS128 - CVE-1999-0067 - CGI phf attempt [**]
05/09-16:58:49.249670 193.205.162.191:1641 -> 193.205.162.192:80
TCP TTL:4 TOS:0xE ID:8991 IpLen:20 DgmLen:79 DF
***AP*** Seq: 0x64ED624B  Ack: 0x73D4EFFA  Win: 0x7D78  TcpLen: 32
TCP Options (3) => NOP NOP TS: 15687837 191167785


[**] spp_portscan: PORTSCAN DETECTED from 193.205.162.191 (THRESHOLD 4
connections exceeded in 2 seconds) [**]
05/09-16:58:50.708946
[**] IDS126 - Outgoing Xterm [**]
05/09-16:58:50.302954 193.205.162.191:6000 -> 193.205.162.192:1677
TCP TTL:64 TOS:0x0 ID:25274 IpLen:20 DgmLen:60 DF
***A**S* Seq: 0x72E7AE1F  Ack: 0xEC25E2D4  Win: 0x7D78  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 663026 15902473 NOP WS: 0


[**] IDS126 - Outgoing Xterm [**]
05/09-16:58:50.302954 193.205.162.191:6000 -> 193.205.162.192:1677
TCP TTL:64 TOS:0x0 ID:25274 IpLen:20 DgmLen:60 DF
***A**S* Seq: 0x72E7AE1F  Ack: 0xEC25E2D4  Win: 0x7D78  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 663026 15902473 NOP WS: 0

[**] IDS187 - DDoS - TrinOO [**]
05/09-16:58:50.829536 193.205.162.191:1249 -> 193.205.162.192:31335
UDP TTL:4 TOS:0xE ID:9848 IpLen:20 DgmLen:33
Len: 13
```

Denial of Service

Snort è stato in grado di rilevare tutti gli attacchi DoS effettuati.

Per uno di questi si presenta un crash di Snort (sottoforma di segmantation fault).

Tale attacco utilizza il protocollo ICMP e può essere considerato simile al più famoso **Ping of Death**.

Viene riportato di seguito il rilevamento dell'attacco prima che Snort vada in crash:

```
[**] IDS246 - MISC - Large ICMP Packet [**]
05/10-14:42:31.084488 77.208.151.47 -> 193.205.162.192
ICMP TTL:255 TOS:0x10 ID:1 IpLen:24 DgmLen:1024
IP Options (1) => Opt 5:
Type:90  Code:10  UNKNOWN

[**] IDS246 - MISC - Large ICMP Packet [**]
05/10-14:42:31.084488 77.208.151.47 -> 193.205.162.192
ICMP TTL:255 TOS:0x10 ID:1 IpLen:24 DgmLen:1024
IP Options (1) => Opt 5:
Type:90  Code:10  UNKNOWN
```

Conclusioni

Pregi:

- **Leggerezza**
- **La modularità**
- **Codice Open Source**

Difetti

- **Tutti quelli comuni agli IDS basti sul pattern matching**
- **Mancanza di livelli per gli alert**

Valutazione come NIDS: **BUONA**

Autori

Gerardo Donnamaria *gdonnam@tin.it*

Luigi Monaco *luimon@libero.it*