



13 Snort-Inline

13.1 Einführung

Snort-Inline ist ein Intrusion-Prevention-System (IPS). Häufig werden diese Systeme auch Gateway-IDS genannt. Snort-Inline ist ein modifiziertes Snort, welches seine Pakete nicht von den *libpcap*-Bibliotheken, sondern von *iptables* erhält. Hierzu verwendet Snort-Inline das *QUEUE*-Target und die *libipq*-Bibliothek.

Für die erweiterte Funktionalität wurde Snort-Inline um einige neue Regeltypen erweitert: *drop*, *sdrop* und *reject*.

Die aktuelle Version von Snort-Inline zum Zeitpunkt der Drucklegung ist 2.2.0 und wird nach Jed Hail und Rob McMillen nun von William Metcalf gepflegt. Sie können diese Version von der Homepage <http://snort-inline.sourceforge.net/> herunterladen.

Da die wesentliche Entwicklung auch durch das HoneyNet Project vorangetrieben wurde, wird auf deren Homepage (<http://www.honeynet.org>) auch ein statisch kompiliertes Snort-Inline mit fertigen Konfigurationsdateien angeboten. Diese Konfigurationsdateien sind allerdings auf den Einsatz in einem HoneyNet optimiert. Das bedeutet, sie analysieren und überwachen den ausgehenden Verkehr.



Achtung:

Snort wird ab der Version 2.4 wahrscheinlich die Snort-Inline-Funktionalität bereits enthalten. Der aktuelle Snort-2.3 CVS-Baum enthält bereits erste Erweiterungen in diese Richtung. Die wesentliche Entwicklung wird weiterhin durch das Snort-Inline-Projekt erfolgen. Neue Funktionen werden dann nach ihrer Stabilisierung in Snort übernommen werden.

**Tipp:**

Dieses Kapitel beschreibt die Installation und Anwendung von *Snort-Inline* in den Punkten, die es von dem Standard-Snort unterscheidet. Bitte lesen Sie auch das Snort-Kapitel in diesem Buch.

13.2 Installation von Snort-Inline

Die Installation von Snort-Inline ist recht einfach. Zunächst sollten Sie sicherstellen, dass Sie über eine aktuelle *iptables*-Installation verfügen. Wichtig ist insbesondere, dass Ihre Installation auch die *libipq*-Bibliothek einschließt. Wenn Sie *iptables* selbst aus den Quellen übersetzen, dann können Sie mit dem Befehl `make install-devel` dafür sorgen. Wenn Sie das *iptables*-Paket einer Distribution nutzen, so sollten Sie auch, wenn vorhanden, das *iptables-devel*-Paket installieren. Dieses enthält bei den meisten Distributionen die *libipq*-Bibliothek.

Zusätzlich benötigen Sie die *Libnet*-Bibliothek. Diese erhalten Sie von <http://www.packetfactory.net>, wenn sie nicht in Ihrer Distribution enthalten ist. Wichtig ist die Version der Bibliothek. *Snort-Inline* verlangt die ältere Version 1.0.x. Die aktuelle Version 1.1.x erzeugt Probleme.

Eine weitere Voraussetzung für den späteren Einsatz von Snort-Inline ist die Unterstützung des *QUEUE*-Targets durch den eingesetzten Linux-Kernel (*IP_NF_QUEUE*). Bei den meisten Distributionen können Sie das mit dem folgenden Befehl überprüfen:

```
$ sudo insmod ip_queue
```

Wenn dieser Befehl eine Fehlermeldung ausgibt, verfügt Ihr Kernel wahrscheinlich nicht über die entsprechende Unterstützung und Sie sollten die Konfiguration überprüfen (siehe Abbildung 13.1).

Anschließend können Sie *Snort-Inline* mit den folgenden Befehlen übersetzen:

```
$ ./configure
$ make
```

Auf Red-Hat-Distributionen tritt häufig der folgende Fehler bei der Übersetzung auf:

```
In file included from /usr/include/linux/netfilter_ipv4/ip_queue.h:10,
                 from /usr/include/libipq.h:37,
                 from ../../src/inline.h:8,
                 from ../../src/snort.h:38,
                 from spo_alert_fast.c:51:
/usr/include/linux/if.h:59: error: redefinition of "struct ifmap"
```

```
/usr/include/linux/ifa.h:77: error: redefinition of "struct ifreq"
/usr/include/linux/ifa.h:126: error: redefinition of "struct ifconf"
```

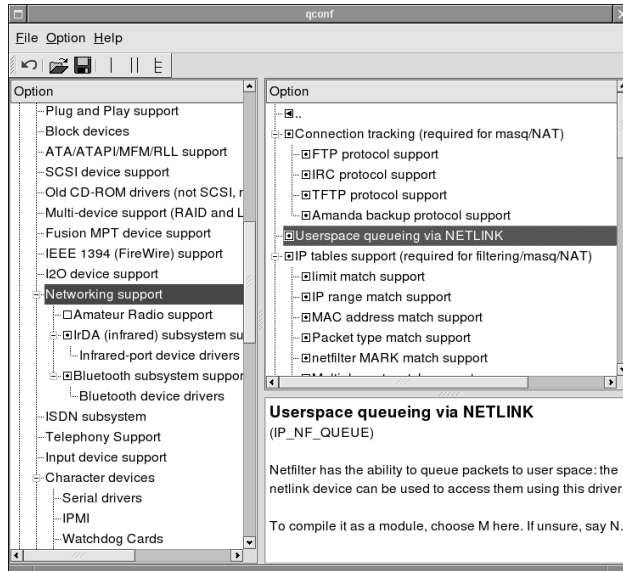


Abbildung 13.1 Der Kernel muss über das QUEUE-Target verfügen.

Hier liegt ein Problem mit den Kernel-Header-Dateien vor. Redhat verwendet nicht die originalen Kernel-Header sondern stellt die Kernel-Header in einem eigenen Paket (*glibc-kernheaders*) zur Verfügung. Sie können das Problem mit den folgenden Befehlen vermeiden.

```
$ cd /usr/include
$ sudo mv linux linux.orig
$ sudo ln /usr/src/linux-version/include/linux linux
```

Anschließend übersetzen Sie Snort-Inline mit `make clean; make`. Ist die Übersetzung erfolgreich, so können Sie die Änderungen an dem `/usr/include/linux`-Verzeichnis wieder rückgängig machen.

Bei der Konfiguration können Sie folgende Argumente dem `configure`-Befehl übergeben:

- `--disable-inline`. Hiermit schalten Sie die Inline-Funktionalität aus.
- `--enable-clamav`. Dies aktiviert den Clamav-Antivirus-Präprozessor (siehe Abschnitt »Clamav« auf S. 470).

Falls die Bibliotheken und ihre Header-Dateien nicht unter `/usr` gespeichert wurden, können Sie mit den folgenden Parametern ihren Ort angeben.

- `--with-libnet-includes=DIR.`
- `--with-libnet-libraries=DIR.`
- `--with-libipq-includes=DIR.`
- `--with-libipq-libraries=DIR.`
- `--with-clamav-includes=DIR`
- `--with-clamav-defdir=DIR`

Alle weiteren Konfigurationsoptionen werden Ihnen bei dem Aufruf von `./configure --help` angezeigt und werden in dem Snort-Kapitel besprochen.

Die Installation von Snort erfolgt nach der Übersetzung mit einem einfachen `make install`.

13.3 Anpassung der Snort-Regeln

Snort-Inline verfügt gegenüber Snort über drei weitere Regeltypen:

- `drop`. Snort-Inline beauftragt *iptables*, das Paket zu verwerfen und protokolliert es gleichzeitig.
- `sdrop`. Snort-Inline beauftragt *iptables*, das Paket zu verwerfen. Es erfolgt keine Protokollierung (silent drop)
- `reject`. Snort-Inline beauftragt *iptables*, das Paket zu verwerfen. Es erfolgt eine Protokollierung. Handelte es sich um ein TCP-Paket, so erzeugt Snort zusätzlich ein TCP-RST-Paket. Im Falle von UDP erfolgt die Ablehnung mit einem ICMP-Port-Unreachable-Paket.

Die Ablehnung einer Verbindung mit `reject` kann zusätzlich mit der Direktive `config layer2resets` konfiguriert werden. Dies ist insbesondere bei dem Einsatz von Snort-Inline auf einer Bridge interessant. Hiermit können Sie die Source-MAC-Adresse für das Reject-Paket angeben.

Die Reihenfolge der Regelabarbeitung ist nun:

```
activation->dynamic->drop->sdrop->reject->alert->pass->log
```

Wird Snort-Inline mit der Option `-o` aufgerufen, so werden `log`-Regeln zwischen `dynamic` und `drop` abgearbeitet.

Snort-Inline verfügt über zwei weitere Optionen, die den Einsatz von Snort-Inline als Gateway-IDS ermöglichen.

13.3.1 Paketmodifikation

Wenn Snort-Inline ein böses Paket erkennt, kann es auch den Inhalt verändern. Damit können Sie auch einen Angriff verhindern. Diese Option ist insbesondere interessant beim Einsatz in einem Honeypot. Ist ein Angreifer auf einem Honeypot ein-

gedrungen und versucht er weitere Rechner anzugreifen, so kann er den Angriff durchführen. Der Angriff wird jedoch durch Snort-Inline entschärft und ist daher nicht erfolgreich. Ein Honeypot kann somit nicht für den Angriff weiterer Systeme eingesetzt werden.

Hierzu müssen Sie mit einem `content`-Attribut zunächst den zu änderenden Inhalt suchen. Das `replace`-Attribut ermöglicht dann den Austausch des Inhalts. Die einzige Einschränkung bei der Anwendung des `replace`-Attributs ist die identische Länge.

Ein Beispiel:

```
alert tcp any any <> any 80 (msg:"Replace GET" content: "GET"; replace: "XXX");
```

13.3.2 TCP-Zustandsüberwachung

Ein Problem, welches den Einsatz des `stream4`-Präprozessors in Snort-Inline immer verhindert hat, ist der niedrige Timeout-Wert des Präprozessors für TCP-Verbindungen. Laut TCP-Standard ist der Timeout einer aufgebauten TCP-Verbindung fünf Tage. Das bedeutet, dass erst nach fünf Tagen, in denen sowohl der Client als auch der Server keine Daten mehr gesendet haben, die Verbindung beendet wird. Snort verwendet hier wesentlich kleinere Werte. Snort-Inline kann für diese Funktion nun direkt die Funktionalität von `iptables` nutzen und auf dessen Zustandsinformationen zugreifen. Die Zustandsüberwachung von `iptables` verwendet Timeout-Informationen entsprechend den Internet-Standards. Die Kommunikation zwischen `iptables` und dem `stream4`-Präprozessor erfolgt über die Paket-Markierung mit dem `iptables`-MARK-Target.

Hierfür hat der `stream4`-Präprozessor die folgenden neuen Parameter:

- `forceiptstate`. Der `stream4`-Präprozessor nutzt die Zustandsinformationen von `iptables`. Ist kein Zustand bekannt, so wird das Paket verworfen.
- `iptablesnewmark mark`. Dies weist den Präprozessor an, für jedes Paket mit der entsprechenden Markierung eine neue Sitzung zu starten. Bei der angegebenen Markierung kann es sich auch um einen Bereich handeln (z.B. 1-5). Wenn Sie hier keine Markierung spezifizieren, ist der Default-Wert 1. Sinnvoll ist diese Markierung für alle Pakete mit dem Zustand `NEW` und `RELATED`.
- `iptablenessmark`. Hiermit können Sie die Markierung für Pakete mit dem Zustand `ESTABLISHED` angeben. Default ist 2.

Ein einfaches Beispiel zeigt die Anwendung. Definieren Sie folgende `iptables`-Regeln:

```
iptables -t mangle -A FORWARD -p tcp --syn -m state --state NEW -j MARK
↳ --set-mark 100
iptables -t mangle -A FORWARD -p tcp --syn -m state --state ESTABLISHED -j
↳ MARK --set-mark 200
iptables -t filter -A FORWARD -j QUEUE
```

Konfigurieren Sie den Präprozessor entsprechend:

```
preprocessor stream4: iptablesnewmark 100, iptablesestmark 200, forceiptstate
```

13.3.3 Automatische Anpassung der Regeln

Brian Caswell hat für das HoneyNet Project das Werkzeug SnortConfig (<http://www.shmoo.com/~bmc/software/snortconfig/>) entwickelt. Dieses Werkzeug nimmt einen vorhandenen Snort-Regelsatz und wandelt diesen in einen Snort-Inline-Regelsatz um. Die Konfiguration erfolgt über eine einfache INI-Datei.

```
[files]
pass: porn.rules, virus.rules
disable: rpc.rules, icmp.rules

[sids]
disable: 2122, 1866, 2108, 2109

[classifications]
replace_or_drop: shellcode-detect
drop: kickass-porn, policy-violation
```

Ein weiteres Skript, welches für diesen Zweck eingesetzt werden kann, ist in dem Snort-Inline-Paket enthalten: *convert.sh*. Dieses Skript wandelt alle Snort-*alert*-Regeln in *drop*-Regeln um. Außerdem wird die Richtung der Regeln ausgetauscht, so dass der entstehende Regelsatz für den Einsatz in einem HoneyNet geeignet ist.

Tipp:

Auch Oinkmaster unterstützt ab der Version 0.8 bereits Snort-Inline.



13.4 Clamav

Clam Antivirus ist ein Open-Source-Antivirus-Scanner (<http://www.clamav.net/>). Snort-Inline kann diesen Antivirus-Scanner für die Analyse des Netzwerkverkehrs nutzen. Hierfür muss der Clam-Antivirus-Scanner auf dem System installiert werden. Die Installation wird sehr gut in dem ClamAV-Paket und auf der Homepage erklärt. Anschließend aktivieren Sie den Präprozessor mit:

```
preprocessor clamav
```

Damit die Aktivierung erfolgreich ist, ist es erforderlich, dass Sie bei der Konfiguration von Snort-Inline die Option `--enable-clamav` angegeben haben.

**Achtung:**

Der Präprozessor erkennt keine OLE2-Viren. Kodierte E-Mail-Anhänge können von ClamAV ebenfalls nicht erkannt werden.

Der Präprozessor verfügt über folgende weiteren Optionen:

- `ports liste`. Der Clamav-Präprozessor überwacht die angegebenen TCP-Ports. Default sind: 21, 25, 80, 81, 110, 119, 139, 143 und 445.
- `toclientonly|toserveronly`. Der Clamav-Präprozessor überwacht nur den Verkehr zum Client oder zum Server. Wird keine Option angegeben, beobachtet der Präprozessor beide Richtungen.
- `action-drop|action-reset`. Pakete, die einen Virus enthalten werden verworfen oder abgelehnt. Wenn Sie weder die eine noch die andere Option angeben, so lösen Viren lediglich eine Alarmierung aus.
- *db-Verzeichnis*. ClamAV verwendet das angegebene Viren-Signatur-Verzeichnis.

Die ClamAV-Erkennung ist selbstverständlich nur so gut, wie die verwendete Signaturrendatenbank. Außerdem ist die Virenerkennung sehr aufwändig. Daher sollte der Präprozessor sparsam eingesetzt werden.

**Tipp:**

Wenn Sie ClamAV nach dem `stream4`-Präprozessor aktivieren, dann kann ClamAV eine ganze TCP-Sitzung nach einem Virus durchsuchen. Ansonsten kann ClamAV maximal die Daten eines Paketes analysieren.

13.5 Einbindung als IPS mit iptables

Snort-Inline wird über das Userspace Queueing Interface via Netlink angesprochen. Das bedeutet, dass Sie mit *iptables*-Regeln den Netzwerkverkehr auswählen können, der von Snort-Inline betrachtet werden soll.

Daher ist die Integration von Snort-Inline in einer Firewall oder einem Router sehr einfach. Hierzu muss lediglich das *ip_queue*-Modul geladen werden und der zu analysierende Verkehr mit dem *iptables*-MARK-Target an Snort-Inline gesendet werden.

Die folgenden beiden Zeilen genügen bereits:

```
# insmod ip_queue
# iptables -A FORWARD -j QUEUE
```

Nun müssen Sie lediglich Snort-Inline starten. Bei den Start-Optionen unterscheidet sich Snort-Inline lediglich in einer Option von Snort:

- `-Q`. Diese Option veranlasst Snort-Inline die Pakete von *iptables* entgegenzunehmen. Die *Libpcap*-Bibliothek wird nicht genutzt.

Wenn nun bei dem Start von Snort-Inline folgender Fehler auftritt, ist das *ip_queue*-Modul nicht geladen. Überprüfen Sie das bitte mit dem `lsmod`-Befehl.

```
Reading from iptables
Running in IDS mode
Log directory = /var/log/snort
Initializing Inline mode
InitInline: : Failed to send netlink message: Connection refused
```

Ist alles in Ordnung, so können Sie Snort-Inline mit dem folgenden Befehl starten:

```
snort_inline -Qdc snort_inline.conf
```



Tipp:

Das Honeynet Project pflegt auf seiner Homepage unter http://www.honeynet.org/tools/dcontrol/snort_inline.sh ein Start-Skript für Snort-Inline.



Tipp:

Snort-Inline kann auch auf einer Bridge eingesetzt werden. Hierzu ist dann aber bei Einsatz des Linux Kernels 2.4 ein spezieller Patch erforderlich (<http://ebtables.sf.net>). Dieser Patch ist in dem Kernel 2.6 bereits enthalten und ermöglicht das Filtern von IP-Paketen auf einer Bridge.

