mxallowd-Dokumentation

Michael Stapelberg

Version 1.7, Dokumentation 24. September 2019

Inhaltsverzeichnis

1 Einleitung

mxallowd ist ein Daemon für Linux/Netfilter (via libnetfilter_queue) oder BSD/pf (via pflog), der eine Verfeinerung der nolisting-Methode darstellt. Hierbei werden für eine Domain zwei MX-Einträge vom Nameserver ausgeliefert, wobei auf der IP-Adresse des ersten MX-Eintrags kein Mailserver läuft. Einige Spammer versuchen nun, nur auf den ersten Mailserver Spam auszuliefern und werden damit keinen Erfolg haben. Auf der IP-Adresse des zweiten MX-Eintrags läuft dann ein richtiger Mailserver, der die E-Mails entgegennimmt. Echte Mailserver probieren – im Gegensatz zu Spammern – alle MX-Einträge in der angegeben Reihenfolge (geordnet nach Priorität) durch, bis sie die Mail zustellen können. Somit kommen echte Mails an und Spam bleibt draußen.

Das Problem beim nolisting ist nun, dass einige Spammer (vermutlich aufgrunddessen) direkt den zweiten MX-Eintrag benutzen („direct-to-second-mx"). Hier kommt nun *mxallowd* ins Spiel: Auf den zweiten Mailserver darf man sich nicht verbinden (das Paket wird einfach via netfilter/iptables verworfen), außer, wenn man es zuvor beim ersten Mailserver probiert hat.

Dieses Problem hätte man prinzipiell auch nur via iptables mit dem Modul *ipt_recent* lösen können, wenn es nicht ein kleines Problem dabei gäbe: Einige Anbieter (wie zum Beispiel Google Mail) verwenden zwar den gleichen DNS-Namen, aber unterschiedliche IP-Adressen im selben Zustellzyklus. Das heißt, dass *ipt_recent*, welches ausschließlich auf IP-Adress-Basis arbeitet, E-Mails von Google nicht durchlässt. *mxallowd* fügt daher alle IP-Adressen des DNS-Eintrags in die Whitelist ein (außer, wenn man die Option −−no-rdns-whitelist angibt).

1.1 Lizenz

mxallowd ist freie Open-Source-Software unter der GPLv2[1].

1.2 Entwicklung

Den momentanen Stand der Entwicklung kann man auf https://code.stapelberg.de/mxallowd/ begutachten.

Über Patches, Feature-Requests oder Bugreports bin ich natürlich erfreut, meine E-Mail-Adresse und meinen GPG-Key findest du auf http://michael.stapelberg.de/Kontakt.

[1] http://www.gnu.org/licenses/old-licenses/gpl-2.0.html

1.3 Funktionsweise

Auf folgender Grafik ist abgebildet, wie ein Verbindungsaufbau ablaufen kann. Dabei ist mit „Spammer“ ein nicht-RFC-konformer Mailsender gemeint und mit „Mailserver“ ein RFC-konformer. Das SMTP-RFC schreibt vor, dass ein Mailsender alle MX-Einträge in der angegebenen Priorität durchprobieren muss, um die Mail zu versenden.

Der Spammer verbindet sich in dieser Grafik entweder direkt mit dem ersten (falschen) Mailserver oder direkt mit dem zweiten, während der Mailserver beide nacheinander durchprobiert.

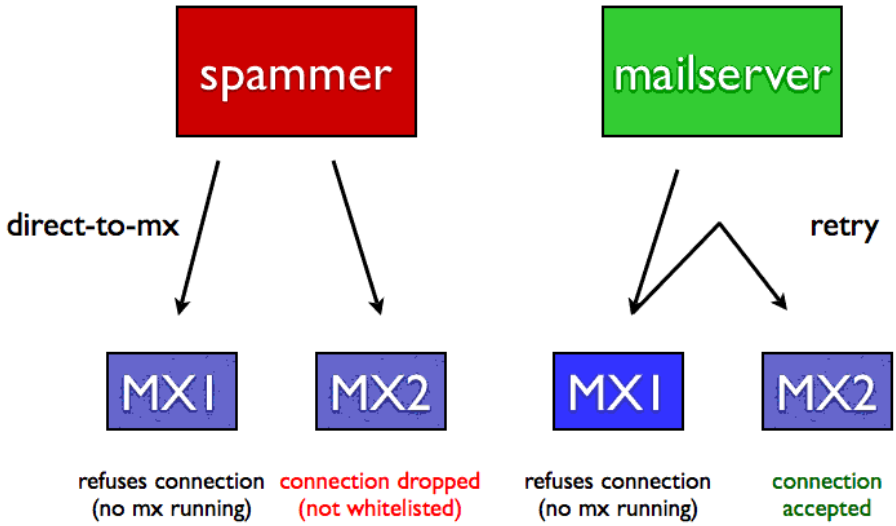

Abb. 1: Funktionsweise

2 Installation unter Linux

Damit neue Verbindungen an *mxallowd* geleitet werden, muss man folgende iptables-Regel hinzufügen:

```
iptables -A INPUT -p tcp --dport 25 -m state --state NEW \
         -j NFQUEUE --queue-num 23
```

Falls das Einfügen dieser Regel nicht klappt, muss zuvor das Queue-Modul geladen werden:

```
modprobe nfnetlink_queue
```

Die Regel kann man selbstverständlich anpassen, sodass zum Beispiel nur an bestimmte IP-Adressen gerichtete Verbindungen gefiltert werden, oder dass Verbindungen von bestimmten IP-Adressen von vorneherein akzeptiert werden (-j ACCEPT am Ende).

Seit Version 1.5 funktioniert das auch mit ip6tables und IPv6-Verbindungen.

3 Installation unter BSD

Eine */etc/pf.conf* könnte so aussehen:

```
table <mx-white> persist

real_mailserver="192.168.1.4"
fake_mailserver="192.168.1.3"

real_mailserver6="2001:dead:beef::1"
fake_mailserver6="2001:dead:beef::2"

pass in quick log on fxp0 proto tcp from <mx-white> \
             to $real_mailserver port smtp
pass in quick log on fxp0 inet6 proto tcp from <mx-white> \
             to $real_mailserver6 port smtp
block in log on fxp0 proto tcp \
              to { $fake_mailserver $real_mailserver } port smtp
block in log on fxp0 inet6 proto tcp \
              to { $fake_mailserver6 $real_mailserver6 } port smtp
```

Wichtig dabei ist, dass die Table *mx-white* existiert und dass sowohl die pass- als auch die block-Regeln loggen.
Wenn man ein anderes pflog-interface verwendet, kann man mxallowd das via Parameter mitteilen.

4 Hilfe, ich kann keine Mails mehr versenden!

Das stimmt – wenn du den selben Mailserver auch verwendest, um Mails zu versenden, probiert dein Mailclient in der Regel nur eine Verbindung. Ich würde raten, die Mails über SMTPS (SSL) zu versenden, denn dieser Port (465) wird nicht von *mxallowd* gefiltert. Ansonsten kannst du deinen Mailserver auch zusätzlich auf einem anderen Port laufen lassen, den nur zu benutzt (Spammer treiben nicht den Aufwand, einen Portscan durchzuführen, wenn sie nicht mal standardkonforme Mailer verwenden...). Falls du eine fixe IP-Adresse hast, kannst du diese natürlich auch via iptables whitelisten:

```
iptables -I INPUT 1 -p tcp --dport 25 --s 192.168.2.3 -j ACCEPT
```