

Greylisting auf dem GUUG-Server

Martin Schulte

Mitte Oktober hat Jochen Topf Greylisting auf dem GUUG-Mailserver installiert - eine guter Anlass, hier mal etwas über dieses interessante Verfahren zu berichten.

Greylisting basiert auf dem Umstand, dass ein seriöser Mailserver eine Zustellung etliche Male wiederholt, bevor er die Zulieferung als gescheitert ansieht.

Spammer hingegen, die ja sonst für ihre Penetranz bekannt sind, geben häufig schon nach dem ersten gescheiterten Versuch auf, denn nach den bekannten Regeln für den Erfolg von Spam ist es letztlich nur entscheidend, möglichst viele E-Mail in kurzer Zeit zu verteilen, die Behandlung von Zustellfehlern hingegen hält nur auf.

Deshalb wird manche Spam-Software ein erneutes Zustellen vermutlich gar nicht unterstützen. Besonders, wenn von gehackten Rechnern Mails verteilt werden, wird sich das nicht lohnen, denn oft genug ist der Angreifer schon rausgeflogen, bevor er einen weiteren Versuch starten könnte.

Verwendete Software

Der GUUG-Mailserver läuft unter Debian sarge, als SMTP-Server wird der exim4 verwendet, zu dem der greylisd mittels aptitude installiert greylisd hinzugefügt wurde.

Dieses Paket liefert direkt ein Kommando greylisd-setup-exim4 mit, das die Konfigurationsdatei des exim entsprechend modifiziert. Ob man nun einem solchen Tool vertrauen kann und will oder nicht, sei dahin gestellt – jedenfalls deutet seine Existenz an, dass es sich hier um eine erprobte Kombination handelt.

Wer statt exim4 und greylisd die Kombination aus postfix und policyd verwenden möchte, finden Hinweise dazu im nächsten Beitrag.

Implementation

Beim Aufbau einer Verbindung sammelt der SMTP-Server die IP-Adresse des Clients, den Absender (MAIL FROM) und den Empfänger (RCPT TO) - alle diese Daten liegen vor, bevor mit dem eigentlichen Datentransfer (DATA) begonnen wird.

Dieses Tupel wird an den Greylisting-Daemon geschickt, der zwei Listen mit solchen Tupeln, die Greylist und die Whitelist, führt.

Ist das Tupel weder in der greylist noch in der Whitelist, fügt der Greylisting-Daemon es in seine Greylist ein und gibt dem SMTP-Server zu verstehen, dass er dem Client antworten soll, dass die Mail momentan nicht angenommen werden kann (SMTP-Antwort 451).

Ist das Tupel in der Greylist, so wird es in die whitelist verschoben und der Greylisting-Daemon teilt dem SMTP-Server mit, dass die Mail akzeptiert werden kann. Der SMTP-Server erhält die gleiche Antwort, wenn das Tupel bereits in der Whitelist war.

Damit ist erreicht, dass eine Mail nur angenommen wird, wenn nach der erste Zustellversuch wiederholt wird, was nach dem eingangs gesagten Spammer sehr häufig eben nicht machen.

Damit ein Spammer nun die Zustellung nicht einfach unmittelbar ein zweites Mal erfolgreich probieren kann, kommt eine positive Antwort von Greylisting-Daemon nur, wenn zwischen den beiden ersten Versuchen eine gewisse Zeit (beim greylisd retryMin genannt), typischerweise im Bereich einiger Minuten, liegt.

Wird ein Tupel innerhalb einer Zeitspanne (retryMax) von größenordnungsmäßig einem Tag kein

zweites Mal „angefragt“, so verschwindet es aus der Greylist.

Ein Tupel wandert also genau dann von der Greylist in die Whitelist, wenn es zwischen retryMin und retryMax ein zweites Mal an den Greylisting-Daemon geschickt wird.

Damit die Whitelist nun nicht über alle Grenzen wächst, gibt es auch hier eine Zeitspanne (expire), nach der ein Eintrag aus der Whitelist wieder verschwindet - diese sollte bei etlichen Wochen liegen.

Darüber hinaus kann man auch manuelle Whitelists konfigurieren, also IP-Adressen angeben, von denen Mails grundsätzlich angenommen werden. Das ist sinnvoll, denn ist eine Spam-Mail erstmal von einem „seriösen“ Mail-Server angenommen worden, helfen folgende Greylisting ohnehin nicht mehr, sondern verzögern nur noch die weitere Mailzustellung.

Nachteile

Bei einem ersten Mailaustausch tritt eine Verzögerung auf, die mindestens retryMin beträgt, allerdings auch deutlich darüber liegen kann, abhängig davon, wann der Client den erneuten Zustellversuch startet. Auch können sich diese Verzögerungen addieren, wenn die Mail weitergeleitet wird und alle beteiligten Stationen Greylisting verwenden.

Während dieses Phänomen eher lästig ist (es sei denn, man wartet dringend auf ein Dokument...), kann das Überholen von Mails - das zwar theoretisch ohnehin möglich ist, in der Praxis aber selten auftritt - schon eher Verwirrung auslösen. Das Szenario ist dabei folgendes: A schickt B von einer Adresse erstmals eine Mail, die

erstmal im Greylisting hängen bleibt. 5 Minuten später fällt A auf, dass er die erste Mail korrigieren muss, und schickt eine zweite hinterher. Da die Parameter übereinstimmen, geht diese sofort durch das Greylisting und erzeugt beim Empfänger Verwunderung, da er die erste Nachricht noch gar nicht erhalten hat. Wenn man dann Mails nur in Kopie erhält, wie das als Leser einer Mailing-Liste schonmal der Fall ist, und ein Thema heiß diskutiert wird, kann der Überblick schonmal komplett verloren gehen.

Der Teufel im Detail

Die reale Welt ist wie so oft mit kleinen Tücken versehen: Große Provider haben manchmal ganze Pools von Maschinen, die für einen dahinterliegenden Datenserver den Versand von Mails übernehmen. Dadurch kann es passieren, dass versucht wird, eine Mail hintereinander von verschiedenen Adressen in den Verkehr zu bringen. Deshalb empfiehlt es sich, den Vergleich nicht auf einzelne IP-Adressen, sondern auf Netzwerke, z.B. auf /24, zu beschränken - exim4 und greylistd unterstützen das direkt.

Was bringt's?

Nach fast 64 Tagen Betrieb sind in der Whitelist des GUUG-Servers fast 20.000 Tupel eingetragen, über die fast 30.000 Mails erfolgreich zugestellt worden sind. In der gleichen Zeit sind aber 240.000 Tupel aus der Greylist rausgeflogen, weil kein zweiter Verbindungsaufbau innerhalb der Frist erfolgt ist. Also sind 8/9 aller E-Mails gar nicht angenommen worden. Das entlastet natürlich ganz erheblich alle nachgeschalteten Antispam-Maßnahmen, auf die man letztlich auch weiterhin nicht verzichten kann, und die zum Beispiel im Falle von Bayes-Filtern erheblich Rechenzeit in Anspruch nehmen.

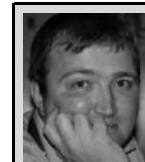
Dabei bleibt natürlich die Frage unbeantwortet bleibt, wieviele false-positives unter den 240.000 Mail waren. Da der greylistd keinen Mechanismus hat, um die aus der greylist verworfenen Einträge zu speichern, ist hier auch ab Werk keine Analyse-Möglichkeit gegeben, selbst wenn man das versuchen wollte.

Nach denen, die gar keinen Spam-Filter haben, sind jene besonders dankbar für die Einführung des greylistings auf dem GUUG-Server, die ihrerseits greylisting als primäre Abwehrmaßnahme nutzen.

Zu Ohren gekommen ist mir auch, dass raffinierte Installationen das greylisting als eine Art Vorwarnstufe für blacklisting verwenden, denn wenn ein Sturm offensichtlicher Spams von einer Adresse losbricht, gewinnt man die beim greylistd mit retryMin konfigurierte Zeit (die man in diesem Fall dann nicht so klein wählen sollte), um den Sender auf eine blacklist zu setzen. Der greylistd in der verwendeten Version unterstützt allerdings keine blacklists.

Und weiter?

Es bleibt natürlich die Befürchtung, dass die Spammer nicht untätig bleiben, wenn sich greylisting immer weiter ausbreitet, und ihrer Software doch die erneuten Zustellversuche beibringen. Das wird man abwarten müssen, und zumindest zur Zeit wirkt das ganze sehr gut.



Martin Schulte (martin.schulte@guug.de) ist seit 2001 Vorsitzender der GUUG und kümmert sich seit 2003 als Geschäftsführer der vereinseigenen OpenServices GmbH hauptberuflich und die GUUG-Veranstaltungen.

Greylisting mit postfix 2.1 und policyd 1.77

Jan Schmidt

Nachdem im vorausgegangen Beitrag auf Funktionsweise von greylisting und Bezugnahme von exim4/greylistd eingegangen worden ist, zeigt der folgende Beitrag, wie sich mit postfix und policyd Greylisting nutzen lässt.

Voraussetzungen

Folgende (Programm-)Pakete sind notwendig, ich gehe im Folgenden von einer Debian 3.1 stable Installation aus:

- postfix >= 2.1
- MySQL 3.x/4.x
- policyd >= 1.77 (aktuell: 1.80); zu beziehen über <http://policyd.sf.net>

Einrichtung

1. policyd herunterladen und mit dem Dreisatz „./configure; make; checkinstall“ installieren. „CheckInstall“ sorgt dafür, dass am Ende die Dateien nicht wild im Dateisystem herumliegen sondern erzeugt ein temporäres ‘make install’, aus dem dann (je nach Distribution) automatisch ein .deb oder .rpm-Paket erzeugt und installiert

wird.

Unter Debian lässt sich CheckInstall leicht mit „apt-get install checkinstall“ installieren.

2. MySQL-Datenbank für policyd erzeugen; ein Dump-File wird mit policyd mitgeliefert.
3. User „postfix“ in mysql.users anlegen (MIT Passwort, aber ohne jegliche Rechte wie INSERT, SELECT usw.)

4. Datenbankberechtigung in mysql.db für den User postfix an DB policyd vergeben:

```
GRANT ALL ON policyd.* \
TO postfix@127.0.0.1 \
IDENTIFIED by 'p0stf1x';
```

5. Standard-Whitelist installieren:

```
mysql policyd < \
docs/WHITELIST.sql -p
```

6. policyd.conf nach eigenen Wünschen anpassen (wichtig natürlich User/Passwort für die Kommunikation mit der MySQL-DB richtig zu setzen); anschließend die Datei am besten mit „chmod 700“ gegen unbefugte Einblicke sichern

7. cleanup-Skript in der crontab verankern (läuft jede Nacht über die Tabellen und löscht veraltete Einträge):

```
0 * * * * /usr/local/\
policyd/cleanup -c\
/usr/local/policyd/\
policyd.conf
```

8. policyd starten:

```
policyd -c policyd.conf\
>/dev/null 2>&1 &
```

9. postfix anpassen:

```
main.cf:
smtpd_recipient_\
restrictions=.\
check_policy_service\
inet:127.0.0.1:10031
```

10. „postfix reload“ ausführen

WICHTIG für alle User, die sich Mails von diversen Freemail-Accounts weiterleiten lassen: die mailout-IP-Adressen bzw. die logischen mailout-Hostnamen UNBEDINGT in die Whitelist mit aufnehmen! Andernfalls

kann es passieren, dass allein aufgrund der Masse an Mails die IP-Adresse des Freemailers auf die Blacklist gesetzt wird (und man sich wundert, dass gar keine Mail mehr ankommt ;-)

Was bietet policyd?

Was bietet policyd im Vergleich zu anderen greylisting-agents (wie z.B. postgrey)?

- Whitelisting (ist bei Mails die von bestimmten Providern wie AOL versendet werden auch ein must-have, da hier Mails z.T. NICHT nochmals verschickt werden)
 - whitelist sender
 - whitelist DNS
 - auto-whitelisting von Netzwerken (nach „x“ erfolgreich zugestellten Triplets wird das Netzwerksegment ge-whitelisted)
 - auto-whitelist expiry (normalerweise wird ein Netzwerksegment nach 28 Tagen aus der auto-whitelist-DB ausgetragen)
- blacklisting
 - blacklisting (de-)aktivieren
 - blacklist DNSNAME
 - blacklist sender
 - permanentes/temporäres Blacklisting von Hosts oder ganzen Netzwerken
 - auto-blacklisting (wenn z.B. mehr als „n“ nicht-erfolgreiche Triplets in den letzten 24h zugestellt wurden)
 - auto-blacklist expiry (normalerweise 7 Tage)
- blacklist HELO (hosts welche versuchen sich als der Empfängerhost auszugeben)

- HELO-check
- spam-trapping: sobald eine Mail an eine „verbrannte“ EMail-Adresse verschickt wird, wird der sendende Host automatisch in die Blacklist eingetragen
- greylisting ;-)
- greylisting training-mode
- greylisting TRIPLET-TIME (in der Praxis haben sich Werte von 3-5 Minuten bewährt)
- greylisting opt-in/opt-in-out-all
- Löschen von „autorisierten“ Triplet-Einträgen nach „n“-Tagen
- Löschen von nicht-autorisierten Triplet-Einträgen nach „n“-Tagen
- sender throttling
 - sender quota rejection: Begrenzung der Gesamtanzahl von Mails von System „XYZ“
 - sender size rejection: bietet postfix allerdings auch (standardmässig 10MB“)
 - sender message limit: Begrenzung der Gesamtanzahl von einem Sender/Tag
- recipient throttling



Jan Schmidt (info@solarisguru.de) ist UNIX Senior Support Analyst bei einem großen Automobilkonzern mit Konzernzentrale in Stuttgart.