Email Virus- und Spamfilter

Sebastian Marius Kirsch
skirsch@moebius.inka.de

Spam, lovely spam. . .

- Statistik am MPI ueber die letzten 2.75 Monate
- Von 178000 Emails wurden 37000 als Spam gewertet
- etwa 20% aller Emails sind Spam
- überschlagsmässig 30% aller eingehenden Emails sind Spam

Möglichkeiten

- ignorieren
- externen Filter benutzen (http://spamcop.net/, GMX, etc.)
- RBL benutzen (http://mail-abuse.org/rbl/, rechtliche Schwierigkeiten)
- lokalen Filter benutzen (SpamAssassin etc.)

SpamAssassin, http://spamassassin.taint.org/

- in Perl geschrieben
- entweder als standalone-Skript oder als Client/Server-Kombination einsetzbar.
- über CPAN installierbar
- 99.94% Trefferquote
- Bewertungen mit realen Samples über Algorithmus festgelegt.

Installation

- `perl -MCPAN -e 'install Mail::SpamAssassin'`
- `apt-get install spamassassin`
- RPMs und `.tgz` von http://spamassassin.taint.org/downloads.html

Einbindung

- in exim als `transport_filter` im geeigneten Transport:

```
local_delivery:
  transport_filter=/usr/local/bin/spamassassin -P
[...]
mail_server:
  driver = smtp
  transport_filter = /usr/local/bin/spamassassin -F0 -P
[...]
```

- über procmail:

  ```
  :0fw
  | /usr/local/bin/spamassassin -P
  ```

  als erste Zeile von `~/.procmailrc`

- Andere Möglichkeiten: http://spamassassin.org/sitewide.html

Auswirkungen

- Zusätzliche Header:

```
Subject: *****SPAM***** UNIQUE INVESTING OPPORTUNITY IN...
X-Spam-Flag: YES
X-Spam-Status: Yes, hits=25.0 required=5.0 tests=SUBJ_H...
X-Spam-Level: *************************
X-Spam-Checker-Version: SpamAssassin 2.20 (devel $Id: S...
X-Spam-Prev-Content-Type: text/html;
        charset="iso-8859-1"
X-Spam-Prev-Content-Transfer-Encoding: quoted-printable
```

- Abschnitt am Anfang des Body

```
SPAM: -------------------- Start SpamAssassin results ----------------
SPAM: This mail is probably spam.  The original message has been alter
SPAM: so you can recognise or block similar unwanted mail in future.
SPAM: See http://spamassassin.org/tag/ for more details.
SPAM:
SPAM: Content analysis details:   (25 hits, 5 required)
SPAM: Hit! (2.7 points)  Subject contains lots of white space
[...]
SPAM: Hit! (-1.0 points) BODY: Gives information about an opportunity
SPAM: -------------------- End of SpamAssassin results ----------------
```

Konfiguration

- `/usr/share/spamassassin` – mitgelieferte Konfiguration, nicht ändern.
- `/etc/mail/spamassassin/local.cf` – lokale Konfiguration
- `~/.spamassassin/user_prefs` – Konfiguration pro Benutzer.

Variablen

- `rewrite_subject` – subject tag hinzufügen
- `report_header` – Report im Header statt am Anfang der Email
- `subject_tag` – `*****SPAM*****` oder `[SPAM]` oder `SPAM!!!!` oder...
- `use_terse_report` – kürzeren Report schreiben (z. B. weil im Header)

- `defang_mime` – Content-Type von Spam auf text/plain setzen (um HTML-Mails zu entschärfen)
- `skip_rbl_checks` – RBL nicht prüfen
- `required_hits` – Wert, ab dem Mail als Spam gewertet wird (default: 8, MPI: 8)
- `score DEAR_SOMEBODY 0.0` – Scores für einzelne Tests anpassen
- `backlist_{from,to} <Adresse>`, `whitelist_{from,to}` – glob patterns, mit Wildcard `*`.

`spamd/spamc`

- `spamd` läuft als Server, ähnliche Optionen wie `spamassassin`
- `spamc` kontaktiert Server auf localhost oder irgendwo im Netzwerk
- `spamc` kann wie `spamassassin -P` eingesetzt werden
- geringerer Overhead, da Perl-Interpreter und Module nur einmal geladen werden
- `spamd` stürzt gerne ab, über inittab oder daemontools starten.

Virenscanner

- Viren in Emails stoppen, bevor sie an die User ausgeliefert werden
- hemmt Verbreitung von Würmern
- Mailserver unter Unix ist fuer Viren/Wuermer schwer anzugreifen (Zielgruppe Windows)
- rechtliche Fragen?
- MPI: ca. zehn Viren am Tag.

amavis

- http://www.amavis.org/
- in Perl geschrieben (erkennen wir ein Muster?)
- benutzt externen Scanner (z.B. NAI VirusScan, H+BEDV Antivir, Sophos, etc.)
- benutzt div. Entpacker und Perl-Module zum Auspacken der Nachrichten.

- Plattformunabhängigkeit stark abhängig von Hilfsprogrammen (Linux ja, Solaris auch, Rest?)

- Implementierung als Daemon (amavisd) existiert, aber noch nicht getestet

Installation

- nur Source-`.tgz`s, keine Pakete verfügbar
- viele Hilfsprogramme müssen mühselig zusammengesucht werden
- Einbindung in sendmail/exim etc. im README beschrieben

Exim

- `transport` section

```
amavis:
  driver = pipe
  command = "/usr/local/sbin/amavis \
    -f <${sender_address}> -d ${pipe_addresses}"
[...]
```

- `directors` section

```
amavis_director:
```

```
  condition = "${if ! eq {$received_protocol}{scanned-ok} {1}{0}}"
  driver = smartuser
  transport = amavis
```

- `router` section

```
amavis_router_incoming:
  condition = "${if ! eq {$received_protocol}{scanned-ok} {1}{0}}"
  driver = domainlist
  route_list = "*"
  transport = amavis_incoming
```

Verhalten

- ankommende Email wird in einem Verzeichnis ausgepackt und gescannt
- kein Virus gefunden: Email wird ausgeliefert
- Virus gefunden: Email wird als Datei in `/var/virusmails` gespeichert, Warnung verschickt ‘to whom it may concern’.
- Default: Warnung wird verschickt an Absender und Admin.

Verhalten am MPI

- für Emails aus dem Institut: Warnung wird an Absender und Admin verschickt.
- für Emails von ausserhalb: Warnung wird an Empfänger verschickt.
- realisiert über zwei unterschiedlich konfigurierte amavis-‘Binaries’, die von exim abhängig vom Absender-Host aufgerufen werden

Probleme

- hohe Systemlast
- dadurch Anfälligkeit für DoS
- Unverständnis der User (‘Ich will aber meine emails!’ ‘Ich benutz aber kein Windows! Mir können Viren doch sowieso nichts anhaben!’)