Apache SpamAssassin

„Mehr als statische Filter“

2. Mailserver-Konferenz, Mai 2005

Malte Stretz <http://msquadrat.de>
Apache SpamAssassin Project Management Committee

http://msquadrat.de/pub/talks/mailserver-konferenz-2005/
http://www.heinlein-support.de/web/konferenzen/mailserver/

Warum „Apache“?

- Lizenz: Ursprünglich PAL/GPL Duallizenz
  - PAL: zu artistisch
  - GPL: zu generell
  - ASL1: gute Grundlage, aber noch Defizite
  - ASL2: perfekt :) naja, fast...
- Infrastruktur: SourceForge 2004 überlastet
  - CVS häufig down
  - Mailinglisten fraßen Mail
  - soll sich inzwischen gebessert haben
- Rechts“sicherheit“: Klagen als mögliche Waffe der Spammer gegen uns
  - Die ASF hat die Rechtsanwälte (und Gelder)
  - Daher: Alle Releases als „Apache SpamAssassin“

Hilfe!

- SpamAssassin ist Open Source
- Das Projekt lebt von den Beiträgen seiner Nutzer
- Alles hilft:
  - Codebeiträge (Bugfixes, neue Features)
  - Beiträge zu DNSDBs (mehr dazu später)
  - Dokumentation (im Wiki fast trivial)

Das Projekt

- Mehr zum Thema ist zu finden unter:

1. SpamAssassin Website
   http://spamassassin.apache.org/
2. SpamAssassin Wiki
   http://wiki.apache.org/spamassassin/FrontPage
3. SpamAssassin FAQ
   http://wiki.apache.org/spamassassin/FrequentlyAskedQuestions
4. SpamAssassin, die Geschichte
   http://wiki.apache.org/spamassassin/SpamAssassinHistory
5. Apache Software License 2.0
   http://www.apache.org/licenses/LICENSE-2.0.html
6. ASL-GPL-Kompatibilität
   http://www.apache.org/licenses/GPL-compatibility.html
7. SpamAssassin Bugzilla
   http://bugzilla.spamassassin.org/
8. SpamAssassin Subversion Repository
   http://svn.apache.org/repos/asf/spamassassin/
9. Beim Projekt mithelfen
   http://wiki.apache.org/spamassassin/WeLoveVolunteers
10. Was ist neu in 3.0?
    http://wiki.apache.org/spamassassin/UpgradeTo300
    http://spamassassin.apache.org/full/3.0.x/dist/UPGRADE
    http://www.onlamp.com/pub/a/onlamp/2004/09/09/spamassassin.html

Mehrbenutzerbetrieb mit Datenbank

- Backends zur Benutzerkonfiguration sind (relativ) austauschbar
- Zur Zeit möglich: Textdateien, SQL (DBI), LDAP
- Vorteile von Datenbanken:
  - (Meistens) Performanter als Dateisystemzugriffe
  - Lastverteilung durch Trennung von Scanning- und Storage-Servern möglich
  - Lastverteilung durch mehrere Scanning-Server möglich
  - Virtuelle Benutzer ohne Home-Verzeichnis möglich
  - Webinterfaces zur Benutzerkonfiguration sind einfacher zu implementieren (bzw. bereits verfügbar)

Mehrbenutzerbetrieb mit Datenbank

- Einfaches Tabellenlayout:

```
mysql> describe userpref;
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| username   | varchar(100) |      | MUL |         |                |
| preference | varchar(30)  |      |     |         |                |
| value      | varchar(100) |      |     |         |                |
| prefid     | int(11)      |      | PRI | NULL    | auto_increment |
+------------+-------------+------+-----+---------+----------------+
4 rows in set (0.02 sec)
mysql> select * from userpref;
+----------------------+-----------------+---------------------+--------+
| username             | preference      | value               | prefid |
+----------------------+-----------------+---------------------+--------+
| foo@example.com      | required_hits   | 5                   |      1 |
| foo@example.com      | x-spam-days     |                     |      2 |
| foo@example.com      | use_terse_report | 0                   |      3 |
| foo@example.com      | ok_languages    | all                 |      7 |
| foo@example.com      | use_dcc         | 1                   |      5 |
| foo@example.com      | use_razor2      | 1                   |      6 |
| foo@example.com      | whitelist_from  | *@example.com       |      8 |
+----------------------+-----------------+---------------------+--------+
7 rows in set (0.00 sec)
```

Mehrbenutzerbetrieb mit Datenbank

- Drei Zeilen in `local.cf`:

```
1 user_scores_dsn          DBI:mysql:spamassassin:localhost
2 user_scores_sql_username spamassassin
3 user_scores_sql_password YourL33tP4ssw0rd
```

  - MySQL-DB „spamassassin“ auf localhost
  - DSN-String unterschiedlich je nach Backend
  - Auf Dateizugriffrechte achten! Evtl. Passwörter in eine extra Datei `db.cf` auslagern

- Und ein paar Optionen beim spamd Aufruf:

```
1 spamd --daemonize --sql-config --setuid-with-sql --nouser-config ...
```

  - `--sql-config` aktiviert den Datenbankzugriff
  - `--setuid-with-sql` wird benötigt, falls `--helper-home-dir` zum Einsatz kommt
  - `--nouser-config` verhindert (nur) den Zugriff auf Konfigurationsdateien in `$HOME/.spamassassin`

Mehrbenutzerbetrieb mit Datenbank

- Mehr zum Thema ist zu finden unter:

1. Einführung SQL im Wiki
   http://wiki.apache.org/spamassassin/UsingSQL
2. Die SQL README in der Distribution
   http://spamassassin.apache.org/full/3.0.x/dist/sql/README
   http://svn.apache.org/repos/asf/spamassassin/branches/3.0/sql/README
3. SQL CREATE TABLE Skripte in der Distribution
   http://spamassassin.apache.org/full/3.0.x/dist/sql/userpref_*.sql
   http://svn.apache.org/repos/asf/spamassassin/trunk/sql/userpref_*.sql
4. Die LDAP README in der Distribution
   http://spamassassin.apache.org/full/3.0.x/dist/ldap/README
   http://svn.apache.org/repos/asf/spamassassin/branches/3.0/ldap/README
5. Links zu verschiedenen Vorträgen über SpamAssassin, u.A. Michael Parker's "Storing SpamAssassin User Data in SQL Databases" (ApacheCon 2004)
   http://wiki.apache.org/spamassassin/PresentationsAndPapers
6. Übersicht über Konfigurationsoptionen
   http://spamassassin.apache.org/full/3.0.x/dist/doc/Mail_SpamAssassin_Conf.html
7. Liste bereits existierender Web-Oberflächen für SpamAssassin
   http://wiki.apache.org/spamassassin/WebUserInterfaces

Lernende Filter: Bayes

- Bayesische Filter sind inzwischen weit verbreitet
- Das Prinzip ist relativ einfach:
  - Der Spamfilter wird trainiert (surprise)
  - Dazu füttert der Benutzer eine Reihe Mails an den Filter und gibt an „dies ist Spam“ oder „dies ist Ham“
  - Der Filter unterteilt die Mail in einzelne „Token“
  - Dies können z.B. Wörter, URLs, Header, etc. sein
  - Zu jedem Token merkt sich der Filter die Häufigkeit, mit der es insgesamt in der jeweiligen Klasse auftaucht
  - Anhand dieser Häufigkeiten kann er dann in Zukunft abschätzen, ob eine Mail Spam oder Ham sein könnte

Lernende Filter: Bayes

- In Wirklichkeit ist das Ganze um einiges komplizierter
- Die Wirklichkeit besteht aus viel Statistik und verschachtelten Formeln mit griechischen Buchstaben und Bruchstrichen, gewürzt mit Datenbanktabellen
- Alle diese Formeln habe ich bis heute auch nicht verstanden
- Die Realität sorgt aber unter Anderem dafür, dass z.B. das beliebte „Bayes-Poisening“ nicht so funktioniert wie gedacht

Lernende Filter: AWL/History

- AWL steht für Autowhitelist
  - Autograylist wäre passender, da es sich sowohl um eine White- als auch eine Blacklist handelt
- Die AWL basiert auf Tripeln, bestehend aus Empfänger, Absender und der IP-Adresse des Absender-Relays
- Der Filter versucht nun, die Punktzahl so zu beeinflussen, dass sie sich über längere Zeit gesehen dem zum jeweiligen Tripel gehörenden Durchschnitt annähert
- Die AWL ist recht effektiv, hat jedoch ein paar Probleme; wird daher in Version 3.2 vom History-Plugin abgelöst

Lernende Filter mit Datenbank

- Auch Bayes und die AWL lassen sich Datenbankgestützt implementieren
- Dazu notwendige Einstellungen:

```
1 bayes_store_module Mail::SpamAssassin::BayesStore::SQL
2 bayes_sql_dsn      DBI:mysql:spamassassin:localhost
3 bayes_sql_username spamassassin
4 bayes_sql_password YourL33tPassw0rd
5
6 auto_whitelist_factory Mail::SpamAssassin::SQLBasedAddrList
7 user_awl_dsn           DBI:mysql:spamassassin:localhost
8 user_awl_sql_username  spamassassin
9 user_awl_sql_password  YourL33tPassw0rd
```

  - DSN und Zugangsdaten müssen wiederholt angegeben werden
  - In Version 3.2 wird hoffentlich eine globale Einstellung verfügbar sein :)

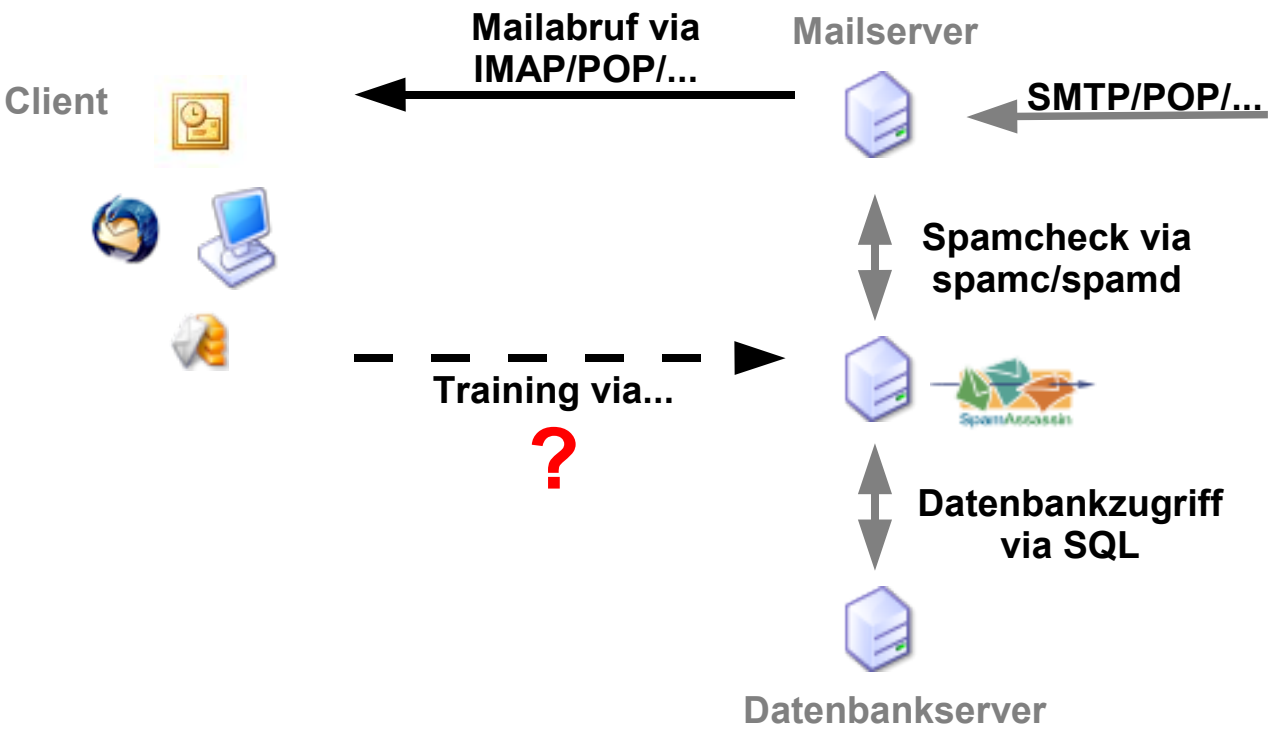
Lernende Filter mit Datenbank

- Die Tabellen sind hier schon umfangreicher:

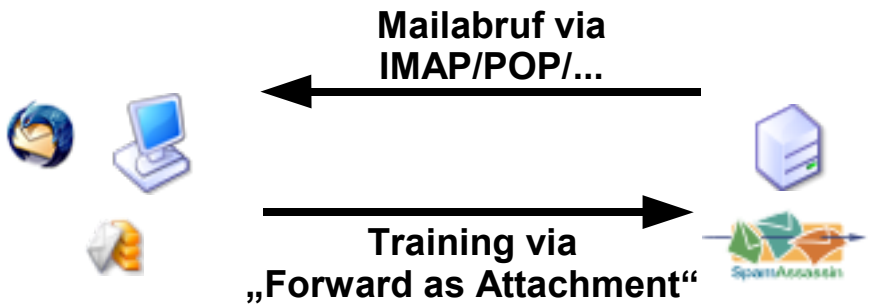
```
mysql> use spamassassin
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-------------------------+
| Tables_in_spamassassin |
+-------------------------+
| awl                     |
| bayes_expire            |
| bayes_global_vars       |
| bayes_seen              |
| bayes_token             |
| bayes_vars              |
| userpref                |
+-------------------------+
7 rows in set (0.00 sec)
```

Serverseitige Filter trainieren

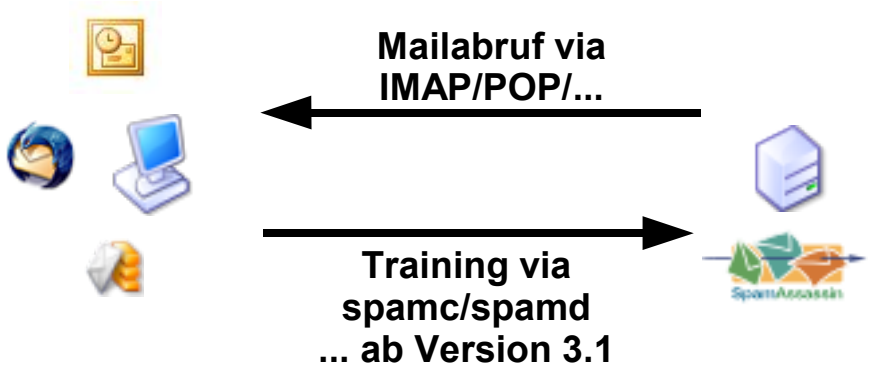


Serverseitige Filter Trainieren (Forward)



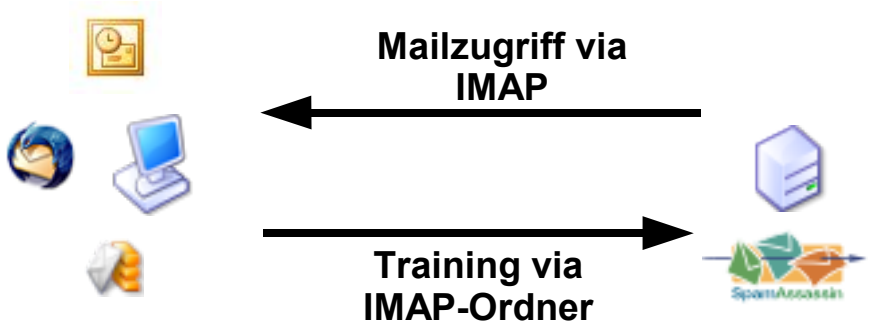
- Alle Header-Informationen müssen erhalten bleiben
  - Weiterleitung muss als Anhang erfolgen
- Nur von wenigen MUAs unterstützt
  - ... zumindest nicht von Outlook
- Sehr Fehleranfällig
  - An die falsche Adresse weitergeleitet? Oops...
- Nicht praktikabel

Serverseitige Filter Trainieren (spamd)



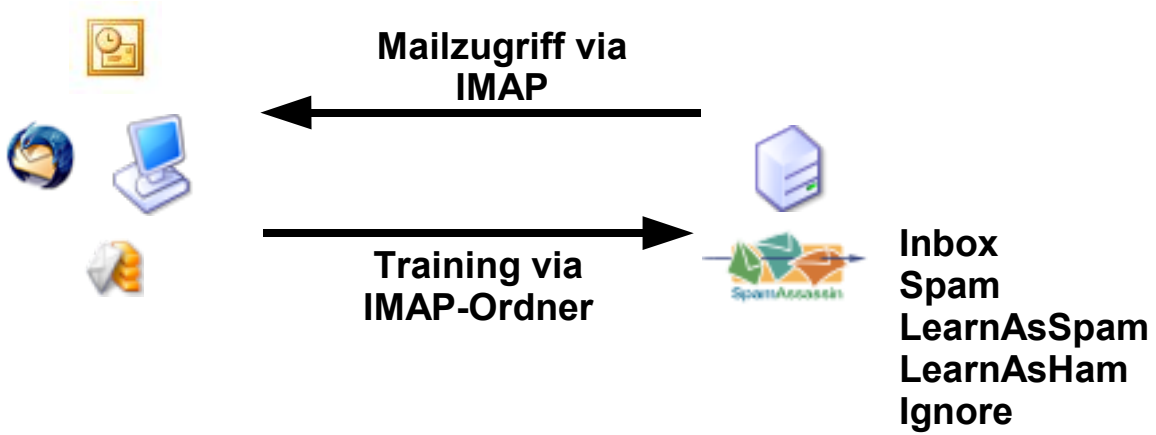
- War Bug 1201, seit November 2002 im Bugzilla
  - Ab Version 3.1 (endlich) implementiert
- Flexibelste Lösung
- Plugins für MUAs möglich
- Zwei Bier für Michael Parker und Nico Prenzel :)

Serverseitige Filter Trainieren (IMAP)



- Die eleganteste Alternative zu spamc/spamd
- Maximal vier Ordner werden benötigt:
  - „Spam“: hier landet der erkannte Spam
  - „LearnAsSpam“: Mail als Spam trainieren
  - „LearnAsHam“: Mail als Ham trainieren
  - „Ignore“: Diese Mail ignorieren

Serverseitige Filter Trainieren (IMAP)



- Relativ einfach zu implementieren:
  - Benutzer verschiebt/kopiert die zu lernende Mail in den jeweiligen Ordner
  - Inhalt des Ordners wird automatisch an sa-learn gefüttert
    - Entweder per Cron-Job
    - Oder per Trigger auf den Ordner
  - Gelernte Mail kann schließlich zurück nach Inbox oder Spam verschoben werden

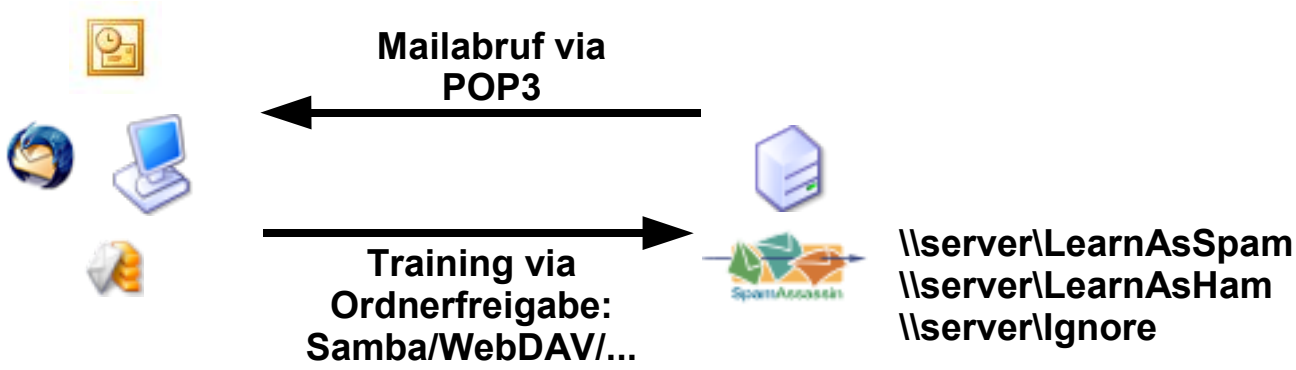
Serverseitige Filter Trainieren (IMAP)

```
#!/bin/bash
# Hacked by mss, 2005-02-01, based on
# <http://www.rootforum.de/faq/index.php?action=artikel&cat=3&id=79&artlang=de>
VBASE=/var/vpopmail/domains
VHOME=/var/vpopmail/domains/$HOST/$EXT

function learn {
  dir="$VBASE/$1/$2/Maildir/.Spam.$4/cur"
  user="$2@$1"
  if [ `find $dir -type f 2>/dev/null | wc -l` != 0 ]; then
    echo -n "$user: $3: "
    nice /usr/bin/sa-learn --username "$user" --$3 "$dir"
  fi
}

for h in $VBASE/*; do
  h=${h###*/}
  for e in $VBASE/$h/*; do
    e=${e###*/}
    if [ -d "$VBASE/$h/$e" ]; then
      learn $h $e forget Ignore
      learn $h $e spam   Undetected
      learn $h $e ham    FalsePositives
    fi
  done
done
```

Serverseitige Filter Trainieren (Freigabe)



- Notlösung, falls kein IMAP verfügbar (z.B. für RoadWarrior)
  - (Web-) Ordner z.B. per VPN verfügbar
  - Benutzer kopiert Mail per Drag&Drop in diesen Ordner
  - Training siehe IMAP
  - Ham kann anschließend wieder per POP3 zugestellt werden

Serverseitige Filter Trainieren

- Mehr zum Thema ist zu finden unter:

1. Bayes FAQ
   http://wiki.apache.org/spamassassin/BayesFaq
2. Einführung in die AWL
   http://wiki.apache.org/spamassassin/AutoWhitelist
3. Trainieren via IMAP
   http://wiki.apache.org/spamassassin/RemoteImapFolder
4. Trainieren via Forward
   http://wiki.apache.org/spamassassin/BayesFeedbackViaForwarding
   http://wiki.apache.org/spamassassin/ProcmailToForwardMail
5. DB_File Oddities
   http://wiki.apache.org/spamassassin/DbDumpAndLoad
6. Zusätzliche Bayes-Informationen mit dem RelayCountry Plugin
   http://wiki.apache.org/spamassassin/RelayCountryPlugin

Netzbasierte Datenbanken

- Eine Reihe netzbasierte Datenbanken werden unterstützt, mit unterschiedlichen Ansätzen
  - Prüfsummen-basiert:
    - Razor
    - Pyzor
    - DCC: Distributed Checksum Clearinghouse
  - Relay-IP basiert (DNSDBs aka RBLs):
    - Spamhaus XBL: Exploits Block List
    - CBL: Composite Blocking List (via XBL)
    - Blitzed OPM: Open Proxy Monitor List (via XBL)
    - NJABL: Not Just Another Bogus List (auch via XBL)
    - DSBL: Distributed Sender Blackhole List
    - AHBL: The Abusive Hosts Blocking List
    - RFC-Ignorant.org
    - SORBS: Spam and Open-Relay Blocking System
    - MAPS: Mail Abuse Prevention Systems
    - SpamCop
    - Habeas
    - Bonded Sender
    - (SPF)

Netzbasierte Datenbanken

- Fortsetzung unterstützte Datenbanken
  - URL-basiert (URIDNSDBs):
    - Überprüfen in Spam beworbene Domains
    - Sehr effektiv
    - SURBL Meta-Liste: Spam URI Realtime Blocklists
      - SURBL SC: SpamCop „Spamvertised Sites“
      - SURBL WS: sa-blacklist Projekt
      - SURBL OB: OutBlaze feed
      - SURBL AB: AbuseButler feed
      - SURBL PH: MailSecurity & MailPolice Phishing feed
      - SURBL JP: jwSpamSpy & Prolocation feed
      - Spamhaus SBL: Spamhaus Block List
- Ein DNS-Cache ist empfehlenswert/gewünscht
  - Die meisten DNSDBs bieten für „große“ Provider zudem rsync oder AFXR/IXFR an

Rückmeldung an Netzdatenbanken

- Auch die Datenbanken benötigen Input
  - Inhaltsbasierte Datenbanken:
    - Für Razor & Pyzor kann der dazugehörige Client die Prüfsummen schicken
    - Jede bei DCC angefragte Prüfsumme wird registriert
    - SURBL SC bekommt die Daten von SpamCop
    - SURBL WS aus dem sa-blacklist Projekt, dort kann per Mail gemeldet werden
    - SURBL ist offen für weitere Listen à la OB und AB
  - Open Relay/Proxy Datenbanken:
    - OPM wird standardmäßig von IRC-Servern gefüttert, akzeptiert aber auch andere Meldungen per E-Mail
    - NJABL hat spezielle DNS-Zonen
  - DNSDBs:
    - SpamCop hat eine SMTP-Schnittstelle
    - Bonded Sender, Habeas, SPF und DomainKeys werden vom Absender individuell „aktiviert“
    - RFC-Ignorant.org hat ein Webformular
    - Die meisten anderen DNSDBs werden von Spamtraps gefüttert, Meldungen evtl. auch per E-Mail oder Webformular

Rückmeldung an Netzdatenbanken

- SpamAssassin kann bereits an einige Datenbanken melden
  - Der Befehl hierzu lautet „spamassassin --report“
    - Die Funktionalität wird in Version 3.2 oder später in ein eigenes Programm „sa-report“ ausgelagert
    - Der alte Weg wird jedoch weiterhin funktionieren
    - Spätestens in Version 3.2 geht dies auch mit spamc/spamd
  - Unterstützte Datenbanken:
    - Razor
    - Pyzor
    - (DCC)
    - SpamCop (und damit SURBL SC)

Rückmeldung an Netzdatenbanken

- Eine Automatisierung wäre möglich
- Alle von den Benutzern als Spam gelernte Mails könnten gemeldet werden
  - Pro:
    - Einfach einzurichten
  - Contra:
    - Wie exakt gehen die Anwender vor?
    - Vielleicht war es ja doch ein legitimer Newsletter, irgendwann mal bestellt
    - Was ist mit versehentlichen Meldungen?
- Mit Vorsicht zu genießen, immer abhängig von der Benutzerschaft

Rückmeldung an Netzdatenbanken

- Das Nonplusultra: Handgeprüfte Meldungen
- Die Datenbanken sind immer nur so gut wie die Leute, die sie warten
  - Pro:
    - Hohe Zuverlässigkeit (hoffentlich)
    - Verbesserte Erkennungsraten für alle Anwender der Software
    - Evtl. werbewirksam (siehe OutBlaze)
  - Contra:
    - Aufwändig
    - Arbeitszeit, die sich nicht (direkt) für die eigene Firma auswirkt, sondern „der Allgemeinheit“ zu Gute kommt

Netzbasierte Datenbanken

- Mehr zum Thema ist zu finden unter:

1. Liste der standardmässig verfügbaren DNSDBs:
   http://svn.apache.org/repos/asf/spamassassin/branches/3.0/rules/20_dnsbl_tests.cf
   http://svn.apache.org/repos/asf/spamassassin/trunk/rules/20_dnsbl_tests.cf
2. Liste der standardmässig verfügbaren URIDNSDBs:
   http://svn.apache.org/repos/asf/spamassassin/branches/3.0/rules/25_uribl.cf
   http://svn.apache.org/repos/asf/spamassassin/trunk/rules/25_uribl.cf
3. SURBL Listen:
   http://www.surbl.org/lists.html

Überlebenstipps

- Play nice, play fair (aka OTFR)
  - RFC 1918: Keine Experimente mit IP-Adressbereichen in lokalen Netzen
  - RFC 2606: Keine Experimente mit Domainnamen
  - RFC 2822: Immer gültige (MIME) Mail verschicken
    - Keine benötigten Header vergessen
    - Keine kaputten Header senden
    - Nicht versuchen, irgendwelche MUAs zu imitieren
  - SPF, MTAMark, DULs: Nur offizielle Mailserver verwenden
  - Kollateralschaden: Keine Bounces generieren
  - Bitte kein Challenge-Response
- Don't spam :)

Ende

Danke für Ihre Aufmerksamkeit

Fragen?