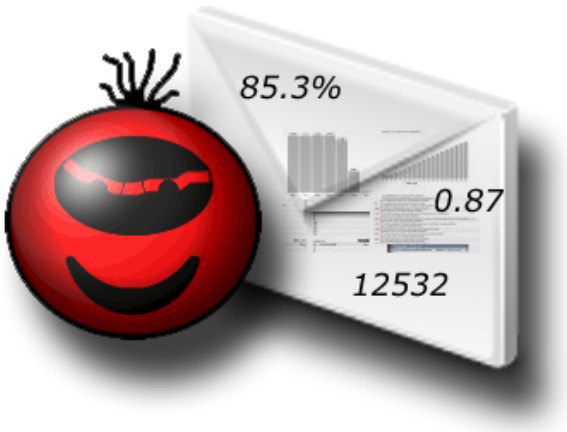
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed
Computing Group

Spamato Statistics

A Statistical Approach towards
Spam Filtering



Christian Wassmer

Diplomarbeit

12. Oktober 2004 – 11. Februar 2005

Betreuender Professor: Prof. Dr. Roger Wattenhofer
Betreuender Assistent: Keno Albrecht

Vorwort

Zusammenfassung

SPAMATO ist ein erweiterbares, kollaboratives Spam-Filter-System der *Distributed Computing Group* der ETH Zürich.

Die vorliegende Diplomarbeit behandelt sowohl die Integration von Erhebungsmechanismen statistischer Daten in das SPAMATO-System als auch deren Auswertung auf einem zentralen Statistik-Server. Statistiken über das Verhalten der SPAMATO-Benutzer oder über die Qualität der SPAMATO-Filter werden dadurch möglich.

Danach wird ein neuartiges Verfahren zur Identifizierung von Spam-Mails vorgestellt und dessen Qualität analysiert. Dieses Verfahren beinhaltet eine statistische Analyse von in Spam-Mails vorkommenden Domains hinsichtlich ihrer Verbreitung im World Wide Web. Die Verbreitung wird mittels Suchabfragen, die diese Domain beinhalten, bei der Suchmaschine *Google* ermittelt. Das Ergebnis der Analyse ist der DOMAINATOR, ein neuer Filter für SPAMATO.

Danksagung

Ich möchte mich bei allen bedanken, die mich während meines Studiums immer unterstützt haben.

Allen voran bei meinen Eltern, die mich immer vorbildlich unterstützt und mir das Studium ermöglicht haben.

Bei Simon Schlachter, der mehr war als nur ein Studienkollege. In all den Semestern konnten wir uns hervorrangend ergänzen, profitierten in idealer Weise voneinander und hatten eine tolle Zeit zusammen an der ETH.

Dank gebührt auch meinem Betreuer Keno Albrecht, der mir in den vier Monaten der Arbeit immer mit Rat und Tat zur Seite stand, und Prof. Roger Wattenhofer, der mir es ermöglicht hat, meine Abschlussarbeit bei seiner Gruppe zu schreiben.

Zu guter Letzt möchte ich mich auch herzlich bei Urs Berger für das Korrekturlesen bedanken.

Inhaltsverzeichnis

Die Zahl ist das Wesen aller Dinge.

Pythagoras (576-496), griech. Philosoph u. Mathematiker)

1 Einführung

1.1 Motivation

Beinahe täglich werden wir mit Statistiken über Wichtiges und weniger Wichtiges konfrontiert. Ein Schweizer verzerrt pro Jahr 11 kg Schokolade, die Krebsgefahr und das Ozonloch nehmen zu, Informatiker leben länger als andere, Landluft ist gesund, Landluft ist ungesund, Eishockeyspieler B schiesst am liebsten Tore gegen rechts fangende Torhüter im ersten Drittel et cetera ...

Nun verfügt die Menschheit, genauer gesagt die Benutzer von SPAMATO, über eine Statistik-Quelle mehr. Nach Abschluss dieser Arbeit existieren auch Statistiken über SPAMATO. Über das Spam-Filter-System der *Distributed Computing Group* der ETH Zürich existierten bis vor Beginn dieser Diplomarbeit keine Statistiken. Dies änderte sich jedoch im Wintersemester 2004/05 als ich mich entschloss, die Abschlussarbeit meines Informatikstudiums auf diesem Gebiet zu schreiben.

Meine persönliche Motivation im Schreiben dieser Arbeit lag darin, dass mich Zahlen schon seit meiner Kindheit interessieren und faszinieren. Seit nun bald zehn Jahren führe ich diverse Statistiken über das Geschehen im Schweizer Eishockey. Es lag nahe, dass ich meine Erfahrungen, die ich in diesem Bereich im Laufe der Zeit gewonnen habe, in meiner Diplomarbeit einsetzen würde.

1.2 Übersicht

Nach dieser Einführung wird in Kapitel 2 ein Statistik-Framework für SPAMATO eingeführt. Es erläutert, welche Datenstrukturen benötigt werden, damit statistische Daten von SPAMATO an einen zentralen Statistik-Server geschickt werden können.

Kapitel 3 behandelt ein neuartiges Verfahren, um Spam-Mails zu identifizieren. Das Resultat einer Analyse von in Spam-Mails vorkommenden Domains war der DOMAINATOR, ein neuer Filter für SPAMATO.

In Kapitel 4 werden verschiedene Statistiken, wie zum Beispiel über das Verhalten der SPAMATO-Benutzer oder über die Qualität der SPAMATO-Filter, vorgestellt.

Schliesslich behandelt Kapitel 5 weiterführende Arbeiten.

In Anhang A befindet sich eine Aufzählung aller in Kapitel 2 definierten Events und in Anhang B die SQL-Tabellendefinitionen der Datenbank des Statistik-Servers.

2 Statistik-Framework

2.1 Einleitung

In diesem Kapitel wird die Problematik dargestellt, mit dem bestehenden SPAMATO-System globale Statistiken über selbiges zu erstellen. Bevor das Statistik-Plugin von SPAMATO vorgestellt wird, wird die dazu notwendige Datenstruktur eingeführt. Das Statistik-Plugin sendet die gesammelten Daten an den Statistik-Server. Dieser wird im darauf folgenden Abschnitt beschrieben.

2.2 Motivation

Über SPAMATO, wie es sich vor Beginn dieser Arbeit präsentierte, konnten nur bedingt Statistiken wie das Benutzerverhalten oder die Filtereffektivität erstellt werden. Einige statistische Erhebungsmechanismen wurden zwar im *URL*-Filter eingebaut, dessen Daten vom *URL-Servers* verarbeitet werden. Jedoch konnten so nur Aussagen über den *URL*-Filter selber gemacht werden, nicht aber über das SPAMATO-System und alle darin enthaltenen Filter. Zudem wurde beim Entwickeln des *URL-Filters* das Augenmerk naturgemäss nicht auf das Sammeln von statistischen Daten gelegt, sondern auf das effiziente und sichere Erkennen von Spam-Mails. Vom anderen Filter - dem *Razor*-Filter - konnten überhaupt keine statistischen Daten erhoben werden. Dieser Filter ist ein kollaboratives Client-Server-System, auf dessen Server die SPAMATO-Entwickler keinen Zugriff haben.

Das Statistik-Framework behebt diese Hindernisse, indem es direkt im SPAMATO-Kern eingreift. Statistische Daten können nun sowohl von allen Filtern als auch vom SPAMATO-System erhoben werden und an einen zentralen Statistik-Server geschickt werden.

2.3 Datenstruktur

Dieser Abschnitt beschreibt die Datenstruktur, die eingeführt wurde, um Statistiken über SPAMATO erheben zu können. Neben der Klasse `StatReport`, die dazu dient, eine komplette SPAMATO-Operation [Bur04] - wie *Check*, *Report* oder *Revoke* - statistisch zu erfassen, dienen die Subklassen von `SpamatoResult` der Speicherung der Ergebnisse der einzelnen Filter. Die Filter dürfen jedoch nicht direkt auf dieser Datenstruktur operieren. Dazu wurde ein Event basiertes System implementiert, das in Abschnitt 2.4.1 beschrieben wird.

2.3.1 StatReport

Diese Klasse repräsentiert sämtliche zu einer SPAMATO-Operation gehörenden Eigenschaften und Ergebnisse.

userName Die vom SPAMATO-Benutzer verwendete Email-Adresse.

results Die Menge der `SpamatoResult`-Objekte. Siehe Abschnitt 2.3.2.

clientId Identifiziert das SPAMATO-Plugin, das vom SPAMATO-Benutzer gebraucht wird. Mögliche Werte sind unter anderem *mozilla*, *outlook* und *proxy*.

mailId Eine Identifzierung für die E-Mail. Aus den Bestandteilen *body*, *subject*, *sentOn* der E-Mail und `userName` berechnet.

firstCheck Falls es sich um einen *Check* handelt, gibt diese Boolsche Variable an, ob es sich um einen *ersten Check* (`true`) oder um einen *Re-Check* (`false`) handelt. Ansonsten besitzt sie den Wert `false`.

minSpamAnswerAvg Bestimmt den minimalen Anteil von Filtern, die eine E-Mail als Spam identifizieren müssen, damit sie von SPAMATO als Spam identifiziert wird.

spam Die Entscheidung, ob die E-Mail Spam ist oder nicht.

mail Die E-Mail, die als Spam gemeldet wurde.[1]

autoReport Bei einem *Report* gibt diese Boolsche Variable an, ob dieser automatisch (`true`) oder vom Benutzer selbst (`false`) erzeugt wurde. Ansonsten besitzt sie den Wert `false`.

notOnSpamCam Falls es sich um einen *Report* handelt, gibt diese Boolsche Variable an, ob die gemeldete E-Mail in der *Spam Cam* [Sch04] erscheinen darf (`true`) oder nicht (`false`). Ansonsten besitzt sie den Wert `false`.

2.3.2 SpamatoResult

Diese Klasse bildet die gemeinsame Superklasse für die Klassen der Ergebnisse der einzelnen SPAMATO-Operationen. Sie beinhaltet allen Subklassen gemeinsame Werte, wie:

success Gibt an, ob der Filter erfolgreich gewesen ist (`SUCCESS`). Falls nicht, wird einer der beiden folgenden Werte angenommen: `FILTER_IS_DISABLED`, falls er deaktiviert ist, oder `ERROR`, falls ein Fehler aufgetreten ist.

errorString Hier kann im Fehlerfall eine Beschreibung des Fehlers angegeben werden.

filterType Die Bezeichnung des Filters, zu dem die `SpamatoResult`-Instanz gehört.

[1] Wird nur bei *Report*s gesetzt

CheckResult

Ein `CheckResult` repräsentiert das Ergebnis einer *Check*-Operation. Neben dem eigentlichen Resultat der Operation (`result`) beinhaltet dieses Objekt eine Menge von Filtereigenschaften sowie die berechneten *Fingerprints*[2] (siehe `FilterProperty` bzw. `SubFilterResult`, Abschnitt 2.3.3).

ActiveResult

Diese Superklasse beinhaltet lediglich die von einem Filter berechneten *Fingerprints*. Die aus einer *Check*-Operation berechneten *Fingerprints* müssen nicht zwingenderweise mit denen einer *Report*- oder *Revoke*-Operation übereinstimmen. Bei einem kollaborativen Filter wie dem *URL*-Filter kann es vorkommen, dass sie dem Server noch nicht als Identifikatoren einer Spam-Mail gemeldet wurden und deshalb dem Client bei einer *Check*-Operation noch keine Identifikatoren mitgeteilt werden können. Die beiden Subklassen `ReportResult` und `RevokeResult` besitzen keine zusätzlichen Felder.

2.3.3 Weitere Klassen

Des Weiteren existieren zwei Klassen für Instanzen von `CheckResult`, sowie eine Klasse für `ActiveResult`, deren Verwendungen nachfolgend erläutert werden.

FilterProperty

Diese Klasse dient der Speicherung von Filtereigenschaften. Die Eigenschaft wird als Werte-Schlüssel-Paar einem `CheckResult` hinzugefügt. Beispiele von Filtereigenschaften beinhalten *Spam Confidence Value* des *Razor*-Filters sowie die *Aggressivness*-Einstellung (siehe Abschnitt 3.5.2) des DOMAINATOR.

SubFilterResult

Erzeugen SPAMATO-Filter eindeutige Identifikatoren (*Fingerprints*) für E-Mails, werden diese als Werte-Schlüssel-Paar in Form eines `SubFilterResult`-Objekts abgespeichert. Ein Filter kann mehrere *Fingerprints* für ein und dieselbe E-Mail bestimmen. Diese Klasse ist für Instanzen von `CheckResult` und beinhaltet ein Feld zur Speicherung der Angabe, ob der *Fingerprint* ein Spam-Mail identifiziert.

Fingerprint

Diese Klasse wird von der Klasse `ActiveResult` zur Speicherung von *Fingerprints* verwendet. Sie beinhaltet lediglich zwei Felder für das Werte-Schlüssel-Paar eines *Fingerprints*.

[2] Fingerprints werden in kollaborativen Filtern verwendet um E-Mails zu identifizieren.

2.3.4 Übersicht

Abbildung 2.1 zeigt das UML-Diagramm der voran beschriebenen Datenstruktur.

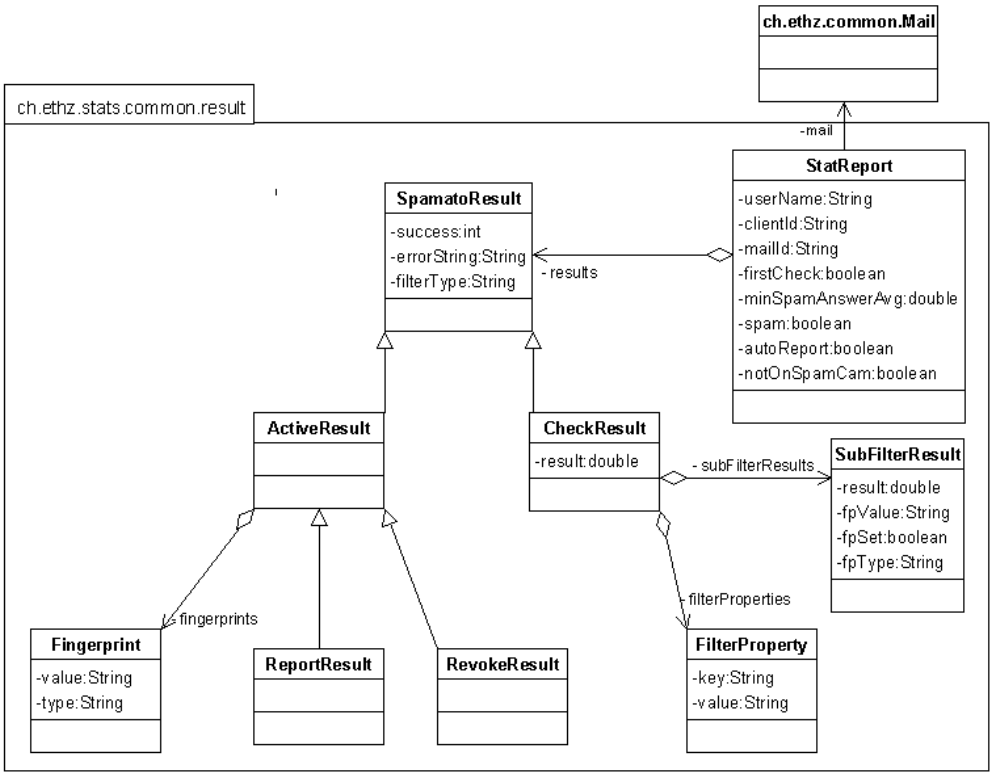


Abbildung 2.1: Datenstruktur zur Erfassung von statistischen Daten über SPAMATO. Die Methoden wurden aus Gründen der Übersichtlichkeit weggelassen.

2.4 Statistik als Spamato-Client-Plugin

Dank Remo Meier [Mei05] ist es möglich, SPAMATO durch zusätzliche Komponenten, wie neue Filter oder eine Komponente zur Erhebung von Statistiken, auf eine einfache Art und Weise zu erweitern. Die Komponenten werden Plugins genannt. Das Statistik-Plugin erhält vom SPAMATO-Kern die statistischen Informationen via Events, die in Abschnitt 2.4.1 beschrieben werden. Abschnitt 2.4.2 behandelt ihre Verarbeitung im Event-Handler.

Neben der Einstellung der Server-Adresse und des Server-Ports kann der Benutzer angeben, ob er Statistiken über seine SPAMATO-Installation erheben lassen will und, falls ja, ob seine als Spam gemeldeten E-Mails nach einer gewissen Verzögerung in der *Spam Cam* [Sch04] erscheinen dürfen.

Eine spezielle Registrierung beim Statistik-Server ist nicht nötig. Der Benutzer wird über seine Email-Adresse identifziert. Es besteht keine Authentifizierung, wie dies zum Beispiel beim *URL*-Filter der Fall ist.

2.4.1 Spamato-Events

Im folgenden werden die verschiedenen Event-Typen und ihre Verwendung vorgestellt. Eine detaillierte Beschreibung mit sämtlichen Argumenten befindet sich in Anhang A.

STAT_PRE_CHECK Wird von SPAMATO vor einer *Check*-Operation erzeugt. Enthält Informationen über den SPAMATO-Benutzer sowie das SPAMATO-Plugin.

STAT_PRE_REPORT Wird von SPAMATO vor einer *Report*-Operation erzeugt. Enthält die gleichen Informationen wie das `STAT_PRE_CHECK`-Event, sowie ob es sich um einen *Auto-Report* handelt.

STAT_PRE_REVOKE Wird von SPAMATO vor einer *Revoke*-Operation erzeugt. Enthält die gleichen Informationen wie das `STAT_PRE_CHECK`-Event.

STAT_CHECK Wird von jedem Filter nach Bestimmung des Ergebnisses einer *Check*-Operation erzeugt. Daneben enthält es Informationen der berechneten *Fingerprint*s sowie der Filtereigenschaften im Falle eines positiven *Checks*.

STAT_REPORT Wird von jedem Filter nach Bestimmung des Ergebnisses einer *Report*-Operation erzeugt.

STAT_REVOKE Wird von jedem Filter nach Bestimmung des Ergebnisses einer *Revoke*-Operation erzeugt.

STAT_POST_CHECK Wird von SPAMATO nach der Bestimmung des globalen Resultats - ob Spam oder nicht - erzeugt.

2.4.2 Event-Handler

Beim Aufstarten von SPAMATO wird das Statistik-Plugin als Event-Handler der im Abschnitt zuvor erwähnten Events registriert. Der Handler erstellt aus den Events den in Abschnitt 2.3.1 beschriebenen `StatReport`.

Da bei einer *Check*-Operation auf die Resultate aller Filter gewartet werden muss, bis eine Entscheidung von SPAMATO getroffen werden kann, kann erst nach dem Erhalt eines `STAT_POST_CHECK`-Events dem Statisitik-Server der `StatReport` geschickt werden. Per Definition muss dabei mindestens ein Filter ein positives Ergebnis[3] geliefert haben, falls es sich um einen *ersten Check* handelt, oder die E-Mail muss als Spam identifiziert worden sein, falls es sich um einen *Re-Check* handelt. Abbildung 2.2 zeigt den Ablauf einer *Check*-Operation und welche Events wann erzeugt werden.

Eine *Report*- bzw. *Revoke*-Operation läuft im Wesentlichen gleich wie eine *Check*-Operation ab. Anstelle des `STAT_PRE_CHECK`-Events wird zu Beginn ein `STAT_PRE_REPORT`- bzw. ein `STAT_PRE_REVOKE`-Event erzeugt und die Filter feuern anstelle des `STAT_CHECK`-Events je ein `STAT_REPORT`- bzw. ein `STAT_REVOKE`-Event. Da eine *Report*- bzw. *Revoke*-Operation asynchron ablaufen kann, entfällt

[3] im Sinne von: "E-Mail ist Spam".

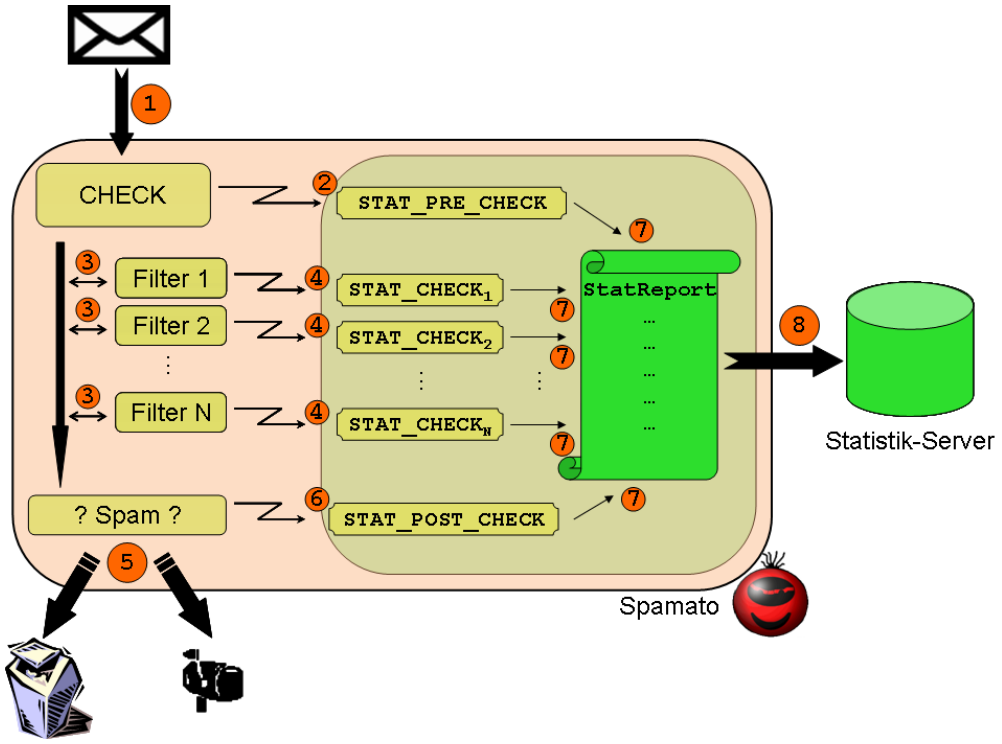


Abbildung 2.2: Ablauf einer *Check*-Operation. Die zu prüfende E-Mail wird von SPAMATO empfangen (1), worauf das `STAT_PRE_CHECK`-Event erzeugt wird (2). Jeder SPAMATO-Filter führt nun seine *Check-Operation* durch (3) und löst dabei je ein `STAT_CHECK`-Event aus (4). Wenn alle Filter ihre Operation beendet haben und SPAMATO die Entscheidung, ob Spam oder nicht, getroffen hat (5), wird das `STAT_POST_CHECK`-Event erzeugt (6). Danach kann das `StatReport`-Objekt aus den Daten der Events gebildet werden (7) und an den Statistik-Server übermittelt werden (8).

das Event am Ende der Operation. Der Event-Handler muss warten, bis alle entsprechenden Events von allen beteiligten Filtern erzeugt worden sind, bevor der `StatReport` verschickt werden kann. Ungeachtet der Vollständigkeit wird er eine bestimmte Zeit nach dem Eintreffen des ersten Events verschickt. Die E-Mail wird nur bei einer *Report*-Operation dem `StatReport` angehängt.

2.5 Statistik-Server

2.5.1 Einführung

Der Statistik-Server empfängt die `StatReport`-Objekte und bearbeitet sie in zwei Schritten.

In einem ersten Schritt wird anhand der `mailID` überprüft, ob in den letzten 60 Sekunden nicht die gleiche SPAMATO-Operation mit derselben E-Mail durchgeführt wurde. Ist dies der Fall, so werden aus dem `StatReport` die allgemeinen Informationen für die entsprechende SPAMATO-Operation gelesen und in eine Datenbank geschrieben. Dabei wird der Benutzername - eine Email-Adresse - mittels des MD5-Hashverfahrens anonymisiert. Falls die gleiche SPAMATO-Operation mit derselben E-Mail durchgeführt wurde, wird der `StatReport` nicht weiter verarbeitet. Dies um zu verhindern, dass versehentlich vom SPAMATO-Benutzer doppelt ausgeführte Operationen nicht mehrfach gespeichert werden und die Statistik verfälschen.

Tabelle 2.1 beschreibt, welche Informationen eines `StatReport` für welche SPAMATO-Operation relevant sind. Die Tabellendefinitionen für die Operationen befinden sich in den Anhängen B.7, B.15 und B.17.

Feld	***Check***	***Report***	***Revoke***
`userName`	ja	ja	ja
`results`	ja	ja	ja
`clientId`	ja	ja	ja
`mailId`	ja	ja	ja
`firstCheck`	ja	nein	nein
`minSpamAnswerAverage`	ja	nein	nein
`spam`	ja	nein	nein
`mail`	nein	ja	nein
`autoReport`	nein	ja	nein
`notOnSpamCam`	nein	ja	nein

Tabelle 2.1: Inhalt eines `StatReport`-Objekts und welche Felder für welche SPAMATO-Operation relevant ist.

Im zweiten Schritt werden die `StatReport`-Objekte Implementierungen des Interfaces `IStatisticHandler` übergeben.

2.5.2 IStatisticHandler

Zur Compile-Time können dem Statistik-Server `IStatisticHandler`-Objekte hinzugefügt werden. Diese Handler verrichten die Hauptarbeit im Server. Sie verarbeiten einen eingehenden `StatReport` auf jeweils ihre eigene Weise, indem sie zum Beispiel die in einer E-Mail enthaltenen Domains analysieren oder die E-Mail als solche abspeichern. Die Handler werden sequentiell abgearbeitet. In Anhang B befinden sich die SQL-Definitionen aller Tabellen des Statistik-Servers.

Der aktuelle Server (Stand: Januar 2005) wurde mit den folgenden vier Handlern kompiliert.

FingerprintLogger

Der `FingerprintLogger` speichert alle *Fingerprints* jedes Filters in die Datenbank. Dazu existieren zwei Tabellen pro SPAMATO-Operation. Für eine *Check*-Operation sind dies `stat_filter_check` (siehe Anhang B.10) und `stat_subfilter_result` (B.19). In der ersteren ist das Ergebnis des Filters gespeichert, während in der letzteren jeder einzelne *Fingerprint*-Wert mit einer eventuell dazugehörenden Spam-Entscheidung[4] gespeichert wird.

Sowohl die Filter- als auch die *Fingerprint*-Typen werden aus Platzgründen durch Integerwerte repräsentiert, die in zwei weiteren Tabellen definiert sind. Erkennt der `FingerprintLogger` einen für ihn neuen und deshalb unbekannten *Fingerprint*-Typen, wird automatisch ein neuer Eintrag für den neuen Typ in der dafür vorgesehenen Tabelle `stat_fingerprint` (B.13) erzeugt. Analog wird für jeden neuen, unbekannten SPAMATO-Filter ein Eintrag in der Tabelle `stat_filter` (B.9) angelegt.

MailLogger

Dieser `IStatisticHandler` speichert die von den SPAMATO-Benutzern gemeldeten E-Mails in der Tabelle `stat_log_mail` (B.14) ab. Aufgrund der `mailID` wird bei einer *Revoke*-Operation die E-Mail wieder gelöscht. Bei einer *Check*-Operation geschieht nichts.

FilterPropertiesLogger

Damit werden die einem `CheckResult` angehängten Filtereigenschaften in der Tabelle `filter_properties` (B.2) gespeichert. Es werden nicht alle Werte, sondern nur diejenigen, die sich gegenüber dem letzten Eintrag des Benutzers verändert haben, gesichert. Die Bestimmung der Filtereigenschaften jeder einzelnen Spam-Erkennung (*positiver Check*) ist somit mit einem geringeren Platzbedarf möglich.

[4] Bei einer *Check*-Operation ist dies der Fall.

GoogleDomainLogger

Der `GoogleDomainLogger` ist für zwei Sachen verantwortlich: Zum einen werden die Domains aus dem Body gemeldeter Spam-Mails extrahiert und gesichert. Zum anderen werden die gesammelten Domains daraufhin untersucht, wie häufig sie in den von Google indizierten Seiten vorkommen. Sprich, es werden verschiedene Suchabfragen mit der Domain im Suchbegriff durchgeführt. Dieser Suche und den daraus gewonnen Erkenntnissen ist Kapitel 3 gewidmet.

3 Domain-Analyse

3.1 Einleitung

In diesem Kapitel geht es um die Analyse von in Spam-Mails vorhandenen Domains. Aus den Erkenntnissen entsteht ein neuartiger Spam-Filter. Dabei werden die Domains nicht wie in anderen Verfahren dazu verwendet, eine Identifizierung (*Fingerprint*) der E-Mail zu erhalten. Sondern es wird versucht, anhand von Eigenschaften einer Domain, wie ihrer Verbreitung im World Wide Web, eine Entscheidung zu treffen, ob die E-Mail, in welcher die Domain enthalten war, Spam ist.

3.2 Hintergrund

Die Domain-Analyse konzentrierte sich darauf herauszufinden, wie sich Spam-Domains von regulären Domains unterscheiden. Eine Spam-Domain ist eine sich in einer Spam-Mail befindenden Domain. Diese Domain stammt meistens von einem Link, der zum vom Spammer beworbenen Produkt führt. Als ein Unterscheidungsmerkmal bot sich die Verbreitung im World Wide Web an. Die Verbreitung wurde mittels der Anzahl Treffer bei Suchabfragen, welche die Domain in irgendeiner Form wiederverwendet, bei der Suchmaschine *Google*[1] bestimmt. Suchabfragen wie zum Beispiel

ethz.ch

oder

site: ethz.ch

wurden betrachtet.

Es stellte sich heraus, dass diese Art von Suchabfragen nicht reichen würde, Spam-Domains zu identifizieren. Einerseits gibt es viele reguläre Domains, unter anderem von Privatpersonen oder wenig bekannten Firmen, die ähnlich niedrige Werte wie Spam-Domains bei der Trefferanzahl aufweisen. Die Folgen wären zu viele *False Positives*. Andererseits gibt es Spam-Domains, die trotz ihrer zweifelhaften Absicht, eine ähnlich hohe Trefferanzahl wie reguläre Domains erzielen. Die Konsequenz wäre, dass etliche Spam-Domains als reguläre Domains identifiziert und sich der Aussortierung unterziehen würden.

[1] `http://www.google.com/`

Abhilfe schafften Suchabfragen, die neben der Domain auch Spam relevante Schlüsselwörter wie *spam* oder *blacklist* enthalten. Daraus ergaben sich Suchabfragen wie

ethz.ch + spam

oder

ethz.ch + blacklist.

Es zeigte sich, dass diese Art der Suchabfragen ein viel höheres Differenzierungspotential aufweisen als solche ohne Zusatzwörter.

3.3 Vorgehen

Um die vorab geschilderten Vermutungen verifizieren zu können, musste eine genügend grosse Menge sowohl von Spam-Domains als auch von regulären Domains bestimmt werden. Mit den gesammelten Domains wurden die in Abschnitt 3.2 erwähnten Suchabfragen durchgeführt und danach mit Hilfe eines selbst geschriebenen Tools (siehe 3.3.3) in verschiedene Kategorien eingeteilt.

3.3.1 Domains sammeln

Um potentielle Spam-Domains zu sammeln, wurden die Domains aus von SPAMATO-Benutzern gemeldeten Spam-Mails extrahiert. Diese Spam-Mails stammen aus dem Zeitraum von Ende November 2004 bis Anfang Januar 2005. Das Sammeln von regulären Domains stellte sich ein wenig schwieriger dar. Das Problem war nicht, dass es zuwenige geben würde, sondern dass die Testmenge auch möglichst wenig bekannte und schwach verbreitete Domains enthalten sollte.

Die potentiellen regulären Domains stammen schliesslich aus drei verschiedenen Quellen:

- aus *Hard-Ham*-Mails von *SpamAssassin* [Spa04]
- von Mitgliedern der *Distributed Computing Group* (DCG) [2] der ETH Zürich gesammelte Domains
- aus dem Linkarchiv von *Hockeyfans.ch*[3]

Bei den *Hard-Ham*-Mails handelt es sich um schwierig bestimmbare, reguläre E-Mails, die von *SpamAssassin* zum Testen von Spam-Filtern zur Verfügung gestellt werden. Im Linkarchiv von *Hockeyfans.ch* befinden sich viele relativ unbekannte Internetadressen von Eishockey-Fanclubs, deren Domains sehr gut den geforderten Kriterien für reguläre Domains - wenig verbreitet und relativ unbekannt - entsprechen. Die DCG-Mitglieder wurden dazu angehalten, Domains zu nennen, die möglichst gut die Kriterien erfüllen.

Insgesamt standen 2406 Domains zur Verfügung.

[2] `http://dcg.ethz.ch/`
[3] `http://www.hockeyfans.ch/`

3.3.2 Domains analysieren

Für jede Domain wurde die Anzahl Treffer bei Google von den in Tabelle 3.1 beschriebenen Suchabfragen ermittelt. Für Domains aus gemeldeten Spam-Mails existierte die Anzahl Treffer bereits, da diese zuvor vom Statistik-Server ermittelt wurde. Siehe dazu Abschnitt 2.5 im Allgemeinen und Abschnitt 2.5.2 im Speziellen. Für die regulären Domains musste die Analyse separat durchgeführt werden. Die Suchabfragen werden nachfolgend auch (*absolutes*) Kriterium genannt.

Kriterium	**Suchabfrage-Muster**
Domain	"<domain>"
Link	link: <domain>
Related	related: <domain>
Site	site: <domain>
Spam	<domain> + spam
Blacklist	<domain> + blacklist
SpamBlacklist	<domain> + spam + blacklist

Tabelle 3.1: Suchabfragen bei Google.

3.3.3 Domains bewerten

Nun ging es darum, die gesammelten Domains in verschiedene Kategorien einzuteilen. Die Kategorien sind:

spam Von Spam-Mail stammende Domains, auf dessen Internetseiten die in Spam-Mails beschriebenen Produkte wie Medikamente, Geld, Sex et cetera beworben werden.

fake Domains in Spam-Mails, um die eigentliche Spam-Domain zu verschleiern[4] oder um Spam typische Wörter maschinen*un*lesbar zu machen[5].

ok Reguläre Domains, z.B. *fam-muller.ch* oder *bluewin.ch*.

borderline_case Grenzfälle. Nicht klar bestimmbar, ob Spam-Domain.

whitelist Domains von den Grossen und Grössten des Internets: Z.B. von Firmen wie *Microsoft* oder *Apple* oder Internet-Behörden (*w3.org*). Diese Domains weisen eine extrem hohe Verbreitung auf und würden in eine globale Whitelist eingetragen.

invalid In diese Kategorie fallen Domains, die zum Zeitpunkt der Analyse nicht mehr aktiv gewesen sind, sowie Reseller-Domains wie *ch.vu* und spezielle Länder-Domains wie *co.uk*.

[4] z.B. mit "unsichtbarem" HTML-Code: `<font color="white">www.fakedomain1.com</font>`
[5] `Via<a href="www.fakedomain2.com"></a>gra`

email Domains, die von Email-Providern und -Gateways stammen (*"Free e-mail services by www.supermail.com"*).

Die Einteilung geschah mit Hilfe des selbst geschriebenen, Browser basierten Tools *Domain-Checker*, das die Domain mit ihrem Vorkommen in den E-Mails, den Suchergebnissen aus Abschnitt 3.3.2 sowie mit Links auf selbige darstellt. Abbildung 3.1 zeigt einen Screenshot dieses Tools.

Die Bewertung ergab die in Tabelle 3.2 dargestellte Verteilung.

Kategorie	**Anzahl Domains**	
spam	781	(32.4%)
fake	312	(13.0%)
ok	1'109	(46.1%)
borderline_case	9	(0.4%)
whitelist	74	(3.1%)
invalid	102	(4.2%)
email	19	(0.8%)
Total	2406	(100%)

Tabelle 3.2: Domain-Verteilung.

3.4 Ergebnisse

Für die Auswertung wird die Anzahl der Kategorien von sieben auf vier reduziert: *spam*, *fake*, *ok* und *whitelist*. Domains aus den drei anderen Kategorien werden von der Auswertung ausgenommen, da ihre Aussagekraft nicht vorhanden (*invalid*), beschränkt (*borderline_case*) oder nicht gewünscht (*email*) ist. Domains der Kategorien *ok* und *whitelist* werden in der Folge als *gute* Domains bezeichnet, solche der *spam*-Kategorie Spam-Domains und solche der *fake*-Kategorie Fake-Domains.

3.4.1 Statistische Kennzahlen

Die statistischen Kennzahlen beschreiben die Anzahl Treffer der Suchabfragen *Domain*, *Link*, *Related* und *Site* für jede Domain-Kategorie (*spam*, *fake*, *ok* und *whitelist*).

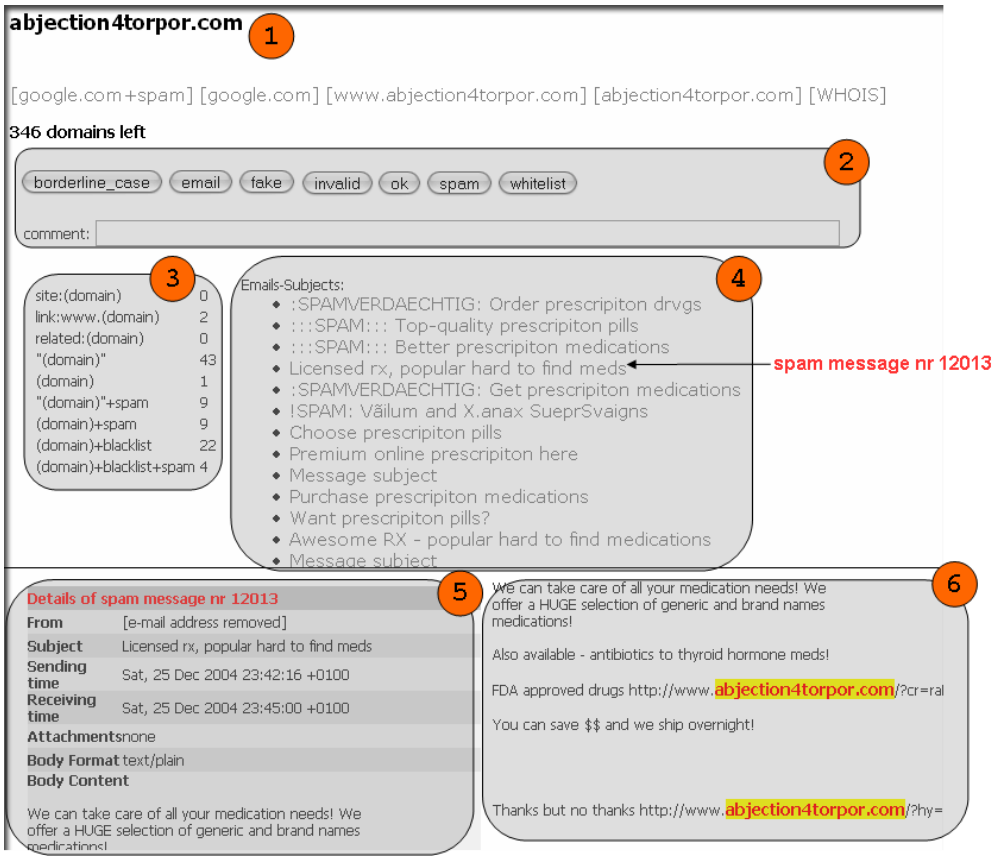


Abbildung 3.1: *Domain-Checker*. Mit diesem selbst geschriebenen Tool wurden die Domains in Kategorien eingeteilt. Legende: 1) Zu bewertende Domain. 2) Domain-Kategorien. 3) Ergebnisse der Suchabfragen bei Google. 4) Betreffzeilen von E-Mails, welche die Domain benutzen. 5) *Spam Cam* auf E-Mail Nr. 12013. 6) Quelletext dieser E-Mail mit hervorgehobener Verwendung der Domain.

Domain

Tabelle 3.3 zeigt die Anzahl Treffer der Suchabfrage *Domain*.

	Domain-Kategorie			
	spam	***fake***	***ok***	***whitelist***
Total N	781	312	1'109	74
Minimum	0	0	0	31'600
Erstes Quantil	10	44	550	1'072'500
Durchschnitt	485	22'596	305'591	5'318'912
Median	23	370	8'230	2'720'000
Drittels Quantil	70	3'865	134'000	7'360'000
Maximum	87'800	1'080'000	12'700'000	27'400'000
Standardabweichung	4'464	109'765	922'707	6'065'269

Tabelle 3.3: Statistische Kennzahlen des *Domain*-Kriteriums. Lesebeispiel: Die grösste Anzahl Treffer einer *Domain*-Suchabfrage mit einer Spam-Domain beträgt 87'800. Die durchschnittliche Anzahl Treffer einer *Domain*-Suchabfrage mit einer *ok*-Domain beträgt 305'591.

Kommentar: Dieses Kriterium zeigt deutlich die geringe Verbreitung von Spam-Domains im Internet. Drei Viertel der Domains der Kategorie *spam* liegen unterhalb von 70. Doch einzig aufgrund dieser Tatsache zu schliessen, eine Domain unterhalb dieses Wertes sei eine Spam-Domain, ist nicht möglich. 87 Domains oder etwa acht Prozent aus der Kategorie der *guten* Domains würden fälschlicherweise als Spam-Domain identifiziert. Dies sind unter anderem Domains von Privatpersonen oder Familien (`http://www.familie-meier.ch/`), die nicht stark verbreitet sind, jedoch öfter, als man meinen könnte, in E-Mails vorkommen. Auch das sichere Bestimmen einer *guten* Domain würde sich schwieriger herausstellen, als man annehmen könnte. Spam-Domains weisen des Öfteren ähnlich hohe Werte wie *gute* Domains auf. Deshalb würden entweder viele Spam-Domains nicht erkannt werden (bei zu hohem Grenzwert) oder viele *gute* Domains würden als Spam-Domains verurteilt (bei zu tiefem Grenzwert).

Fazit: Das Potential ist sowohl für die Erkennung von Spam-Domains als auch für *gute* Domains gering.

Link

Tabelle 3.4 zeigt die Anzahl Treffer der Suchabfrage *Link*.

	Domain-Kategorie			
	spam	***fake***	***ok***	***whitelist***
Total N	781	312	1'109	74
Minimum	0	0	0	0
Erstes Quantil	0	0	9	7'580
Durchschnitt	757	1'670	4'222	128'853
Median	0	0	137	25'600
Drittels Quantil	0	11	2'240	99'250
Maximum	581'000	403'000	458'000	3'030'000
Standardabweichung	20'790	23'006	20'857	372'467

Tabelle 3.4: Statistische Kennzahlen des *Link*-Kriteriums.

Kommentar: Hier zeigt sich deutlich, dass Spam-Domains nur sehr gering verlinkt sind. Einzelne Spam-Domains erreichen dennoch einen erstaunlich hohen Wert. Dies sind zum Beispiel Domains von Webspace Hosting Providern, die den Spammern nur Webspace jedoch keine eigene Domain zur Verfügung stellen, was zur Folge hat, dass die von einer Spam-Mail beworbene URL die Form `http://spam.hostingprovider.com/` hat. Nichtsdestotrotz sind mindestens drei Viertel von keiner anderen Webseite von durch Google indizierten Webseiten verlinkt. Jedoch stellt sich auch hier das Problem der möglicherweise nicht indizierten, vorwiegend privaten und schwach verbreiteten Webseiten der Form `http://www.familie-meier.ch/`.

Fazit: Das Potential zur Erkennung von Spam-Domains ist gering. *Gute* Domains lassen sich - mit Ausnahmen - erfolgreich erkennen.

Related

Tabelle 3.5 zeigt die Anzahl Treffer der Suchabfrage *Related*.

	Domain-Kategorie			
	spam	***fake***	***ok***	***whitelist***
Total N	781	312	1'109	74
Minimum	0	0	0	0
Erstes Quantil	0	0	0	13.3
Durchschnitt	0.4	3.4	11.2	21.9
Median	0	0	0	30
Drittels Quantil	0	0	27	31
Maximum	31	31	31	31
Standardabweichung	3.4	9.2	13.4	11.8

Tabelle 3.5: Statistische Kennzahlen des *Related*-Kriteriums.

Kommentar: Beim *Related*-Kriterium zeigt sich möglicherweise ein spezielles Phänomen der Google spezifischen Suche mit dem *Related*-Schlüsselwort. Suchabfragen mit diesem Schlüsselwort liefern deren Argumenten ähnliche Webseiten. Domains aus den Kategorien *ok* und *whitelist* haben mehrheitlich entweder einen *Related*-Wert von Null oder fallen in den Bereich von 17 bis 32. Dazwischen liegen nur wenige (45 bzw. 3.8%). Das Spezielle zeigt sich auch bei Suchabfragen mit Spam-Domains, jedoch im geringeren Ausmass: Die wenigen Domains mit einem *Related*-Wert grösser als Null (12 bzw. 1.5%) fallen allesamt in den Bereich von 17 bis 32. Da dies nur wenige Spam-Domains sind, können Domains mit einem hohen *Related*-Wert relativ problemlos als *gut* betrachtet werden.

Fazit: Das Potential zur Erkennung von Spam-Domains ist gering. *Gute* Domains lassen sich äusserst erfolgreich erkennen.

Site

Tabelle 3.6 zeigt die Anzahl Treffer der Suchabfrage *Site*.

	Domain-Kategorie			
	spam	***fake***	***ok***	***whitelist***
Total N	781	312	1'109	74
Minimum	0	0	0	0
Erstes Quantil	0	1	108	160'750
Durchschnitt	139	2'811	101'392	1'592'555
Median	2	4	1'050	531'500
Drittels Quantil	4	59	53'900	1'705'000
Maximum	52'900	212'000	6'620'000	23'000'000
Standardabweichung	2'231	18'281	344'322	3'168'290

Tabelle 3.6: Statistische Kennzahlen des *Site*-Kriteriums.

Kommentar: Drei Viertel der Spam-Domains haben einen kleineren *Site*-Wert als Vier. Dies lässt vermuten, dass man aus einem geringen solchen Wert, auf eine Spam-Domain schliessen könnte. Jedoch ergibt sich wiederum dasselbe Problem wie beim *Domain*- und *Link*-Kriterium. Je höher aber dieser Wert ist, desto eher handelt es sich um eine *gute* Domain. Suchabfragen bei Google mit dem *Site*-Schlüsselwort schränken die Suche auf Webseiten der angegebenen Domain ein. Spam-Domains umfassen stark mehrheitlich nur ein paar wenige Webseiten. Je höher der Wert, desto eher handelt es sich um eine *gute* Domain.

Fazit: Das Potential zur Erkennung von Spam-Domains ist gering. *Gute* Domains lassen sich äusserst erfolgreich erkennen.

Die Kriterien, die Zusatzwörter wie *spam* enthalten, haben ohne Vergleichswerte keine Aussagekraft. Deshalb wurden drei neue Kriterien eingeführt, die aus dem Verhältnis der jeweiligen Ergebnissen der Suchabfrage mit dem Zusatzwort zum Ergebnis des Kriteriums *Domain* gebildet werden. Die drei neuen, *relativen* Kriterien sind in Tabelle 3.7 definiert. Tabellen 3.8, 3.9 und 3.10 zeigen ihre statistischen Kennzahlen.

Kriterium	**Verhältnis**
SpamRelativ	*Spam*- zu *Domain*-Kriterium
BlacklistRelativ	*Blacklist*- zu *Domain*-Kriterium
SpamBlacklistRelativ	*SpamBlacklist*- zu *Domain*-Kriterium

Tabelle 3.7: Die drei *relativen* Kriterien.

	Domain-Kategorie			
	spam	***fake***	***ok***	***whitelist***
Total N	781	312	1'109	74
Minimum	.000	.000	.000	.000
Erstes Quantil	.226	.000	.001	.002
Durchschnitt	.451	.029	.027	.012
Median	.409	.007	.008	.067
Drittels Quantil	.667	.021	.024	.055
Maximum	1.000	.778	1.000	1.000
Standardabweichung	.363	.074	.064	.160

Tabelle 3.8: Statistische Kennzahlen des *SpamRelativ*-Kriteriums.

	Domain-Kategorie			
	spam	*fake*	*ok*	*whitelist*
Total N	781	312	1'109	74
Minimum	.000	.000	.000	.000
Erstes Quantil	.043	.000	.000	.000
Durchschnitt	.248	.005	.001	.002
Median	.143	.000	.000	.001
Drittels Quantil	.364	.001	.001	.002
Maximum	1.000	.377	.318	.070
Standardabweichung	.27	.026	.011	0.009

Tabelle 3.9: Statistische Kennzahlen des *BlacklistRelativ*-Kriteriums.

	Domain-Kategorie			
	spam	*fake*	*ok*	*whitelist*
Total N	781	312	1'109	74
Minimum	.000	.000	.000	.000
Erstes Quantil	.034	.000	.000	.000
Durchschnitt	.205	.001	.001	.001
Median	.111	.000	.000	.000
Drittels Quantil	.300	.000	.000	.000
Maximum	1.000	.130	.231	.028
Standardabweichung	.244	.008	.008	.004

Tabelle 3.10: Statistische Kennzahlen des *SpamBlacklistRelativ*-Kriteriums.

Kommentar: Alle drei relativen Kriterien weisen ein hohes Differenzierungspotential bezüglich der Spam-Domain-Identifizierung auf. Es ist nicht entscheidend, wie hoch die Trefferanzahl beim *Domain*-Kriterium ist. Vielmehr ist entscheidend, wie viele dieser Treffer beim Zusatz eines Spam bezogenen Wortes zur Suchabfrage übrig bleiben. Je höher der Anteil der verbleibenden Treffer ist, desto öfter steht die Domain in Zusammenhang mit Spam. Über einem bestimmten Schwellenwert befinden sich fast keine *guten* Domains mehr. Das Differenzierungspotential dieser Kriterien zeigt sich deutlich in Abbildung 3.2.

Vereinzelt weisen an und für sich *gute* Domains einen hohen Wert beim *Spam-Relativ*-Kriterium auf. Dies schränkt das Differenzierungspotential dieses Kriteriums ein. Vor allem Domains von Webseiten, die zum Beispiel ein Produkt zur Abwehr von Spam beschreiben (z.B. *SpamAssassin*) und deshalb, gezwungenermassen, oft mit dem Wort *spam* in Zusammenhang stehen, sind davon betroffen.

Das Differenzierungspotential bezüglich Erkennung einer *guten* Domain ist hingegen schwach, da bei diesen Kriterien viele Spam-Domains ähnliche Werte wie *gute* Domains aufweisen.

Fazit: Die relativen Kriterien eignen sich gut zur Identifizierung von Spam-Domains, jedoch nur bedingt zur Erkennung von *guten* Domains.

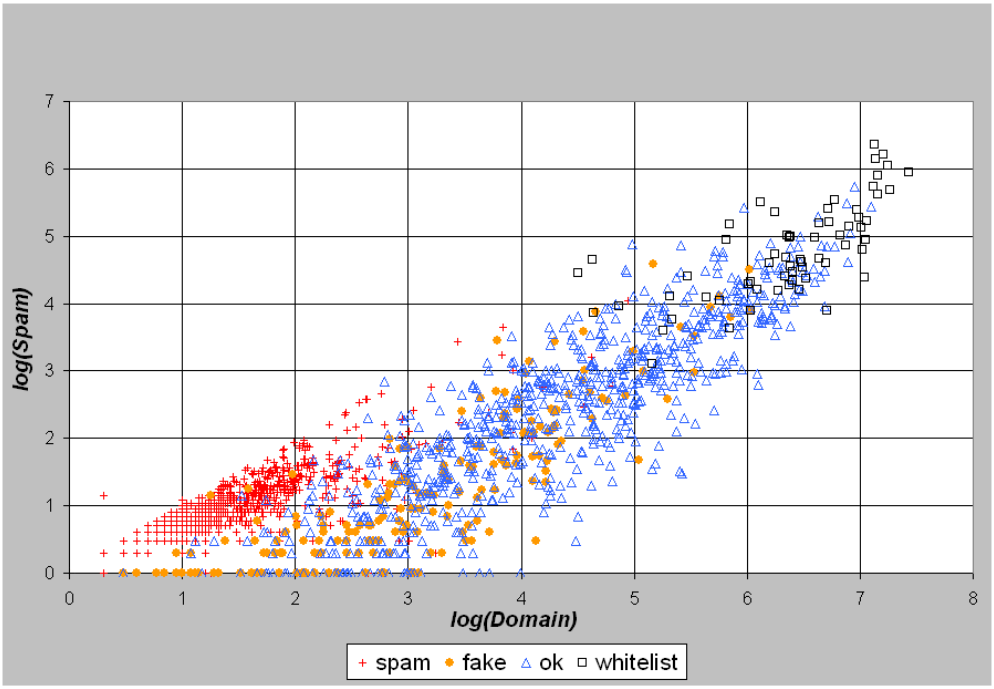


Abbildung 3.2: Clusterbildung anhand des Domain-Kriteriums (X-Achse) und des Spam-Kriteriums (Y-Achse). Während die Spam-Domains (+ *spam*) sich in einem beschränkten Bereich befinden, sind die *guten* Domains (△ *ok* und □ *whitelist*) auf der ganzen X-Achse verteilt und zugleich signifikant von den Spam-Domains getrennt. Die Fake-Domains (• *fake*) befinden sich sowohl in der von den Spam-Domains dominierten Region als auch in der Region der *guten* Domains.

3.4.2 Folgerungen

Die Erkenntnisse sind in Tabelle 3.11 zusammengefasst.

Kriterium	Eignung zur Erkennung von ...	
	... Spam-Domains	**... *guten* Domains**
Domain	gering	gering
Link	gering	mittel
Related	gering	hoch
Site	gering	hoch
Spam	ohne Vergleichswert nicht vorhanden	
Blacklist	siehe *Spam*	
SpamBlacklist	siehe *Spam*	
SpamRelativ	mittel bis hoch	gering, aber höher als bei *Domain*
BlacklistRelativ	hoch	siehe *SpamRelativ*
SpamBlacklistRelativ	sehr hoch	siehe *SpamRelativ*

Tabelle 3.11: Differenzierungspotential: Welches Kriterium eignet sich zur Identifizierung von Spam-Domains? Welches zur Identifizierung von *guten* Domains.

3.5 Domainator

3.5.1 Einleitung

Gestützt auf den Ergebnissen aus Abschnitt 3.4 implementierte ich einen Filter für Spamato. Dieser extrahiert aus den URLs einer E-Mail die Domains und trifft für jede Domain eine Entscheidung *Spam oder Nicht-Spam*. Wird eine Spam-Domain erkannt, so wird die E-Mail als Spam identifiziert und der entsprechende Spamato-Rückgabewert retourniert. Kann keine Spam-Domain gefunden werden, wird die E-Mail nicht als Spam angesehen. Falls die E-Mail keine Domains enthält, kann der Domainator keine Entscheidung treffen.

Der Algorithmus zur Identifizierung einer Spam-Domain wird im folgenden Abschnitt dargestellt. Abschnitt 3.5.3 präsentiert Testergebnisse aus Probeläufen des Domainator mit ausgewählten Domains und mit von Spamato-Benutzern gemeldeten Spam-Mails. Ein Vergleich mit den anderen Filtern von Spamato befindet sich in Kapitel 4, Abschnitt 4.2.1.

3.5.2 Algorithmus

In diesem Abschnitt wird der Algorithmus des Domainator zur Identifizierung von Spam-Domains erläutert.

Suchabfragen

In einem ersten Schritt werden die sieben, aus Abschnitt 3.3.2 bekannten Suchabfragen durchgeführt. Für die Beispiel-Domain *medzstoreonline.com* ergibt das die Ergebnisse aus Tabelle 3.12.

Da die Kriterien *Spam*, *Blacklist* und *SpamBlacklist* ohne Vergleichswerte nichts aussagen, werden aus ihnen neue Kriterien berechnet (siehe Abschnitt 3.4). Für die berechneten Kriterien ergeben sich die Werte aus Tabelle 3.13.

Kriterium	**Suchabfrage**	**Wert** (x)
Domain	"medzstoreonline.com"	124
Link	link: medzstoreonline.com	0
Related	related: medzstoreonline.com	0
Site	site: medzstoreonline.com	38
Spam	medzstoreonline.com + spam	34
Blacklist	medzstoreonline.com + blacklist	3
SpamBlacklist	medzstoreonline.com + spam+ blacklist	2

Tabelle 3.12: Ergebnisse der Suchabfragen bei Google mit der Beispiel-Domain *medzstoreonline.com*. Der Wert bezeichnet die Anzahl Treffer einer Suchabfrage.

Kriterium	**Wert** (x)
SpamRelativ	0.274
BlacklistRelativ	0.024
SpamBlacklistRelativ	0.016

Tabelle 3.13: Ergebnisse der relativen Kriterien anhand der Beispiel-Domain *medzstoreonline.com*.

Punktzahlberechnung

Im nächsten Schritt wird für jede der sieben Kriterien (K_i) entweder eine positive (p_i) oder eine negative Punktzahl (n_i) berechnet. Eine negative Punktzahl resultiert, falls der Wert eines relativen Kriteriums einen bestimmten Grenzwert (*Ham-Threshold*, HT_i) unterschreitet oder der Wert eines absoluten Kriteriums den Grenzwert überschreitet. Eine positive Punktzahl ergibt sich, wenn der Wert eines relativen Kriteriums einen bestimmten Grenzwert (*Spam-Threshold*, ST_i) überschreitet oder der Wert eines absoluten Kriteriums den Grenzwert unterschreitet. Falls die Treffermenge der Suchabfragen leer ist, wird der Algorithmus abgebrochen und die Domain nicht bewertet.

Tabelle 3.14 zeigt die verwendeten Grenzwerte. Diese wurden anhand der Ergebnisse der Domain-Analyse (siehe Abschnitt 3.4) bestimmt. Dabei wurde nicht nach mathematischen Kriterien vorgegangen. Es wurde darauf geachtet, dass nicht zu viele Domains in den jeweils falschen Bereich - zum Beispiel *gute* Domains in den Bereich oberhalb des *Spam-Tresholds* des *SpamRelativ*-Kriteriums - zu liegen kommen.

Die positive Punktzahl p_i berechnet sich für ein relatives Kriterium K_i mit Wert x gemäss Formel (3.1) bzw. Formel (3.2) für ein absolutes Kriterium.

$$p_i(x) = \begin{cases} \frac{s_{\leq x, \geq ST_i}}{s_{\geq ST_i}} & \text{falls } x \geq ST_i \\ 0 & \text{sonst} \end{cases} \tag{3.1}$$

$$p_i(x) = \begin{cases} \frac{s_{\geq x, \leq ST_i}}{s_{\leq ST_i}} & \text{falls } x \leq ST_i \\ 0 & \text{sonst} \end{cases} \tag{3.2}$$

Kriterium K_i	**Typ**	HT_i	ST_i
Domain	absolut	8	9
Link	absolut	2	3
Related	absolut	0	1
Site	absolut	64	65
SpamRelativ	relativ	0.109	0.059
BlacklistRelativ	relativ	0.015	0.007
SpamBlacklistRelativ	relativ	0.015	0.007

Tabelle 3.14: Grenzwerte für die absoluten und relativen Kriterien.

S : Menge aller Spam-Domains der Domain-Analyse.
$s_{\leq x, \geq ST_i}$: Anzahl Domains aus S, deren Kriteriumwert $\leq x$ und $\geq ST_i$ ist.
$s_{\geq ST_i}$: Anzahl Domains aus S, deren Kriteriumwert $\geq ST_i$ ist.
$s_{\geq x, \leq ST_i}$: Anzahl Domains aus S, deren Kriteriumwert $\geq x$ und $\leq ST_i$ ist.
$s_{\leq ST_i}$: Anzahl Domains aus S, deren Kriteriumwert $\leq ST_i$ ist.

Die negative Punktzahl n_i berechnet sich für ein relatives Kriterium K_i mit Wert x gemäss Formel (3.3) bzw. Formel (3.4) für ein absolutes Kriterium.

$$n_i(x) = \begin{cases} \frac{d_{\geq x, \leq HT_i}}{d_{\leq HT_i}} & \text{falls } x \leq HT_i \\ 0 & \text{sonst} \end{cases} \tag{3.3}$$

$$n_i(x) = \begin{cases} \frac{d_{\leq x, \geq HT_i}}{d_{\geq HT_i}} & \text{falls } x \geq HT_i \\ 0 & \text{sonst} \end{cases} \tag{3.4}$$

D : Menge aller *guten* Domains der Domain-Analyse.
$d_{\geq x, \leq HT_i}$: Anzahl Domains aus D, deren Kriteriumwert $\geq x$ und $\leq HT_i$ ist.
$d_{\leq HT_i}$: Anzahl Domains aus D, deren Kriteriumwert $\leq HT_i$ ist.
$d_{\leq x, \geq HT_i}$: Anzahl Domains aus D, deren Kriteriumwert $\leq x$ und $\geq HT_i$ ist.
$d_{\geq HT_i}$: Anzahl Domains aus D, deren Kriteriumwert $\geq HT_i$ ist.

Gewichtung

Nun werden die positiven (p_i) und die negativen Punktzahlen (n_i) mit einem spezifischen Gewicht versehen, welches das Differenzierungspotential eines Kriteriums widerspiegelt, und getrennt aufsummiert.

$$P = \frac{\sum_{1 \leq i \leq 7} (p_i * w_{Pos_i})}{\sum_{1 \leq i \leq 7} w_{Pos_i}} \tag{3.5}$$

$$N = \frac{\sum_{1 \leq i \leq 7} (p_i * w_{Neg_i})}{\sum_{1 \leq i \leq 7} w_{Neg_i}} \tag{3.6}$$

Tabelle 3.15 gibt die Gewichte w_{Pos_i} und w_{Neg_i} an. Diese wurden aus den Erkenntnissen von Tabelle 3.11 und aus Tests mit den analysierten Domains aus Abschnitt 3.3 abgeleitet.

Kriterium	w_{Pos_i}	w_{Neg_i}
Domain	4	4
Link	4	20
Related	4	80
Site	4	80
SpamRelativ	60	10
BlacklistRelativ	100	10
SpamBlacklistRelativ	150	10

Tabelle 3.15: Gewichte für die positiven und die negativen Punktzahlen.

Aggressivität

Mit der Einstellung der Aggressivität zwischen 0 und 1 kann der SPAMATO-Benutzer einstellen, ob die positive (P) oder die negative Punktzahl (N) mehr berücksichtigt werden soll. Eine Aggressivität von 0 hat zur Folge, dass keine Spam-Domains, und somit keine Spam-Mails, identifiziert werden können. Da umgekehrt eine Aggressivität von 1 jede Domain inklusive der *guten* als Spam-Domain erkennen würde, wurde eine *maximale Aggressivität* ($A_{max} = 0.9$) eingeführt. Oberhalb dieses Wertes würde die Anzahl der *False Positives* zu stark beginnen anzusteigen.

Die Aggressivität wird folgendermassen verwendet: Als erstes wird die vom SPAMATO-Benutzer einstellbare Aggressivität A mit dem Wert der maximalen Aggressivität multipliziert.

$$A' = A * A_{max} = A * 0.9 \tag{3.7}$$

Anschliessend werden die aufsummierten Punktzahlen P bzw. N gemäss Formel (3.8) bzw. Formel (3.9) skaliert.

$$P' = P * \begin{cases} \dfrac{A'}{1 - A'} & \text{falls } A' \leq 0.5 \\ 1 & \text{sonst} \end{cases} \tag{3.8}$$

$$N' = N * \begin{cases} \dfrac{1 - A'}{A'} & \text{falls } A' \geq 0.5 \\ 1 & \text{sonst} \end{cases} \tag{3.9}$$

Entscheidung

Formel (3.10) berechnet den Wert V, der darüber entscheidet, ob die Domain eine Spam-Domain ist oder nicht.

$$V = P' - N' \rightarrow \begin{cases} V > 0 & \text{Spam-Domain} \\ V \leq 0 & \textit{keine} \text{ Spam-Domain} \end{cases} \tag{3.10}$$

3.5.3 Tests

Test mit gesammelten Domains

Ein mit der Aggressivität $A = 0.5$ durchgeführter Test des DOMAINATOR-Algorithmus' mit den in Abschnitt 3.3 gesammelten Domains ergab folgende Ergebnisse:

Domain-Kategorie	**Anzahl Domains**	**Fehl-Identifikationen**
fake	312	9 (2.9%)
ok	1'109	4 (0.4%)
whitelist	74	0 (0.0%)
Total (*False Positives*)	1'495	13 (0.9%)
spam (False Negatives)	781	92 (11.8%)

Tabelle 3.16: Anzahl *False Negatives* und *False Positives* eines DOMAINATOR-Tests mit 2'276 Domains. Während circa 12% der Spam-Domains nicht erkannt wurden, liegt der Anteil falsch erkannter, *guter* Domains bei weniger als einem Prozent (0.9%). Der gegenüber der *False Positive*-Rate grosse Anteil nicht erkannter Spam-Mails ist insofern nicht schlimm, als dass ein nicht erkanntes Spam-Mail weit weniger folgenreich ist wie ein fälschlicherweise als Spam identifiziertes E-Mail.

Abbildung 3.3 zeigt, wie sich die Fehl-Identifikations-Rate in Abhängigkeit der Aggressivität verhält.

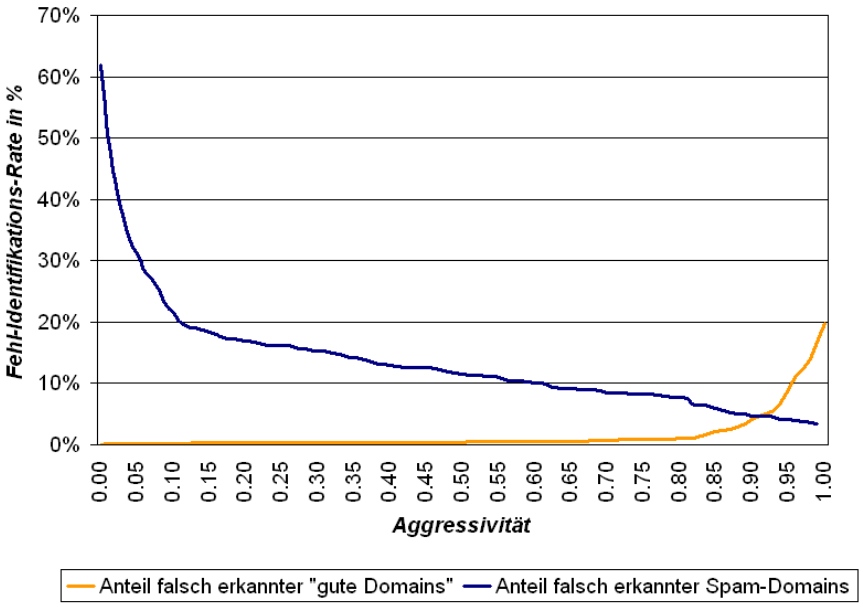


Abbildung 3.3: Fehl-Identifikations-Rate (Y-Achse) in Abhängigkeit der Aggressivität (X-Achse). Zwischen einer Aggressivität von circa 0.1 und 0.8 liefert der DOMAINATOR akzeptable Raten für *False Positives* und *False Negatives*. Ausserhalb dieses Bereich steigen bzw. sinken die Raten exponentiell.

Test mit gemeldeten Spam-Mails

Um das DOMAINATOR-Spam-Erkennungspotential von echten Spam-Mails herauszufinden, wurde er mit allen, von SPAMATO automatisch per *Auto-Report* mitgeteilten und von SPAMATO-Benutzern gemeldeten, E-Mails getestet. Diese Mails stammen vom 19. November 2004 (Inbetriebnahme Statistik-Server) bis zum 7. Februar 2005 (Stichtag). Total wurden 20'363 Spam-Mails durch den DOMAINATOR überprüft.

Der Test ergab, dass 13'542 (66.5%) E-Mails als Spam identifiziert wurden. Bei 4'301 (21.1%) E-Mails konnte der DOMAINATOR nichts aussagen, da sie keine Domains enthalten hatten. Die restlichen 2520 (12.4%) E-Mails wurden nicht als Spam identifiziert. Würde man nur diejenigen Mails betrachten, die Domains enthalten, würde die Trefferquote identifizierter Spam-Mails bei 84.3% (13'542 von 16'062) liegen.

Dieser Test sagt nur etwas über die Trefferquote bezüglich Spam-Identifizierung aus, jedoch nicht über die Anzahl *False Positives*, weil die Menge der Test-Mails nur solche enthielt, die nie widerrufen (*Revoke*) worden sind. Widerrufene Spam-Mails werden aus der Datenbank gelöscht.

3.5.4 Diskussion

Bekanntheit

Um brauchbare Suchergebnisse zu erhalten, müssen die Domains in irgendeiner Form von der verwendeten Suchmaschine indiziert sein. Entweder muss die Domain selber besucht worden sein oder eine andere, bereits indizierte Webseite muss die Domain verlinken oder erwähnen. Doch auch dann muss die Webseite von der Suchmaschine erneut besucht werden, damit die Domain in den Suchmaschinen-Index gelangt. Vom Versenden der Spam-Mail bis zum Auftauchen der Domain in der Treffermenge einer Suchmaschine vergeht eine gewisse Zeit - meist sind dies ein paar Wochen.

Man könnte annehmen, dass der Ansatz des DOMAINATOR scheitern muss, weil vom Bekanntwerden der Spam-Domain bis zum Empfang einer Spam-Mail, die diese beinhaltet, nur wenige Stunden vergehen. Die Zeit also nicht reicht, dass die Domain überhaupt in einem Suchmaschinen-Index auftauchen kann. Diese Annahme trifft jedoch nicht zu. Es hat sich gezeigt, dass eine genügend grosse Anzahl von in Spam-Mails enthaltenen Domains beim Empfang der E-Mail bekannt sind, mit denen eine Analyse der E-Mail - ob Spam oder nicht - möglich sind.

Nichtsdestotrotz bleibt die Voraussetzung, dass eine Spam-Domain in irgendeiner Form von einer Suchmaschine indiziert sein muss, damit die Domain als Spam-Domain identifiziert werden kann. Der momentane Ansatz des DOMAINATOR kann nicht verwendet werden, um sämtliche Spam-Mails zu erkennen. Jedoch könnte eine Weiterentwicklung des DOMAINATOR zusätzliche Domain-Eigenschaften in die Spam-Entscheidung miteinbeziehen, die nicht den genannten Voraussetzungen unterlegen sind. Als Beispiel möge das Alter der Domain oder ihre Erreichbarkeit (Ping-Zeit) dienen.

Falscher Ansatz?

Wie wir gesehen haben, ist der DOMAINATOR sehr stark davon abhängig, wie oft eine Domain im Index einer Suchmaschine vertreten ist, insbesondere im Zusammenhang mit den Wörtern *spam* und *blacklist*. Dies kann zur Annahme führen, es würde reichen, die Domain bei entsprechenden Blacklists und/oder Spam-Verzeichnissen abzufragen. Eine Annahme, die nicht ganz falsch ist, aber den grossen Vorteil einer Suchmaschine unterschlägt. Diese Verzeichnisse müssten von Hand administriert werden. Die Suchmaschine hingegen hat einen Grossteil dieser Listen bereits in ihrem Index. Des Weiteren sind andere Entscheidungshilfen in Form von *site*-Abfragen und dergleichen möglich.

Es spricht jedoch nichts dagegen, Abfragen bei Blacklists mit in die Bewertung einer Domain aufzunehmen, um die Erkennungsrate des DOMAINATOR zu erhöhen.

Ungenauigkeit der Suchergebnisse und Trefferanzahl

Eine Suchabfrage mit *ofa.ch* im Suchbegriff liefert neben Webseiten, die direkt oder indirekt etwas mit dieser Domain zu tun haben, auch Webseiten, die diese

Zeichenfolge eher zufällig enthalten. Dies kann die Wahrscheinlichkeit der Identifikation einer Spam-Mail sowohl erhöhen als auch verringern. Verfälschungen treten gleichermassen auf, wenn eine Domain einen Postfix einer anderen bildet (*oca.ch* ist Postfix von *kkf-oca.ch*).

Weitere Problempunkte

Reseller-Domains, spezielle Top Level Domains: Domains aus Grossbritanien haben die Endung *co.uk*, Reseller-Domains zum Beispiel *ch.vu*. Diese Domains werden vom DOMAINATOR nicht richtig erkannt. [6]

Unerkennbare Domains: Oft sind URLs anzutreffen, die dahingehend manipuliert wurden, dass sie nicht mehr automatisch erkennbar sind. [7]

Domain-Missbrauch Tritt dann auf, wenn Spammer fremde Webseiten missbrauchen, um Email-Adressen zu verifizieren. Dies geschieht durch einen eindeutig identifizierbaren Link auf eine fremde Webseite. Die Domains dieser Webseiten geraten in ein falsches Licht.

3.5.5 Fazit

Der DOMAINATOR analysiert nur eine Eigenschaft einer Domain und erzielt dadurch bereits beachtliche Ergebnisse, wie die Tests in Abschnitt 3.5.3 und die Statistiken in Abschnitt 4.2.1 zeigen. Durch Hinzunahme von weiteren Eigenschaften wäre es möglich, eine höhere Spam-Erkennungsrate bei einer tieferen *False Positive*-Rate zu erreichen und SPAMATO noch besser zu machen.

[6] In den SPAMATO-Versionen nach dem 21. Januar 2005 existiert dieses Problem nicht mehr, da die Domain-Extraktion verbessert wurde und nun auch diese Art von Domains erkennt werden können. Sämtliche Analysen und Tests wurden jedoch ohne diese Domains durchgeführt.

[7] Diese Tatsache kann als Erfolg der URL-basierten Spam-Filter betrachtet werden. Derart manipulierte URLs können von den Email-Programmen weder erkannt noch hervorgehoben und von den Empfängern nicht angeklickt werden. Der "Erfolg" von Spam-Mails mit solchen URLs dürfte geringer sein als von denjenigen mit erkennbaren URLs.

4 Statistiken

4.1 Einleitung

Damit Aussagen über SPAMATO wie zum Beispiel über das Benutzerverhalten oder über die Filterqualität gemacht werden können, müssen die vom Statistik-Plugin gesammelten und an den Statistik-Server geschickten Daten ausgewertet und aufbereitet werden. Diese Auswertungen in Form von verschiedenen Grafiken und Statistiken werden in diesem Kapitel behandelt.

Die Statistiken können auf der SPAMATO-Homepage[1] unter *Statistics* sowohl für einen bestimmten Benutzer (*Personal Usage Statistics*) als auch für alle SPAMATO-Benutzer zusammen (*Global Usage Statistics*) betrachtet werden.

Sämtliche in diesem Kapitel abgebildeten Statistiken wurden am 7. Februar 2005 um 12:30 Uhr erstellt und beziehen sich auf alle SPAMATO-Benutzer. Der bislang letzte SPAMATO-Release wurde am 21. Januar 2005 durchgeführt. Diese Version beinhaltet unter anderem zwei neue Filter: den DOMAINATOR und den *Ruleminator*. Aufgrund von Tests vor dem eigentlichen Release reichen die Daten der neuen SPAMATO-Filter jedoch bis zum 19. Januar zurück.

Die Grafiken wurden mit der für Forschungszwecke kostenlos nutzbaren PHP-Bibliothek *JpGraph* [Adi04] erstellt.

4.2 Statistiken

Folgende Statistiken sind verfügbar:

Filter Comparison Filtervergleich. Zeigt, wie sich ein bestimter SPAMATO-Filter bei einer Spam-Mail-Identifikation im Vergleich zu den anderen SPAMATO-Filtern verhält.

Filter Statistics Filter-Statistik. Zeigt, was die SPAMATO-Filter entschieden haben.

Number of Spam Detections / Reports / Revokes Beantwortet Fragen wie "Wie viele Spam-Mails wurden von SPAMATO erkannt (*Spam Detections*)?", "Wie viele wurden dem System gemeldet (*Reports*)?" oder "Wie viele Entscheide wurden widerrufen (*Revokes*)?".

Success Rate Erfolgsrate. Wie sieht das Verhältnis der Anzahl identifizierter zur Anzahl nicht identifizierter Spam-Mails aus?

Daily Distribution Tagesverteilung. Zeigt die Verteilung der SPAMATO-Operationen während eines (Wochen/Arbeits)-Tages.

[1] `http://www.spamato.net/`

4.2.1 Filter Comparison

Mit dieser Statistik lässt sich herausfinden, wie sich ein bestimmter SPAMATO-Filter bei einer Spam-Mail-Identifikation im Vergleich zu den anderen SPAMATO-Filtern verhält. Daraus lassen sich Aussagen über die Qualität der Spam-Identifikation eines Filters ableiten. Über die Anzahl *False Positives*, die ein Filter (mit)verursacht, sagt diese Statistik jedoch nichts aus. Abbildung 4.1 vergleicht den in Abschnitt 3.5 vorgestellten Filter DOMAINATOR mit den anderen SPAMATO-Filtern für den Monat Januar 2005.

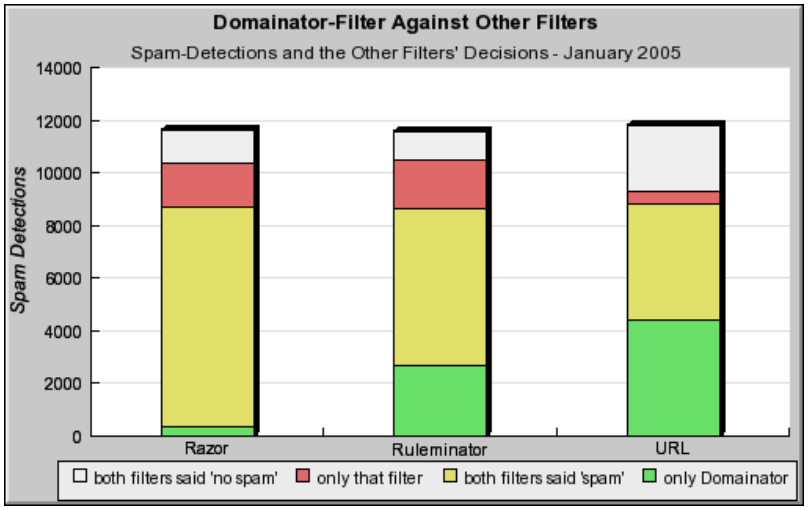


Abbildung 4.1: Der DOMAINATOR-Filter im Vergleich mit den anderen SPAMATO-Filtern.

Tabelle 4.1 zeigt die der Abbildung 4.1 zu Grunde liegenden Werte.

Filter	nur Domainator	beide Filter: "Spam"	nur dieser Filter	beide Filter: "kein Spam"
Razor	365	8'328	1'692	1'252
Ruleminator	2'679	5'944	1'881	1'079
URL	4'402	4'411	499	2'490

Tabelle 4.1: Der Abbildung 4.1 zu Grunde liegende Werte. Die Grundmenge bilden alle Spam-Identifizierungen von SPAMATO, an welchen *beide* Filter im Januar 2005 beteiligt waren. Im Vergleich mit dem *URL*-Filter identifizierte 4'402 Mal lediglich der DOMAINATOR eine Spam-Mail, während der *URL*-Filter keine Aussage machen konnte oder die E-Mail nicht als Spam identifizierte. In 4'411 Fällen erkannten beide und in 499 Fällen erkannte nur der *URL*-Filter die Spam-Mail. In den restlichen 2'490 Fällen konnten beide Filter die E-Mail nicht als Spam identifizierten.

Kommentar: Im Vergleich mit dem *URL*-Filter schneidet der DOMAINATOR besser ab. Während er dem *Razor*-Filter deutlich unterlegen ist, ist er dem *Ruleminator*-Filter leicht überlegen.

4.2.2 Filter Statistics

Diese Statistik stellt die Verteilung der Filterentscheidungen - "E-Mail ist Spam" (*spam*), "E-Mail ist *kein* Spam" (*ok*) und "unbekannt" (*unknown*) - dar. Abbildung 4.2 zeigt diese Verteilung im Januar 2005. Eine Verteilung der globalen SPAMATO-Entscheidungen kann aus den zum Statistik-Server geschickten Daten nicht gewonnen werden, da lediglich bei einer Spam-Identifikation Daten an den Statistik-Server geschickt werden. Der Statistik-Server wird nicht über alle *Check*-Operationen in Kenntnis gesetzt (siehe Abschnitt 2.4.2).

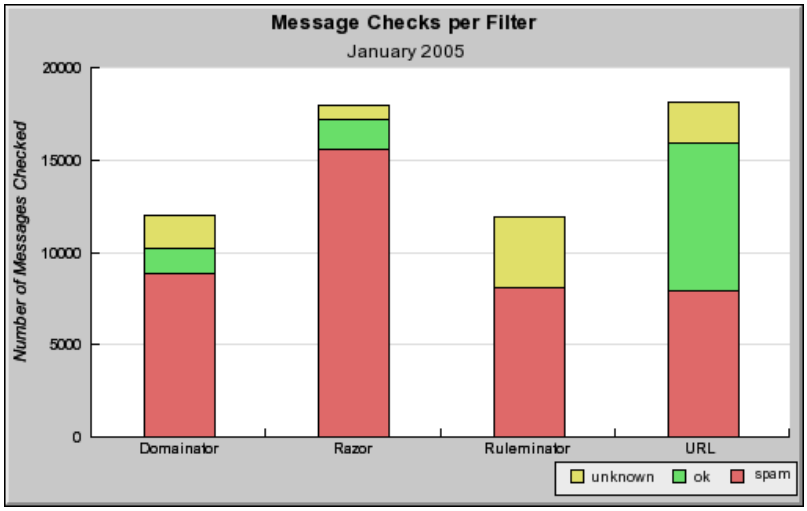


Abbildung 4.2: Verteilung der Filterentscheidungen im Januar 2005.

Tabelle 4.2 zeigt die der Abbildung 4.2 zu Grunde liegenden Werte.

Filter	**spam**	**ok**	**unknown**
Domainator	8'828	1'420	1'762
Razor	15'540	1'621	822
Ruleminator	8'117	0	3'824
URL	7'926	7'981	2'223

Tabelle 4.2: Der Abbildung 4.2 zu Grunde liegende Werte. Würde SPAMATO nur aus dem DOMAINATOR-Filter bestehen, wären im Januar 8'828 oder knapp drei Viertel der 12'010 Spam-Mails immer noch als solche identifiziert geworden. 1'420 (11.8%) Spam-Mails wären nicht identifiziert worden und bei 1'762 (14.7%) der E-Mails hätte keine Aussage getroffen werden können. Die unterschiedlichen Zeilentotale sind dadurch erklärbar, dass die Filter DOMAINATOR und *Ruleminator* erst seit dem bis anhin letzten Release von SPAMATO am 21. Januar 2005 verwendet und dass nur Entscheidungen von aktivierten Filtern miteinbezogen werden.

4.2.3 Number of Spam Detections / Reports / Revokes

Diese Statistik zeigt die Anzahl erkannter Spam-Mails (*Spam Detections*), die Anzahl von SPAMATO-Benutzern gemeldeter (*Reports*) oder die Anzahl widerrufener Spam-Mails (*Revokes*) an den Tagen eines bestimmten Monates, den letzten 30 Tage oder als Monats-Total. Nachfolgend die Definitionen dieser Begriffe.

Spam Detections Die Anzahl sämtlicher, von SPAMATO identifizierter und herausgefilterter Spam-Mails. Ein späteres Widerrufen dieser Entscheidung durch den SPAMATO-Benutzer (*Revoke*) hat keinen Einfluss auf diese Zahl.

Reports Die Anzahl sämtlicher, von SPAMATO-Benutzern gemeldeter Spam-Mails, die nicht als Spam identifiziert worden sind. E-Mails, die automatisch gemeldet werden, haben keinen Einfluss auf diese Zahl.

Revokes Die Anzahl sämtlicher, falsch identifizierter (*Spam Detections*) und von einem SPAMATO-Benutzer widerrufenen Spam-Mails. Eine zuvor gemeldete, jedoch danach widerrufene, Spam-Mail hat keinen Einfluss auf diese Zahl.

Abbildung 4.3 zeigt die Anzahl gemeldeter Spam-Mails im Januar 2005.

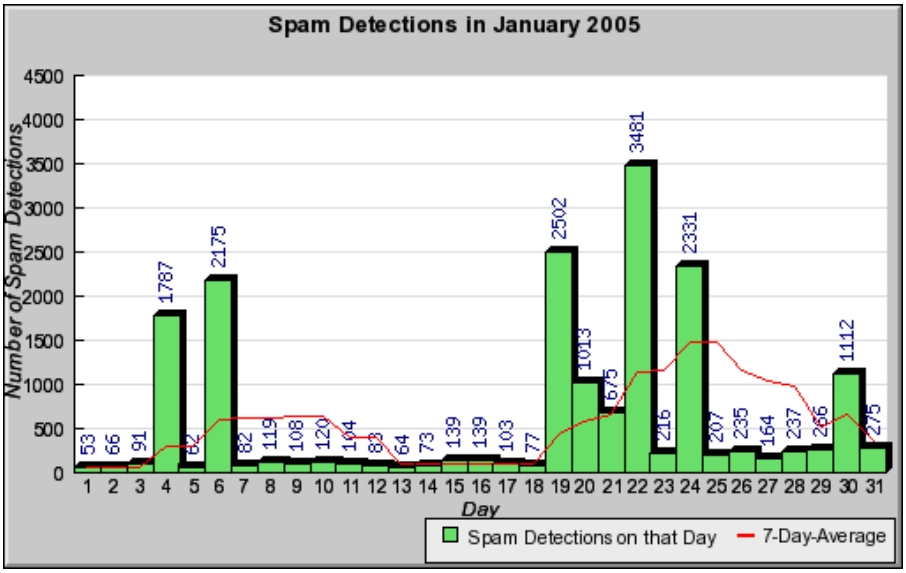


Abbildung 4.3: Gemeldete Spam-Mails im Januar 2005. Am 30. Januar 2005 wurden 1112 E-Mails von den SPAMATO-Benutzern als Spam gemeldet. Die Peaks stammen von Überprüfungen der *Honeypots*. Dies sind extra eingerichtete Email-Adressen ausschliesslich um damit Spam zu sammeln. Die Linie stellt den jeweiligen Durchschnitt der letzten sieben Tage dar. Der Durchschnitt pro Tag in diesem Monat betrug 586. Der Median lag bei 139.

4.2.4 Success Rate

Diese Statistik stellt die Anzahl identifizierter Spam-Mails (*Spam Detections*, S) der Anzahl als Spam gemeldeter E-Mails (*Reports*, R) gegenüber. Die Erfolgsrate (*Success Rate*, σ) ist gemäss Formel 4.1 definiert. Abbildung 4.4 zeigt die Erfolgsrate im Januar 2005.

$$\sigma = \frac{S}{S+R} \tag{4.1}$$

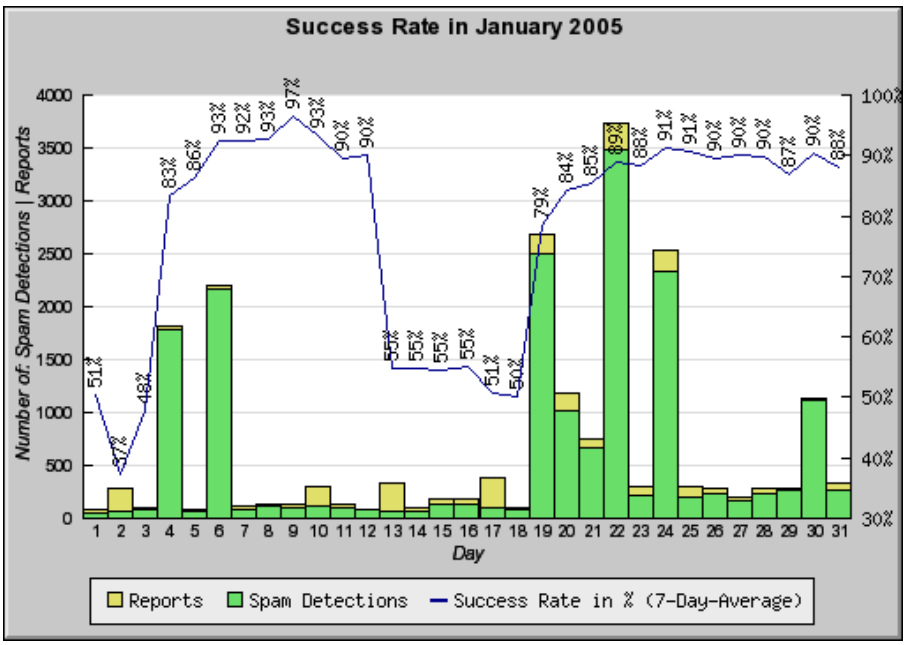


Abbildung 4.4: Der untere Teil der Balken repräsentiert die Anzahl von SPAMATO identifizierter Spam-Mails (*Spam Detections*) an einem bestimmten Tag und der obere Teil die Anzahl gemeldeter Spam-Mails (*Reports*) an jenem Tag. Die Linie entspricht dem Durchschnitt der Erfolgsrate σ, berechnet aus den *Reports* und *Spam Detections* der jeweils letzten sieben Tage.

4.2.5 Daily Distribution

Bei dieser Statistik wird die Verteilung der SPAMATO-Operationen eines Tages dargestellt. Dabei kann zwischen Wochentagen und Tagen des Wochenendes unterschieden werden. Abbildung 4.5 zeigt die Verteilung von *Spam Detections* eines Wochentages. Die zu Grunde liegenden Daten reichen bis zur Inbetriebnahme des Statistik-Servers am 19. November 2004 zurück.

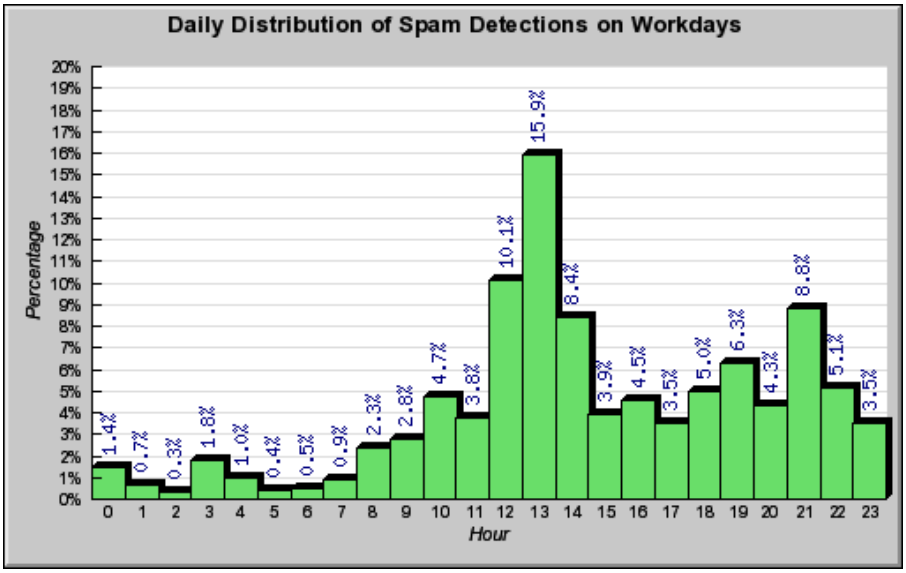


Abbildung 4.5: Verteilung von *Spam Detections* während eines Wochentages seit dem 19. November 2004. *Hour 3* entspricht der Zeit von drei bis vier Uhr nachts. Deutlich zu sehen ist die Spitze kurz nach der Mittagspause (13-14 Uhr, 15.9%)

5 Weiterführende Arbeiten

Die Zeit dieser Diplomarbeit war auf vier Monate beschränkt. Nicht allen Ideen und Vorstellungen konnten nachgegangen werden. Dieses Kapitel gibt Anregungen für weiterführende Arbeiten im Bereich *Statistik und* Spamato.

5.1 Erweiterung von Spamato

Spamato gewichtet alle Filterentscheidung gleich stark. Es identifiziert eine E-Mail als Spam, falls mindestens ein Filter diese E-Mail als Spam erkennt hat, ungeachtet dessen, wie sich die anderen Spamato-Filter entschieden haben.

Dieses triviale System könnte durch ein (automatisches) Gewichtungssystem ersetzt werden, das die Entscheidungen der Spamato-Filter je nach der Qualität ihrer bisherigen Spam-Identifikationen unterschiedlich stark gewichtet. Zum Beispiel würde ein Spamato-Filter, dessen Spam-Mail-Erkennungen nie durch einen *Revoke* widerrufen wurde, ein höheres Gewicht erhalten, als einer, dessen Entscheidungen des Öfteren widerrufen werden.

Für die Realisierung eines Gewichtungssystemes müsste eine statistische Auswertung folgender Fragestellungen durchgeführt werden:

- Welche Spamato-Filter sind wie oft an Fehl-Identifikationen beteiligt? Sind es immer die gleichen?
- Welcher Filter identifiziert wie viele Spam-Mails?
- Sind immer die gleichen Filter an einer Spam-Identifikation beteiligt? Sind zwei oder mehrere Filter immer der gleichen Meinung? Widersprechen sich die Filter?

5.2 Verbesserung des Domainator

Neben der Verbreitung der Domain wäre es denkbar, weitere Domain-Eigenschaften in die Filter-Entscheidungen miteinzubeziehen.

Round-Trip-Zeit, Anzahl Hops Lassen sich aus der Round-Trip-Zeit oder der Anzahl Hops zur Domain Aussagen machen, ob es sich um eine Spam-Domain handeln könnte?

Domain-Alter Spam-Domains werden oft nur für eine bestimmte Spam-Mail erzeugt. Die Vermutung liegt nahe, dass zwischen der Registrierung der Domain und dessen Gebrauch in einer Spam-Mail nicht allzu viel Zeit liegt. Daraus könnten Rückschlüsse auf ihre Eigenschaft als Spam-Domain gezogen werden. Der Zeitpunkt der Registrierung würde mittels einer

WHOIS-Abfrage durchgeführt werden. Doch genau hier liegt das Problem. Es existiert keine einheitliche Struktur der WHOIS-Einträge, was ein automatisches Auslesen des Registrierungsdatums der Domain erschwert.

Automatisch berechnete Parameter Die Gewichte und Grenzwerte sind fix im DOMAINATOR-Quellcode verankert und begründen auf einer Analyse von *alten* Spam-Mails. Sie könnten durch dynamisch berechnete Werte ersetzt werden, die aus einer Analyse *aktueller* Spam-Mails stammen.

Weitere Verhältnisse Es könnten weitere Verhältnisse aus den Kriterien gebildet werden. Anstatt nur Verhältnisse aus dem *Domain*-Kriterium zu bilden, wären auch Verhältnisse zum Beispiel des *Site*-Kriteriums zum *Link*-Kriterium denkbar.

Blacklists Um die Spam-Domain-Identifizierung zu ergänzen, könnten fragliche Domains bei bekannten Verzeichnissen von Spam-Domains (Blacklists) nachgefragt werden.

5.3 Effizienter erstellte Statistiken

Die Statistiken über SPAMATO auf `http://www.spamato.net/` werden bei jedem Aufruf neu berechnet. Dies kann je nach Art der Statistik mehrere Sekunden dauern.

Abhilfe schaffen würde ein intelligentes Caching-Verfahren auf Datenbankebene, das oft benötigte Zwischenergebnisse abspeichert und bei Bedarf wieder verwendet. Zum Beispiel könnte die Anzahl Spam-Identifikationen für jeden Benutzer und jeden Tag in einer separaten Tabelle abgespeichert werden.

A Spamato-Events

Dieser Anhang beschreibt die in Kapitel 2 beschriebenen Events für das Statistik-Framework.

Argumente in eckigen Klammern müssen nicht angegeben werden. Solche in geschweiften Klammern bilden zusammen ein Objekt und sind mehrwertig. Die Datentypen der Paramter wurden aus Gründen der Übersichtlichkeit weggelassen. Sie entsprechen jeweils den korrespondierenden Felder in der Datenstruktur.

Der `mail`-Parameter ist bei jedem Event vorhanden, weil im Mail-Objekt eine Laufnummer enthalten ist, die eine SPAMATO-Operation identifiziert. Anhand dieser Laufnummer kann der Event-Handler die eintreffenden Events dem richtigen `StatReport` zuordnen.

A.1 STAT_PRE_CHECK-Event

Verwendung	von SPAMATO vor einer *Check*-Operation	
abgebildet auf	`StatReport`	
Vorbedingung	keine	
Argumente	`mail`	E-Mail
	`username`	Benutzername
	`client_id`	Client-Identifikation

Tabelle A.1: `STAT_PRE_CHECK`-Event.

A.2 STAT_PRE_REPORT-Event

Verwendung	von SPAMATO vor einer *Report*-Operation	
abgebildet auf	`StatReport`	
Vorbedingung	keine	
Argumente	`mail`	E-Mail
	`username`	Benutzername
	`client_id`	Client-Identifikation
	`auto_report`	Gibt an, ob es sich um einen *Auto-Report* handelt

Tabelle A.2: `STAT_PRE_REPORT`-Event.

A.3 STAT_PRE_REVOKE-Event

Verwendung	von Spamato vor einer *Revoke*-Operation	
abgebildet auf	`StatReport`	
Vorbedingung	keine	
Argumente	`mail`	E-Mail
	`username`	Benutzername
	`client_id`	Client-Identifikation

Tabelle A.3: `STAT_PRE_REVOKE`-Event.

A.4 STAT_CHECK-Event

Verwendung	von jedem Filter nach Bestimmung des Ergebnisses der *Check*-Operation	
abgebildet auf	`CheckResult`	
Vorbedingung	`STAT_PRE_CHECK`-Event erhalten	
Argumente	`mail`	E-Mail
	`filter_type`	Filtertyp
	`filter_success`	Gibt an, ob Operation erfolgreich war oder nicht
	`filter_result`	Ergebnis der Operation
	`[errorString]`	Fehlerbeschreibung
	{`fingerprint_type`,	Typ eines *Fingerprints*
	`fingerprint_value`,	Wert eines *Fingerprints*
	`fingerprint_result`}	Ergebnis eines *Fingerprints*
	{`filter_property_type`,	Typ einer Filtereigenschaft
	`filter_property_value`}	Wert einer Filtereigenschaft

Tabelle A.4: `STAT_CHECK`-Event.

A.5 STAT_REPORT-Event

Verwendung	von jedem Filter nach Bestimmung des Ergebnisses der *Report*-Operation	
abgebildet auf	`ReportResult`	
Vorbedingung	`STAT_PRE_REPORT`-Event erhalten	
Argumente	`mail`	E-Mail
	`filter_type`	Filtertyp
	`filter_success`	Gibt an, ob Operation erfolgreich war oder nicht
	`[errorString]`	Fehlerbeschreibung
	{`fingerprint_type`, `fingerprint_value`}	Typ eines *Fingerprints* Wert eines *Fingerprints*

Tabelle A.5: `STAT_REPORT`-Event.

A.6 STAT_REVOKE-Event

Verwendung	von jedem Filter nach Bestimmung des Ergebnisses der *Revoke*-Operation	
abgebildet auf	`RevokeResult`	
Vorbedingung	`STAT_PRE_REVOKE`-Event erhalten	
Argumente	`mail`	E-Mail
	`filter_type`	Filtertyp
	`filter_success`	Gibt an, ob Operation erfolgreich war oder nicht
	`[errorString]`	Fehlerbeschreibung
	{`fingerprint_type`, `fingerprint_value`}	Typ eines *Fingerprints* Wert eines *Fingerprints*

Tabelle A.6: `STAT_REVOKE`-Event.

A.7 STAT_POST_CHECK-Event

Verwendung	von SPAMATO nach *Check*-Operation	
abgebildet auf	`StatReport`	
Vorbedingung	von allen Filtern `STAT_CHECK`-Event erhalten	
Argumente	`mail`	E-Mail
	`username`	Benutzername
	`client_id`	Client-Identifikation
	`first_check`	Gibt an, ob es sich um einen *ersten* Check oder um einen *Re-Check* handelt
	`min_spam_answer_average`	Minimaler Anteil von Filtern, die eine E-Mail als Spam identifizieren müssen, damit sie als Spam identifiziert wird
	`spam`	Gibt an, ob die E-Mail als Spam identifiziert worden ist

Tabelle A.7: `STAT_POST_CHECK`-Event.

B SQL-Definitionen

Dieser Anhang beschreibt alle Tabellen der MySQL-Datenbank [mys04] des Statistik-Servers. Für ihre jeweilige Verwendung sei auf den englischen Kommentar innerhalb der SQL-Tabellendefinition hingewiesen.

Abbildung B.1 gibt eine grafische Übersicht der Tabellen-Struktur.

B.1 SQL-Tabellendefinition für `common_filters`

```
1  CREATE TABLE common_filters (
2    name varchar(32) NOT NULL default '',
3    filter_id tinyint(1) NOT NULL default '0',
4    PRIMARY KEY  (filter_id),
5    KEY name (name)
6  ) TYPE=MyISAM COMMENT='to map MANUALLY a filter_id to a
       common filter name';
```

B.2 SQL-Tabellendefinition für `filter_properties`

```
1  CREATE TABLE filter_properties (
2    check_id int(1) NOT NULL default '0',
3    user_id varchar(32) NOT NULL default '',
4    check_time timestamp(14) NOT NULL,
5    filter_id tinyint(1) NOT NULL default '0',
6    property_id varchar(128) NOT NULL default '',
7    property_value text,
8    PRIMARY KEY  (check_id,filter_id,property_id),
9    KEY filtertype (filter_id),
10   KEY property_type (property_id),
11   KEY user_id (user_id),
12   KEY user_filter_property_type (user_id,filter_id,
         property_id)
13 ) TYPE=MyISAM COMMENT='stores properties from _all_
       filters';
```

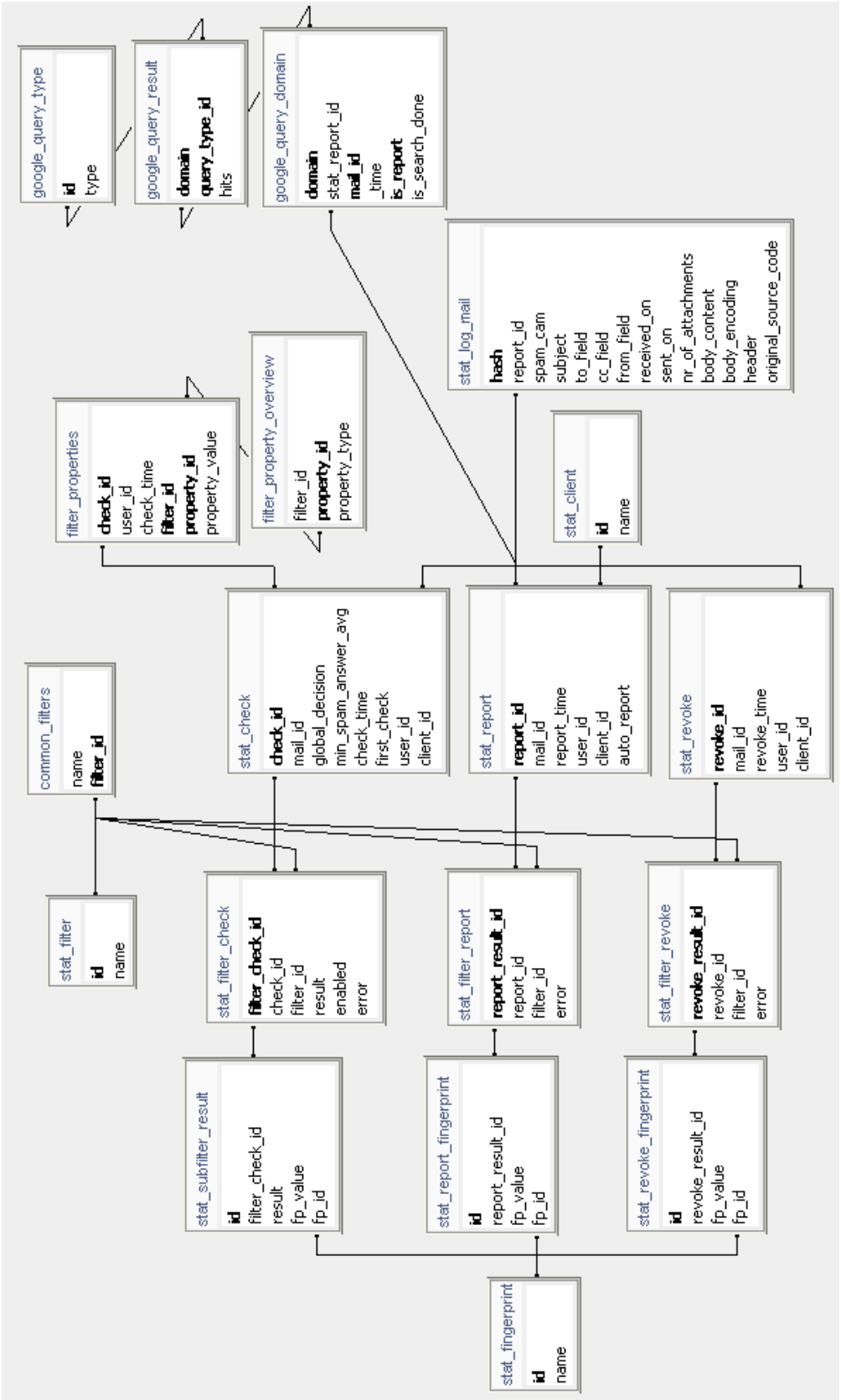


Abbildung B.1: Tabellen-Struktur der MySQL-Datenbank des Statistik-Servers.

B.3 SQL-Tabellendefinition für filter_property_overview

```
CREATE TABLE filter_property_overview (
  filter_id tinyint(1) NOT NULL default '0',
  property_id tinyint(1) unsigned NOT NULL
      auto_increment,
  property_type varchar(255) NOT NULL default '',
  PRIMARY KEY  (property_id),
  UNIQUE KEY filter_id_property_type (filter_id,
      property_type)
) TYPE=MyISAM COMMENT='a list of all possible filter
    property types';
```

B.4 SQL-Tabellendefinition für google_query_domain

```
CREATE TABLE google_query_domain (
  domain varchar(255) NOT NULL default '',
  stat_report_id int(1) unsigned NOT NULL default '0',
  mail_id varchar(32) NOT NULL default '',
  _time timestamp(14) NOT NULL,
  is_report tinyint(1) unsigned NOT NULL default '0',
  is_search_done tinyint(1) unsigned NOT NULL default '0
      ',
  PRIMARY KEY  (domain,mail_id,is_report),
  KEY domain (domain),
  KEY mail_id (mail_id),
  KEY is_report (is_report),
  KEY stat_report_id (stat_report_id)
) TYPE=MyISAM COMMENT='stores a search-set {site:domain
    , link:domain, ...} at google';
```

B.5 SQL-Tabellendefinition für google_query_result

```
CREATE TABLE google_query_result (
  domain varchar(255) NOT NULL default '',
  query_type_id tinyint(1) unsigned NOT NULL default '0'
      ,
  hits int(1) NOT NULL default '-1',
  PRIMARY KEY  (domain,query_type_id),
  KEY query_type_id (query_type_id),
  KEY domain (domain)
) TYPE=MyISAM COMMENT='stores the number of hits of a
    google_query using a domain';
```

B.6 SQL-Tabellendefinition für google_query_type

```
CREATE TABLE google_query_type (
  id tinyint(1) unsigned NOT NULL auto_increment,
  type varchar(255) NOT NULL default '',
  PRIMARY KEY  (id),
  UNIQUE KEY type (type)
) TYPE=MyISAM COMMENT='holds information about different
    query-types (patterns)';
```

B.7 SQL-Tabellendefinition für stat_check

```
CREATE TABLE stat_check (
  check_id int(1) unsigned NOT NULL auto_increment,
  mail_id varchar(255) default NULL,
  global_decision tinyint(1) NOT NULL default '-1',
  min_spam_answer_avg double default '-1',
  check_time timestamp(14) NOT NULL,
  first_check tinyint(1) unsigned default NULL,
  user_id varchar(32) NOT NULL default '',
  client_id tinyint(1) default NULL,
  PRIMARY KEY  (check_id),
  KEY user_id (user_id),
  KEY client_id (client_id),
  KEY first_check (first_check),
  KEY mail_id (mail_id)
) TYPE=MyISAM COMMENT='contains all checks of mails by
    all spamato-clients';
```

B.8 SQL-Tabellendefinition für stat_client

```
CREATE TABLE stat_client (
  id tinyint(1) unsigned NOT NULL auto_increment,
  name varchar(255) NOT NULL default '',
  PRIMARY KEY  (id),
  UNIQUE KEY name (name)
) TYPE=MyISAM COMMENT='contains ids for pamato-clients (
    mozilla, outlook, ...)';
```

B.9 SQL-Tabellendefinition für stat_filter

```
CREATE TABLE stat_filter (
  id tinyint(1) unsigned NOT NULL auto_increment,
  name varchar(255) NOT NULL default '',
  PRIMARY KEY  (id),
  UNIQUE KEY name (name)
) TYPE=MyISAM COMMENT='mapping of a filter name to a
    filter number to save storage space';
```

B.10 SQL-Tabellendefinition für stat_filter_check

```
1  CREATE TABLE stat_filter_check (
2    filter_check_id int(1) unsigned NOT NULL
       auto_increment,
3    check_id int(1) unsigned NOT NULL default '0',
4    filter_id tinyint(1) unsigned NOT NULL default '0',
5    result double NOT NULL default '-1',
6    enabled tinyint(1) unsigned NOT NULL default '0',
7    error varchar(255) default NULL,
8    PRIMARY KEY  (filter_check_id),
9    KEY check_id (check_id,filter_id),
10   KEY enabled (enabled),
11   KEY error (error)
12 ) TYPE=MyISAM COMMENT='stores check results for each
     filter';
```

B.11 SQL-Tabellendefinition für stat_filter_report

```
1  CREATE TABLE stat_filter_report (
2    report_result_id int(1) unsigned NOT NULL
       auto_increment,
3    report_id int(1) unsigned NOT NULL default '0',
4    filter_id tinyint(1) unsigned NOT NULL default '0',
5    error varchar(255) default NULL,
6    PRIMARY KEY  (report_result_id),
7    KEY report_id (report_id,filter_id),
8    KEY error (error)
9  ) TYPE=MyISAM COMMENT='stores report results for each
     filter';
```

B.12 SQL-Tabellendefinition für stat_filter_revoke

```
1  CREATE TABLE stat_filter_revoke (
2    revoke_result_id int(1) unsigned NOT NULL
       auto_increment,
3    revoke_id int(1) unsigned NOT NULL default '0',
4    filter_id tinyint(1) unsigned NOT NULL default '0',
5    error varchar(255) default NULL,
6    PRIMARY KEY  (revoke_result_id),
7    KEY revoke_id (revoke_id,filter_id),
8    KEY error (error)
9  ) TYPE=MyISAM COMMENT='stores revoke results for each
     filter';
```

B.13 SQL-Tabellendefinition für stat_fingerprint

```
1  CREATE TABLE stat_fingerprint (
2    id tinyint(1) unsigned NOT NULL auto_increment,
3    name varchar(255) NOT NULL default '',
4    PRIMARY KEY  (id),
5    UNIQUE KEY name (name)
6  ) TYPE=MyISAM COMMENT='mapping of a fp_name to a fp_id
       to save storage space';
```

B.14 SQL-Tabellendefinition für stat_log_mail

```
1  CREATE TABLE stat_log_mail (
2    hash varchar(32) NOT NULL default '',
3    report_id int(1) unsigned NOT NULL default '0',
4    spam_cam tinyint(1) unsigned NOT NULL default '1',
5    subject varchar(255) default NULL,
6    to_field varchar(255) default NULL,
7    cc_field varchar(255) default NULL,
8    from_field varchar(255) default NULL,
9    received_on timestamp(14) NOT NULL,
10   sent_on timestamp(14) NOT NULL default '00000000000000
         ',
11   nr_of_attachments tinyint(1) unsigned default '0',
12   body_content text,
13   body_encoding varchar(255) default NULL,
14   header text,
15   original_source_code text,
16   PRIMARY KEY  (hash),
17   UNIQUE KEY report_id (report_id),
18   KEY spam_cam (spam_cam)
19 ) TYPE=MyISAM COMMENT='all reported messages with all
       available data';
```

B.15 SQL-Tabellendefinition für stat_report

```
1  CREATE TABLE stat_report (
2    report_id int(1) unsigned NOT NULL auto_increment,
3    mail_id varchar(255) default NULL,
4    report_time timestamp(14) NOT NULL,
5    user_id varchar(32) NOT NULL default '',
6    client_id tinyint(1) default NULL,
7    auto_report tinyint(1) unsigned NOT NULL default '0',
8    PRIMARY KEY  (report_id),
9    KEY user_id (user_id),
10   KEY client_id (client_id),
11   KEY auto_report (auto_report),
12   KEY mail_id (mail_id)
13 ) TYPE=MyISAM COMMENT='contains all reports of mails by
       all spamato_clients';
```

B.16 SQL-Tabellendefinition für stat_report_fingerprint

```
1  CREATE TABLE stat_report_fingerprint (
2    id int(1) unsigned NOT NULL auto_increment,
3    report_result_id int(1) unsigned NOT NULL default '0',
4    fp_value varchar(255) NOT NULL default '',
5    fp_id tinyint(1) unsigned default '0',
6    PRIMARY KEY  (id),
7    KEY report_result_id (report_result_id)
8  ) TYPE=MyISAM COMMENT='stores fingerprints of report
       results';
```

B.17 SQL-Tabellendefinition für stat_revoke

```
1  CREATE TABLE stat_revoke (
2    revoke_id int(1) unsigned NOT NULL auto_increment,
3    mail_id varchar(255) default NULL,
4    revoke_time timestamp(14) NOT NULL,
5    user_id varchar(32) NOT NULL default '',
6    client_id tinyint(1) default NULL,
7    PRIMARY KEY  (revoke_id),
8    KEY user_id (user_id),
9    KEY client_id (client_id),
10   KEY mail_id (mail_id)
11 ) TYPE=MyISAM COMMENT='contains all revokes of mails by
       all spamato-clients';
```

B.18 SQL-Tabellendefinition für stat_revoke_fingerprint

```
1  CREATE TABLE stat_revoke_fingerprint (
2    id int(1) unsigned NOT NULL auto_increment,
3    revoke_result_id int(1) unsigned NOT NULL default '0',
4    fp_value varchar(255) NOT NULL default '',
5    fp_id tinyint(1) unsigned default '0',
6    PRIMARY KEY  (id),
7    KEY revoke_result_id (revoke_result_id)
8  ) TYPE=MyISAM COMMENT='stores fingerprints of revoke
       results';
```

B.19 SQL-Tabellendefinition für `stat_subfilter_result`

```
1 CREATE TABLE stat_subfilter_result (
2   id int(1) unsigned NOT NULL auto_increment,
3   filter_check_id int(1) unsigned NOT NULL default '0',
4   result double NOT NULL default '0',
5   fp_value varchar(255) default NULL,
6   fp_id tinyint(1) unsigned default '0',
7   PRIMARY KEY  (id),
8   KEY filter_check_id (filter_check_id)
9 ) TYPE=MyISAM COMMENT='stores results/fingerprints of
    subfilters';
```

Literaturverzeichnis

[Adi04] Aditus Consulting. Jpgraph - oo graph library for php. http://www.aditus.nu/jpgraph/, 2004.

[Bur04] Nicolas Burri. Spamato: A collaborative spam filter system. Diploma thesis, Swiss Federal Institute of Technology Zurich, 2004.

[Mei05] Remo Meier. Ein PlugIn-System für SPAMATO. Semesterarbeit, Eidgenössische Technische Hochschule Zürich, 2005.

[mys04] MySQL: The World's Most Popular Open Source Database. http://www.mysql.com/, 2004.

[Sch04] Simon Schlachter. Spamato spam cam. http://spamato.ethz.ch/index.php?option=com_newsfeeds&task=view&feedid=19&Itemid=47, 2004. View spam mails reported by SPAMATO users.

[Spa04] The apache spamassassin project. http://www.spamassassin.org/, 2004. SpamAssassin is an extensible email filter which is used to identify spam.