Diplomarbeit

zum Thema

Ein Internetdienst zur Vermeidung von unerwünschter Reklamepost

Frank Herrmann

Technische Universität Dresden
Fakultät Informatik
Institut für Systemarchitektur
Professur Betriebssysteme

14.07.2003

Verantwortlicher Hochschullehrer:
Prof. Dr. Hermann Härtig

Betreuer:
Dr.-Ing. Andreas Westfeld

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbstständig und unter ausschließlicher Verwendung der angegebenen Literatur, Verweise und Hilfsmittel erstellt zu haben. Verwendete Quellen wurden als solche gekennzeichnet.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

.....
Dresden, 14.07.2003

“So eine Arbeit wird eigentlich nie fertig,
man muß sie für fertig erklären,
wenn man nach Zeit und Umständen
das Mögliche getan hat.”

Johann Wolfgang von Goethe (Italienische Reise, 1787)

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Abschnitt 1

Einleitung

1.1 Einführung

Das Kommunikationsmittel E-Mail hat sich weit verbreitet. Es ermöglicht den schnellen Austausch von Informationen aller Art. Zu welcher Zeit unerwünschte E-Mails das erste Mal über das Internet transportiert wurden, vermag heute wohl niemand mehr zu sagen. Unerwünschte E-Mail – im Internetjargon kurz als SPAM (Akronym für SPiced HAM [1]) oder offiziell als *unsolicited commercial email* (UCE, unerwünschte Werbemail) bezeichnet – erregte 1994 weltweites Aufsehen. Es ist der Fall des Rechtsanwalt-Ehepaars Laurence Canter und Martha Siegel aus Phoenix, Arizona [1]. Das Paar verschickte an sämtliche der zu dieser Zeit existierenden Diskussionsforen (also etwa 8000 internationale sowie lokal verteilte Foren) eine Nachricht, die ein auf den ersten Blick kostenloses Angebot für die rechtliche Unterstützung bei der Teilnahme an einer staatlichen Green-Card-Verlosung enthielt. Dafür benutzten sie ein eigens für diesen Zweck bei einem UNIX-Hacker in Auftrag gegebenes Shell-Skript. Die Nachricht wurde dabei eklatanterweise nicht mit Hilfe des Crossposting-Mechanismus[2] verbreitet – stattdessen wurde sie an jedes Forum einzeln verschickt. Dadurch vervielfachte sich die erzeugte Datenmenge enorm, in diesem Fall um den Faktor 8000. Die Nachricht konnte nicht mehr entfernt werden.

Was einmal im Usenet begann, setzt sich heute in zunehmendem Maße bei der Kommunikation mit Hilfe von E-Mail fort. In einem Bericht des Marktforschungsinstituts *International Data Corporation* (IDC) wird davon ausgegangen, dass im Jahre 2006 von den schätzungsweise 9,2 Billionen E-Mails, die über das Internet transportiert werden, fast die Hälfte UCE beinhaltet [2].

1.2 Auswirkungen auf das Medium Internet

Die Auswirkungen von Spam auf das Internet teilen sich in 2 Kategorien. Zum einen wirkt sich UCE auf die technische Seite des Internets aus, also auf die Zugangsanbieter. Bei diesen entstehen eine Vielzahl von Kosten. Datenvolumen, benötigte Bandbreite, Rechenkapazität und Speicherkosten steigen an. Hinzu kommen Kosten für die Prävention gegen UCE, beispielsweise durch aufwändig installierte Filter.

[1] engl. für Würzfleisch

[2] Crossposting ermöglicht, nur den Titel der Nachricht in der entsprechenden Gruppe zu veröffentlichen. Die eigentliche Nachricht wird jedoch nur einmal übertragen.

	1996	1999	2002	2006
geschäftlich	130 Mrd.	920 Mrd.	3.330 Mrd.	5.580 Mrd.
privat	100 Mrd.	660 Mrd.	2.150 Mrd.	3.570 Mrd.
gesamt	230 Mrd.	1.580 Mrd.	5.480 Mrd.	9.150 Mrd.
davon UCE	50 Mrd.	290 Mrd.	1.500 Mrd.	2.920 Mrd.

Tabelle 1.1: IDC E-Mail Usage Forecast 2002-2006

Wenn es einem UCE-Versender gelingt, einen Zugangsanbieter als Weiterleitung für einen Massenversand von E-Mail zu missbrauchen, so wird einerseits dessen Ruf stark geschädigt und andererseits das E-Mailsystem durch zurückkommende Nachrichten wegen nicht existenten Empfängern oder zahlreichen Beschwerden belastet.

Zum zweiten hat UCE aber auch einen Einfluss auf die gesellschaftliche Seite des Internets, also auf die Netzgemeinschaft. Der vom Benutzer primär erkennbare Einfluss von UCE ist ein von nutzlosen E-Mails überfüllter elektronischer Briefkasten. Nutzlose E-Mails müssen vom Benutzer zunächst als solche identifiziert und danach manuell von legitimen E-Mails getrennt werden. Dadurch entsteht bei einer höheren Anzahl von Nachrichten eine wesentliche Belastung auf der Empfängerseite. Im weiteren zwingt UCE dem Benutzer ein vergrößertes Datenvolumen auf und die Verbindungszeit für die Übertragung der E-Mails erhöht sich. Zusätzlich wird Speicherplatz benötigt. Mobile Benutzer, welche oft nur über eine schmalbandige und volumenbegrenzte Verbindung verfügen, erfahren diese Auswirkungen am deutlichsten. Eine weitere Auswirkung stellt die Verletzung der Privatsphäre des Benutzers dar. Es ist für den Empfänger selten nachvollziehbar, woher ein UCE-Versender die E-Mailadresse bekommen hat und wieso er gerade an den konkreten Benutzer anonym bestimmte Informationen verschickt.

Benutzer haben in der Vergangenheit zahlreiche eigene Strategien entwickelt, um die Flut von Nachrichten in ihren elektronischen Briefkästen zu vermeiden. So werden beispielsweise ständig neue E-Mailzugänge bei Zugangsanbietern beantragt, sobald einer E-Mailadresse zu viel UCE zugestellt wird. Andere Benutzer legen sich mehrere E-Mailadressen an und teilen die "private Adresse" nur engsten Bekannten mit. Das dieses Vorgehen schon bei einer einfachen E-Mail des Bekannten mit *carbon copy* (CC, Mehrfachadressierung) zum Scheitern verurteilt ist, wissen nur die wenigsten.

Es existieren derzeit bereits eine Vielzahl von Lösungen gegen Spam. Die meisten E-Mail-Anbieter stellen zwar webbasiert unterschiedliche Methoden für den Benutzer zur Verfügung, diese sind jedoch – abhängig vom jeweiligen Anbieter – festgelegt und können nicht oder nur unzureichend an die Belange des Benutzers angepasst werden. Für den Benutzer, der klientbasierte Antispam-Verfahren einsetzt, gestaltet sich die Handhabung der Problematik um ein Vielfaches schwieriger. Abhängigkeiten vom verwendeten E-Mail-Programm sowie lediglich Schutz bei expliziter Verwendung dieses Programmes erschweren die Benutzung des Kommunikationsmediums E-Mail.

In dieser Arbeit wird ein Internetdienst vorgestellt, der eine gemeinsame Integration mehrerer Schutzkonzepte in ein webbasiertes Weiterleitungssystem erlaubt. Zusätzlich wird eine Kombination von mehreren Schutzkonzepten ermöglicht. Ergebnis der Arbeit ist ein Prototyp, der zwei Methoden zum Schutz

gegen Spam beinhaltet. Zum einen die von mir innerhalb des Großen Beleges untersuchte capability-basierte Methode und zum zweiten eine einfache Filtermethode.

1.3 Gliederung

Im folgenden Abschnitt der Arbeit werden die Gründe für den massenhaften Versand von E-Mail untersucht und gesellschaftliche, rechtliche und technische Lösungen gegen Spam vorgestellt. Im dritten Abschnitt wird der Entwurf des Internetdienstes vorgestellt und Überlegungen hinsichtlich der Kombinationsmöglichkeiten der im Abschnitt "Grundlagen" vorgestellten Methoden dargelegt. Abschnitt 4 behandelt ausschnittsweise Implementationsdetails und die programmtechnische Umsetzung einiger Aspekte des Entwurfes. In Abschnitt 5 wird der Ablauf eines typischen Szenarios vorgestellt und es erfolgt eine Bewertung der praktischen Realisierung und eine Einschätzung der erreichten Ergebnisse. Den Abschluss der Arbeit bildet Abschnitt 6. Dieser beinhaltet eine Zusammenfassung und mögliche Ansatzpunkte für weiterführende Arbeiten werden aufgezeigt.

Nachfolgend seien einige Hinweise zu den Typographie-Konventionen in dieser Arbeit gegeben:

Fachwörter sind beim ersten Auftreten im Text *kursiv* gesetzt. In Klammern folgen Abkürzung und deutsche Erklärung. Ein Beispiel ist *application programming interface* (API, Programmierschnittstelle). Programmanweisungen und Programmausschnitte sind in Schreibmaschinenschrift gedruckt, beipielsweise die Methode `getSubscriberState(String recipientEmailAddress)`. Zustände, Parameter, Argumente oder Konstanten sind mit KAPITÄLCHEN hervorgehoben.

1.4 Danksagung

An erster Stelle gilt mein Dank Prof. Dr. Hermann Härtig, welcher mir die Möglichkeit eröffnete, an einem für die Betriebssystemgruppe eher ungewöhnlichen Thema zu arbeiten. Weiterhin danke ich meinem Betreuer Dr. Ing. Andreas Westfeld für die Unterstützung, die er mir gewährte. Meinen Eltern möchte ich an dieser Stelle dafür danken, dass sie mir dieses Studium ermöglichten. Dank an die geduldigen Korrekturleser der Betriebssystemgruppe (Michael Hohmuth, Jean Wolter) für die konstruktiven Hinweise sowie an Adam Lackorzynski für die technische Unterstützung. Zum Abschluss möchte ich noch allen weiteren Personen meinen Dank ausdrücken, welche zum Gelingen dieser Arbeit beigetragen haben, – meiner Freundin Anne Corsing für die Erstkorrektur und Steffen Trümper und Gerhard Walter für den "Software-Beta-Test".

Abschnitt 2

Grundlagen

2.1 Gründe für den Versand von unerwünschter E-Mail

Die Absichten eines UCE-Versenders sind kommerzieller oder störungsgerichteter Natur. Der Versender möchte ein Produkt anpreisen, den normalen Betrieb im Internet stören oder mit einer Information überschwemmen. Dieser Abschnitt legt dar, warum der Versand von E-Mail über das Internet profitabel ist.

2.1.1 Spamkategorien

Anzahl und Häufigkeit unerwünschter E-Mails werden von dem Spamfilter-Hersteller und Dienstleister *Brightmail* [19] monatlich in einer Statistik [20] veröffentlicht. Aus Tabelle 2.1 ist ersichtlich, dass Spam sich in fast jedem Lebensbereich ausgebreitet hat. Am häufigsten vertreten sind Werbungen für Produkte, Finanzdienstleistungen und Erotikangebote.

2.1.2 Adressquellen der Spamversender

Ein Versender von Massenmail benötigt in erster Linie eine große Anzahl von E-Mailadressen. Die Quellen dafür sind vielfältig. Die wichtigsten sollen im folgenden benannt werden.

Die globale Hierarchie der Diskussionsforen, das Usenet, ist eine der wichtigsten Quellen. Suchagenten durchforsten alle Nachrichten nach verwertbaren E-Mailadressen. Dabei ist sowohl die Adresse selbst, als auch das konkrete Diskussionsforum (zur Ermittlung des Interessensgebietes) interessant. Archivsammlungen von Mailinglisten sind ebenfalls eine Quelle zahlreicher Adressen, des weiteren Gästebücher von Homepages sowie öffentlich zugreifbare Adressverzeichnisse (“Gelbe Seiten”). Eine E-Mailadresse ist ebenfalls Bestandteil der Registrierinformationen bei dem *network information center* (NIC, Registrierungsstelle), die automatisiert abgefragt werden kann. Beispiele für solche Suchwerkzeuge sind *Atomic Harvester 2000*, ein reines Suchwerkzeug und *Target 98*. Letzteres integriert Suchmaschine und Massenmailer in einem Werkzeug. Die Software ist in der Lage, stichwortabhängige E-Mailadressen zu finden. Bei all den bisher benannten Methoden lassen sich die benötigten Informationen vollautomatisch abfragen bzw. ausfiltern. Ein anderer, manueller Weg viele E-Mailadressen zu erhalten, besteht in dem Versenden von Kettenbriefen mit für den Empfänger interessant erscheinendem

Kategorie	Beschreibung	Beispiel	Anteil
Produkt-werbung	Waren, Dienstleistungen	"Get your Free satellite TV system now"	19 %
Erotik	Kontaktvermittlung, Pornographie	"Watch Me Free On Webcam."	19 %
Finanzdienst-leistungen	Kredite, Geldanlagen	"Get Cash, No Equity Required! Apply Today!"	17 %
Betrug	Kettenbriefe, Pyramiden-Systeme	"Urgent Business Letter"	10 %
Internet	Webhosting, Webdesign Spamware-Angebote	"High Speed Internet Access is Here!"	10 %
Gesundheit	Medikamente, Potenzmittel	"Enlarge your penis!"	9 %
Freizeit	Urlaub, Casino, Spiele	"Collect $150 Free Chips for Onlinegaming!"	9 %
geistlich	Astrologie, Religion, Spirituelles	"The truth About Your Power"	3 %
sonstige			4 %

Tabelle 2.1: Statistik unerwünschter Email von Brightmail, Stand: März 2003

Inhalt. Es könnte beispielsweise das Versprechen gemacht werden, dass der Sender für jede weitergeleitete E-Mail eine gewisse Geldsumme bekommt – unter der Bedingung, dass zum Nachweis dafür eine Kopie aller E-Mails an den Versender zurückgeschickt wird.

Diese Beispiele zeigen, wie leicht es ist, eine große Anzahl von E-Mailadressen aus dem Internet zu extrahieren.

2.1.3 Versand

Für den Versand bedienen sich viele UCE-Versender Mailserver unbeteiligter Dritter. Voraussetzung dafür ist, dass diese ihren Mailtransport für *mail relaying* (Weiterleiten jeglicher E-Mail) konfiguriert haben. Beim Versand werden spezielle Versandwerkzeuge eingesetzt. Der populärste Massenmailer ist der *Stealth MassMailer*. Er ist in der Lage mehr als 200.000 Nachrichten pro Stunde per *mail relaying* über ein 28.8K Modem zu senden. Bei Nutzung des eigenen Rechners für den Versand sind es bei gleicher Konfiguration immer noch mehr als 5.000 Nachrichten pro Stunde [34]. Zusätzliche eingebaute Eigenschaften wie zufällige Generierung der Versenderadresse, Statusverfolgung und Protokollierung machen das Versenden zum Kinderspiel.

2.1.4 Kostenbetrachtung

Die anfallenden Kosten für die Übertragung einer Nachricht sind zusammengesetzt aus den Kosten für die Adressenbeschaffung, dem Internetzugang sowie Kosten für den eigentlichen Versand. Ein gängiges Suchwerkzeug kostet etwa 150 Dollar [11]. Ein temporärer Internetzugang ist leicht und kostengünstig bei einem *internet service provider* (ISP, Zugangsanbieter) zu erhalten. Weitere Kosten entstehen für das Versandwerkzeug, welches etwa 300 Dollar kostet [11]. Bei einer Erfolgsquote von nur 0,2 bis 2 Prozent, die bei postalischen Versendungen erreicht wird, kann der Versender mit beispielsweise 100.000 versendeten E-Mails bei 200 bis 2000 Empfängern Erfolg haben. Laut einer EU-Studie im Jahr 2001 liegt die Erfolgschance [1] des "Spam-Direktmarketing" mit 5 bis 15 Prozent weit höher als vergleichbare konventionelle postalische Mailings [12]. Wenn ein Spammer also 100.000 E-Mails versendet, so hat er im Durchschnitt bei 5.000 bis 15.000 Empfängern mit seiner E-Mail Erfolg. In beiden Fällen ist das Ergebnis verglichen mit einer Aktion über das Briefsystem, das eine ähnliche Resonanz erzeugen soll, profitabel.

2.2 Tarnung gegen Spam

Die zugrundeliegende, einfache Idee ist, die Versenderadresse so zu verfälschen, dass diese Fälschung für einen Rechner möglichst nicht erkennbar, jedoch für einen Menschen möglichst leicht erkennbar ist. Ein einfaches Beispiel:

```
adresse.REMOVETHIS@domain.org
```

In diesem Beispiel lautet die echte Adresse `adresse@domain.org`. Der Empfänger muss anhand des Hinweises REMOVETHIS erkennen, dass er die Zeichenkette .REMOVETHIS vor Beantwortung der Nachricht manuell entfernen muss. Da die benutzten Suchwerkzeuge der UCE-Versender solche Adressen inzwischen erkennen und "intelligent" modifizieren können, sind einige Benutzer obiger

[1] d.h. der Anteil an tatsächlichen Verkäufen

Methode auf die Idee gekommen, Adressen solcher Form als legitime Adressen zu verwenden. Das Kommunikations- und Adresschaos ist somit vorprogrammiert. Benutzer, welche in solche Prozeduren nicht eingeweiht sind, werden von der Kommunikation gänzlich ausgeschlossen.

Hierzu ein komplexes Beispiel, welches mit Zeichenersetzung arbeitet:

```
#/bin/bash
#meine E-Mailadresse lautet
echo zcqdrrd@cnlzhm.nqf | tr a-z b-za
```

Um diese E-Mailadresse zu extrahieren, wird eine *bourne-again shell* (BASH, Kommandozeileninterpeter) sowie das Werkzeug *translate* (tr, Zeichenersetzungswerkzeug) benötigt. Wird obiger Zweizeiler unter dieser Umgebung ausgeführt, ensteht die Ausgabe:

```
adresse@domain.org
```

Es ist offensichtlich, dass selbst Eingeweihten mit solchen Konstruktionen die Kommunikation durch manuelle Bearbeitung oder Gewinnung der Adresse wesentlich erschwert wird.

2.3 Rechtliche Schritte gegen Spam

Trotz des Fehlens einer zentralen Kontrolle über das Internet sind in der Vergangenheit viele Versuche unternommen worden, das Verschicken von Spam unter Strafe zu stellen. In Deutschland ist das unaufgeforderte Versenden von kommerziellen Angeboten nach herrschender juristischer Meinung gegenüber Unternehmen, Gewerbetreibenden und Freiberuflern ein rechtswidriger Eingriff in den Gewerbebetrieb. Bei Privatpersonen liegt eine Verletzung des allgemeinen Persönlichkeitsrechtes vor. Wenn keine ausdrückliche Erlaubnis seitens des Empfängers vorliegt, ergeben sich für Betroffene Unterlassungsansprüche nach Paragraph 1004 BGB (Beseitigungs- und Unterlassungsanspruch bei Ansprüchen aus Eigentum) und Paragraph 823 I BGB (Schadenersatzpflicht bei unerlaubten Handlungen).

Der hierfür relevante Teil des Paragraphen 101 des Telekommunikationsgesetzes (TKG) lautet: *"Anrufe - einschließlich das Senden von Fernkopien - zu Werbezwecken ohne vorherige Einwilligung des Teilnehmers sind unzulässig. Der Einwilligung des Teilnehmers steht die Einwilligung einer Person, die vom Teilnehmer zur Benützung seines Anschlusses ermächtigt wurde, gleich. Die erteilte Einwilligung kann jederzeit widerrufen werden; der Widerruf der Einwilligung hat auf ein Vertragsverhältnis mit dem Adressaten der Einwilligung keinen Einfluss. Die Zusendung einer elektronischen Post als Massensendung oder zu Werbezwecken bedarf der vorherigen - jederzeit widerruflichen - Zustimmung des Empfängers."*

Der Empfänger kann eine Unterlassungsklage erwirken, wenn er nachweist, dass er die E-Mail erhalten hat und das Wiederholungsgefahr besteht. Die Beweislast für das gegebene Einverständnis der Zusendung trägt in jedem Fall der Versender. Seit Juli 2002 gilt eine, verglichen mit deutschem Recht, abgeschwächte Regelung auch innerhalb der EU. Die Durchsetzbarkeit ist aufgrund der globalen Struktur des Internets fraglich – nationale Grenzen können mit Leichtigkeit überwunden werden. Und selbst wenn der Versender ausfindig gemacht werden konnte, so muss der Rechtsweg erfolgreich beschritten werden. Ein solcher ist, wenn überhaupt, nur sehr schwierig zu führen.

Das kalifornische Unternehmen Habeas [13] gewährleistet Schutz durch ein urheberrechtlich geschütztes Gedicht (“Haiku”). Dieses Gedicht wird inklusive eines Hinweises auf das Urheberrecht dem Kopf jeder E-Mail beigefügt. Habeas selbst tritt bei Verwendung des Gedichtes in Spamnachrichten als Kläger mit dem notwendigen finanziellen Hintergrund auf. E-Mails, welche das Gedicht beinhalten, erhalten beim Empfänger eine hohe Priorität. Das Unternehmen verlangt von geschäftlichen Nutzern eine Gebühr für das Einfügen.

2.4 Technische Lösungen gegen Spam

Der folgende Abschnitt gibt eine Übersicht über die Vielzahl der vorhandenen Lösungen. Methoden gegen Spam wurden in 3 Gruppen klassifiziert: Filterung, Adressvalidierung/Adressgenerierung und Protokollmodifikation. Zu der Gruppe Filterlösungen gehören Inhaltsfilter, regelbasierter Filter, Bayes-Filter sowie verteilter Prüfsummenfilter. Die Gruppe Adressvalidierung teilt sich in Blacklist, Whitelist, Channels und Wegwerfadresse auf. Die dritte Gruppe Protokollmodifikationen besteht aus dem Teergruben-Verfahren, SMTP nach POP sowie SMTP-AUTH.

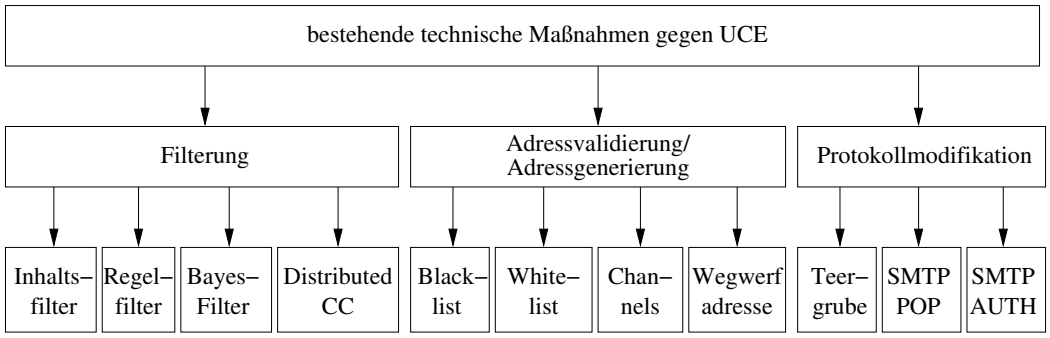

Abbildung 2.1: Übersicht der bestehenden technischen Lösungen gegen UCE

2.4.1 Inhaltsfilter

Inhaltsfilter untersuchen automatisch den Inhalt einer E-Mail nach nicht akzeptablen Wörtern und filtern Nachrichten mit solchem Inhalt aus. Man unterscheidet halbautomatische und vollautomatische Filter. Erstere sortieren erkannte Nachrichten in einen als UCE gekennzeichneten Bereich, während letztere die als Spam erkannten Nachrichten sofort löschen oder einen *bounce* (engl., prellen) auslösen. Unter einem *bounce* versteht man in diesem Zusammenhang das Zurücksenden der Nachricht, sodass beim ursprünglichen Sender der Eindruck entsteht, der Empfänger existiert nicht.

Reine Inhaltsfilter sind in dieser Form selten anzutreffen. Sie kommen beispielsweise in Unternehmen zum Einsatz, welche ein bestimmtes Wortrepertoire als inakzeptabel betrachten und daher sämtliche Nachrichten, die eines oder mehrere solcher Wörter enthalten konsequent ausfiltern.

2.4.2 Regelbasierender Filter

Ein regelbasierender Filter erkennt UCE anhand von Regeln, die meist mit Hilfe von regulären Ausdrücken formuliert werden. Diese sollen die Charakteristik von Spam möglichst genau wiedergeben. Dabei werden *Header* (Kopf, enthält u.a. Transportinformationen wie Sender, Empfänger und Quellrechner), Betreff, Nachrichtenkörper, Inhaltstyp sowie eventuelle Anhänge der E-Mail untersucht.

Beispiel eines Ausschnittes aus dem Regelsatz für *sendmail* (sendmail MTA, Mailtransportsystem):

```
###---------->HTML<---------
:0
*^Content-Type:.*multipart/alternat*
spam

###---------->Subject<---------
:0
*^Subject:.make.*money.*fast
spam
```

Die erste Regel filtert sämtliche E-Mails aus, die als Inhaltstyp *hypertext markup language* (HTML, Standard-Seitenbeschreibungssprache des Internets) besitzen. Die zweite Regel filtert Nachrichten aus, die im Betreff MAKE, MONEY und FAST enthalten. Nachrichten, die nach diesen Regeln positiv als Spam bewertet sind, werden sicherheitshalber in einem Ordner SPAM platziert.

Ein populäres Beispiel für einen regelbasierenden Filter ist *SpamAssassin* (Spam-Attentäter, [3]). Spam–Assassin untersucht Kopf und Körper der E-Mail auf

- Stichworte
- ausschließliche Großschreibung
- undruckbare Zeichen
- HTML-Inhalt
- Javascript-Code

Die erhaltenen Ergebnisse werden mit einer vorgegebenen Wichtung aufsummiert. Falls die Summe einen Schwellwert überschreitet, fügt die Software einen Hinweis auf Spam in die Betreffzeile ein. Dem Körper der Nachricht werden die Auswertungsdetails hinzugefügt, die zu der Entscheidung geführt haben. Zusätzliche Änderungen an der Nachricht, wie z.B. die Änderung des *multipurpose internet mail extension*-Typs (MIME, universelle E-Mail-Erweiterung) sollen zusätzlich der Ausführung von aktiven Inhalten oder der Ausspähung durch Webbugs[2] vorbeugen.

2.4.3 Bayes-Filter

Paul Graham beschreibt in "A Plan for Spam" [5] ein Verfahren, das mit Wahrscheinlichkeiten arbeitet. Dazu wird die gesamte E-Mail, d.h. Kopf, Körper und Rumpf zunächst nach Wörtern durchsucht und

[2] winzige – in der E-Mail transparent versteckte – Grafiken, die auf einen Webserver referenzieren und somit dem Versender das Lesen einer E-Mail bestätigen

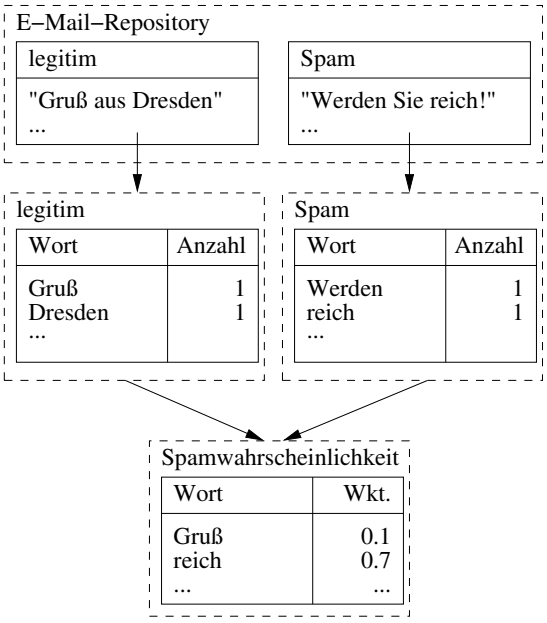

Abbildung 2.2: Speicherung der Wahrscheinlichkeiten bei dem Bayes-Verfahren

ihre Anzahl in Tabellen gespeichert. Die erste Tabelle speichert Wort und Anzahl für legitime Nachrichten, die zweite für UCE. Eine weitere enthält die berechnete Zuordnung zwischen den Wörtern und der Wahrscheinlichkeit ihres Auftretens in einer Spam-Nachricht. Unbekannte Wörter erhalten eine geringe UCE-Wahrscheinlichkeit. Beim Eintreffen einer Nachricht wird mit Hilfe des Satzes von Bayes die Wahrscheinlichkeit ihrer Zuordnung als Spam berechnet. Gegeben ist die Wahrscheinlichkeitsverteilung auf Ω. Es gilt weiterhin

$$\Omega = \bigcup_{i=1}^{n} B_i \text{ mit } B_i \cap B_j = \emptyset \text{ für } i \neq j.$$

Dann gilt für jedes j und jedes $A \subseteq \Omega$ mit $P(A) \neq 0$

$$P(B_j/A) = \frac{P(B_j)*P(A/B_j)}{\sum_{i=1}^{n} P(B_i)*P(A/B_i)}$$

Es sind also zu einem gegebenen Ereignis A die bedingten Wahrscheinlichkeiten $P(A/B_i)$ sowie die Wahrscheinlichkeiten $P(B_i)$ gegeben und die bedingte Wahrscheinlichkeit $P(B_j/A)$ gesucht. Anders ausgedrückt heisst das, man kann mit Hilfe von a-priori-Wahrscheinlichkeiten (B_j) und einem eintretenden Ereignis A die a-posteriori-Wahrscheinlichkeiten $P(B_j/A)$ berechnen. Der Filter "lernt" also durch Erfahrung dazu. Daher ist im Vorfeld ein Repository von legitimen sowie UCE-Nachrichten notwendig, damit der Bayes-Filter zuverlässig arbeitet. Der Filter passt sich je nach angelernten Nachrichten an die Benutzercharakteristik an.

Seit der Veröffentlichung von Paul Graham wurden zahlreiche Varianten von Bayes-Filtern implementiert. Beispiele sind *ifile* [15] und der E-Mail-Klient von *Mozilla* [16]. Der ifile-Filter wurde an mehrere

E-Mail-Klienten und sämtliche *procmail* [17] Klienten angepasst. Der Filter des Mozilla-Projektes ist im Gegensatz dazu direkt in den E-Mail-Klienten integriert.

2.4.4 Verteilter Prüfsummenfilter

Der *Distributed Checksum Clearinghouse* (DCC, verteilter Prüfsummenfilter) wird in [21] vorgestellt. Das Verfahren erzeugt für jede empfangene E-Mail eine Prüfsumme, welche an ein verteiltes Netzwerk von Rechnern weitergeleitet wird. Aus der Antwort der anderen Rechner wird anschließend ermittelt, wie oft diese Prüfsumme bereits registriert wurde und erhöht gleichzeitig den entsprechenden Zählerstand. Wird ein bestimmter Schwellwert überschritten und ist diese E-Mail nicht in einer Positivliste gemeldet, so wird die Nachricht als Spam klassifiziert. Streng genommen erkennt das Verfahren nicht nur Spam, sondern allgemein in hoher Anzahl versendete E-Mails. Spam ist häufig personalisiert, zum Beispiel könnte die Anrede in jeder Nachricht leicht verändert sein. Weiterhin versuchen viele UCE-Versender durch Einfügen zufälliger Zeichenketten einer Kategorisierung zu entgehen. Somit würde ein konventioneller Prüfsummenalgorithmus ein völlig anderes Ergebnis liefern, sobald sich lediglich ein Zeichen in der Nachricht ändert. Daher benutzt das Verfahren *fuzzy checksums* (unscharfe Prüfsummen), die für ähnliche E-Mails gleiche Prüfsummen erzeugen.

Die praktische Einbindung eines solchen Verfahrens erfolgt entweder als Server-Komponente (beispielsweise über *sendmail*) oder als Klient-Komponente (Quellkode erhältlich unter [21]).

2.4.5 Blacklists

Eine *blacklist* (Sperrliste), die lokal oder zentral im Internet geführt wird, beinhaltet alle Rechner, die Spam-Nachrichten versendet haben und dadurch an diese Liste gemeldet worden sind.

Ein praktisches Anwendungsbeispiel für die Methode ist die *realtime blackhole list* (RBL, in "Echtzeit" aktualisierte Sperrliste). Diese wird auf einem zentralen Server gespeichert und enthält alle Versender, die in der Vergangenheit als Ursprung von UCE erkannt wurden. Dabei können sowohl einzelne Versender als auch ganze Domains vom E-Mail-Verkehr ausgeschlossen werden. Klienten, die sich mit Hilfe von RBL schützen möchten, starten beim Empfang von E-Mail eine Anfrage an den RBL-Server, ob der Versender gelistet ist. Um die Anzahl der Anfragen möglichst gering zu halten, benutzt der Klient einen Zwischenspeicher für bereits stattgefundene Anfragen.

Eine Implementation dieses Verfahrens nutzt beispielsweise der Dienst *mail abuse prevention system* (MAPS, System zur Vorkehrung von E-Mail-Missbrauch) [18].

2.4.6 Whitelists

Bei dieser Methode wird eine Liste aller legitimen Versender geführt. Im einfachsten Fall kann die Eintragung der Versender durch den Empfänger selbst vorgenommen werden. Dazu wird jeder erwünschte Kontakt in diese Liste aufgenommen und alle anderen E-Mails werden vom Zustellsystem in einem niedrig priorisierten Postfach des Benutzers abgelegt. Dieses Verfahren ist eine vereinfachte Variante des im nächsten Abschnitt vorgestellten Verfahrens.

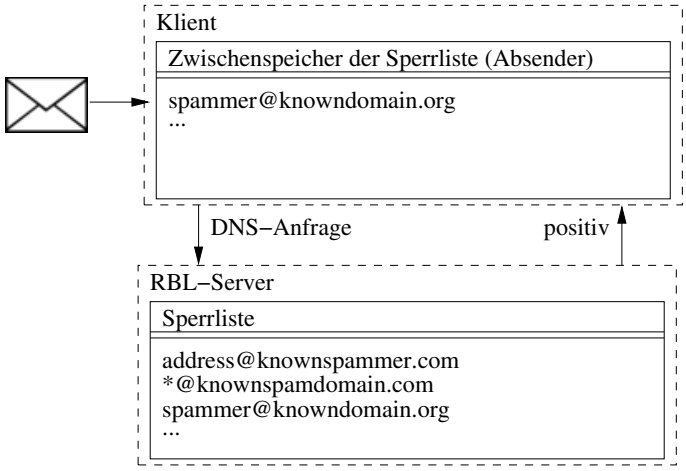

Abbildung 2.3: realtime blackhole list-Serveranfrage

2.4.7 Channels

Das *whitelist*-Verfahren kann man verbessern, indem die Einträge dynamisch über ein Versenderverifikationssystem vorgenommen werden. Dazu wird einem unbekannten Versender zunächst eine E-Mail als Bestätigung zugesendet. Wenn eine zweite E-Mail eintrifft – und der Versender sich somit als antwortfähig herausgestellt hat – wird die E-Mail zugestellt. Die Idee, ein System zu schaffen, in dem sich die Kommunikationsteilnehmer untereinander *Channels* (Kommunikationskanäle) zuweisen, wird von Robert Hall in [4] beschrieben. Jedem Kontakt wird dabei ein eigenständiger Kanal zugewiesen. Es existieren im weiteren ein oder mehrere öffentliche Kanäle, die für neue Kontakte benötigt werden und jederzeit zurückgezogen werden können. Ein *personal channel agent* (PCA, persönliche Kanalverwaltung) unterstützt dabei das Erstellen, Verfolgen und Entfernen von Kanälen. Das Kanalisieren ermöglicht es dem Benutzer, seine Kommunikation auf verschiedene Postfächer aufzuteilen, verursacht jedoch im Gegenzug einen relativ hohen Verwaltungsaufwand.

Weitere Beispiele für solche Systeme sind *tagged message delivery agent* (TMDA) [6] und das von mir innerhalb des Großen Beleges implementierte *capability*-basierte Verfahren *SpamStop* [7].

2.4.8 Wegwerfadressen

Wegwerfadressdienste ermöglichen eine Beschränkung der Gültigkeit einer E-Mailadresse. Begrenzende Faktoren können Zeit oder Anzahl sein. Solche Adressen sind auch nach Ablauf der Gültigkeit noch gültig im technischen Sinne, d.h. es können zwar noch E-Mails an diese Adresse gesendet werden, diese werden aber beim Empfänger automatisch verworfen und es erfolgt keine Zustellung an den Empfänger.

Ein Beispiel für einen Wegwerfadressdienst ist *SpamGourmet* [14]. Für die Benutzung dieses Diens-

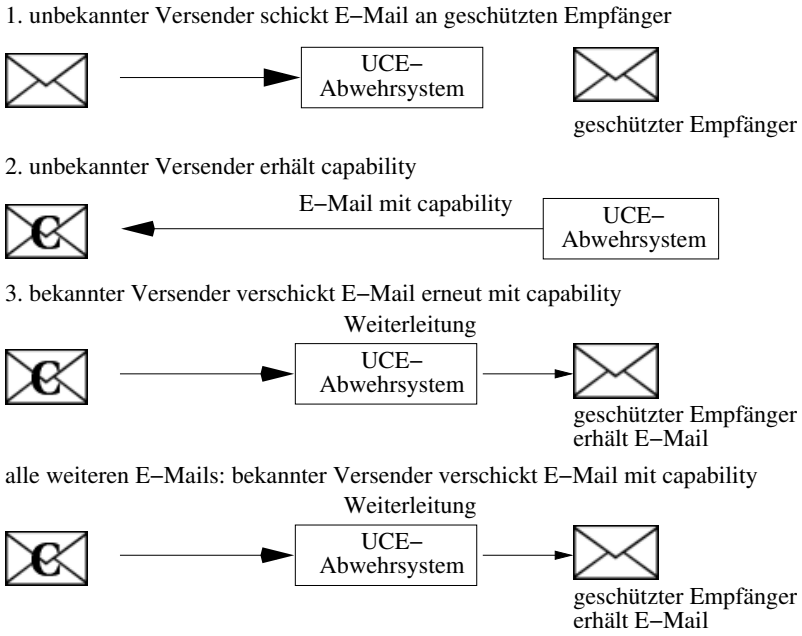

Abbildung 2.4: Kommunikationsablauf bei SpamStop

tes ist es zunächst notwenig, eine gültige E-Mailadresse anzugeben, an die Weiterleitung erfolgen soll. Danach können beliebig viele neue E-Mailadressen der Form

`wort.x.nutzer@spamgourmet.com`

angelegt werden. Dabei ist WORT ein beliebiges (jedoch zuvor noch nicht benutztes) Wort und X die Gültigkeitsanzahl, welche im Bereich von 1 bis 20 Nachrichten liegen kann. NUTZER ist der Benutzername, welcher bei der Registrierung für den Dienst festgelegt wurde.

2.4.9 Teergrube

In [9] wird das Teergruben-Verfahren beschrieben, welches direkt auf dem E-Mail-Transportprotokoll *simple mail transfer protocol* (SMTP, einfaches E-Mail-Versandprotokoll) [22] aufsetzt. Der Ablauf des Protokolls wird dabei so verändert, dass ein Port des Versandrechner im Idealfall über mehrere Stunden hinweg künstlich offen gehalten wird. Ein Beispiel für den normalen Ablauf des Protokolls soll im Folgenden vorgestellt werden. Es soll eine E-Mail von SENDER@SENDER.DE an EMPFAENGER@BEISPIEL.DE über den SMTP-Server SMTP.BEISPIEL.DE verschickt werden. Dabei entsteht folgender Ablauf zwischen dem Klient (K) und dem Server (S):

```
S: 220 smtp.beispiel.de ESMTP server ready
K: HELO sender.de
S: 250 smtp.beispiel.de Hello sender.de pleased to meet you
K: MAIL FROM: sender@sender.de
S: 250 sender@sender.de... Sender ok
K: RCPT To: empfaenger@beispiel.de
S: 250 empfaenger@beispiel.de... Recipient ok
```

```
K: DATA
S: 354 Enter mail, end with "." on a line by itself
K: Subject: Beispiel
K: Test
K: .
S: Message accepted for delivery
K: QUIT
S: 221 smtp.beispiel.de closing connection
```

Eine Teegrube verzögert den Ablauf des Protokolls durch Schicken von sogenannten Fortsetzungszeilen. Dies zwingt den Klienten zu einer durch den Server frei bestimmbaren Wartezeit, bevor die Nachricht übermittelt werden kann. Ein Bindestrich nach dem Identifikationscode am Beginn jeder vom Server übermittelten Zeile bedeutet, dass der Server noch nicht mit seiner Antwort fertig ist und der Klient warten soll. Falls der Klient versucht durch vorzeitiges Eingreifen diesen Prozess zu verkürzen, so wird das Protokoll mit einer Warnung abgebrochen und eine weitere Wartezeit eingefügt. Das folgende Beispiel zeigt eine Verletzung des Protokollablaufs.

```
S: 220-smtp.beispiel.de ESMTP server ready
S: 220-delay is required
K: HELO sender.de
S: 220-You violated the SMTP protocol!
S: 220-Teergrubing delay extended
```

Nach Ablauf der Wartezeit verläuft das Protokoll nach dem gewohnten Schema. Unter der Voraussetzung, dass sich ein solches System genügend weit verbeitet, wird ein Massenversand durch SMTP-konforme Absender verhindert. Meist erfolgt beim Teergruben-Verfahren eine Kombination mit dem RBL-Verfahren.

Ein Beispiel ist Jackpot [10].

2.4.10 Authentifizierung vor dem Versand

Das Standardprotokoll SMTP ermöglicht in der ursprünglichen Version keine Authentifizierung beispielsweise mit Benutzername und Passwort. Das Empfangsprotokoll *post office protocol 3* (POP3, Empfangsprotokoll Version 3) [23] jedoch fordert eine explizite Authentifizierung. Daher sind einige E-Mail-Anbieter dazu übergegangen, eine Authentifikation mit Hilfe von *SMTP after POP* (SMTP nach POP) durchzuführen. Die Idee ist, dass die Identifikation mit Hilfe von POP den Ursprung der Anfrage für eine bestimmte Zeit zwischenspeichert. SMTP-Anfragen werden generell nur von zwischengespeicherten Versendern angenommen.

2.4.11 Erweiterung des Versandprotokolls

Da das SMTP-Protokoll in seiner ursprünglichen Form keine Authentifizierung unterstützt, wurde dieser Mangel später als Option ergänzt (RFC 2554) [24]. Der Mailserver kann mehrere Authentifikationsmethoden anbieten, gebräuchlich ist *challenge-response authentication mechanism* (CRAM-MD5, Forderung-Antwort-Authentifikation Typ MD5) [25].

```
S: 220 smtp.beispiel.de ESMTP server ready
K: EHLO beispiel.de
S: 250-smtp.beispiel.de
S: 250 AUTH CRAM-MD5 DIGEST-MD5
K: AUTH CRAM-MD5
S: 334
PENCeUxFREJoU0NnbmhNWitOMjNGNndAZWx3b29kLmlubm9zb2Z0LmNvbT4=
K: ZnJlZCA5ZTk1YWVlMDljNDBhZjJiODRhMGMyYjNiYmFlNzg2ZQ==
S: 235 Authentication successful.
```

Bei diesem Ablauf zwischen dem Klient (K) und dem Server (S) wurde eine E-Mail über den SMTP-Server SMTP.BEISPIEL.DE verschickt. Der Server bot AUTH CRAM-MD5 und DIGEST-MD5 als mögliche Protokolle an. Der Klient benutzte ersteres und authentifizierte sich erfolgreich.

2.5 Kombinationssysteme in der Praxis

Während der Erstellung dieser Arbeit begannen einige Anbieter von E-Mail neue Kombinationssysteme in ihre Systeme einzubauen. Es seien hier der persönliche Internet-Assistent (PIA) von Arcor und der Spamschutz der GMX AG als Beispiele genannt. Das Kombinationssystem von GMX soll im folgenden näher vorgestellt werden.

Das System schützt den Anwender mit 7 einzeln aktivierbaren Modulen vor Spam. Diese werden in die Gruppen “Analyse-Tools”, “Sperrlisten” und “Sonstige” eingeteilt.

Erkannter Spam wird dabei in einem speziellen Ordner für Spam abgelegt und für maximal einen Monat in diesem zwischengespeichert. Um dem Verlust von Nachrichten wegen einer Falscherkennung vorzubeugen, erhält der Anwender regelmäßig einen Bericht über die einsortierten Nachrichten. Zusätzlich markiert das System das Modul, welches der Auslöser für die Erkennung war. In Tabelle 2.2 sind alle Module mit den Markierungskürzeln und dem benutzten Verfahren ersichtlich.

Der “Textmuster-Profiler” ist nicht in der kostenfreien Registrierung enthalten. Werden die E-Mails nicht über die Internetschnittstelle abgerufen sondern per Fernabfrage, so erhält der Anwender täglich als erste E-Mail einen Bericht zugeschickt, in dem alle als Spam erkannten E-Mails mit Versender, Betreff und Größe aufgeführt sind. Um fälschlicherweise als unerwünscht gekennzeichnete Nachrichten zu erhalten muss also das Internetangebot von GMX genutzt werden. Zusätzlich werden bei einem Überlaufen des Zwischenspeichers für unerwünschte Nachrichten überzählige E-Mails unwiderbringlich gelöscht.

Gruppe	Modulname	Kürzel	Verfahren
Analyse-Tools	Textmuster-Profiler	T	Bayes-Filter
Analyse-Tools	Briefkopf-Analyzer	H	SpamAssassin
Analyse-Tools	Spamserver-Blocker	S	Blacklist (Server)
Sperrlisten	Persönliche Blacklist	B	Blacklist (Adressen)
Sperrlisten	GMX Team AntiSpam-Liste	G	Blacklist (Adressen)
Sperrlisten	Globale AntiSpam-Liste	A	DCC
Sonstige	"Nur Freunde"-Option	F	Whitelist
Sonstige	Manuell verschoben	M	Bayes-Filter

Tabelle 2.2: Module des GMX-Spamschutzes

Abschnitt 3

Entwurf

3.1 Ziele

Für den nachfolgend vorgeschlagenen Entwurf eines Schutzsystems gegen Spam ergeben sich drei wesentliche Zielstellungen: Integration, Standardisierung und Offenheit.

Die Integration ermöglicht die Einbindung von mehreren Methoden gegen UCE innerhalb einer Plattform. Die erwünschte Funktionalität kann nicht von einem einzelnen System erreicht werden. Daher versuche ich, mehrere unterschiedliche Methoden gegen Spam miteinander zu kombinieren und so ihre Vorzüge zu vereinen.

Standardisierung ermöglicht eine gemeinsame, einheitliche Schnittstelle zum Benutzer. Es existieren bereits zahlreiche Methoden, welche auf unterschiedlichsten Standards und Werkzeugen basieren. Diese Methoden sollen nun unter einer gemeinsamen Schnittstelle angeboten werden.

Offenheit ermöglicht eine einfache Einbindung neuer Funktionalität. Bewährte Methoden waren in sich abgeschlossen. Nun können jederzeit neue Entwicklungen eingebunden werden.

3.2 Architektur

3.2.1 Bestehende Architekturvarianten

In Abbildung 3.1 sind die Architekturvarianten bisheriger Anti-Spam-Systeme dargestellt. Der E-Mail-Kommunikation liegt jeweils ein E-Mail-Dienst zugrunde, der bei Variante a entweder über ein klientseitig vorhandenes E-Mail-Programm oder direkt mit Hilfe eines Internetbrowsers angesprochen wird. Bei letzterem Zugriff liegt die Verantwortlichkeit für den Schutz gegen UCE vollständig beim Anbieter des E-Mail-Dienstes. Der Dienstanbieter kann eine oder mehrere Anti-Spam-Maßnahmen im Portfolio haben, die meist aus technischen Gründen für den Benutzer nicht konfigurierbar oder erweiterbar sind. Besitzt der benutzte E-Mail-Dienst, wie in Variante b, integrierte Anti-Spam-Maßnahmen und ermöglicht eine freie Konfiguration des Anti-Spam-Repertoires, so besteht dennoch eine Beschränkung auf die verfügbaren Anti-Spam-Maßnahmen des gewählten Dienstanbieters (Variante c). Wird der Dienst mit

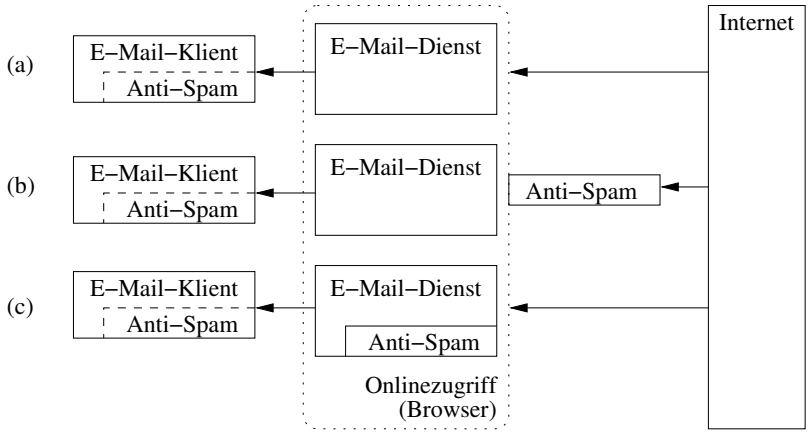

Abbildung 3.1: Architektur bestehender Anti-Spam-Lösungen

Hilfe eines E-Mail-Klienten angesprochen, so ergibt sich eine zusätzliche Möglichkeit, einen klientbasierten Schutz zu installieren. Bei letzterem ist der Schutz auf den entsprechenden Klienten beschränkt.

3.2.2 Vergleich von Klient- und Serverarchitektur

In diesem Abschnitt soll der klient- und der serverbasierte Ansatz noch einmal direkt miteinander verglichen werden. Generell ist es für das Funktionieren des Schutzes notwendig, dass die gesamte Kommunikation von einer zusätzlichen Zwischenschicht abgefangen wird.

Die Realisierung als Service direkt auf dem Klientrechner erfordert eine Installation einer eigenständigen Anwendung, welche dauerhaft als Hintergrundprozess ausgeführt wird. Der Klient sollte eine ständige Netzverbindung haben oder die E-Mailadressen müssen alle auf eine zentrale Adresse kanalisiert werden. Dies ist zwar technisch möglich, aber praktisch aufwändig oder oft gar nicht erreichbar. Hierzu zählen zum Beispiel private E-Mail-Anbieter, welche lediglich eine bzw. eine begrenzte Anzahl von Adressen vergeben. Weitere Anforderungen sind Betriebssystemunabhängigkeit sowie notwendige Installationen innerhalb jedes Betriebssystems bei mehreren Betriebssystemen auf ein- und demselben Rechner. Hinzu kommt, dass die benötigten Datenbankinformationen in diesem Fall eine gemeinsame Ressource nutzen müssen, um Konsistenz zu gewährleisten. Das gravierendste Problem besteht im fehlenden Schutz gegen Spam, wenn der E-Mail-Zugang direkt im Internet, also mit Hilfe eines Browsers, abgefragt wird.

Im Vergleich dazu benötigt der Betrieb als Internetdienst einen dedizierten Server, welcher den Dienst für eine bestimmte Anzahl von Benutzern anbietet. Es besteht für den Benutzer keine Notwendigkeit für die Installation von zusätzlicher Software auf dem Endrechner, da eine Nutzung des Dienstes mit jedem Betriebssystem unter Zuhilfenahme eines herkömmlichen Browsers möglich ist. Eine sich daraus ergebende Anforderung ist der Zugriffsschutz des öffentlichen Systems. Benutzer müssen sich registrieren und authentifizieren. Der Betrieb auf einem zentralen Serverrechner fordert zusätzliche Betrachtungen hinsichtlich Skalierbarkeit, da Benutzerdaten, E-Mails sowie Sitzungsdaten zentral auf dem Server aufbewahrt und verwaltet werden müssen.

3.2.3 Architektur des Internetdienstes

Aufgrund der überwiegenden Vorteile der serverbasierten Lösung lege ich den Dienst als Internetdienst aus. Dieser ist in Abbildung 3.2 dargestellt. Er fungiert sowohl als E-Mail-Dienst als auch als umfassen-

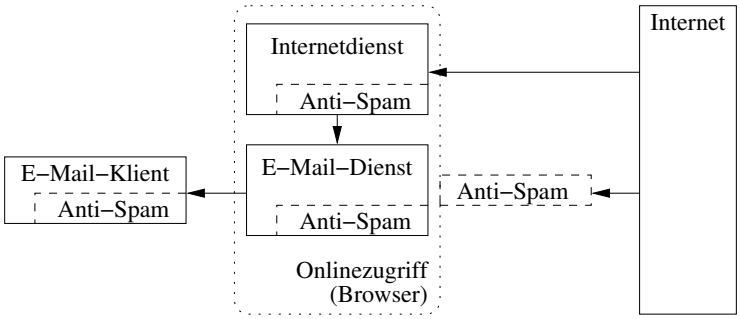

Abbildung 3.2: Architektur der Anti-Spam-Lösung mit Hilfe eines Internetdienstes

der Schutzwall gegen unerwünschte E-Mail. Legitime sowie als Spam erkannte Nachrichten werden an einen herkömmlichen E-Mail-Dienstanbieter weitergeleitet. Abbildung 3.3 verdeutlicht diese Weiterleitung, welche legitime sowie unerwünschte Nachrichten voneinander trennt.

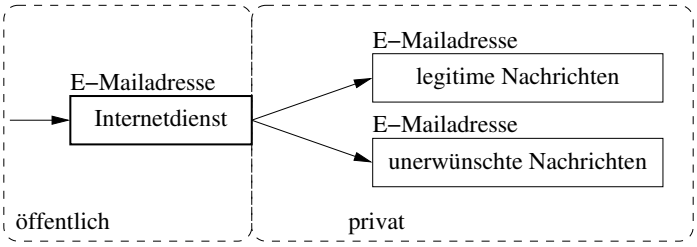

Abbildung 3.3: Auslegung als Weiterleitungssystem

3.3 Kombination der Anti-Spam-Maßnahmen

Im folgenden Abschnitt soll untersucht werden, ob eine Kombination von mehreren Anti-Spam-Maßnahmen möglich ist und welche Konsequenzen sich daraus ergeben.

3.3.1 Kombinationsmöglichkeiten

Um alle Kombinationsmöglichkeiten zu erfassen, soll zunächst das Kreuzprodukt über alle Maßnahmen gebildet werden, dies ist in Tabelle 3.1 dargestellt. Gültige Verknüpfungen sind durch ein Plus (+) und ungültige durch ein Minus (-) gekennzeichnet. Das entstandene Kreuzprodukt ist symmetrisch bezüglich der Hauptdiagonalen, zur Vereinfachung ist die Darstellung reduziert. Ein Verfahren mit sich selbst zu kombinieren, ist nicht sinnvoll, da durch ein erneutes Anwenden des gleichen Verfahren keine Vorteile zu erwarten sind. Daher ist die Hauptdiagonale der Tabelle als ungültig markiert. Das DCC-Verfahren

	Inhaltsfilter	Regelfilter	Bayesfilter	DCC	Blacklist	Whitelist	Channels	WW-Adresse	Teergrube	SMTP nach POP	SMTP-AUTH
Inhaltsfilter	-	+	+	+	+	+	+	-	+	+	+
Regelfilter		-	+	+	+	+	+	-	+	+	+
Bayesfilter			-	+	+	+	+	-	+	+	+
DCC				-	*	*	*	-	+	+	+
Blacklist					-	*	*	-	+	+	+
Whitelist						-	*	-	+	+	+
Channels							-	-	+	+	+
WW-Adresse								-	+	+	+
Teergrube									-	+	+
SMTP nach POP										-	-
SMTP-AUTH											-

Tabelle 3.1: Kombination der Anti-Spam-Verfahren

sowie das Channels-Verfahren sind Kombinationen aus den Verfahren Blacklist und Whitelist. Dieser Sachverhalt ist in der Tabelle durch einen Stern (*) kenntlich gemacht. Kombinationen mit den basierenden Verfahren sind möglich, es handelt sich dann um eine persönliche Black- bzw. Whitelist. Eine weitere Sonderstellung nimmt die Wegwerfadresse ein, da diese im allgemeinen jegliche Nachrichten empfangen soll und lediglich die Anzahl der empfangbaren E-Mails oder den Gültigkeitszeitraum begrenzt. Eine Kombination ist daher nicht sinnvoll. Protokollbasierte Maßnahmen wie Teergrube, SMTP nach POP und SMTP-AUTH greifen, wie auf Seite 26 beschrieben, direkt auf das E-Mail-Versandprotokoll SMTP zurück und eignen sich somit für die Kombination mit allen anderen Verfahren unter der Bedingung, dass der Klient direkt mit dem Transportsystem Kontakt aufnimmt. Bei der Kommunikation der *mail transfer agents* (MTA, Transportsysteme für E-Mail) untereinander können sie nicht verwendet werden. Zu beachten ist, dass SMTP nach POP nicht mit SMTP-AUTH kombiniert werden sollte, da eine zusätzliche Authentifizierung bei dieser Kombination nicht sinnvoll ist.

3.3.2 Abhängigkeitsanalyse

Bei der Kombination der Methoden sind nicht nur die Verknüpfungsvarianten sondern auch die Reihenfolge der Verknüpfungen innerhalb einer Kombinationskette interessant. Abbildung 3.4 zeigt ein Kombinationsmodell. In diesem Modell wurden alle für die Kombination geeigneten Schutzmethoden in einer Ablaufkette dargestellt.

SMTP nach POP, SMTP AUTH und das Teergruben-Verfahren werden bei einem direkten Transport zwischen Klienten und MTA aktiv. SMTP nach POP und SMTP AUTH sind beides Erweiterungen des SMTP-Protokolls und somit steht das Teergruben-Verfahren als Modifikation des SMTP-Protokolls näher am Empfang.

Trifft eine E-Mail mit Wegwerfadresse ein, so durchläuft sie das Verfahren Wegwerfadresse. Handelt es sich nicht um eine Nachricht mit Wegwerfadresse, so wird der Versender zunächst mit den Adressen der persönlichen White- und Blacklist verglichen. Wird eine Übereinstimmung gefunden, so kann die Nachricht sofort weitergeleitet bzw. als Spam deklariert werden. Die Reihenfolge der nachfolgenden Methoden Inhaltsfilter, Regelfilter, Bayesfilter und DCC ergibt sich aus dem zu erwartenden Aufwand. Eine Abfrage des DCC ist wesentlich aufwändiger als der Bayes-Filter. Dies ist in der notwendigen DNS-Abfrage begründet. Der Bayes-Filter ist aufwändiger als der Regelfilter, denn das Bayes-Verfahren benötigt Berechnungen sowie viele Vergleichsoperationen mit den gespeicherten Daten. Der Regelfilter ist wiederum aufwändiger als der Inhaltsfilter, da reguläre Ausdrücke ausgewertet werden müssen. Im Vergleich dazu nimmt der Inhaltsfilter eine Suche nach Stichwörtern vor. Es ist von Vorteil, wenn unerwünschte Nachrichten im Vorfeld aus der Kette entfernt werden. Das Channels-Verfahren, als Verfahren mit dem höchsten Aufwand, sollte somit den letzten Platz einnehmen. Der Aufwand setzt sich bei diesem Verfahren aus den notwendigen Ressourcen für die Zwischenspeicherung der E-Mail, der Antwort an den Versender und dem für den Versender entstehenden Aufwand für den Versand der zweiten E-Mail zusammen.

3.3.3 Schlussfolgerungen

Das entwickelte Modell ist nach Abbildung 3.4 in der Auswahl der Schutzmechanismen und ihrer Reihenfolge festgelegt. Je nach gewünschter Priorität sind Vertauschungen innerhalb der Gruppe Filterung und Adressvalidierung/Adressgenerierung möglich. Wegwerfadressen nehmen aufgrund ihrer Semantik als Universaladressen eine Sonderstellung ein. Sie können trotzdem mit der Gruppe Protokollmodifikation kombiniert werden. SMTP-Protokollmodifikationen schützen generell nur bei direktem Kontakt zwischen Klienten und Transportsystem. Aufgrund der Auslegung als ausschließliches Weiterleitungssystem kann diese Gruppe nicht eingebunden werden.

3.4 Teil 1: Mailserver

Die Entscheidung für die Nutzung eines bestehenden Mailservers als Entwicklungsgrundlage legt die Mehrzahl der Entwurfsüberlegungen bereits im Vorfeld fest. Die Gründe dafür werden in diesem Abschnitt dargelegt.

3.4.1 Nutzung eines bestehenden Mailsystems

Der Ansatz wurde gewählt, da der zu erwartende Programmieraufwand für ein neues System sehr hoch ist. Der Grund liegt in der Komplexität von einem eigenständigen Sende- und Empfangssystem. Hervorgerufen wird diese durch notwendige Ausnahmebehandlungen bei Ablauf der Empfangs- und Versandprotokolle, Warteschlangenmechanismen, Fehlerbehandlungen bei Zustellungsproblemen und vielen anderen Faktoren. Bei der Suche nach einem geeigneten Projekt stellte sich schnell heraus, dass von den derzeitig existenten freien Mailservern das Projekt *Java Apache Mail Enterprise Server* (JAMES, Apache Mailserver) geeignet ist. Programmiert in der Sprache Java und lizenziert unter der Apache Software Lizenz[33] bietet das Projekt Eigenschaften wie Portabilität, Protokoll- und Ressourcenabstraktion sowie das Konzept der Matcher und Mailets. Dieses Konzept ist für den Entwurf sehr wichtig, daher soll es an dieser Stelle vorgestellt werden.

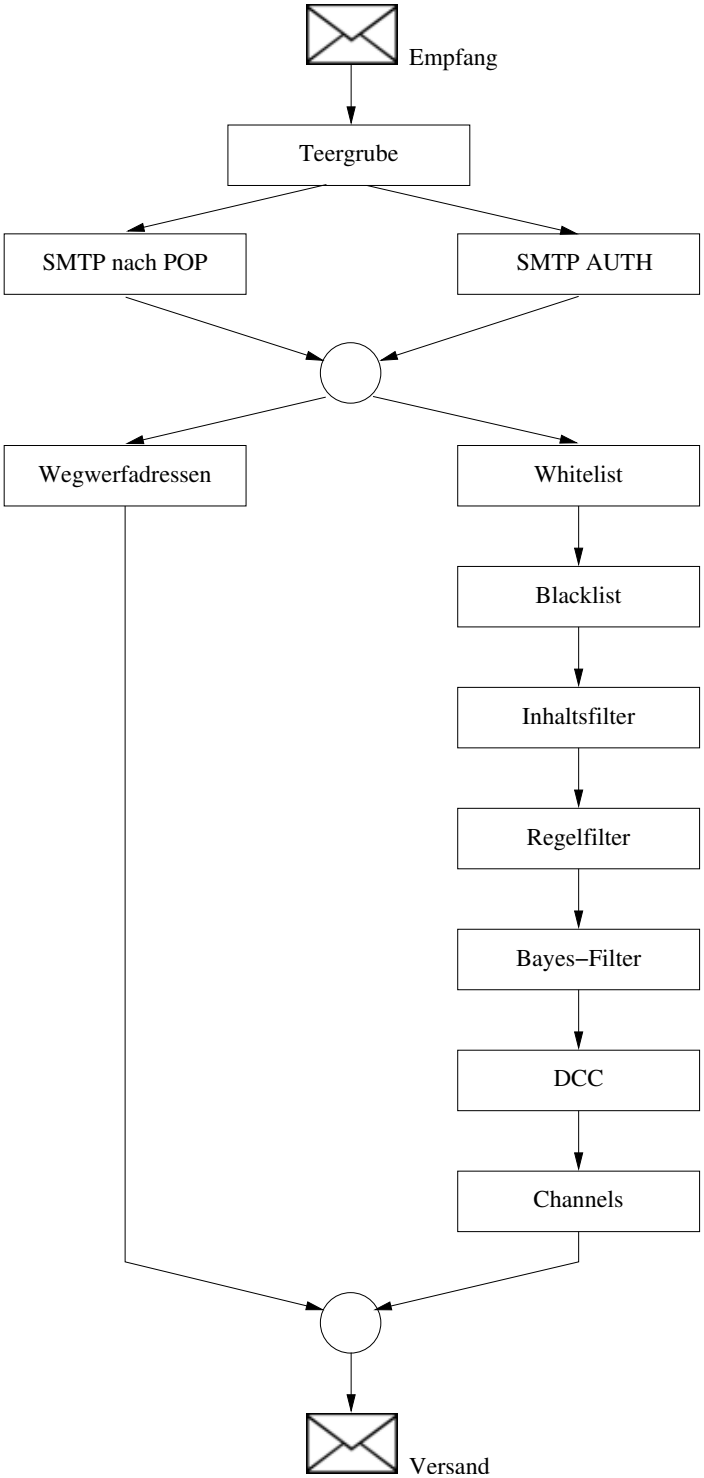

Abbildung 3.4: Kombinationsmodell der Anti-Spam-Maßnahmen

3.4.2 Matcher und Mailets

Matcher und Mailets sind zwei essentielle Konzepte, welche die Flexibilität des Mailservers ermöglichen [27]. Die Mailet-API baut auf der JavaMail-API [28] auf. Ein Matcher ermöglicht das Extrahieren von Nachrichten aus der internen Warteschlange des Mailservers nach frei definierbaren Gesichtspunkten. Nachrichten, welche ein Matcher-Kriterium erfüllen, werden zur Erledigung der entsprechenden resultierenden Aktion an das korrespondierende Mailet weitergeleitet. Dieses verarbeitet die E-Mail. In Abbildung 3.5 ist die Topologie für zwei Matcher dargestellt. Abhängig von der gewünschten Funktionalität kann die Nachricht nach Bearbeitung durch den Matcher zurück in die Warteschlange eingereiht oder komplett verworfen werden. Aus programmiertechnischer Sicht umfasst die API die beiden *interfaces*, also Schnittstellen, Matcher und Mailet mit den abstrakten Implementationen `GenericMatcher` und `GenericMailet`. Dabei übernimmt der Mailserver die Instantiierung und Verwaltung der Objekte.

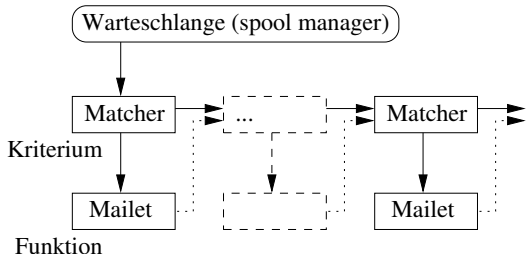

Abbildung 3.5: Flexibilität durch Matcher und Mailets

Der Matcher ist für die Wegeleitung innerhalb des Servers zuständig. Dazu existiert eine abstrakte Klasse `GenericRecipientMatcher`, die speziell Empfängeradressen auswertet. Streng genommen bekommt der Matcher eine ganze Sammlung von Objekten übergeben. `GenericRecipientMatcher` iteriert über diese Sammlung und erzeugt daraus ein einzelnes Objekt. Dadurch muss nur noch die `matchRecipient(MailAddress recipient)` Methode überschrieben werden. Abbildung 3.6 demonstriert die Erstellung des Matchers BEISPIELMATCHER.

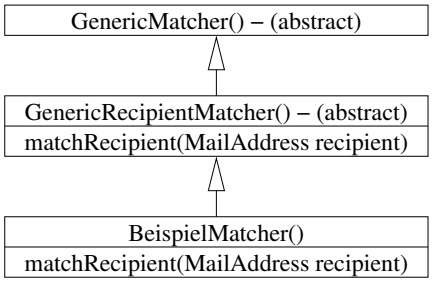

Abbildung 3.6: UML-Diagramm für den Matcher “BeispielMatcher”

3.4.3 Entwurf der Matcher- und Mailettopologie

In Abbildung 3.7 ist die Umsetzung auf das Mailet/Matchermodell dargestellt. Die aus der Warteschlange ankommenden Nachrichten müssen zunächst auf den Empfänger geprüft werden. Dies ist notwendig, um festzustellen, ob der Empfänger bei dem Dienst registriert ist und somit bei Missbrauchsversuchen weitere aufwändige Prüfungen zu vermeiden. Wenn der Empfänger beim Dienst registriert ist, so wird die Nachricht innerhalb der Warteschlange an das nächste Matcher/Mailet-Paar weitergereicht. Bei positiver Prüfung auf UCE wird die Nachricht innerhalb des Mailets zur Spamadresse weitergereicht und verworfen,d.h. aus der Kette entfernt. Bei negativer Prüfung erfolgt eine erneute Weiterleitung an weitere Verfahren. Nach dem letzten Verfahren sorgt ein abschließendes Mailet für die ordnungsgemäße Zustellung an die legitime Adresse. Dies ist notwendig, da Nachrichten welche den Weg durch sämtliche Verfahren nehmen legitime Nachrichten sind.

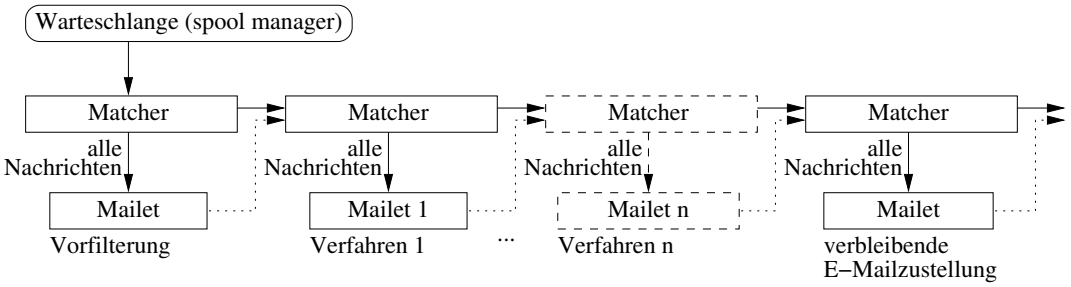

Abbildung 3.7: Entwurf der Mailet/Matcher-Topologie

3.5 Teil 2: Internetdienst

3.5.1 Benutzerschnittstelle

Der Dienst soll im Internet einer möglichst breiten Masse von Benutzern zur Verfügung gestellt werden. Primär muss die Schnittstelle zum Benutzer also möglichst einfach sein, um dieser Anforderung gerecht zu werden. Dennoch sollen die unterschiedlichen Ansprüche der Anwender Berücksichtigung finden.

Folgende Punkte sind maßgeblich für den Entwurf der Benutzerschnittstelle:

- Authentifikation und Zugriffsschutz
- Anbindung an Schutzkonzepte und deren Optionen
- Anpassbarkeit an Bedürfnisse des Benutzers (Beispiel: Landessprache)

Einerseits ermöglicht Authentifikation einen passwortgeschützten Zugang. Andererseits wird durch Zugriffsschutz gewährleistet, dass die Privatsphäre der Benutzer erhalten bleibt. Innerhalb des Dienstes werden beispielsweise Kontaktadressen und E-Mails gespeichert, die nur vom Benutzer selbst eingesehen werden dürfen. Weiterhin soll die Benutzerschnittstelle Zugriff auf alle Schutzmethoden und der zugehörigen Einstellungen bieten. Die Anpassbarkeit an die Bedürfnisse des Benutzers stellt sicher, dass der Dienst komfortabel genutzt werden kann. Auch Nicht-Experten sollen beispielsweise Black-

oder Whitelists festlegen können oder Landessprache und Weiterleitungsadressen für legitime und unerwünschte E-Mails anpassen.

3.5.2 Webserver

Ein Webserver bildet die Betriebsbasis für den Internetdienst. Dieser bildet eine standardisierte Schnittstelle zwischen dem Dienst als Serveranwendung und dem Klienten des Benutzers. Der Apache Webserver [32] der *Apache Software Foundation* (ASF) [29], bietet die notwendigen Eigenschaften. Unterstützung einer dynamischen Skriptsprache, Stabilität und Flexibilität sowie der modulare Aufbau waren die Kriterien für diese Enstcheidung.

3.5.3 Seitengenerierung

Der Dienst soll für zukünftige Methoden offen sein und eine leichte Integration ermöglichen. Daher soll die Generierung der Internetseiten weitesgehend dynamisch erfolgen und somit der Aufwand für Erweiterungen gering gehalten werden.

Als Sprache für die dynamische Generierung der HTML-Seiten wird *PHP hypertext preprocessor* (PHP[1], PHP Hypertext-Präprozessor) genutzt. Es handelt sich um eine serverseitige Skriptsprache, deren Entwicklung 1995 von Rasmus Lerdorf initiiert wurde. Die Anweisungen der Sprache sind direkt in den HTML-Quellcode der Internetseite eingebettet. Die für die Wahl der Sprache ausschlaggebenden Eigenschaften der Sprache sind die ausgezeichnete Anbindung an eine Datenbank, nahtlose Integration in den Webserver Apache sowie eine Sitzungsverwaltung. PHP ist ein Bytecode-Compiler, der das zu verarbeitende Script zum Verarbeitungszeitpunkt compiliert. PHP kann als *common gateway interface script* (CGI-Skript) oder als Bestandteil des Webservers ausgeführt werden. Das notwendige Modul mod_php ist für eine Reihe von populären APIs verfügbar. PHP unterstützt Datenbankzugriffe über ODBC. Das für die Authentifizierung der Benutzer notwendige Session-Management ist seit Version 4 fester Bestandteil von PHP.

3.5.4 Datenverwaltung

Für die Datenhaltung wird eine Datenbank genutzt. Datenbanken bieten Vorteile gegenüber eigenen Speicherlösungen wie Abwicklungen paralleler Zugriffe, Zugriffschutz (Rechte), integrierte Transaktionsverwaltungen, Fernzugriff und Datenkompatibilität. Die Datenbank MySQL [30] erfüllt diese Anforderungen hervorragend. Sie unterliegt der *general public license* (GPL, Softwarelizenz) [31] und ist damit kostenlos für die private Nutzung. Der Zugriff auf die Datenbank erfolgt mit Hilfe von *structured query language* (SQL, strukturierte Abfragesprache) und *Java DataBase Connectivity* (JDBC, plattformunabhängige Schnittstelle zur Datenbanksoftware).

[1] Anmerkung: diese Abkürzung ist rekursiv definiert

Abschnitt 4

Implementierung

In diesem Kapitel wird ein lauffähiges System – im folgenden SpamStopService genannt – vorgestellt, in welchem die im Entwurf aufgeführten Ideen umgesetzt sind. Das Gesamtsystem teilt sich in den auf dem Java Apache Mailserver aufbauenden sowie in den auf den Webserver Apache nutzenden Teil. Es wurde ein Framework von häufig benutzten Funktionen implementiert. Dies betrifft Zugriffe auf E-Mails, Adressen, Datenbank und Administration.

4.1 Teil 1: Mailserver

In diesem Abschnitt werden wesentliche praktische Aspekte des ersten Teils vorgestellt.

4.1.1 Abbildung des Architekturmodells auf den Java Apache Mailserver

Aufbauend auf JAMES wurde die Funktionalität in einem *Matcher* sowie mehreren *Mailets* programmiert. Abbildung 4.1 zeigt den Matcher SPAMSTOPMATCHER sowie die Mailets ADMINMAILET, CONTENTMAILET, CAPABILITYMAILET und FORWARDMAILET. Wie im Entwurf bereits dargestellt, übernimmt das erste Mailet die Vorsortierung. Darüber hinaus ist es möglich, mit Hilfe dieses Mailets den SpamStopService zu administrieren. Das erste Verfahren, die Inhaltsfilterung, prüft eingehende E-Mails auf UCE-verdächtige Wörter und sortiert erkannte Nachrichten aus, d.h. leitet diese an die Spamadresse. Wird kein verdächtiges Wort gefunden, so treffen die Nachrichten beim zweiten Verfahren, dem capability-Verfahren, ein. Dieses reagiert entsprechend dem capability-Protokoll, beschrieben in Abschnitt 2.4.7. Ist das Verfahren nicht aktiviert, so wird die Nachricht vom FORWARDMAILET an die legitime Adresse weitergeleitet und aus der Kette entfernt.

4.1.2 Datenhaltung

Für das Speichern der Benutzerdaten, der nutzerspezifischen Optionen und sonstiger Metadaten wurde ein Datenbankmodell entworfen, das in Abbildung 4.2 dargestellt ist. Die Tabelle SUBSCRIBERS enthält sämtliche registrierte Benutzer. Beim Registrierungsvorgang wird für jede Methode eine korrespondierende Tabelle angelegt, deren Namen sich aus dem Namen der Methode, einem Unterstrich und dem registrierten Loginnamen zusammensetzt. Bei der capability-Methode wird zusätzlich eine weitere Tabelle pro Benutzer zur Zwischenspeicherung der Nachrichten benötigt.

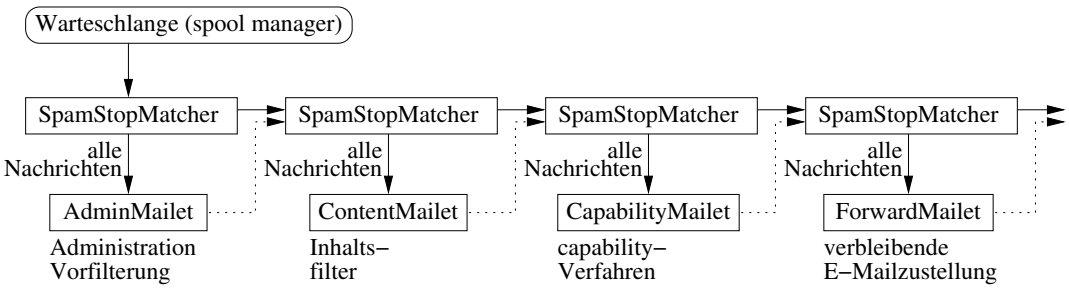

Abbildung 4.1: Implementation der Mailet/Matcher-Topologie

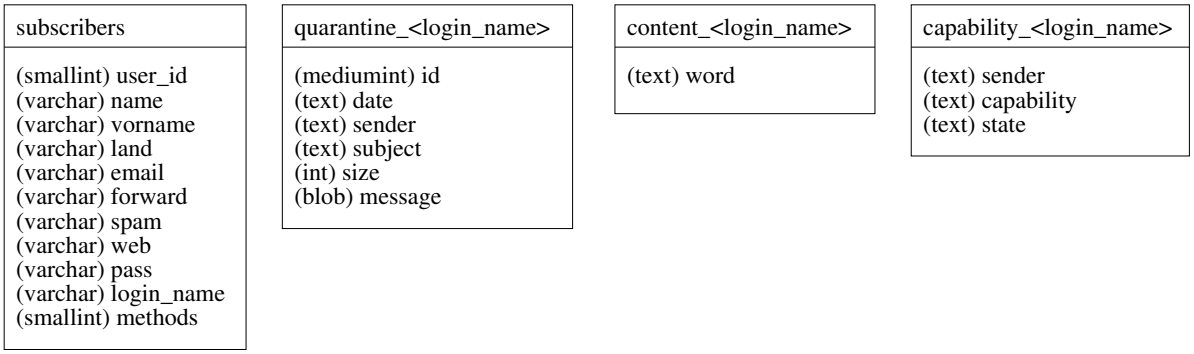

Abbildung 4.2: Datenbankmodell

4.1.3 Entwurfsdiagramm

Abbildung 4.3 zeigt das vollständige UML-Diagramm der praktischen Umsetzung. Wie in Abschnitt 3.4.3 dargelegt, implementiert jedes Mailet die abstrakten Methoden `init()`, `service()` und `getMailetInfo()`. Die Klassen `AdminService`, `MailetService`, `UtilAddress` und `SQLService` enthalten eine wiederverwendbare Bibliothek von Hilfsfunktionen für den Zugriff auf administrative Nachrichten, MIME-Nachrichten, von Sende- und Empfangsadressen und Datenbank.

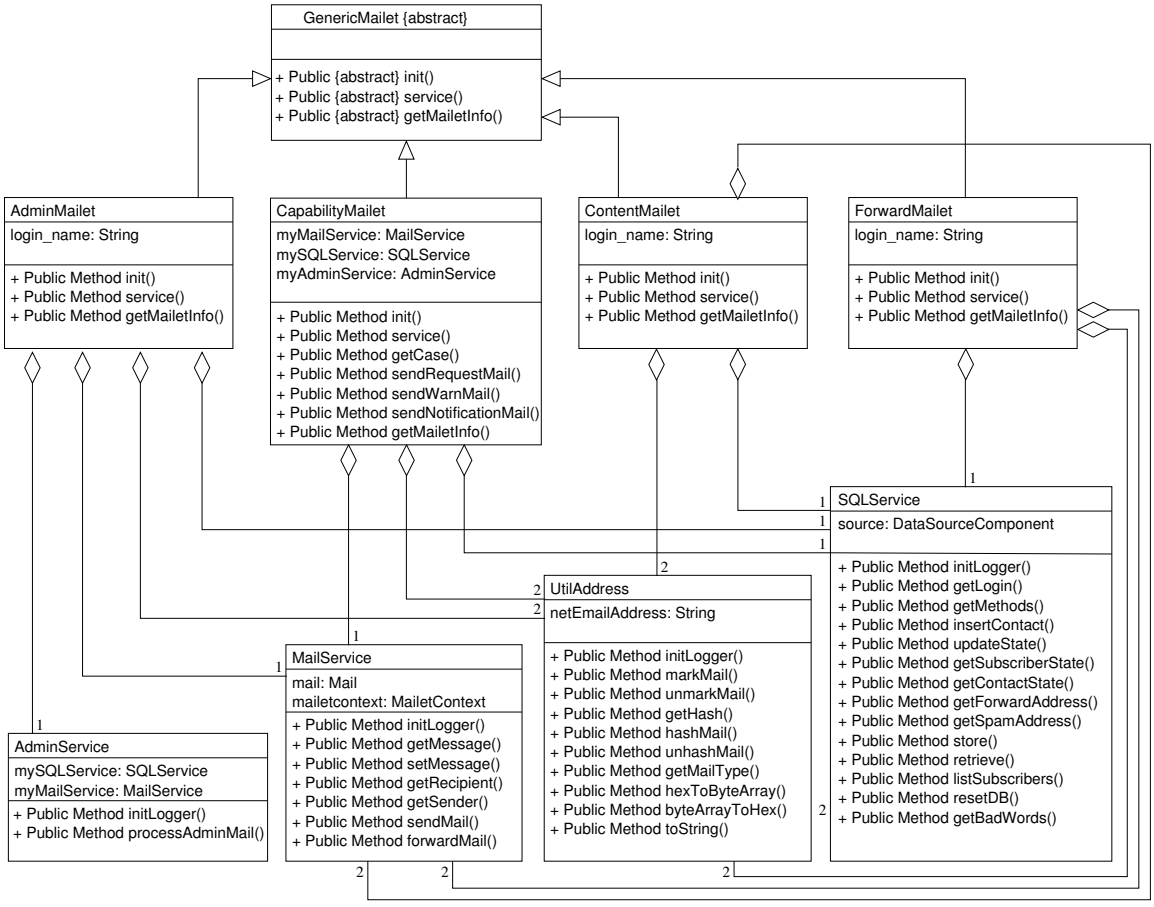

Abbildung 4.3: UML-Diagramm des Entwurfs

4.1.4 Protokollierung

Um eine detaillierte Aufzeichnung aller Ereignisse vornehmen zu können, wurde eine Protokollierungsfunktion in jedes Objekt der Software eingebaut. Unter Benutzung des Paketes `java.util.logging` der Java Logging-API wurden Protokollklassen festgelegt. Java unterstützt sieben verschiedene Standard-Protokollstufen, welche von SEVERE (höchste Priorität bzw. kritische Fehlermeldung) bis FINEST (niedrigste Priorität bzw. feiner Informationstext) reichen. In Tabelle 4.1 sind die Objekte mit den zugehörigen Protokolldateien zusammengestellt. Jeder Dateiname wird nach dem Präfix zusätzlich mit einem Zeitstempel versehen. Die Ausgabe der Daten kann im *extensible markup language* (XML)-Format, in Textform oder einem individuell anpassbaren Format erfolgen. Standardmäßig erfolgt die

Dateipräfix	Objekt
admin	AdminMailet.class
cap	CapabilityMailet.class
content	ContentMailet.class
forward	ForwardMailet.class
mail	MailService.class
sql	SQLService.class
util	UtilAddress.class

Tabelle 4.1: Protokollierungsklassen

Ausgabe im Textformat.

4.2 Teil 2: Internetdienst

Der zweite Teil der Implementierung erzeugt die eigentliche Benutzerschnittstelle über das Internet. Im folgenden Abschnitt werden wichtige Bestandteile der Implementierung vorgestellt.

4.2.1 Datenbankzugriff

Der Zugriff auf die MySQL-Datenbank wird mit Funktionen des PHP-Interpreters ermöglicht. Zunächst fordert

```
mysql_connect("rechnername","nutzer","pass");
```

eine neue Datenbankverbindung zum Rechner RECHNERNAME mit Benutzerkennung NUTZER und Passwort PASS an.

```
mysql_connect("localhost","root","passwort");
mysql_select_db("james");
```

Letzterer Befehl wählt als aktuelle Datenbank JAMES aus. Nun kann auf diese mit Hilfe von SQL-Anfragen zugegriffen werden.

```
$c=mysql_query("CREATE TABLE testtab (spalte1 TEXT NULL);");
if (!$c) echo "Fehler beim Anlegen !";
```

4.2.2 Sitzungsverwaltung

Um einen Benutzer während der Benutzung des Internetdienstes eindeutig identifizieren zu können, ist es notwendig, für die gesamte Nutzungsdauer eine Identität zu vergeben. Eine solche muss auch über weitere verlinkte Seiten innerhalb des Dienstes erhalten bleiben. Dafür bietet sich die Nutzung der in PHP eingebauten Sitzungsverwaltung an. Die Unterstützung einer Sitzungsverwaltung in PHP besteht darin, dass eine oder mehrere Variablen auch über mehrere Seitenanfragen hinweg in einer Sitzungsvariablen gespeichert werden können. Die Sitzungsvariable selbst kann dabei in einem *Cookie* (Keks) oder über den *uniform resource locator* (URL, einheitliche Adressierungsangabe) von Seite zu Seite

mitgeführt werden. Ein Cookie ist eine auf Klientseite abgelegtes Objekt, welches lokal auf dem Benutzerrechner abgelegt wird. Es besitzt ein Verfallsdatum, welches die Gültigkeitsdauer begrenzt. Dadurch kann die Identität für eine bestimmte Zeit zwischengespeichert werden.

Beide Möglichkeiten kommen also für die Speicherung der Identität in Betracht, wobei letztere voraussetzt, dass der Benutzer die Annahme von Cookies akzeptiert. Trotz dieser Einschränkung fiel die Entscheidung auf die letztere Variante, da die Mitführung der Sitzungsidentität in der URL ein Sicherheitsrisiko birgt. Bei Anwahl eines externen Verweises ausserhalb des Dienstes oder beim Besuch einer nachfolgenden Seite im Internet kann die verweisende URL ausgelesen werden. Diese enthält die Identifikation im Klartext. Dies ist auch der Fall, wenn eine URL als Lesezeichen abgelegt wird.

Seit PHP Version 4 sind globale Variablen aus Sicherheitsgründen deaktiviert. Im Gegensatz dazu ist das assoziative Datenfeld `$_SESSION` immer global verfügbar. Bevor diese Variable benutzt werden kann, muss die Sitzung mit dem Befehl `session_start()` initialisiert werden. Danach können dem Datenfeld beliebig viele Variablen übergeben werden. Das folgende Codefragment übergibt den Inhalt der Variable LOGIN_USER als Sitzungsvariable.

```
session_start();
session_register("user");
$_SESSION['user']=$login_user;
```

Wie bei allen HTTP-Kopfzeilen, dürfen vor dem Setzen von Cookies keine Daten an den Klient gesendet worden sein. Erst wenn der Versand aller Kopfzeilen erfolgte, können weitere Nutzdaten, wie zum Beispiel HTML, folgen. Ob eine Sitzungsvariable USER in der aktuellen Sitzung existiert, wird mit dem Kommando `session_is_registered(user)` überprüft. Eine Überprüfung erfolgt auf jeder Webseite des Dienstes. Falls die Sitzungsvariable USER nicht gefunden werden kann, bricht das HTTP-Protokoll mit einer Fehlermeldung ab.

```
session_start();

if (!session_is_registered('user')) {
    die("Nicht registriert!");
    }
```

Das Beenden der Sitzung erfolgt durch Löschen der aktuellen Sitzung sowie dem Entfernen aller mit der Sitzung verbundenen Variablen.

```
session_unset();
session_destroy();
```

4.2.3 Internationalisierung

Um den Internetdienst für eine Internationalisierung in vielen Sprachen vorzubereiten, wurde ein einheitlicher Mechanismus entwickelt. Der folgende Abschnitt behandelt die vorhandenen Möglichkeiten und legt dar, weshalb der benutzte Ansatz gewählt wurde. Für die praktische Umsetzung ergeben sich mehrere Varianten, die im folgenden kurz vorgestellt werden sollen.

Erstellt werden soll eine Internetseite, deren Inhalt in Deutsch sowie Englisch vom Benutzer abgerufen werden kann. Als Beispiel dient eine kurze HTML-Datei, welche lediglich das Wort AUSGABE bzw. OUTPUT darstellt. Die einfachste Möglichkeit wäre, diese Seite in mehrfacher Ausführung zu erstellen und z.B. durch einen entsprechenden Suffix des Dateinamens die Landessprache zu kennzeichnen. Die Datei würde also in zweifacher Fassung erstellt und in verschiedene Dateien gespeichert, beispielsweise AUSGABE_DEUTSCH.HTML und AUSGABE_ENGLISCH.HTML:

```
<HTML>
     <BODY>
          Ausgabe
     </BODY>
</HTML>
```

und

```
<HTML>
     <BODY>
          Output
     </BODY>
</HTML>
```

Diese Möglichkeit ist aus mehreren Gründen sehr aufwändig. Jede Änderung am Quelltext muß in allen Dateien durchgeführt werden. Falls eine neue Sprache hinzugefügt werden soll, ergeben sich ebenfalls notwendige Änderungen in allen HTML-Dateien. Es entsteht ein enormer Verwaltungsaufwand um das gesamte Projekt konsistent zu halten. Eine weitere Idee ist es, PHP für die dynamische Generierung der Texte einzusetzen. Bei Verwendung von PHP für die dynamische Generierung der Ausgabetexte sinkt der Verwaltungaufwand, da nur noch eine Seite geändert werden muss. Trotzdem ist das Hinzufügen von weiteren Sprachen kompliziert, da wiederum alle beteiligten Projektseiten um den entsprechenden PHP-Quelltext erweitert werden müssen. Hinzu kommt, dass jede Projektdatei für die Ermittlung der Sprache eine Datenbankabfrage enthält. Daher entstand die Idee, diese Abfrage global durchzuführen. Trotz dieser Optimierung besteht weiterhin das Problem der notwendigen Änderungen im Quelltext. Der optimierte Ansatz, der auch in der praktischen Implementation eingesetzt wurde, besteht in einer globalen Abfrage der Sprache, der Einführung einer globalen Sprachdatei für jede Sprache, die dynamisch eingebunden wird. Auf das Beispiel angewendet, ergeben sich vier notwendige Dateien. Zum einen die zwei internationalisierte Dateien. Die Datei SPRACHE_DEUTSCH.PHP enthält

```
<?PHP
 $beispiel_ausgabetext="Ausgabe";
 ?>
```

und Datei SPRACHE_ENGLISH.PHP

```
<?PHP
 $beispiel_ausgabetext="Output";
 ?>
```

Weiterhin wird die globale Abfrage benötigt, welche in jede Projektseite eingebunden wird INCLUDE_SPRACHE.PHP:

```
<?PHP
    /* Datenbank abfragen, Ergebnis in Sprache ablegen */
    $abfrage = mysql_query("SELECT * from database
         WHERE login_name='login';");
    $reihen = mysql_num_rows($abfrage);
    $row=mysql_fetch_object($abfrage);
    $sprache=$row->language;

    /* entsprechende Sprachdatei einbinden */
    if ($sprache=="Deutsch") include "sprache_deutsch.php";
    if ($sprache=="English") include "sprache_english.php";
 ?>
```

Die eigentliche Projektseite hat den Inhalt

```
<?PHP include "include_sprache.php"; ?>

<HTML>
     <BODY>
          echo $beispiel_ausgabetext;
     </BODY>
</HTML>
```

Dies ermöglicht einerseits eine wesentliche Minimierung des Arbeitsaufwandes bei anstehenden Änderungen von Seiten und andererseits ein umkompliziertes Hinzufügen von weiteren Sprachen. Die Änderungen beschränken sich auf die einzubindene Datei. Dieser Ansatz wurde für die Internationalisierung des Dienstes verwendet. Dabei wurden die Variablen konsequent nach der enthaltenen Internetseite benannt. Momentan werden als Sprachen Deutsch und Englisch unterstützt.

Abschnitt 5

Bewertung

5.1 Beispielszenario

Der folgende Abschnitt beschreibt die notwendigen Schritte, um den Internetdienst nutzen zu können. Die Benutzerschnittstelle wird in einem typischen Nutzungsablauf vorgestellt.

5.1.1 Ausgangssituation

Abbildung 5.1 zeigt die Eingangsseite, welche der Benutzer bei Aufruf der Basisadresse des Webservers angezeigt bekommt. Eine kurze Einführung mit häufig gestellten Fragen und den entsprechenden Antworten kann bei Bedarf abgerufen werden. Dies soll den Einstieg für das Verständnis des Dienstes erleichtern. Damit der Dienst benutzt werden kann, muss vorher eine Registrierung erfolgen. Im folgenden wird davon ausgegangen, dass der Beispielnutzer MAX MUSTER bereits bei einem (oder alternativ mehreren) Anbietern zwei E-Mailadressen beantragt hat. Diese lauten MAX.MUSTER@PROVIDER.DE und MAX.MUSTER_SPAM@PROVIDER.DE. Sie sollten geheim gehalten und unter keinen Umständen unerwünschten Versendern zugänglich gemacht werden.

5.1.2 Anmeldung für den Dienst

Nachdem "Anmeldung" angewählt wurde erscheint ein Registrierungsformular, dieses ist in Abbildung 5.2 ersichtlich. Für die Registrierung sind einige zusätzliche Informationen notwendig. Dazu gehört der vom Benutzer gewünschte Loginname sowie ein persönliches Passwort.

5.1.3 Login

Nach erfolgreicher Anmeldung und einem anschließenden Login auf der Basisseite wird der Benutzer in den sitzungsabhängigen und damit geschützten Bereich weitergeleitet. Abbildung 5.3 zeigt die Ausgabe direkt nach erfolgreichem Login. Das Menü auf der linken Seite ist für den schnellen Zugriff auf alle Seiten reserviert. Es enthält die Punkte Übersicht, Anti-Spam Einstellungen, Anti-Spam Verfahren, Nachrichtenspeicher, SpamStopService-Optionen und Beenden. Auf der Übersichtsseite wird jede Funktion kurz beschrieben. Anti-Spam Einstellungen ermöglichen die Auswahl der Verfahren und globale Einstellungen, welche für alle Verfahren zutreffen. Dies beinhaltet die zwei Weiterleitungsadressen für legitime und unerwünschte Nachrichten. Unter Anti-Spam Verfahren können Beschreibungen

Abbildung 5.1: Eingangsseite

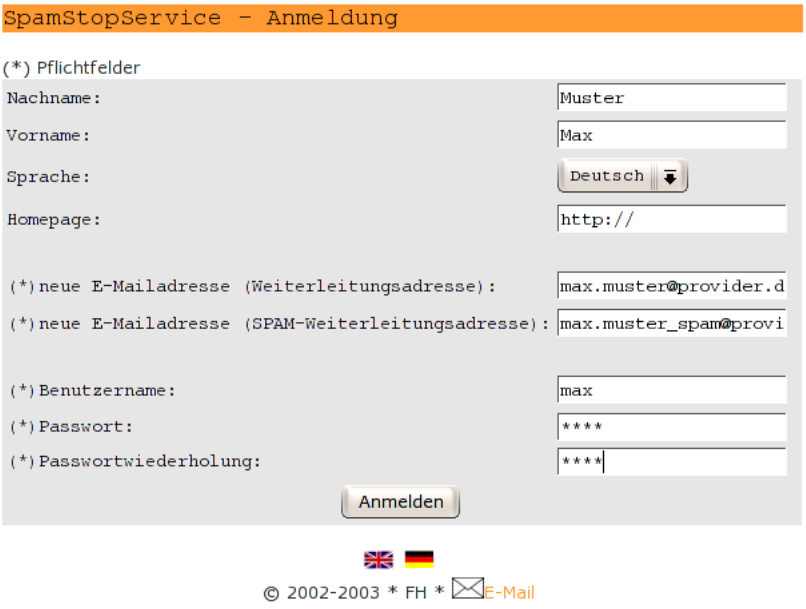

Abbildung 5.2: Registrierung

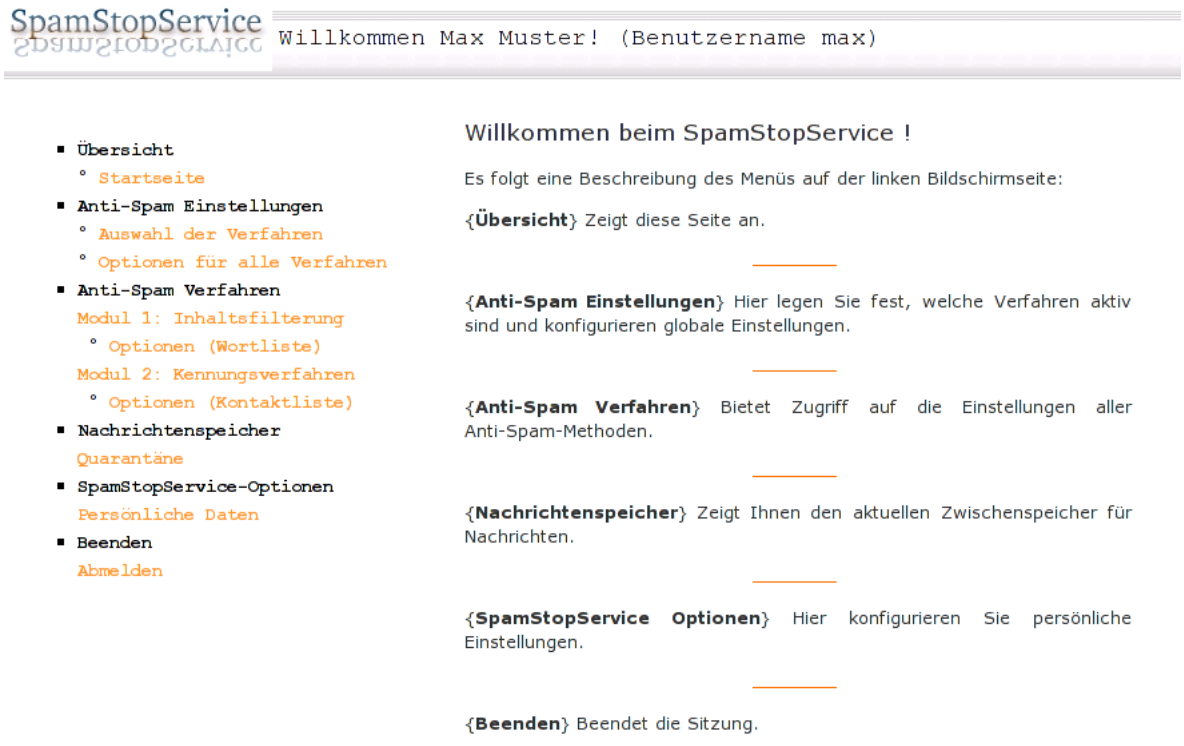

Abbildung 5.3: Start der Sitzung

der Anti-Spam Module abgefragt sowie notwendige spezifische Einstellungen vorgenommen werden. Ein Beispiel für den capability-basierten Ansatz ist die Kontaktliste. Der Nachrichtenspeicher dient zur temporären Speicherung von Nachrichten. Im Moment wird dieser nur von der capability-basierten Methode genutzt. Unter SpamStopService-Optionen besteht die Möglichkeit, persönliche Optionen wie Sprache, Vor- und Zuname festzulegen. Beenden schließt die Sitzung ab.

5.1.4 Auswahl der Verfahren

Der Benutzer entscheidet sich im Beispiel für das Verfahren “Inhaltsfilterung”. Nachdem das Verfahren markiert und übernommen wurde, ist dieses für den Benutzer aktiv. Abbildung 5.4 zeigt die Ausgabe. Die verfügbaren Optionen sind für jedes Verfahren zunächst leer initialisiert worden. Im konkreten Fall, also dem Inhaltsfilter, ist eine Wortliste als Option verfügbar. Im folgenden sollen nun die Wörter STRENG und VERBOTEN in diese Liste aufgenommen werden. Das Ergebnis ist in Abbildung 5.5 dargestellt. Hinzugefügte Wörter können jederzeit durch Anwahl von Löschen wieder aus der Liste entfernt werden.

5.1.5 Logout

Das Abmelden erfolgt durch Anwahl des Menüpunktes Abmelden. Die Sitzung wird damit geschlossen.

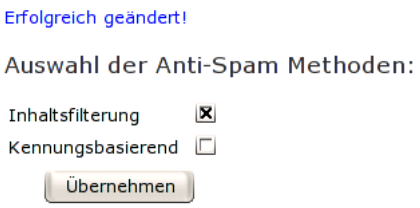

Abbildung 5.4: Auswahl von Inhaltsfilterung als gewünschtes Verfahren

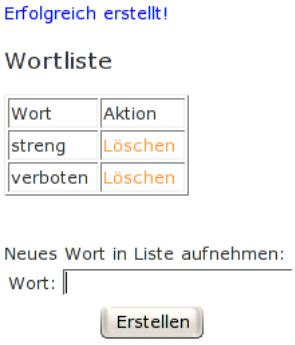

Abbildung 5.5: Aufnahme von Wörtern in die Wortliste

5.1.6 Sonstige Funktionen

Ein Sprachwechsel kann im Anmeldebereich durch Anwahl der Landesflagge erfolgen. Im Mitgliederbereich kann die Sprache durch Veränderung der persönlichen Optionen gewechselt werden. Falls der Benutzer sich nicht mehr an das vergebene Passwort erinnern kann, so besteht die Möglichkeit sich dieses per E-Mail an die legitime Weiterleitungsadresse zusenden zu lassen. Dazu wird sowohl der Loginname als auch die zugehörige Weiterleitungsadresse benötigt. Es besteht ausserdem die Möglichkeit, sich vom Dienst wieder abzumelden. Es werden dann alle gespeicherten Daten inklusive der zwischengespeicherten Nachrichten aus der Datenbank entfernt.

5.2 Ergebnisse

5.2.1 Missbrauch des Servers für Weiterleitungen

Während der Programmierung und des Tests des praktischen Teils der Arbeit, wurde die gesamte E-Mail-Kommunikation von und zum Testrechner protokolliert. Dabei wurde festgestellt, dass bereits einige Tage nach dem Öffnen von Port 25 und 110 des Servers Kontaktversuche von aussen stattfanden.

Die Anzahl der Kontaktversuche nahm stetig zu. Fast alle Nachrichten waren darauf ausgerichtet, den Mailserver als Weiterleitung zu missbrauchen. Dies ist damit zu erklären, dass die öffentliche Adresse FRANK@ALBRECHT.INF.TU-DRESDEN.DE erst seit Juni 2003 im Rahmen der Tests benutzt wurde. In Tabelle 5.1 sind einige der Kontaktversuche mit zugehörigem Zeitpunkt ersichtlich.

Datum	Versender	Empfänger
8.Apr 2003	consumerreply08@flashmail.com	tupperhorse5@aol.com
22.Apr 2003	francisf062000@yahoo.com	tupperhorse5@aol.com
27.Apr 2003	diamondmktg@netzero.net	d01m@firemail.de
3.Mai 2003	wealthnow7676@zapo.net	pengwinnfoot@hotmail.com
22.Mai 2003	d7t5he4472@ananzi.co.za	kihytf@yahoo.com
24.Mai 2003	lfi7hr@ananzi.co.za	xunestone@yahoo.com
25.Mai 2003	d7t5he7344@inbox.lv	sticway@yahoo.com
25.Mai 2003	l97n6f@runbox.com	zappa638@yahoo.com
23.Jun 2003	ladybeate@tu-dresden.de	theism2@sbcglobal.net
25.Jun 2003	katharina.fahrner@generali.at	frank@albrecht.inf.tu-dresden.de
02.Jul 2003	flight@tu-dresden.de	theism2@sbcglobal.net
04.Jul 2003	jkennihan@tu-dresden.de	theism2@sbcglobal.net
06.Jul 2003	as4f@postino.ch	fh8ju7h@yahoo.com

Tabelle 5.1: Server-Mißbrauchsversuche von Spammern

5.2.2 Leistungsmessung

Als Testrechner stand ein Rechner mit Pentium Pro-Prozessor (200 MHz), 64 MB RAM und 300 MB Festplattenkapazität zur Verfügung. Auf dem Rechner war das Linuxderivat Debian in der Version 3.0

(Kern 2.4.21) installiert. Das Java Software Development Kit in Version 1.4, sowie Apache James in Version 1.2.4 bildeten die Entwicklungsgrundlage. MySQL Version Version 9.38, Distribution 3.22.32 kam als Datenbank zum Einsatz.

Sowohl James als auch MySQL arbeiteten als dedizierte Serverdienste. Für die Versandtests wurden verschiedene universitätsexterne Rechner eingesetzt. Für Leistungstests wurde ein universitätsinterner Rechner verwendet. Als Internetverbindung stand ein 10 MBit Zugang zur Verfügung. Linux, und das Kommandozeilenwerkzeug *mail* übernahmen den Versand der Emails. Ein BASH-Skript versendete die Emails:

```
#/bin/bash

# Zieladresse
export TARGET="frank@albrecht.inf.tu-dresden.de";

# Schleife
for i in ‘seq 1 $1‘
do
# Anzahl auf Konsole ausgeben
echo "email $i";
# E-Mail abschicken
mail -s "mail no $i" $TARGET < ".";
done
```

Die Ausführungszeit des Skriptes wurde mit Hilfe des Kommandos *time* (Werkzeug zur Messung von Ausführungszeiten) gemessen. Die Bearbeitungsdauer ist die Zeit zwischen dem Eintreffen der Nachricht und dem erfolgreichen Bearbeiten. Die Zeiten sind anhand der Einträge in den Protokolldateien berechnet. Alle Zeiten sind in Sekunden angegeben.

Gegenstand der Messung war zunächst die reine Weiterleitungsleistung des Systems. Wenn alle Schutzsysteme deaktiviert sind, so werden die Nachrichten lediglich an die legitime Adresse weitergeleitet. Die Ergebnisse der Messungen sind in Tabelle 5.2 zusammengefasst.

Anzahl	t_v	t_b
1	0,01	2
2	0,04	4
3	0,06	6
4	0,08	9
5	0,1	14

Tabelle 5.2: Leistungsmessung der Weiterleitung

Zwei weitere Messung wurde für die einzeln aktivierten Verfahren Inhaltsfilterung und capability se-

parat durchgeführt. Die Ergebnisse sind in Tabelle 5.3 und 5.4 dargestellt.

Die Aktivierung beider Verfahren ergab Leistungswerte, die in Tabelle 5.5 dargestellt sind.

Anzahl	t_v	t_b
1	0,01	2
2	0,04	9
3	0,06	11
4	0,08	16
5	0,1	21

Tabelle 5.3: Leistungsmessung der Inhaltsfilterung

Die Versanddauer verhielt sich erwartungsgemäß etwa linear. Abbildung 5.6 zeigt die graphische Dar-

Anzahl	t_v	t_b
1	0,01	2
2	0,04	9
3	0,06	12
4	0,08	16
5	0,1	22

Tabelle 5.4: Leistungsmessung des capability-Verfahrens

Anzahl	t_v	t_b
1	0,01	2
2	0,04	9
3	0,06	13
4	0,08	18
5	0,1	28

Tabelle 5.5: Leistungsmessung der Kombination von Inhaltsfilterung und capability-Verfahren

stellung in einem Diagramm. Das Diagramm in Abbildung 5.7 vergleicht die Bearbeitungszeiten für die gemessenenen Kombinationen graphisch. Deutlich sichtbar ist der exponentielle Rechenzeitbedarf, der bei dem gleichzeitigen Eintreffen von Nachrichten entsteht. Die praktischen Tests haben vor allem Probleme bei der Leistungsfähigkeit des Systems gezeigt. Die bei jedem Eintreffen einer E-Mail notwendigen Datenbankabfragen machen das System langsam und anfällig gegen Attacken von außen. Beispielsweise würde ein konstanter Nachrichtenstrom mit zufällig generierten Versenderadressen von mehreren E-Mails pro Sekunde das System funktionsunfähig machen. An dieser Stelle müssen Optimierungen vorgenommen werden. Ein Zwischenspeicher könnte die Abfragen beschleunigen. Weitere Probleme tauchten bei dem Test der Formulare auf. Alle Eingaben werden momentan ungefiltert als

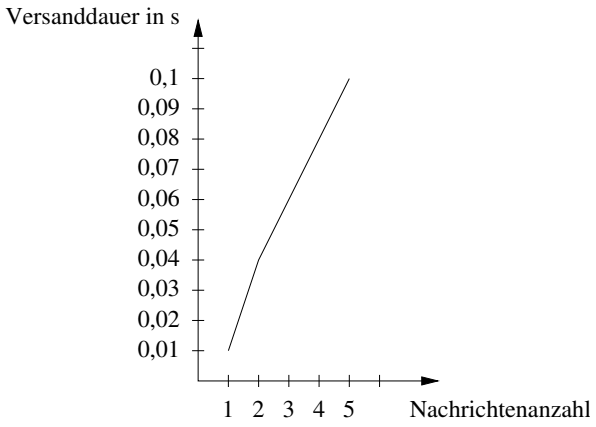

Abbildung 5.6: Versanddauer

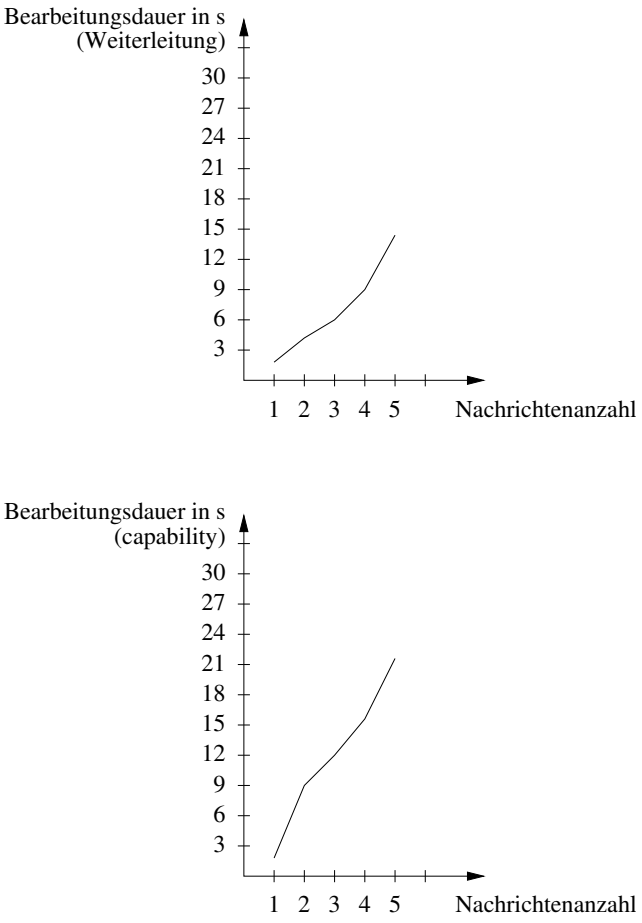

Abbildung 5.7: Bearbeitungsdauer für Weiterleitung und capability-Verfahren

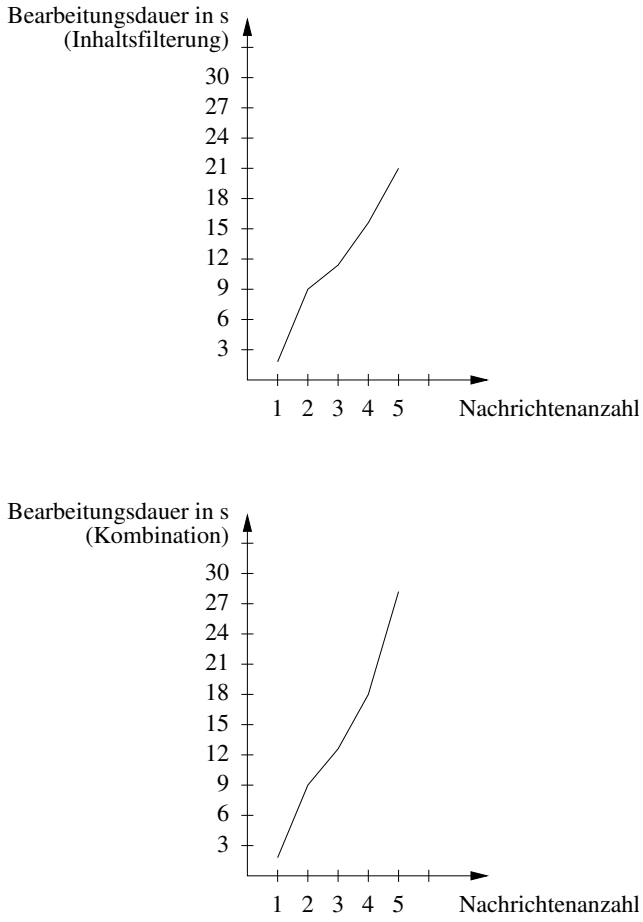

Abbildung 5.8: Bearbeitungsdauer für Inhaltsfilterung und Kombination aller Verfahren

Parameter an die Datenbank übergeben. Böswillige Benutzer könnten hier ansetzen und durch *SQL injection* (geschickte Konstruktion der SQL-Parameter) [35] fremde Informationen erhalten oder verändern.

Abschnitt 6

Zusammenfassung und Ausblicke

Dieses Kapitel fasst die Ergebnisse der Arbeit noch einmal zusammen und behandelt weiterführende Ideen.

6.1 Zusammenfassung

In dieser Arbeit wurde die Kombination von mehreren Systemen gegen unerwünschte Werbemail beschrieben. Es wurde beschrieben, dass es möglich ist, mehrere unterschiedliche Systeme miteinander zu kombinieren und so ihre Vorzüge zu vereinen. Um dies zu erreichen wurde eine einheitliche Schnittstelle herausgearbeitet. Der praktische Teil enthält 2 Bestandteile. Ein Framework, welches als Rahmen für die Einbindung aller Methoden dient und das Einbinden zukünftiger Methoden vereinfacht sowie einen webbasierten Teil, der die Benutzung des Dienstes auch für Nicht-Experten ermöglicht.

Der Test des Systems hat seine Funktionsfähigkeit demonstriert, im Gegenzug aber auch Probleme der Leistungsfähigkeit für eine große Anzahl von Nachrichten aufgezeigt. Weiterhin wuchs die Komplexität der Anwendung auf ein Maß, das es innerhalb des zur Verfügung stehenden Implementationszeitraumes nicht ermöglichte, alle Optimierungs- und Sicherheitsaspekte im praktischen Teil zu berücksichtigen. Der Umfang des Frameworks beträgt etwa 2500 Zeilen Quelltext, der des webbasierten Teils etwa 3000 Zeilen inklusive der Sprachdateien. Es konnten aus Aufwandsgründen lediglich zwei Methoden gegen UCE implementiert werden. Aspekte der möglichen Reaktionen bei Falscherkennung sowie die sich daraus ergebende Beeinflussung der Sortierung für die Zukunft sind ebenfalls Themen für weiterführende Arbeiten.

6.2 Ausblicke auf die Zukunft des Direktmarketing

Um den Anforderungen der Wirtschaft in Bezug auf Direktmarketing gerecht zu werden, ist es notwendig, Firmen Modelle aufzuzeigen, welche legitim für Direktmarketing genutzt werden können. Spam ist zugleich die aggressivste und am wenigsten von den Empfängern akzeptierte Methode. In Tabelle 6.1 werden die vorgeschlagenen Modelle *Opt-Out*, *Opt-In* und *Double Opt-In* mit dem Spam-Modell verglichen. Das Modell *Opt-Out* geht davon aus, dass eine Liste von E-Mail-Adressen beim Anbieter hinterlegt ist, welche innerhalb früherer Geschäftsbeziehungen geführt wurde. Dadurch ist die Adress-

	Spam	Opt-Out	Opt-In	Double Opt-In
Adressquelle	unbekannt	ehemalige Geschäfts-beziehungen	Erlaubnis durch Formular	Erlaubnis und Bestätigung
Adressqualität	niedrig	hoch	hoch, eventuell Fälschung	sehr hoch
Beschwerde-wahrschein-lichkeit	sehr hoch	hoch	niedrig	fast Null
Wahrschein-lichkeit der Spamdeklaration	sehr hoch	hoch	fast Null	fast Null
Kunden-akzeptanz	niedrig	niedrig	hoch	vollständig

Tabelle 6.1: mögliche Modelle für E-Mail-Direktmarketing, Quelle: MessageMedia (MESG)

qualität hoch.

Trotzdem möchten die wenigsten Kunden ohne ihre Einstimmung zukünftig ständig Werbeangebote erhalten. Hier greift das Modell *Opt-In*, welches vorsieht, dass lediglich die Adressaten angesprochen werden, die ausdrücklich durch das Ausfüllen eines Formulars oder das Anklicken einer entsprechenden Dialogbox in dieses Vorgehen eingewilligt haben. Dadurch wird gewährleistet, dass eine ausdrückliche Einwilligung in Werbeaktionen gegeben wurde. Der Nachteil des Modells besteht darin, dass in solche Formulare hin und wieder falsche oder nicht benutzte E-Mail-Adressen eingetragen werden.

Das *Double Opt-In* Modell stellt dies durch eine explizite Bestätigungsnachricht sicher. Kunden, die einen solchen Aufwand in Kauf nehmen, sind mit hoher Wahrscheinlichkeit an dem Angebot interessiert und zusätzlich ist die Adressqualität durch den Überprüfungsvorgang sichergestellt. Die Zukunft des E-Mail-Marketing wird also in diesem Modell liegen.

6.3 Ausblicke auf zukünftige Methoden gegen Spam

Das gravierendste Problem aller in dieser Arbeit behandelten Verfahren ist, dass ihre Verbreitung unter allen beteiligten Kommunikationspartnern nicht gegeben ist. Einige Benutzer, deren Mailklienten bzw. benutzten Provider die neuen Verfahren nicht oder nur unzureichend unterstützen, sind von der Kommunikation ausgeschlossen.

Beispiel hierfür ist ein Versender, welcher der Antwortaufforderung eines Channel-Systems mit zusätzlicher Aufwandsfunktion (z.B. E-Cash) aus technischer Hinsicht nicht nachkommen kann. Zukünftige

Techniken werden plattfomunabhängig und ebenfalls unabhängig vom verwendeten Mailklienten sein. Durch automatisch in die E-Mail eingefügte Programme (z.B. Java-Applet), welche sich im Browser bzw. Mailklient starten und die erforderlichen Aktionen selbst durchführen können, reduziert sich der Aufwand für Empfänger. Statt des Programmes selbst kann auch nur ein Verweis auf eine Internetseite angefügt sein, von der das benötigte Programm heruntergeladen werden kann. Solche dynamischen Nachrichten sowie die daraus resultierenden notwendigen Sicherheitsbetrachtungen werden Gegenstand der zukünftigen Forschung sein.

Literaturverzeichnis

[1] *EFF "Canter & Siegel Green Card Lottery Net Spam Case" Archive*, 1994, http://www.eff.org/Legal/Cases/Canter_Siegel/

[2] *Email Usage Forecast 2002-2006 Know whats coming your way*, 2002, IDC, Framingham Mass.

[3] *Welcome to SpamAssassin*, 2003, http://spamassassin.org/

[4] *Channels: Avoiding Unwanted Electronic Mail*, 1996, http://dimacs.rutgers.edu/Workshops-/-Threats/Hall.html

[5] *A Plan for Spam*, 2002, http://www.paulgraham.com/spam.html

[6] *Tagged Message Delivery Agent (TMDA)-Homepage*, 2003, http://tmda.net/

[7] *Großer Beleg zum Thema "Eine capability-basierte Lösung zur Unterbindung von Spam"*, 2002, Frank Herrmann

[8] *MAPS Realtime Blackhole List*, 2003, http://mail-abuse.org/rbl/

[9] *Teergrubing FAQ*, 2003, http://www.faqs.org/faqs/net-abuse-faq/teergrube-faq/

[10] *Jackpot*, 2003, http://jackpot.uk.net/

[11] *Bulk E-mail Software SUPERSTORE*, 2003, http://www.americaint.com/

[12] *Unerbetene kommerzielle Kommunikation und Datenschutz*, 2001, http://europa.eu.int/comm/internal_market/privacy/docs/studies/spamsum_de.pdf

[13] *Habeas*, 2003, http://www.habeas.com/

[14] *spamgourmet - disposable email addresses, spam filtering*, 2003, http://www.spamgourmet.com/

[15] *Savannah: Project Info - The ifile e-mail filtering system*, 2003, http://savannah.nongnu.org/projects/ifile/

[16] *mozilla.org*, 2003, http://www.mozilla.org/

[17] *Procmail Homepage*, 2003, http://www.procmail.org/

[18] *MAPS Realtime Blackhole List*, 2003, http://mail-abuse.org/rbl/

[19] *Brightmail Inc.*, 2003, http://www.brightmail.com/

[20] *March 2003 Spam Category Data*, 2003, http://www.brightmail.com/pdfs-/-0303_spam_definitions.pdf

[21] *Distributed Checksum Clearinghouses*, 2003, http://www.rhyolite.com/anti-spam/dcc

[22] *RFC 821: Simple Mail Transfer Protocol*, 1982, http://asg.web.cmu.edu/rfc/rfc821.html

[23] *RFC 1725: Post office Protocol - Version 3*, 1994, http://asg.web.cmu.edu/rfc/rfc1725.html

[24] *RFC 2554: SMTP Service Extension for Authentication*, 1999, http://asg.web.cmu.edu/rfc/rfc2554.html

[25] *RFC 2195: IMAP/POP AUTHorize Extension for Simple Challenge/Response*, 1997, http://asg.web.cmu.edu/rfc/rfc2195.html

[26] *Java Apache Mail Enterprise Server*, 2003, http://jakarta.apache.org/james/

[27] *Apache Jakarta Mailet API*, http://jakarta.apache.org/james/mailet/

[28] *JAVA Mail API*, 2003, http://java.sun.com/products/javamail/

[29] *The Apache Software Foundation*, 2003, http://www.apache.org/

[30] *MySQL: The Worldt's Most Popular Open Source Database*, 2003, http://www.mysql.com/

[31] *GNU General Public License*, 2003, http://www.gnu.org/copyleft/gpl.html

[32] *The Apache HTTP Server Project*, 2003, http://httpd.apache.org/

[33] *The Apache Software License, Version 1.1*, http://jakarta.apache.org/james/license.html

[34] *Spamhaus*, 2003, http://www.emailsgalore.com/

[35] *Direct SQL Command Injection*, 2003, http://www.owasp.org/asac/input_validation/sql.shtml

Anhang A

Glossar

API	*application programming interface*, Programmierschnittstelle
ASF	*Apache Software Foundation*, Softwaregremium
BASH	*bourne-again shell*, Kommandozeileninterpreter
CGI	*common gateway interface*, Skriptsprache
DCC	*distributed checksum clearinghouse*, verteilter Prüfsummenfilter
HTML	*hypertext markup language*, Standard-Seitenbeschreibungssprache des Internets
ISP	*internet service provider*, Internet-Zugangsanbieter
JAMES	*Java Apache Mail Enterprise Server*, Apache Mailserver
JDBC	*Java DataBase Connectivity*, Java-spezifische Schnittstelle für den Datenbankzugriff
MAPS	*mail abuse prevention system*, System zur Vorkehrung von E-Mail-Missbrauch
MIME	*multipurpose internet mail extension*, universelle E-Mail-Erweiterung
NIC	*network information center*, Registrierungsstelle
ODBC	*open database connectivity*, plattformunabhängige Schnittstelle für den Datenbankzugriff
PHP	*PHP hypertext preprocessor*, PHP Hypertext-Präprozessor
POP3	*post office protocol 3*, E-Mail Empfangsprotokoll Version 3
RBL	*realtime blackhole list*, in “Echtzeit” aktualisierte Sperrliste
RFC	*request for comments*, Diskussionspapier für Internetstandards, Quasistandard
SMTP	*simple mail transfer protocol*, E-Mail Versandprotokoll

SMTP-AUTH	*simple mail transfer protocol authenticated*, SMTP mit Authentifikationserweiterung
SQL	*structured query language*, strukturierte Abfragesprache
UCE	*unsolicited commercial email*, unerwünschte Werbemail, Spam
XML	*extensible markup language*, Beschreibungssprache