# Diplomarbeit

# Webbasiertes Auffinden maliziöser Software mit fortschrittlichen heuristischen Verfahren

„MWC - Malware Crawler“
Sönke Freitag

Betreut durch

Prof. Dr. Klaus Brunnstein
Prof. Dr. Klaus von der Heide

Arbeitsbereich
Anwendungen der Informatik in
Geistes- und Naturwissenschaften
Universität Hamburg
Vogt-Kölln Straße 30
22527 Hamburg

## Zusammenfassung

Die vorliegende Arbeit befaßt sich mit der automatischen Suche nach Malware (Viren, Trojaner, etc.) im World Wide Web. Hierbei werden fortschrittliche Verfahren der Linkverfolgung (Crawling) sowie der heuristischen Textanalyse (Heuristik) verwendet. Des weiteren findet eine Analyse der Dependenzen der einzelnen Inhalte untereinander statt. Die Arbeit ist in einen theoretischen und einen praktischen Teil untergliedert. Im theoretischen Teil werden die Grundlagen der Heuristik und der Suchverfahren erschlossen. Im praktischen Teil wird die Implementation in der Programmiersprache Visual Foxpro diskutiert. Als Anlage zu dieser Arbeit existiert ein Programm (der sogenannte „AGN-Malware Crawler“), welches wesentliche Teile der in dieser Arbeit beschriebenen Funktionen vollführt.

## Abstract

This thesis is about the automatic search of Malware (Viruses, Trojans, etc.) in the World Wide Web. To acomplish this job there will be used advanced methods of crawling and text-analysis (heuristics). Furthermore there will be an analysis of dependencies between the pages. The thesis is divided in a theoretical part and a practical part. In the theoretical part the basics of the heuristic and of the search-process will be defined. In the practical part the realisation in the programming-language Visual Foxpro is the topic of discussion. This work is accomplished with the programme (the so-called „AGN-Malware Crawler“) that does some of the essential work discussed in this thesis.

# Danksagung

Mein Dank gilt allen, die sich die Zeit für die Ausfüllung des Fragebogens zur Heuristikverfeinerung genommen haben sowie denen, die die Diplomarbeit mit neuen Ideen und Vorschlägen angereichert haben. Ebenfalls danke ich auch denjenigen, die die orthographische Endkontrolle vorgenommen haben.

In alphabetischer Reihenfolge:

AGN-Firewall Team (Andreas Lessig, Karim Senoucci)
AGN-Netz Team (Karim Senoucci)
Dipl. Inform. Arslan Brömme
Prof. Dr. Klaus Brunnstein
Prof. Dr. Klaus von der Heide
Dipl. Inform. Stefan Kelm
Dipl. Inform. Markus Schmall

# Inhalt

# Abbildungsverzeichnis



# Abkürzungsverzeichnis

| | |
|---|---|
| AGN | - Arbeitsbereich „Anwendungen der Informatik in Geistes- und Naturwissenschaft“ |
| ASP | - Active Server Pages |
| BSI | - Bundesministerium für Sicherheit der Informationstechnik |
| CGI | - Common Gateway Interface |
| DHTML | - Dynamic HTML |
| CSS | - Cascading Style Sheets |
| HTML | - Hyper Text Markup Language |
| H/P/V/A | - Selbstbezeichnung einiger Malware-Sites (Hacking / Phreaking / Virii / Anarchy) |
| LAN | - Local Area Network |
| MWC | - Malware Crawler (Programm zu dieser Arbeit) |
| OCR | - Optical Character Recognition – Texterkennung aus Grafiken |
| PHP3 | - Serverbasierte Scriptsprache |
| URL | - Universal Resource Locator |
| VBS | - Visual Basic Script |
| VFP | - Visual Foxpro |
| VO | - Visual Objects |
| VTC | - Virus Test Center |
| WAN | - Wide Area Network |
| WWW | - World Wide Web |

# Legende

Text - Standard Text
`Text` - Programmcode oder HTML-Code
**`Text`** - Hervorgehobener Programmcode oder HTML-Code
*Text* - Zitate
**Text** - URL-Aufrufe

# 1. Einleitung

Diese Arbeit beschäftigt sich mit dem automatischen Auffinden von Webseiten, die maliziöse Inhalte zum Download anbieten. Hierbei werden verschiedene Verfahren der Textanalyse, Heuristik und Crawling-Techniken eingesetzt.

Bereits 1990 hat Dr. Alan Solomon in einem Whitepaper [SOLOMON90,S.3ff] folgende Aussage formal bewiesen:

> *Early detection is a very effective way to reduce the incidence of viruses in a population of computers. Reducing the probability of infection would also be useful, but this requires controls over the flow of diskettes and files between the computers, and one of the major advantages of computers is their ability to communicate information.*

Inzwischen ist der Austausch von Disketten immer mehr in den Hintergrund getreten – das Internet (insbesondere E-Mail, WWW, FTP und Newsgroups) wird für den Datenaustausch verwendet. Für die Distribution neuer Software werden fast ausschließlich CD-Roms benutzt, die durch ihre Beschaffenheit der nicht-Wiederbeschreibbarkeit fast keine Angriffsfläche für Viren bieten.[1] Kann man nun den maliziösen Inhalt von E-Mail Inhalten erkennen (womit sich derzeit die Antivirenhersteller beschäftigen) und den versehentlichen oder intentionalen Zugriff auf Websites, FTP-Server und Newsgroups mit maliziösen Inhalten verhindern, so werden beide Punkte (*„early detection“ und „control over the flow“*) abgedeckt.

Es gibt bereits Ansätze, bestimmte URLs zu blockieren – technisch ist dies kein Problem, da unter Windows-Systemen lediglich eine DLL[2] maskiert, und nicht direkt aufgerufen werden muß. Diese Funktion hat zum Beispiel das Programm SCAN bereits in der Corporate-Lizenz implementiert. Diese Programmoption von NAI-Scan enthält jedoch nur zwei [3] vordefinierte URLs [NAI]. Als eines der Produkte dieser Arbeit wird nun eine Liste derjenigen URLs entstehen, die maliziöse Software bereitstellen. Diese Liste könnte nun sehr leicht in die Produkte integriert werden [4].

## 1.1. Malware – Definition und Historie

Unter den Oberbegriff „Malware“ fällt jegliche Art von Software, die für den Verwender dieser Software nicht gewünschte Funktionen intentiell ausführt, beziehungsweise Funktionen ausführt, die das Programm laut seiner Dokumentation nicht vorgibt zu besitzen. Die ausgeführten Funktionen besitzen einen für den Benutzer der Software direkt oder indirekt schädlichen (engl „malicious“) Charakter, woraus sich der Begriff „Malware“ von „malicious software“ ableitet.

Unter dem Begriff „Malware“ werden die folgenden Typen von schädlichen Programm-Codes zusammengefaßt: Viren, Trojaner (Hostile-Applets, Bombs, Backdoors), Würmer (LAN, WAN).

[1] Sieht man einmal von der relativ selten vorkommenden Virendotierung beim Hersteller oder im Presswerk ab (wie es zum Beispiel bei Microsoft Demo-CD's vorgekommen ist).

[2] WSOCK32.DLL beziehungsweise WSOCK.DLL

[3] Stand Januar 2000, Scan-Version 4.0.4062 [NAI]

[4] Hierbei müßten jedoch die bestehenden Produkte ihre offene Anzeige der Liste ändern, da die Ergebnisse dieser Arbeit keinesfalls als Auskunftquelle für Virensuchende mißbraucht werden sollten.

Malware ist nicht auf ein bestimmtes Computersystem angewiesen – auf jedem programmierbaren Computersystem sind maliziöse Programme realisierbar. Trotzdem findet sich die meiste Malware auf den heutigen PC-Systemen, was nicht zuletzt an der leichten Zugänglichkeit dieser Systeme für jedermann liegt.[5]

Zur Geschichte der Malware findet sich auf der Website des BSI folgendes:

> *1980 verfaßte Jürgen Kraus am Fachbereich Informatik der Universität Dortmund eine Diplomarbeit mit dem Titel „Selbstreproduktion bei Programmen". In dieser Arbeit wurde zum ersten Mal auf die Möglichkeit hingewiesen, daß sich bestimmte Programme ähnlich wie biologische Viren verhalten können [...] 1984 veröffentlichte der Amerikaner Fred Cohen seine Arbeit mit dem Titel „Computer Viruses – Theory and Experiments* [BSI]

Fred Cohen definierte relativ unscharf als erster den Begriff „Computer - Virus":

> *A „computer virus" is a program that can „infect" other programs by modifying them to include a possibly evolved version of itself* [COHEN 94].

Eine schärfere formale Definition von Viren findet sich in der Doktorarbeit von Vesselin Bontchev. Der Begriff „Malware", der jegliche Art von intentiell schädlicher Software umfaßt, wurde von Prof. Dr. Klaus Brunnstein 1997 etabliert.

Für die Aufgliederung von Malware in Ihre Untertypen Viren, Würmer, Trojaner etc. sowie ihre speziellen Eigenschaften (Stealth, Anti-Debugging, Polymorphie, ...) sei auf die Arbeiten von Dr. Vesselin Bontchev (zum Beispiel auf der AGN-Website [VTC]) verwiesen.

[5] Zu Zeiten der Nutzung von Commodore Amiga Systemen fand sich auf diesen Systemen lediglich die Menge von ca. 500 maliziösen Programmen [FREITAG-AV]

## 1.2. Problematik von Malware

> *„...1986 erschienen zum ersten Mal auf IBM-kompatiblen Personalcomputern Computer - Viren..."* [BSI].

Seit dem ersten Erscheinen von Malware ist die Zahl der Viren[6] und die der anderen Malware nahezu exponentiell angestiegen.

Das aktuelle Problem: Webseiten, die jedermann Zugriff auf Malware bieten:

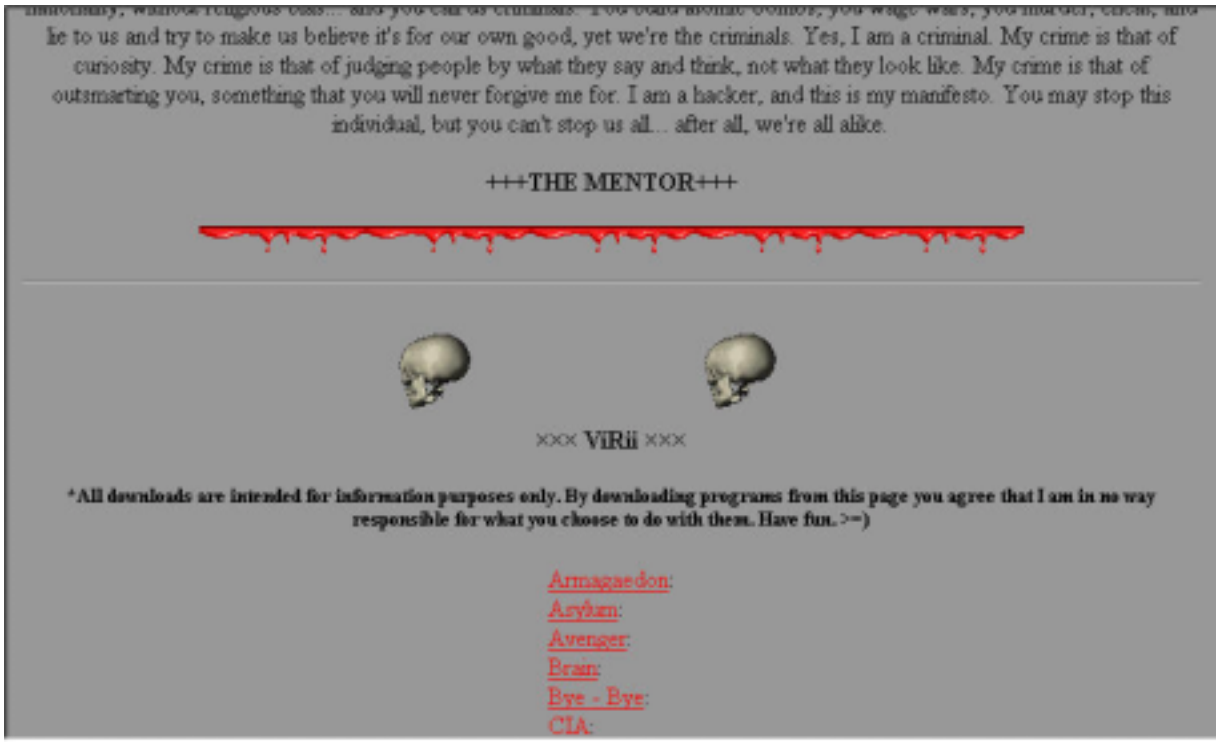



**Abbildung 1 – Malware-Site - Beispiel 1**

Während die Autoren von Malware diese meist ohne jegliche Einschätzung der Auswirkungen ihres Handelns in Umlauf bringen[7], erzeugen die maliziösen Programme in der Wirtschaft real monetär meßbaren Schaden. Dieser Schaden entsteht nicht nur durch die in einigen Malwaretypen implementierte Schadensfunktion[8], sondern auch durch die Kosten für den Ausfall durch präventive Außerbetriebsetzung von Geräten und Entfernung der Malware durch Techniker. Zusätzlich sind die Präventivkosten der Vorhaltung eines aktuellen Virenscanners und entsprechender Fachleute ebenfalls nicht unbeträchtlich.

[6] auf über 40.000, Stand 3/2000

[7] M.Schmall nennt in seiner Diplomarbeit *technisches Interesse* und *eine Form von Hobby* sowie eine Form von *elektronischem Wettkampf* und *Animositäten* gegenüber bestimmten Personen als mögliche Motivationsgründe für die Autoren [SCHM98,S.15].

[8] Mögliche Schadensfunktionen sind zum Beispiel Ausgabe falscher Daten auf Bildschirm oder Festplatte, Veränderung von Daten, Löschen von Daten, Übermittlung von Daten an Dritte,... [FREITAG-WP]

Nach Studien und Interviews (zum Beispiel von Sarah Gordon in [GORDON 94]) schreiben die verschiedensten Personengruppen Viren:

- (Meist) männliche Jugendliche, um ihren Freunden oder der Gesellschaft zu imponieren (zum Beispiel Tequila-Virus / Schweiz) – teilweise sind diese auch in sogenannten „Cliquen“ oder „Gruppen“ organisiert.
- Personen mit geschäftlichem Interesse (zum Beispiel Brain-Virus)
- Experimentierfreudige Programmierer
- Arbeitslose Programmierer (meist aus Ostblock-Ländern oder 3. Welt Ländern)
- Weltverbesserer etc.
- Mitläufer (zum Beispiel Clone-Produzenten, sogenannte „Script-Kiddies“ [SCHM98])

Die Beweggründe einen Virus zu schreiben sind dieser Aufstellung zufolge weit gestreut und das entsprechende Malware-„Produkt“ ist entsprechend den Fähigkeiten des Autors als gefährlich oder als harmlos einzuschätzen. In Viren - Analysen (zum Beispiel in Analysen des VTC) finden sich dementsprechend die verschiedensten Virentypen von einfachen überschreibenden Viren bis hin zu polymorphen Tarnkappenviren (zum Beispiel Crime 92, [FREITAG-CR])

Neuere Malware überlastet vermehrt auch Netzkomponenten und stört somit die Unternehmenskommunikation. So wird zum Beispiel der Schaden durch den am 4. Mai 2000 in Umlauf gekommenen „Love Letter“ Wurm von Experten (gemäß Internet World Newsletter vom 4.5.2000, der sich auf ddp/dpa Meldungen stützt [IW040500]), auf mehrere Milliarden US$ geschätzt. Betroffen waren zum Beispiel die niedersächsische Landesregierung mit 16.000 Rechnern, das britische Unterhaus, Siemens, Microsoft und weitere namhafte Unternehmen.

Im aktuellen Scannertest des Arbeitsbereichs AGN finden sich neben den Bewertungen der aktuellen Virenscanner hinsichtlich deren Erkennungsrate auch die Zahlen der im VTC-Labor verfügbaren Viren – hier ist insbesondere der starke Anstieg an Macroviren bemerkenswert:

*Table ES0: Development of threats as present in VTC test databases:*

| | = File viruses= | | = Boot Viruses= | | =Macro Viruses= | | == Malware == | |
|---|---|---|---|---|---|---|---|---|
| Test# | Number Viruses | Infected objects | Number viruses | Infected objects | Number viruses | Infected objects | file | macro |
| 1997-07: | 12,826 | 83,910 | 938 | 3,387 | 617 | 2,036 | 213 | 72 |
| 1998-03: | 14,596 | 106,470 | 1,071 | 4,464 | 1,548 | 4,436 | 323 | 459 |
| 1998-10: | 13,993 | 112,038 | 881 | 4,804 | 2,159 | 9,033 | 3,300 | 191 |
| 1999-03: | 17,148 | 128,534 | 1,197 | 4,746 | 2,875 | 7,765 | 3,853 | 200 |
| | + 5 | 146,640 | (VKIT+4*Poly) | | | | | |
| 1999-09: | 17,561 | 132,576 | 1,237 | 5,286 | 3,546 | 9,731 | 6,217 | 329 |
| | + 7 | 166,640 | (VKit+6*Poly) | | | | | |
| 2000-04: | 18,359 | 135,907 | 1,237 | 5,379 | 4,525 | 12,918 | 6,639 | 39 |
| | + 7 | 166,640 | (VKit+6*Poly) | | | | | |

*Remark: Before test 1998-10, an ad-hoc cleaning operation was applied to remove samples where virality could not be proved easily. Since test 1999-03, separate tests are performed to evaluate detection rates of VKIt-generated and selected polymorphic file viruses.*

> *With annual deployment of more than 5,000 viruses and several 100 Trojan horses, many of which are available from Internet, and in the absence of inherent protection against such dysfunctional software, users must rely on AntiMalware and esp. AntiVirus software to detect and eradicate - where possible - such malicious software. Hence, the detection quality of AntiMalware esp. including AntiVirus products becomes an essential prerequisite of protecting customer productivity and data.* [BRU2000]

In der folgenden Abbildung nun ein weiteres Beispiel einer Website, die Malware zum Download anbietet. Diese Seite zeigt die Möglichkeit, gleichgeartete Seiten durch Verfolgung der angebotenen Hyperlinks zu finden:

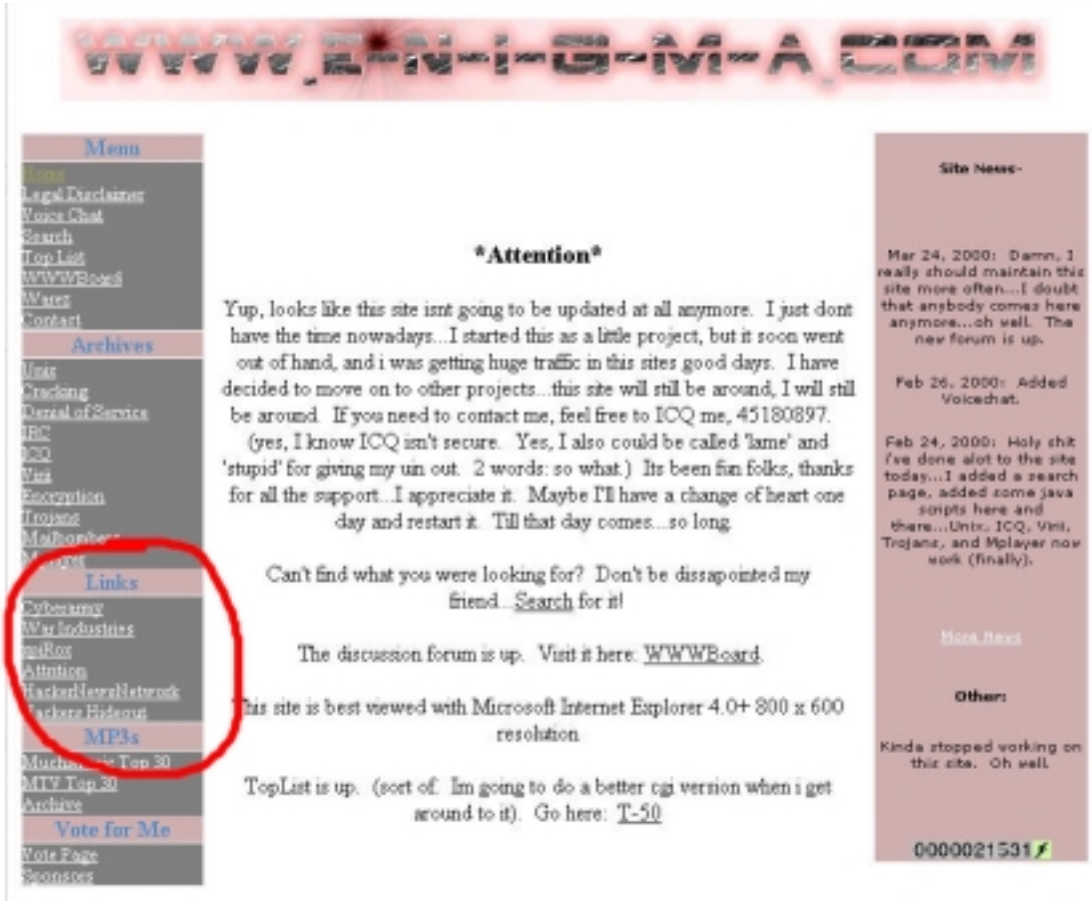



**Abbildung 2 - Beispiel einer Malware-Site mit Hyperlinks**[9]

Die (hier rot eingekreist) dargestellten Hyperlinks sind - neben den enthaltenen maliziösen Programmen - der interessante Part dieser Seiten, auf denen der Malwarecrawler aufsetzt.

[9] Die Seite wurde zum Sparen von Toner - Patronen farblich invertiert wiedergegeben.

# 2. Zielsetzung und Methoden

Bisherige Suchengines sind aufgrund Ihrer Konzeption auf eine Abdeckung jeglicher Suchanfragen im World Wide Web ausgerichtet. Selbst die größten Indizes (Inktomi, Fast, AltaVista) der weltweit mehr als tausend Suchengines [CT 98/13] decken nicht einmal 50% der verfügbaren Webseiten ab[10]. Hinzu kommt noch, daß die Viren-Websites meist gar nicht gefunden werden wollen, also sich nicht selbst in den Suchengine-Websites eintragen und die Robots Exclusion Standards [EXCL] benutzen, um die Crawler der Suchengines vom Indizieren der Seiten abzuhalten. Die URLs werden dann in IRC-Channels oder per E-Mail ausgetauscht, so daß ein Auffinden dieser URLs noch schwieriger ist als eine herkömmliche Internet-Recherche.

Folgende von „Search Engine Watch“ [SEW] entnommene Grafiken verdeutlichen dieses:

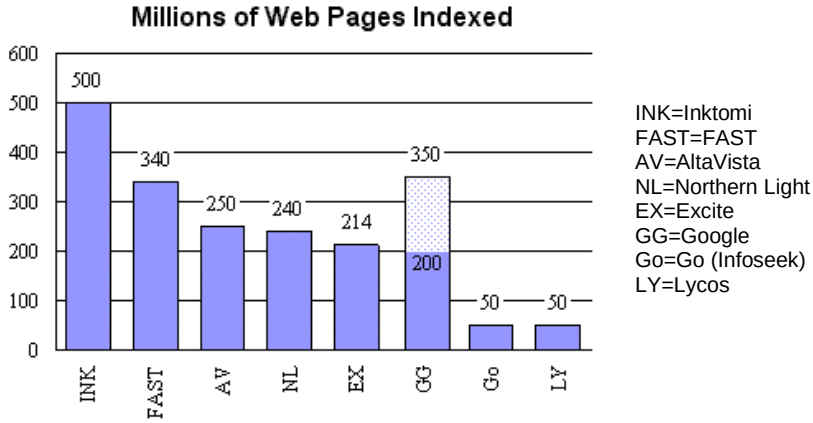



**Abbildung 3 – Anzahl der von Suchmaschinen indizierten Webseiten**

Hinzu kommt der hohe Anstieg an Webseiten, welcher sich nicht zuletzt auch in den Größen der Indizes wiederspiegelt:

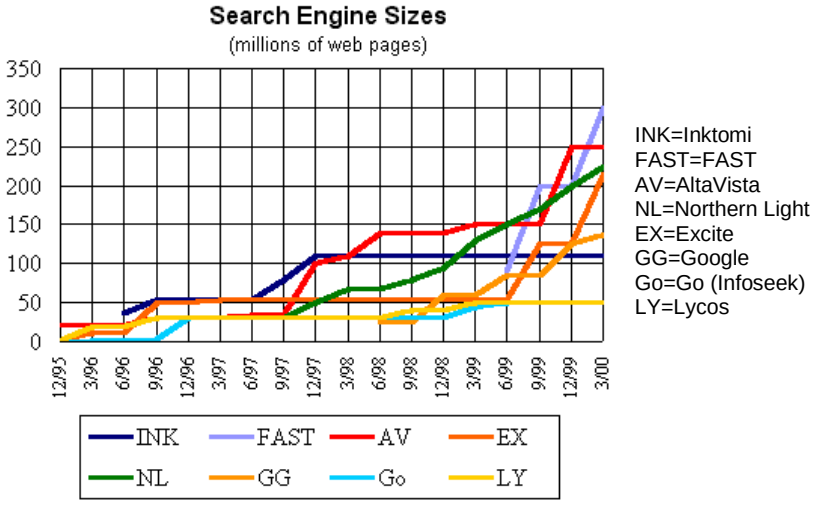



**Abbildung 4 –Anstieg der Suchengine - Verzeichnisgrößen**

[10] [SEW] schätzt, daß ca. 1 Milliarde Websites im April 2000 existierten

Nach einer Studie von 1998 [CT 98/13] decken die besten Suchengines dort sogar nur maximal 34% Prozent der im Web befindlichen HTML-Dokumente ab. Mit Hilfe einer kombinierten Suche über mehrere Suchengines (Meta-Suche[11]) kann das Ergebnis weiter verbessert werden. ([CT 98/13] schreibt hier einen Faktor von 3.5%, der allerdings als unrealistisch erachtet wird.)

Man kann im wesentlichen zwischen folgenden Suchengines und Suchverfahren unterscheiden:

**Indexbasierte Suchengines (Automaten und Kataloge)**

Indexbasierte Suchengines nehmen in ihrer Datenbank die textuellen Inhalte der besuchten Webseiten auf. Hierbei können entweder die realen Textinhalte oder die sogenannten Meta-Tags[12] Verwendung finden. Anhand eines spezifisch für die jeweilige Suchengine erstellten Lexikons werden die irrelevanten „Rausch"-Wörter (zum Beispiel: „and", „und", „oder", etc.) aussortiert. Ein prominenter Vertreter für diese Art Suchverfahren ist die Suchengine Altavista (**www.altavista.com**).

**Online-Meta-Suchengines**

Die Online-Meta-Suchengines fragen parallel indexbasierte Suchengines (und teilweise auch Verzeichnisdienste) ab, und zeigen deren Ergebnisse auf ihren eigenen Seiten. Durch die Berücksichtigung mehrerer Suchengines werden meist bessere Ergebnisse erzielt als durch die Abfrage von nur einer einzelnen Suchengine. Ein Nachteil sind jedoch die reduzierten Möglichkeiten bei der Suche logische Verknüpfungen einzusetzen. Ein Vertreter dieser Gattung ist zum Beispiel der Dienst MetaGer[13].

**Desktop-Meta-Suchengines**

Die Technik der Online-Meta-Suchengines wurde nahezu identisch in Desktop-Programme implementiert, die ausgehend vom Computer des Anwenders, die einzelnen Suchengines abfragen und dann dort die Ergebnisse verarbeiten. Anders als bei Online-Meta-Suchengines erfolgt hier die Verarbeitung nicht auf dem Rechner des Anbieters sondern auf dem Rechner des Suchenden). (Beispiele für ein derartiges Programm sind „Webplanet Tools" von **www.webplanet.com** und „SSSpider" von Fa. Kryltech (**www.kryltech.com**)).

**Verzeichnisdienste**

Verzeichnisse enthalten manuell katalogisierte Verweise zu Seiten im Internet. Die Qualität und Präzision der von den Verzeichnissen auf eine Suchanfrage zurückgelieferten Websites ist naturgemäß besser, da eine manuelle Vorsortierung stattfindet. Verzeichnisdienste können jedoch aufgrund des starken Anstiegs der Anzahl an Webseiten nicht mithalten und verweisen so auf einen wesentlich geringeren Teil des Internets als indexbasierte Suchdienste.

Ein prominenter Vertreter für Verzeichnisdienste im WWW ist **www.yahoo.com**. Verzeichnisse sind aufgrund der manuellen Vorsortierung für diese Arbeit kein geeigneter Ansatzpunkt, da dort kaum Viren - Sites aufgeführt sind. Für ähnlich gelagerte Arbeiten der Internetrecherche sollten diese jedoch an aller erster Stelle ausgewertet werden.

Verteilte Systeme (wie beispielsweise Harvest) konnten sich bisher nicht durchsetzen und spielen derzeit eine untergeordnete Rolle. (Zum Beispiel in einer Testinstallation der Universität Oldenburg: www.gerhard.de).

[11] zum Beispiel www.metacrawler.com oder www.dogpile.com

[12] Bestimmte unsichtbare Texte (Schlüsselworte und Beschreibungen) in der HTML-Codierung, die speziell für die Suchengines eingebracht werden.

[13] http://meta.rrzn.uni-hannover.de

Eine Auflistung und einen Vergleichstest der aktuellen Suchengines findet sich in der Zeitschrift C't [CT 99/23, S.170f]. Der Artikel der C't betrachtet hierbei speziell die deutschen Verzeichnisdienste und Suchengines.

Auf der Website von „Search Engine Watch" [SEW] befindet sich eine nahezu komplette Aufstellung aller allgemeinen Suchengines. Eine weitere Auflistung aller Suchengines hat T. Koch [KOCH99] auf seinen Seiten veröffentlicht, wobei hier die einzelnen Suchengines in Ihrer Funktion beschrieben werden und nach Themengebiet und Suchart katalogisiert sind.

Ein etwas älterer Artikel, der jedoch detaillierter die Fähigkeiten der einzelnen Suchengines analysiert, befindet sich ebenfalls auf den Webseiten von T.Koch [KOCH96]. Hier werden auch Simultan-Suchengines, regional begrenzte Suchengines und Suchengines für bestimmte Informationen Berücksichtigt (siehe [KOCH96_2]).

Weiterführende Informationen über die Leistungsfähigkeit der einzelnen Suchengines sowie Check-Services für Metatags, Quelltexte, Suchparameter der Engines etc. wurden auf der Website von Marc Bauer [MARB] veröffentlicht.

## 2.1. Zielsetzung

Filippo Menczer et.al. schreiben in einer Publikation über adaptive Agenten im Internet [MENCZER_ADP], daß das World Wide Web (WWW) eine Informationsumgebung ist, welche aus einer sehr großen verteilten Datenbank von heterogenen Dokumenten besteht. Diese Datenbank benutzt ein Wide-Area Netzwerk (WAN) und ein Client-Server Protokoll. Die Struktur dieser Umgebung[14] ist die eines Graphen, in dem die Knoten (Dokumente) durch Hyperlinks verbunden sind. Die typische Strategie, um auf Informationen im WWW zuzugreifen, ist, über die gesetzten Hyperlinks zu navigieren.

In einer Studie aus dem Jahr 1999 von der Fakultät Physik an der Universität Notre Dame („*Diameter of the World Wide Web",* Indiana [BARABASI]) werden nur maximal 19 Querverbindungen im WWW benötigt, um über Hypertext-Links auf dem kürzesten Weg zu einer gewünschten Information beziehungsweise Seite zu gelangen[15]. Dieses liegt daran, daß nahezu jede Website mit weiterführenden Hyperlinks ausgestattet ist. Hierbei wurde beobachtet, daß inhaltlich und geographisch verwandte Themen häufiger verlinkt werden als weit entfernte Webseiten. Selbst bei einer angenommenen Zuwachsrate von 1000% der Webseiten würde nach Barabási dieser Distanzwert lediglich von 19 auf 21 Hyperlinks ansteigen.

Ziele dieser Arbeit sind die Bereitstellung der theoretischen und teilweise auch praktischen Methoden für

- das Auffinden von Websites, die bekannte und neue Malware zum Download bereitstellen. (Meistens zusammengefaßt unter dem Begriff H/P/A/V = Hacking, Phreaking, Anarchy, Virii Sites)
- die Zuführung neuer Malware zu der VTC-Datenbank.
- die Bereitstellung von Daten, die zu einem späteren Zeitpunkt statistisch ausgewertet werden können (zum Beispiel für Dependenz-Graphen oder Aktivitätsprotokolle).
- die Kontrolle anhand der URL-Liste, welche Malware-Sites vom Netz entfernt wurden.

[14] „environment" kann in diesem Zusammenhang nur schwer übersetzt werden.

[15] Ein entsprechender Web-Roboter zur Verifizierung dieser Theorie wurde von Albert-László Barabási an der Universität Notre Dame entwickelt. [BARABASI], [ND]

- die Erstellung einer Liste von URLs, die maliziöse Inhalte anbieten (zum Beispiel für universitäre WWW-Ausschlußlisten).

Hierbei sind die Erkenntnisse der Querverweise der Universität Notre Dame von großem Interesse. Wenn wirklich nahezu jede Seite über Hyperlinks erreichbar ist, so sind auch die Virenseiten prädestiniert für eine derartige Verknüpfungstheorie. Findet man nun zum Beispiel durch eine gängige Suchmaschine einen Einstiegspunkt in einen entsprechenden Teil-Graphen der, thematisch nah an dem gewünschten Suchergebnis liegt, so kann man sich nach dieser Theorie anhand der Hyperlinks mit wenigen „hops“[16] komplett zu allen Seiten bewegen, die im identischen Interessengebiet liegen. Dieses ist allerdings aufgrund der Größe des WWW-Raumes ein sehr zeitaufwendiges Unterfangen (Albert-László Barabási [BARABASI] schätzt so (abweichend von den in Kapitel 2. genannten aktuellen Zahlen von [SEW]), daß 1999 mindestens $8x10^8$ Dokumente im WWW existieren ).

Eine manuelle Auswertung dieser Teilgraphen und eine manuelle Linkverfolgung ist dementsprechend kaum möglich. In dieser Arbeit wird ein automatisches Verfahren beschrieben, das genau diese Aufgabe vollführt.

## 2.2. Bestehende Verfahren

Die Schwächen von bestehenden Standardverfahren der Internetrecherche (Suchengines, Meta-Suchengines und Verzeichnisse) wurden bereits in Kapitel 2 erläutert. Nachfolgend werden die Verfahren betrachtet, die derzeit zur systematischen Virensuche eingesetzt werden können.

***Manuelle Suche***

Ausgehend von bekannten Virenseiten werden Hyperlinks verfolgt, welche zu weiteren potentiell Malware enthaltenden Seiten führen. Ebenfalls wird den Suchengineergebnissen für einschlägige Schlagworte wie zum Beispiel „virii“ manuell durch Linkverfolgung nachgegangen. Dieses Verfahren ist personalintensiv und ineffektiv.

Personalintensiv sind diese Verfahren aufgrund der Anzahl der zu durchsuchenden Websites.

Ineffektiv ist dieses Verfahren, da eine einzelne Person aufgrund der Menge an Daten nicht alle Hyperlinks verfolgen kann. Wird diese Aufgabe hingegen auf mehrere Personen aufgeteilt, so besuchen diese Personen zwangsweise entweder dieselben Webseiten mehrfach, oder sie müssen sich vor jeder Link-Verfolgung von einem zentralen Katalog die Gewißheit verschaffen, daß die gewünschte Seite noch nicht besucht wurde.
Der mehrfache Besuch einer entsprechenden URL resultiert hierbei aus der Redundanz der verschiedenen Suchenginekataloge und der von Virenseiten weiterführenden Hyperlinks.

[16] Web-Ausdruck für das Verfolgen eines Hyperlinks.

***Automatische Suche***

Es besteht nach Kenntnisstand des Autors nur ein einziges Verfahren speziell zur Virensuche im Internet (von Network Associates), welches sich auf ein Scannen der Binärfiles in Newsgroups mit dem eigenen Virenscanner[17] beschränkt. Eine heuristische Bewertung findet hier nur auf der Ebene der binären Files statt, die in der Newsgroup gepostet wurden. Diese Bewertung wird zudem lediglich mit der eigenen Heuristik Scan-Engine vorgenommen. Der WWW-Raum, der Ziel dieser Arbeit ist, wird bei diesem Programm nicht untersucht.

Bestehende universelle Crawler-Verfahren suchen zudem meist nur nach einem Schlüsselwort im Text, beziehungsweise lediglich nach einfachen UND / ODER-Verknüpfungen von wenigen Schlüsselworten. Die ebenfalls aussagekräftigen Informationen, wie zum Beispiel der Verweis auf eine Seite, die bereits als relevant identifiziert wurde, die Domain der Seite sowie mehrere Schlüsselwörter mit einer Bewertung in Abhängigkeit ihres gemeinsamen Auftretens, werden nicht berücksichtigt. Eine mehrstufige Heuristik, wie sie in dieser Arbeit beschrieben wird, findet sich nur ansatzweise bei einem Personen-Suchdienst[18].

Eine nahezu vollständige Liste von 226 Crawlern / Robots unterschiedlichster Ausprägung findet sich bei: **http:// info.webcrawler.com/mak/projects/robots/active/html/index.html**

## 2.3. Methoden

> *If one is not sure what one is looking for, browsing works best, and Yahoo presents the right approach. If one is looking for unusal words or names or wants everything known about something, then the spider-based engines cannot be beaten. - Udi Manber* [MANBER_USI]

Der in dieser Arbeit realisierte Web-Roboter (kurz MWC – Malware Crawler) arbeitet sukzessive automatisch vorgegebene URLs ab und bewertet diese anhand der mehrstufigen Heuristik. Hierbei schreitet der MWC anhand der innerhalb der URL gefundenen Hyperlinks auf weitere Seiten selbständig durch das WWW, wobei er autark die Bewertung vornimmt, welche Hyperlinks für ihn relevant (und somit zu verfolgen) sind. In den Kapiteln 3 und 4 der Arbeit wird auf die beiden wesentlichen Teile „Crawling“ und „Heuristik“ näher eingegangen. Zunächst jedoch folgt die Betrachtung der zum Einsatz kommenden Systeme.

## 2.4. Vorbetrachtungen

Für die Realisierung der Arbeit wird ein entsprechend zuverlässiger und schneller Internetanschluß benötigt, wie dieser an Universitäten allgemein üblich ist. Ferner soll das Programm auf einem handelsüblichen PC installiert werden. Der PC wird über einen Firewall-Rechner direkt an das Internet angeschlossen. Der Firewall blockiert alle Zugriffe auf den MWC-Rechner, die nicht auf den benötigten definierten Ports stattfinden. Das Crawler - Programm benötigt die Ports:

| | |
|---|---|
| **80, 443, 8080** | (HTTP, HTTPS, HTTP auf Port 8080 zum Aufrufen der Web-Seiten) |
| **20, 21** | (FTP für Dateiübertragungen) |
| **110** | (POP3 zum Empfangen von E-Mails für Wartungszwecke) |
| **25** | (SMTP zum Versenden von E-Mail Warnungen) |

[17] Das Programm lag beim Verfassen dieser Arbeit nicht vor und wird von NAI nicht herausgegeben, weswegen hier keine genaueren Angaben über die Funktionsweise erfolgen können.

[18] „Ahoy“ der Universität Washington

Alle weiteren Ports werden vom Firewall blockiert. Ein entsprechender Firewall - Rechner wurde am 22.12.1999 im AGN-Labor aufgesetzt. Vorhergehende Tests von anderen PC's haben ergeben, daß einige Anbieter von Malware im Netz auf den Besuch eines Crawlers durch sogenanntes „Backfire" reagieren – das heißt, daß diese Seiten gezielt den Rechner, auf dem der Crawler beheimatet ist, über die IP-Verbindung angreifen[19]. Der eingerichtete Firewall verhindert die meisten solcher Angriffe.

## 2.5. Benötigtes System und Programmiersprache

Da sich auf dem für Internetprogramme eigentlich prädestinierten Betriebssystem Linux keine einfach handhabbaren Datenbanksysteme anbieten[20], wird der Malware Crawler für ein Windows NT System konzipiert. Eine verteilte Lösung ,zum Beispiel auf Basis von SQL-Serversystemen mit mehreren Clienten wie zum Beispiel moderne Suchengines ausgestattet sind, ist im universitären Kontext aus Gründen des hohen Resourcenbedarfs nicht machbar. Ein Vorteil der Entscheidung für eine Einzelplatz - Windows NT Lösung ist, daß nahezu jeder am Netz angeschlossene PC innerhalb kürzester Zeit zum Suchen von Malware eingesetzt werden kann.

***Vorbedingungen für die Programmiersprache:***

- Die benötigte Programmiersprache muß in der Lage sein, große Datenmengen schnell zu verarbeiten.
- Die benötigte Programmiersprache muß in der Lage sein, Datensätze schnell aufzufinden und von der Festplatte zu lesen.
- String-Operationen müssen unterstützt werden.
- Die Programmiersprache muß lauffähig auf Windows-Systemen sein.

***Nachstehend die Abwägung der Programmiersprachen:***

| Programmiersprache | Bemerkungen |
|---|---|
| Access | Möglicher Kandidat, Datenbank |
| C++ | Keine native Datenbankunterstützung, schnell |
| Clipper | Dos – nicht NT geeignet |
| Delphi | Keine Datenbankunterstützung (wenn man von BDE absieht) |
| Dbase | Ungewisse Zukunft, langsam |
| Java | Langsam, keine native Datenbankunterstützung (nur JDBC) |
| Perl | Windows-GUI nicht unterstützt, Sprache dem Autor nicht geläufig |
| Visual Objects 2.5 | Möglicher Kandidat, schnell, Datenbank |
| Visual Foxpro 6 | Möglicher Kandidat, schnell, Datenbank |
| Visual Basic | Langsam, keine native Datenbankunterstützung |

Also kämen Visual Objects 2.5, Visual Foxpro 6 und Access in Frage. Von Access sind Instabilitäten und Performanceprobleme bekannt. Bei Visual Foxpro ist aus Veröffentlichungen zu entnehmen [DFPUG], daß dieses Produkt Datenmengen von immensen Größen (zum Beispiel die Daten des Eurotunnel-Projekts [CIS-VFP]) verwalten kann. Aufgrund der Beschäftigung mit Visual Objects und Visual Foxpro wurde dann die Wahl Visual Foxpro getroffen, da hier die Microsoft-eigenen Controls für Internetzugriffe leicht einbindbar erschienen[21].

[19] Zum Beispiel durch bekannte Schwächen im Windows - Kernel (Ping of Death, Teardrop, Netbios-Fehler)

[20] Eine Realisierung in Perl und mySQL hätte den Programmieraufwand zumindest verdoppelt..

[21] Dieses stellte sich während der Realisierungsphase allerdings als Trugschluß heraus (vgl. Kapitel 2.6)

### 2.5.1. Strukturen der Datenvorhaltung

Visual Foxpro (VFP) ist eine relationale Datenbanksprache. Als eine solche bietet VFP die nötigen Strukturen für die Datenvorhaltung an wie zum Beispiel Tabellen, freidefinierbare Indizes, Relationen (1:n, n:m). Ebenfalls sind die Basisfunktionen für Textmanipulation und Textauswertung in Form von Stringoperationen (Vergleich, Pattern-Matching, Stringextraktion,...) vorhanden. Die anfallenden URL-Daten des Malware Crawlers werden komplett in VFP-Tabellen gespeichert, was diese schnell durchsuchbar und verwaltbar macht.

Auf die Syntax und die Befehle von Visual Foxpro soll hier nicht weiter eingegangen werden – an den entsprechenden Stellen werden die Codesamples einzeln dokumentiert.

Literatur zur Programmierung in Visual Foxpro findet sich in den Büchern: [ANTONOVICH], [GRIVER], [MS-PRESS], [MS-VFP6], [MS-VFP5], [MS-VFP3], [MS-FP26] sowie in den Foren / Newsgroups / Webforen [CIS-VFP], [DFPUG], [MSDN], [MS-FPG], [MS-MAPI], [UT].

## 2.6. Eingesetzte Tools

In der Konzeption der Arbeit und in der ersten Realisation des MWC wurde das beim Internet Explorer beigefügte IE-Control der Firma Microsoft zum Download der URLs mittels des HTTP-Protokolls benutzt. Dieses Control stellte sich in Last-Tests jedoch als unzuverlässig und fehlerhaft heraus[22]. Die in diesem Tool vorhandenen Funktionen (zum Beispiel zum Download einer URL und zur Wandlung HTML->Text etc..) konnten so nicht wie geplant benutzt werden.

Als Ersatz für dieses Control wurde die WWIPSTUFF-DLL der Firma West-Wind verwendet [WWIND]. Dieses Control stellte jedoch lediglich eine Methode für den HTTP-GET Request (also das Download einer HTML-Seite) bereit. Die Extraktion des Textes und weitere komfortable Funktionen des IE-Controls mußten deshalb in Visual Foxpro nachprogrammiert werden. Dieses führte zu einer Verlangsamung der Extraktion – der MWC wurde aber dadurch wesentlich ausfallsicherer. Eventuell kann in Zukunft mittels der Umsetzung zeitkritischer Routinen in C++ die Geschwindigkeit des MWC weiter gesteigert werden.

Das jetzt eingesetzte Control zeigte während weiterer Tests Probleme im Umgang mit dem „Chunked“ HTTP - Transfer-Encoding Format[23], so daß eine weitere Umcodierung dieses Parts notwendig wurde. Die wesentlichen Teile der aktuellen (inzwischen dritten) Code-Implementation der Get-Routine wird nachfolgend aufgelistet, wobei die DLL-Aufrufe gesperrt gedruckt sind.

[22] Das Programm funktionierte – hatte aber unerklärliche Memory-Resource-Leaks und Abstürze nach wenigen gecrawlten Webseiten.

[23] Das Chunked HTTP-Format ist zum Beispiel auf der Malware-Site www.attriton.org zu finden.

```
&& ---------------------------------------------------------------------------------
&& URL laden und HTML-Sourcecode in Formular stellen
&& Input:              Formular Property tx_starturl
&& Direkter Output:   .T. - Erfolg, .F. – Fehler,
&& Indirekter Output:  Formular Propertys HTML-Ausgabe und HTTP-Headers
&& Fr. 12/99
&&
&& ---------------------------------------------------------------------------------

PRIVATE lcURL                                                  && URL-Teil
PRIVATE lcPath                                                 && Pfad in dieser URL
PRIVATE lcHTML                                                 && HTML-Output Buffer
PRIVATE lnText                                                 && Buffer-Größe
LcURL=""
LcPath=""
lcHTML=""
lnText=0

IF AT("/",THISFORMSET.mwc_form1.pg_frames.pg_crawler.;
      tx_starturl.Value,3)=0                                   && Unterpfad vorhanden ?
   lcURL=THISFORMSET.mwc_form1.pg_frames.;
      pg_crawler.tx_starturl.Value                             && Nein
ELSE
   lcURL=LEFT(THISFORMSET.mwc_form1.pg_frames.;
      pg_crawler.tx_starturl.Value,AT("/",THISFORMSET.;
         mwc_form1.pg_frames.pg_crawler.;
         tx_starturl.Value,3)-1)                               && Unterpfad extrahieren
      lcPath=STRTRAN(THISFORMSET.mwc_form1.pg_frames.;
         pg_crawler.tx_starturl.Value,lcURL,"")                && URL-Pfad entfernen
ENDIF
lcURL=STRTRAN(lcURL,"http://","")                              && und HTTP-Header entfernen
lcURL=STRTRAN(lcURL,"https://","")                             && und HTTPS-Header entfernen

oWwip.HTTPConnect(lcURL)                                       && Server connecten

lnResult =oWwip.HTTPGetEx(lcPath,@lcHTML,@lnText)              && HTML holen

IF lnResult # 0
   THISFORMSET.mwc_form1.pg_frames.pg_crawler.;
      Outputframe.Page1.Edit1.VALUE="Error " +;
      ALLTRIM(STR(lnResult))+" "+oWwip.cErrorMsg               && Ausgabe Fehler in Property
      llRetval=.F.
ELSE
   THISFORMSET.mwc_form1.pg_frames.pg_crawler.;
      Outputframe.Page1.Edit1.VALUE=ALLTRIM(lcHTML)            && Ausgabe HTML in Property
   THISFORMSET.mwc_form1.pg_frames.pg_crawler.;
      Outputframe.Page6.Edit1.VALUE=;
      ALLTRIM(oWwip.cHTTPHeaders)                              && Server-HTTP-Header eintragen

   oWwip.HTTPClose()                                           && Connection schließen

   llRetval=.T.
ENDIF
RETURN llRetval
```

## 2.7. Objektkatalog der Basisklasse des MWC

**Class : cmain** (abgeleitet aus: custom)

Beschreibung : Hauptklasse des Malware-Crawlers

Methods created in class cmain

| Methoden | Beschreibung |
|---|---|
| mvrun() | Error-Handling und Main-Loop |
| Mvshutdown() | Shutdown des Systems |

**Class : link**
Based on : Hyperlink
Beschreibung : Hyperlink zum Aufruf von URLs

**Class : mailbtn**
Based on : container
Beschreibung :E-Mail Versand via MAPI

Properties created in class mailbtn

| Eigenschaften | Beschreibung |
|---|---|
| Logsession | Session ID |
| Messagetext | Text der Message |
| Messagesubject | Subject der Message |
| Recipaddress | Empfänger (vrawl@informatik.uni-hamburg.de) |
| Recipdisplayname | Name des Empfängers |

Methods created in class mailbtn

| Methoden | Beschreibung |
|---|---|
| Signon() | MAPI Session öffnen |
| Strippath() | MAPI Filepfad bereinigen für Attachments |

**Class : mycombo**
Based on : combobox
Beschreibung : Combo-Box

Properties created in class mycombo

| Eigenschaften | Beschreibung |
|---|---|
| sf_helpid | ID für Online - Hilfe |
| sf_reftab | Eingabehilfe |
| sf_tooltip | Tooltip-Text |

**Class : textbx**
Based on : textbox
Beschreibung : Texteingabefeld

Properties created in class textbx

| Eigenschaften | Beschreibung |
|---|---|
| sf_helpid | Hilfe - ID Name für Online-Hilfe |
| sf_reftab | Eingabehilfe |
| sf_tooltip | Tooltip-Text |

**Class : gauge**
Based on : form
Beschreibung : Wartezähler

Properties created in class gauge

| Eigenschaften | Beschreibung |
|---|---|
| sf_helpid | Hilfe - ID Name für Online-Hilfe |
| sf_reftab | Eingabehilfe |
| sf_tooltip | Tooltip - Text |

Methods created in class gauge

| Methoden | Beschreibung |
|---|---|
| sf_step() | Prozentzahl und Bar weiterschieben |

## 2.8. Objektkatalog des MWC Basisformulars

**Form : mwc** Beschreibung: MWC Basisformular

Properties created in form mwc

| Eigenschaften | Beschreibung |
|---|---|
| scanlistpos | Position des Satzzeigers in der Scanlist |
| extlistpos | Externe Scanliste - Position des Satzzeigers |
| filelistpos | Position des Satzzeigers in der Filelist |
| nrecnoint | Record-Nummer des Grids für die interne Linkliste |
| nrecnoext | Externe Linkliste - Recordnummer |
| nrecnofile | Filelist – Recordnummer |
| keywords_obj_id | Objekt-Nummer (Schlüssel) in Keywords-Tabelle |
| qualify_obj_id | Objekt-Nummer (Schlüssel) in Qualify-Tabelle |

Methods created in form mwc

| Methoden | Beschreibung |
|---|---|
| sf_htm2txt() | HTML-Text in Rohtext verwandeln |
| sf_htmlinks() | Links aus einem HTM-Text extrahieren |
| sf_quotsonly() | Text in Anführungszeichen aus einem String holen |
| sf_urlonly() | URL vom Pfad befreien |
| sf_onlydir() | Dokument-Link aus URL entfernen (nur Directory-Pfad + "/" ) |
| sf_addextlinks() | Externe Links in der Linkliste eintragen, sofern sie nicht in der Goodlist stehen. |
| sf_addlinks() | Internen Link in die Liste aufnehmen |
| sf_heur() | HEURISTIK – Routine |
| sf_killdots() | Rekursive Verzeichnisse a'la ../../ beseitigen |
| sf_geturl() | URL incl. Header etc... holen |

# 3. Linkverfolgung (Crawling)

In dieser Arbeit wird „Crawler“ in Bezug auf Zugriffe im World Wide Web synonym zu „Spider“ beziehungsweise „Robot“ betrachtet. In einigen Arbeiten finden Abgrenzungen statt – in anderen werden diese Begriffe ebenfalls als identisch betrachtet.

Nachstehend eine Definition von „Crawler“: der Website whatis.com:

> *A crawler is a program that visits Web sites and reads their pages and other information in order to create entries for a search engine index. The major search engines on the Web all have such a program, which is also known as a "spider" or a "bot." Crawlers are typically programmed to visit sites that have been submitted by their owners as new or updated. Entire sites or specific pages can be selectively visited and indexed. Crawlers apparently gained the name because they crawl through a site a page at a time, following the Hyperlinks to other pages on the site until all pages have been read.* [WHATIS]

Eine weitere Definition liefert Martijn Koster 1995:

> Ein Web-Roboter ist ein Programm, welches die Web- Hypertextstruktur durchwandert und Dokumente sowie rekursiv alle referenzierten Dokumente lädt. Diese Programme werden auch „spiders“, „web wanderers“ oder „web worms“ genannt. Diese Namen, obwohl sie besser wirken, sind irreführend weil, die Termini „spider“ und „wanderer“ den falschen Eindruck erwecken, daß der Roboter selbst sich bewegt. Der Begriff „worm“ könnte implizieren, daß der Roboter sich selbst vervielfältigt (wie z.B. der Internet Wurm). In der Realität werden Roboter als einzelne Softwaresysteme implementiert, die Informationen von entfernten Sites durch Benutzung von standard Web-Protokollen abrufen. [KOSTER95, übersetzt aus dem Englischen]

Zusätzlich ordnet Koster [KOSTER95] die verschiedenen Begriffe:

> *So what are Robots, Spiders, Web Crawlers, Worms, Ants*
> *They're all names for the same sort of thing, with slightly different connotations:*
>
> | | |
> |---|---|
> | *Robots* | *the generic name [...]* |
> | *Spiders* | *same as robots [...]* |
> | *Worms* | *same as robots, although technically a worm is a replicating program, unlike a robot.* |
> | *Web crawlers* | *same as robots, but note WebCrawler is a specific robot* |
> | *WebAnts* | *distributed cooperating robots.* |

Zu der Begriffsklasse der Robots könnte man auch noch noch die Begriffe „Web Walker“, „Bots“, „Web Wanderer“ hinzuzählen.

In dieser Arbeit soll nachstehende Definition verwendet werden:

**Crawler** (synonym zu „Spider“, „Robot“, „Web Wanderer“)
(in Verwendung mit Internet-Zugriffen)

> Computer – Programm, welches auf einem Start-Dokument aufgesetzt, durch sukzessive, rekursive Verfolgung von Verweisen (sogenannten Hyperlinks) in Dokumenten des WWW-Raumes des Internets referenzierte Dokumente einlädt und verarbeitet, um die entsprechenden Inhalte einer weiteren Verwertung zuzuführen.

Häufige Arten der Verarbeitung sind Indizierung in Form von Suchengine-Einträgen oder Extraktion bestimmter Inhalte der Dokumente.

Auf der BotSpot-Website [BOTSPOT] findet sich eine detaillierte Aufgliederung der verschiedenen „Bots“ (auf der BotSpot Website werden jedoch nicht nur „Bot“‘s im WWW-Raum betrachtet, sondern unter anderem auch Programme im E-Mail / News - Bereich):

| | |
|---|---|
| Bot Design | - Designstudien für Bots – freie Projekte |
| Chatter Bots | - Bots mit „Chat“ - Eigenschaften |
| Commerce Bots | - Komerzielle Bots |
| Data Mining Bots | - Roboter zum Extrahieren wichtiger Daten aus Strukturen |
| E-Mail Bots | - Autoresponder, POP-Sammler etc. |
| Fun Bots | - Spaßprogramme, Studien, Entertainment... |
| Game Bots | - Spiel-Roboter |
| Government Bots | - Regierungs-Roboter (zum Beispiel Suchroboter Ferret) |
| Knowledge Bots | - Intelligente Agenten |
| Misc. Bots | - Diverse |
| News Bots | - Bots für Online-Nachrichten |
| Newsgroup Bots | - Bots für USENET-Newsgroups (zum Beispiel mailmenews.com) |
| Research On Bots | - Wissenschafts – Roboter, Informationen |
| Search Bots | - Suchengine Roboter |
| Shopping Bots | - Roboter für Preissuche |
| Software Bots | - Roboter zur Softwaredistribution, Überwachung etc. |
| Stock Bots | - Roboter für den Aktienmarkt |
| Update Bots | - Roboter zur Webseitenüberwachung etc. |
| WebDevelopment | - Roboter für Webseitenvalidierung etc. |

Martijn Koster beschreibt bereits 1994 die Aufgaben und Fähigkeiten derartiger Crawler beziehungsweise „Web-Walker“ folgendermaßen:

> Diese „web-walkers“ oder „spiders“ haben den Vorteil, daß sie sehr zielstrebig vorgehen können und potentiell alle durchsuchbaren Teile des WWW-Raumes besuchen können. Sie werden für viele unterschiedliche Zwecke eingesetzt; um ungültige Referenzen zu finden, um neue Server zu finden, um die Größe des Web’s abzuschätzen, um Datenbanken abzufragen und um das Web zu katalogisieren. [KOSTER94, übersetzt aus dem Englischen]

Koster diskutiert dabei ebenfalls die Probleme, die solche Systeme für das WWW und die Betreiber von Websites bringen. Hierbei weist er auch darauf hin, daß diese Probleme nicht nur für den WWW-Raum, sondern auch für den FTP-Raum (mit dem Index Archie) und den Gopher-Bereich (mit dem Index Veronica) gelten.

***Netzverkehr (Traffic)***

Wenn ein Crawler-Programm einen Webserver abfragt, so kann es durch die automatische Verarbeitung der Webseiten eine wesentlich höhere Netzlast erzeugen als es ein normaler Browserbenutzer mit derselben Internetbandbreite erzeugen könnte. Insbesondere gilt dies für schnelle verteilt arbeitende Crawler von großen Suchengines, die mehrere Seiten gleichzeitig übertragen können. Die so geartete Überlastung kann bis zu einer „Denial of Service“ – Attacke ausarten, bei der der betroffene Server keine Anfragen mehr annehmen kann.

***Zugriff auf temporäre und ungewünschte Bereiche***

Herkömmliche Crawler können anders als ein menschlicher Besucher der Website nicht feststellen, ob ein Link auf eine temporäre oder irrelevante Datei verweist. Derartige Dateien werden deshalb optional bei den allgemeinen Crawlern mit Hilfe der Robot-Exclusion Standards von der Suche ausgeschlossen.

***Zugriff auf irrelevante Informationen***

Gewöhnliche Crawler werten die Inhalte der übertragenen Dokumente nicht aus – sie erkennen keine Log-Dateien oder andere irrelevante Dateien.

Weitere Probleme werden im Kapitel 6 (Ökonomische Betrachtungen) diskutiert.

## 3.1. Vorgaben für die Implementation eines Crawlers

Im Artikel *„Guidlines for Robot Writers“* schreibt Martijn Koster (damals bei Firma Nexor, inzwischen bei Metacrawler.com) folgende aus dem Englischen übersetzten Anforderungen für Crawler [KOSTER95]: Die bei der Realisation des Malware Crawlers intentiell nicht beachteten Vorgaben wurden in der nachstehenden Auflistung gesperrt gedruckt (vgl. auch Kapitel Crawler – Ethik)

***Identifikation***

- **Der Crawler muß sich selbst identifizieren (mittels des HTTP USER-AGENT Tags).**
- **Der Crawler sollte von einem Rechner agieren, der einen gültigen DNS-Namen besitzt, damit er leicht identifizierbar ist.**
- **Das HTTP „from“ Feld sollte eine Kontaktadresse des Crawler-Betreibers enthalten**
- **Das HTTP „referrer“ Feld kann benutzt werden, um den Websitebetreibern weitere Informationen (wie zum Beispiel eine Informations – URL) zu übermitteln.**

***Ankündigung***

- **Der Crawler sollte in der Newsgroup comp.infosystems.www.providers angemeldet werden, bevor er gestartet wird.**
- **Wenn eine bestimmte Zielgruppe besucht wird, so sollte der Crawler dort angekündigt werden.**
- **Eine URL mit Informationen zum Crawler sollte aufgesetzt werden.**

### *Überwachung / Tests*

- **Während der Crawler läuft, sollte ein Ansprechpartner erreichbar sein.**
- Die lokalen Netzbetreiber sollten über den Crawler informiert sein.
- Der Crawler sollte lokal gut ausgetestet sein.
- Logfiles über die Aktivitäten sollten erstellt werden.
- **Der Crawler sollte während des Crawlens gesteuert werden können.**
- Der Crawler sollte während des Crawlens abgebrochen werden können.

### *Ausführung*

- Der Crawler sollte keine große Serverlast bei dem Zielsystem erzeugen. („*Walk, don't run*“)
- Der HTTP-if-modified-since Header sollte ausgewertet werden.
- Der HTTP-Accept Header sollte nur die Dokumente enthalten, die auch wirklich verarbeitet werden können.
- Die URLs müssen strikt überprüft werden, diverse fehlerhafte Link-Realisierungen müssen abgefangen werden (vgl. auch nächstes Kapitel)
- Die Ergebnisse müssen auf Fehler und Server-Antworten überprüft werden (im MWC teilweise realisiert).
- Schleifen müssen vermieden werden, bereits besuchte URLs gespeichert werden.
- **Verschiedene URLs für den selben Server müssen erkannt werden und dementsprechend nicht erneut indiziert werden.**
- Der Zugriff sollte möglichst während Zeiten geringer Netzlast erfolgen.
- **Crawlen bereits besuchter Seiten sollte nicht zu häufig erfolgen (>2 Monate)**
- **Formulare sollten nicht verfolgt werden**

### *Allgemeines*

- **Die Exclusion-Standards sollten respektiert werden.**
- **Die Rohergebnisse und der überarbeiteten Ergebnisse sollten veröffentlicht werden.**
- **Die durch den Crawler gefundenen „dead-links“ und andere gefundene Fehler sollten an den Betreiber der aktuellen Site geschickt werden.**

Diese Liste stellt eine ziemlich idealisierte Sichtweise für die Gestaltung von Crawlern dar. Dem Autor ist nicht ein Crawler bekannt, der alle diese Standards einhält. Der Malware-Crawler ist aufgrund seiner besonderen Aufgabe gezwungen, bestimmte Standards (Auskunft über Funktion und Ergebnisse) zu ignorieren. Einige andere Punkte hingegen (zum Beispiel Steuerung während des Crawlens) können als Erweiterung realisiert werden.

## 3.2. Arten der Realisierung eines Hyperlinks im WWW

Um eine Linkverfolgung zu realisieren, muß zuerst einmal ein Link als ein solcher identifiziert werden. In HTML gibt es durch die historische Entstehung der Sprache und durch die verschiedenen Implementationen in den WWW-Browsern diverse Möglichkeiten, einen Hyperlink zu erzeugen. Dementsprechend ist es nicht möglich, allen Hyperlinks zu folgen. Hyperlinks die in Java oder in bestimmten Plug-In Typen (wie zum Beispiel Macromedia Flash) realisiert sind, können nur teilweise erkannt werden. Ein weiteres Problem entsteht dadurch, daß sich die Programmierer von Webseiten[24] nicht an die Standards des W3 Consortiums [W3C] halten. (Dieses läßt sich leicht mit dem vom Consortium angebotenen Validator-Programm für Webseiten [W3C-VAL] anhand diverser Seiten überprüfen.)

### 3.2.1. Standard-Link

```
<a href="neueseite.htm">Link-Text</a> oder
<a href='neueseite.htm'>Link-Text</a> oder
<a href=neueseite.htm>Link-Text</a>
```

Diese drei Ausprägungen eines Standard-Hyperlinks führen zu einer Verzweigung, wenn der Besucher der Website über den Text (oder die Grafik) des Bereichs zwischen **<a... >** und **</a>** klickt. Hierbei handelt es sich um die am häufigsten anzutreffende Art der Seitenverlinkung. Die letztgenannten zwei Ausprägungen werden nicht von allen Browsern akzeptiert.

### 3.2.2. Areamaps

```
<IMG src="map.jpg" border="0" width="123" height="45" ismap usemap="#welcome">
<MAP name="welcome">
    <AREA shape="RECT" coords="6,0,92,21" href="Seite1.htm" alt="">
    <AREA shape="RECT" coords="285,0,382,21" href="Seite2.htm" alt="">
</MAP>
```

Bei Areamaps handelt es sich um Grafiken, in denen bestimmte Bereiche als Hyperlinks gekennzeichnet werden. Ein Bild wird so in mehrere Polygonflächen aufgeteilt. Suchengines wie „Excite“ und „Lycos“ verfolgen diese Arten von Hyperlinks nicht (Quelle: [CT 99/23]). Der MWC verfolgt derartige Hyperlinks, sofern diese auf HTML-Dateien zeigen.

### 3.2.3. Framesets

```
<frameset framespacing="0" border="false" frameborder="0" cols="200,*">
       <frame name="Links" src="links.htm" target="_top" scrolling="auto"
              marginwidth="0"  marginheight="0">
        <frame name="Rechts" src="rechts.htm" scrolling="auto"
               marginwidth="0" marginheight="0">
</frameset>
```

[24] Nicht nur die Programmierer von „Malware“ – Webseiten ignorieren teilweise die W3C Standards.

Bei Framesets findet eine Unterteilung des Bildschirms in mehrere HTML-Dateien statt. Im vorstehenden Beispiel wird auf der linken Seite ein schmaler Navigationsrahmen aus der Datei **links.htm** angezeigt, während auf der rechten Seite der eigentliche Inhalt der Seite (**rechts.htm**) dargestellt wird. Problematisch an der Extraktion der Hyperlinks aus einem Frameset ist zum einen die fehlende Endmarke (wie sie bei normalen Hyperlinks mit **</a>** vorliegt) und zum anderen die Möglichkeit, Framesets beliebig zu verschachteln. Suchengines wie „Inktomi“ (Suchengine hinter diversen bekannten Engines) und „Go“ verfolgen zum Beispiel keinerlei Frame-Hyperlinks aus diesen Gründen. (Quelle: [CT 99/23]). Der MWC verfolgt derartige Hyperlinks problemlos und behandelt diese in einer eigenen Routine.

## 3.2.4. Formulare

```
<FORM NAME=adresse ONSUBMIT="neueseite.htm">
      <INPUT TYPE=SUBMIT VALUE=Click>
</FORM>
```

Mit Formularen wird zur Tarnung vor Suchengines bei einigen Malware-Sites eine Schaltfläche realisiert, obwohl diese Funktion eigentlich für Benutzereingaben in Textfeldern vorgesehen ist und vorrangig derart auch verwendet wird. Problematisch ist hierbei, daß der Crawler auf keinen Fall Eingaben in eventuelle Datenbanken etc. vornehmen darf. Dementsprechend muß innerhalb eines Formulars geprüft werden, ob eine Verlinkung auf ein CGI oder eine normale HTML-Datei vorgenommen wurde. Normale Suchengines fassen derartige Hyperlinks nicht an – wegen der vermehrten Nutzung dieser Tarnverfahren durch Malware-Seiten[25] wurde dieses im Malware-Crawler jedoch implementiert.

## 3.2.5. Javascript, DHTML, CSS, ASP, PHP3 und Varianten

```
<body onload="neueseite.htm">
```

Es gibt kein allgemeingültiges Javascript / ASP,... – Link – Format. In Javascript ist es möglich, aus Teilstrings einen gültigen URL-Namen zusammenzusetzen und diesen dann mit Hilfe von Javascript Befehlen anzuspringen. Simple Verweise auf HTML-Seiten kann der MWC (über die Fähigkeiten normaler Webcrawler hinausgehend[26]) anhand der Endung und der Einschließung in Anführungszeichen erkennen – bei zusammengesetzten Stringteilen kann der MWC jedoch keinen Erfolg erzielen. In diesem Falle müßte eine Codeemulation vorgenommen werden, was aus Sicherheitsgründen ebenfalls nicht sinnvoll wäre (zum Beispiel wegen bestehender JavaScript-„Bomben“, die Endlos-Schleifen erzeugen).

## 3.2.6. Java

```
<APPLET codebase="../ " code="applet.class" width=123 height=456>
   <PARAM name=url1 value="link1.htm">
   <PARAM name=url2 value="link2.htm">
</APPLET>
```

[25] Beispiel: http://www.coderz.net/tally

[26] „Infoseek“ und viele andere Crawler können keinerlei JavaScript verarbeiten [CT 99/23]

Bei Java-Applets kommt es während der Analyse darauf an, ob der Link dem Applet als Parameter übergeben wird (der Malware Crawler erkennt dann anhand der Anführungszeichen und der HTML-Endung, daß es sich um einen Link handeln muß) oder ob der Link im Applet-Code enthalten ist. Wenn der Link im Applet selbst enthalten ist, kann der MWC den Link nicht extrahieren, da das Applet selbst nicht geladen und ausgeführt beziehungsweise emuliert wird.

### 3.2.7. Browser - Plug Ins (Flash, Shockwave etc.)

Diverse Firmen haben Erweiterungen für Browser hergestellt. Der prominenteste Vertreter dieser Gattung ist sicherlich Flash der Firma Macromedia. Hier interpretiert ein sogenanntes Plug-In im Browser den übermittelten Code. Da es sich hierbei um eine Programmausführung handelt, gelten dieselben Einschränkungen, die auch für Java und Javascript gelten.

### 3.2.8. Serverbasiertes CGI

**`<a href="out.cgi?12345">Link-Text</a>`** oder

Das serverseitige CGI präsentiert anhand der Nummer / der Parameter dem Browser eine neue Website, die sich sowohl auf dem internen als auch auf einem externen Server befinden kann. Dementsprechend funktioniert bei serverbasierten cgi's die Aufspaltung der Hyperlinks in externe und interne Hyperlinks nicht. Einige Suchmaschinen verwenden diese Technik, um zu überprüfen, welche Fundstellen der Internetsuche angewählt wurden[27]. Bei Viren-Sites findet sich diese Technik seltener. Das Crawlen bereitet keine Probleme, allerdings ist die Zuordnung und Abgrenzung der einzelnen Sites nicht möglich.

## 3.3. Datei- und Dokumenttypen (im WWW-Raum)

Beim Crawlen ist es wichtig, daß nur diejenigen Dateien übertragen werden, die HTML-Inhalte besitzen.

> Der HTML-Tag: **`<a href=“”></a>`** kann einerseits auf ein Bild verweisen (zum Beispiel in der Ausprägung **`<a href=“datei.jpg“>Text</a>`** ).
>
> Andererseits kann auch ein (zu verfolgender) Hyperlink referenziert werden (zum Beispiel in der Form **`<a href=“datei.htm“>Text</a>`**).

Anhand der Endung der verlinkten Datei wird nun im MWC die Beschaffenheit und Relevanz der Datei erkannt. Hierfür benutzt der Malwarecrawler eine interne Tabelle der Dateitypen, die weiter verfolgt werden. Da sich hinter einer .jpg Datei unter Umständen auch eine versteckte HTML-Seite verbergen kann, scheitert diese Technik des Crawlers an dieser Stelle dann jedoch.

Im Browser ohne Plug-In darstellbare (Text)-Dokumente

| Endung | Dateityp |
|---|---|
| ASP | Active Server Pages – Dynamisch |
| AFP | Active Foxpro Pages – Dynamisch |
| HTM | HTML-Hypertext |
| HTML | HTML-Hypertext |

[27] Dies gilt zum Beispiel für die Suchengine HOTBOT (www.hotbot.com)

| | |
|---|---|
| PHP3 | PHP – Dynamisch |
| SHTML | S-HTML |
| WWS | West Wind Web – Dynamisch |
| XML | XML |
| .PL, .CGI, ... | Sonderfall: Serverbasierte Anwendungen (CGI) liefern HTML-Code |
| TXT | Sonderfall: Textdateien |

Nachfolgend findet sich eine Liste der möglichen Dokumenttypen im Internet. Diese Liste wurde der Dokumentation der Programme Inso-Quickview [INSO] (Anzeigeprogramm für diverse Dokumente), dem Programm ACDSEE (Bildbetrachter, www.acdnet.com) sowie den gescannten Filetypen des Programms NAI-Scan [NAI] entnommen. Im weiteren Verlauf der Arbeit konnte diese Liste an vielen Stellen manuell ergänzt werden. Die Dateitypen die (nach aktuellem Wissensstand) Malware enthalten können, wurden **gesperrt** gedruckt. [28]

### *Ausführbare Programme*

| Endung | Dateityp |
|---|---|
| **EXE** | Ausführbare Datei |
| **COM** | DOS-Befehlsdatei <64k |

### *Programm – Librarys*

| Endung | Dateityp |
|---|---|
| **DLL** | Dynamic Link Library |
| **OCX** | Active-X Objekt |
| **VXD** | Device-Treiber |
| **386** | Geräte- Treiber |
| **SYS** | System- Treiber |
| **OBD** | ? – Entnommen aus NAI-Scan |
| **OBT** | ? – Entnommen aus NAI-Scan |
| **OLE** | Object Linking and Embedding |
| **SHS** | ? – Entnommen aus NAI-Scan |
| **MPP** | ? – Entnommen aus NAI-Scan |
| **MPT** | ? – Entnommen aus NAI-Scan |
| **XTP** | ? – Entnommen aus NAI-Scan |
| **XLB** | ? – Entnommen aus NAI-Scan |
| **CMD** | ? – Entnommen aus NAI-Scan |
| **OVL** | Overlay |
| **DEV** | Device |
| **MD?** | ? – Entnommen aus NAI-Scan |

### *Scripts*

| Endung | Dateityp |
|---|---|
| **VBS** | Visual Basic Script |
| **SCR** | Script (zum Beispiel Modem) |

### *Präsentationsformate*

| Endung | Dateityp |
|---|---|
| **PPT** | Microsoft Powerpoint |

[28] Die hypothetische maliziöse Ausnutzung von zum Beispiel Audio- und Multimediaformaten für „Bomben“ in Form von bestimmten Frequenzen, die für den Menschen oder die angeschlossene Technik maliziös wirken, soll hier nicht weiter betrachtet werden.

| | |
|---|---|
| POT | Microsoft Powerpoint Dokumentvorlage |
| SHW | Corel Show |
| PRE | Freelance Präsentation |
| PRZ | ? |

### *Musik Formate*

| Endung | Dateityp |
|---|---|
| AU | Audio - Datei |
| SND | Sound – Datei |
| WAV | Wave-Datei |
| MID | Midi-Datei |
| MP3 | MP3-Komprimierte Musikdatei |
| KAR | ? |

### *Gepackte Dateien*

| Endung | Dateityp |
|---|---|
| **ARJ** | Archivformat (Robert K. Jung) |
| **LHA, LHZ, LZH** | LH – Archiv (Haruyasu Yoshizaki) |
| **Z** | UNIX. Z |
| **ZIP** | ZIP Archiv (Pkware / diverse) |
| **TAR, TAZ** | Tape Archive (Unix) |
| **GZ** | Gnu Zip |

### *Im Browser ohne Plug-In darstellbare Grafik-Dokumente*

| Endung | Dateityp |
|---|---|
| JPG, JPEG, JPE, JIF, JFIF | JPG – Grafik, Independent JPG-Group |
| GIF | Compuserve Graphic-Interchange-Format |
| PNG | Portable Network Graphics (nicht von allen Browsern unterstützt) |

### *Bildformate, die nur mittels Plug-In oder externem Betrachter dargestellt werden können*

| Endung | Dateityp |
|---|---|
| **EPS** | Encapsulated Postscript |
| PCT | Picture Format |
| PICT | Picture Format |
| CDW | ? |
| SGI, BW, RGB, RGBA | SGI Image Format |
| WMF | Windows Metafile |
| SDW | ? |
| WPG | ? |
| PCX | Zsoft's Paintbrusch |
| CUR | ? |
| DIB | ? |
| CGM | ? |
| TGA, TARGA | Targa |
| ICO | Windows Icon |
| BMP, RLE | Run Length Encoded Bild (Windows, Windows Bitmap) |
| PSD | Photoshop |
| **CDR** | Corel Draw |
| CUR | Windows Cursor |
| EMF | Enhanced Metafile |

| | |
|---|---|
| IFF, LBM, ILBM | Amiga ILBM |
| KDC | Kodak Bildformat |
| PCD | Kodak PhotoCD |
| DCX | Zsoft's Paintbrusch |
| TIF, TIFF | Tag Image File Format |
| PIC | Softimage – PIC Format |
| PIX | Alias Wavefront Image |

### *Multimedia Formate*

| Endung | Dateityp |
|---|---|
| AIF, AIFF | Amiga- Videoformat |
| AIFC | ? |
| MIDI | MIDI – Datei |
| MPG | Motion Picture MPEG |
| MPE | Motion Picture MPEG |
| MOV | Quicktime Movie |
| QT | Quicktime |
| RA | Real Audio |
| RM | Real Media |
| AVI | Video |
| RMM | ? |

### *Dokument-Formate*

| Endung | Dateityp |
|---|---|
| PDF | Adobe Acrobat |
| WRI | Windows Write |
| WTA | ? |
| **DOC** | Word Dokument |
| **DOT** | Word Dokumentvorlage |
| **RTF** | Rich Text Format (Infiziert z.B. als getarntes „doc“) |
| WPD | Word Perfect |
| TSV | ? |
| LWP | ? |
| ETX | ? |
| DX | ? |
| DCA | ? |
| FFT | ? |
| RFT | ? |
| **HLP** | Windows Hilfedatei |
| **CHM** | Neuer Windows-Hilfetyp |
| SAM | Amipro Dokumentendatei |

### *Tabellenkalkulations – Formate*

| Endung | Dateityp |
|---|---|
| WK1 | ? |
| WK3 | ? |
| WK4 | ? |
| 123 | Lotus |
| WT4 | ? |
| WG2 | ? |
| WKS | ? |
| XLC | ? |
| **XLS / XLT** | Excel |

| | |
|---|---|
| XLB | ? |
| WDB | ? |
| WPS | ? |
| WB1 | ? |
| WB2 | ? |
| WB3 | ? |
| WQ1 | ? |
| WQ2 | ? |

***Sourcecodes – Die Dateien können zwar Malware enthalten, bedürfen jedoch einer Compilierung oder Interpretation durch einen Interpreter.***

| Endung | Dateityp |
|---|---|
| BAS | Basic Source |
| C | C-Programm |
| H | Iclude Datei |
| CPP | C++ Programm |
| MDB / MDZ | Access Programm / Vorlage |
| PAS | Pascal Programm |
| PL | Perl Programm |
| PRG | Clipper / Dbase / Visual Objects |
| FXP / APP / PRG / SCX / SCT / VCT / VCX | Visual Foxpro |
| ... | Es existieren noch diverse weitere Sprachen mit teilweise überlappenden Dateiendungen |

***Spezialfälle***

| Endung | Dateityp |
|---|---|
| **REG** | Registry Datei |
| **DAT** | Datei – diverse Verwendung |

## 3.4. Ausschluß von Crawlern (Robots Exclusion Standards)

> *„WWW Robots (also called wanderers or spiders) are programs that traverse many pages in the World Wide Web by recursively retrieving linked pages .[ ...]*
>
> *In 1993 and 1994 there have been occasions where robots have visited WWW servers where they weren't welcome for various reasons. Sometimes these reasons were robot specific, e.g. certain robots swamped servers with rapid-fire requests, or retrieved the same files repeatedly. In other situations robots traversed parts of WWW servers that weren't suitable, e.g. very deep virtual trees, duplicated information, temporary information, or cgi-scripts with side-effects (such as voting).*
>
> *These incidents indicated the need for established mechanisms for WWW servers to indicate to robots which parts of their server should not be accessed. This standard addresses this need with an operational solution*. – Martijn Koster [KOSTER_ROB]

Es gibt im wesentlichen zwei etablierte Verfahren, um Web-Crawlern das Aufsuchen beziehungsweise das Indizieren bestimmter Seiten zu untersagen.

### 3.4.1. Robots Exclusion im HTML-Code

Zum einen gibt es die Möglichkeit, im HTML-Code mittels folgendem Meta-Tag Seiten von der Indizierung durch Roboter auszuschließen (dieses HTML-Tag wird jedoch nur von wenigen Suchengines berücksichtigt):

Das Robots-META-Tag muß wie alle Meta-Tags in dem **<HEAD> </HEAD>** Klammernpaar abgelegt werden:

```
<html>
<head>
      <meta name="robots" content="noindex,nofollow">
      <meta name="description" content="This page ....">
      <title> Seitentitel </title>
</head
<body>
</body>
</html>
```

In diesen Tag wird folgendes geschrieben (Definition nach Martijn Kosters Web Robots Page [KOSTER_ROB]).
:

> Der Inhalt des Robots META Tags enthält Direktiven, die durch Kommatas getrennt werden. Die derzeit definierten Direktiven sind „[NO]INDEX“ und „[NO]FOLLOW“. Die „INDEX“ Direktive spezifiziert, ob ein indizierender Roboter die Seite indizieren soll. Die „FOLLOW“ Direktive spezifiziert, ob ein Roboter den Links der Seite folgen soll. Die Default-Werte sind „INDEX“ und „FOLLOW“. Die Werte „ALL“ und „NONE“ setzen alle Direktiven in Kraft oder außer Kraft. „ALL“ entspricht „INDEX, FOLLOW“ und „NONE“ entspricht „NOINDEX, NOFOLLOW“.
>
> Beispiele:

```
<meta name="robots" content="index,follow"
<meta name="robots" content="noindex,follow"
<meta name="robots" content="index,nofollow"
<meta name="robots" content="noindex,nofollow"
```

Das Robots Namens-Tag und der Inhalt sind unabhängig von Groß- und Kleinschreibung. Offensichtlich sollten keine wiedersprüchlichen oder wiederholten Direktiven wie die folgende angegeben werden:

***<meta name="robots" content="INDEX,NOINDEX,NOFOLLOW,FOLLOW,FOLLOW">***

Die formale Syntax für das Robots – META Tag ist:

```
content    = all | none | directives
all        = "ALL"
none       = "NONE"
directives = directive ["," directives]
directive  = index | follow
index      = "INDEX" | "NOINDEX"
follow     = "FOLLOW" | "NOFOLLOW"
```

Der Begriff „follow“ bedeutet hierbei, daß den Hyperlinks dieses Dokuments bis zum nächsten referenzierten Dokument gefolgt werden soll bzw. darf. Der Begriff „index“ bedeutet, daß der Inhalt dieses Dokuments von Suchmaschinen indiziert werden soll bzw. darf.

### 3.4.2. Robots Exclusion im Web- Rootverzeichnis

Zum anderen gibt es den „Robots Exclusion Standard“, welcher in der Datei „**`robots.txt`**“ im Root-Verzeichnis des jeweiligen Servers definiert wird. Im folgenden ein Auszug aus dem Exclusion-Standard [EXCL]:

> *The method used to exclude robots from a server is to create a file on the server which specifies an access policy for robots. This file must be accessible via HTTP on the local URL "`/robots.txt`". The contents of this file are specified below.*
> *This approach was chosen because it can be easily implemented on any existing WWW server, and a robot can find the access policy with only a single document retrieval.*

In den Robots Exclusion Standards wird weiterhin erwähnt, daß ein möglicher Nachteil dieser Methode ist, daß lediglich der Server-Administrator eine Datei im Root-Verzeichnis des Servers anlegen kann und demenstsprechend eine Liste dort ablegen darf. Der einzelne Benutzer des Webservers kann auf seinen Seiten keine derartige Datei ablegen, wenn er sich unterhalb der WWW-Hauptdirectoryebene befindet. Dieses ist durch eine automatische Generierung einer solchen Liste umgehbar – ein allgemeingültiger Standard für die automatische Erstellung einer solchen Liste existiert hingegen nicht.

Folgende Überlegungen waren bei der Wahl des Exclusion-Standards. [EXCL] wesentlich:

- Der Dateiname sollte in die Restriktionen aller üblichen Betriebssysteme passen.
- Die Dateinamenserweiterung sollte keine Umkonfigurierung des Servers erfordern.
- Der Dateiname sollte den Sinn der Datei wiedergeben und leicht merkbar sein
- Die Übereinstimmung des gewählten Namens mit bestehenden Dateinamen sollte minimal sein.

Beispiele für solche **`robots.txt`** Exclusion-Dateien:

```
# Alle Robots ausschließen
User-agent: *
Disallow: /


# Alle Robots einladen
User-agent: *
Disallow:


# Bestimmte Verzeichnisse sperren
User-agent: *
Disallow: /temp/
Disallow: /incoming/


# Bestimmten Robot (Altavista) sperren
User-agent: scooter
Disallow: /
```

Der Malware Crawler wurde als sogenannter „Rude Robot“ konzipiert, was bedeutet, daß sich der MWC nicht an die Exclusion-Standards hält. Auf den Malware – Sites wird sehr häufig von den Exclusion-Befehlen Gebrauch gemacht, um eine entsprechende Tarnung zu erzielen. Eine Berücksichtigung der Exclusion-Standards wäre daher kontraproduktiv. Dies führt direkt zum nächsten Kapitel – der „Ethik“ des Crawler-Programms.

## 3.5. Crawler - Ethik

David Eichmann [EICHMANN94] und Martijn Koster [KOSTER95] definieren für einen ethisch arbeitenden Crawler, beziehungsweise einen gerechtfertigten Einsatz eines Crawlers, folgendes:

- Der Autor braucht für das Projekt definitiv einen Crawler – es gibt keine bestehenden Verfahren, die adäquat sind.
- Der Crawler muß sich selbst bei den Server-Betreibern zu erkennen geben, der Autor muß erreichbar sein. (vgl. Kapitel 3.1.)
- Der Crawler muß ausgiebig getestet sein, bevor er auf eine reale Umgebung aufgesetzt wird. (vgl. Kapitel 3.1.)
- Wiederholte Dokumentübertragungen und sehr schnelle parallele Übertragungen sollten unterbleiben.
- Der Robot Exclusion Standard sollte eingehalten werden.
- Die Log-Dateien sollten ständig überprüft werden.
- Die Ergebnisse sollten sowohl im Rohzustand als auch ausgewertet der Internet-Gemeinde zur Verfügung gestellt werden.

Crawler, die der Allgemeinheit dienen (zum Beispiel Suchengine-Crawler), dürfen mehr Resourcen beanspruchen als zum Beispiel Crawler eines einzelnen Desktop-Users. Engines von Desktop-Usern sollten die erzeugte Netzwerklast nicht über die Last eines normalen Browse Vorgangs steigen lassen.

Der Malware-Crawler kann mit einigen dieser postulierten Eigenschaften nicht konform arbeiten. Die Positionen im einzelnen:

1. Die Suche nach Viren im WWW-Raum des Internets ist aus den in Kapitel 2 genannten Gründen nicht durch andere Verfahren zu ersetzen – hier ist der MWC „ethik-konform“

2. Der Malware Crawler gibt sich selbst nicht zu erkennen; ebenfalls fehlt die E-Mail Adresse etc. Dies ist beabsichtigt, um gezielte Gegenangriffe nicht zu vereinfachen.

3. Der Crawler wurde auf einem lokalen IIS4 (Internet Information Server unter Windows NT) – System ausgiebig getestet – hier ist der MWC „ethik-konform“.

4. Dokumente werden nicht innerhalb kürzerer Interwalle wiederholt übertragen; parallele Übertragungen finden gar nicht statt – hier ist der MWC „ethik-konform“.

5. Der Exclusion Standard wird aus den im vorigen Kapitel genannten Gründen nicht eingehalten.

6. Die Log-Dateien werden derzeit 1x pro Woche geprüft – hier ist der MWC „ethik-konform“.

7. Die Ergebnisse (also Viren, Malware und Hyperlinks auf entsprechende Seiten) können (ebenfalls aus ethischen Gründen) nicht für die Allgemeinheit veröffentlicht werden.

Der Malware-Crawler geht nach dieser Aufzählung mit drei der sieben genannten Ethikaspekte bewußt nicht konform. Dies liegt an der spezifischen Aufgabe des Malwarecrawlers.

***Zu Position 7 – Ethische Verwendung der MWC Daten.***

Die vom MWC erzeugten Daten sollten der Allgemeinheit nicht zur Verfügung gestellt werden. Die vom Malware Crawler erzeugten URL-Listen stellen in kriminellen Händen ein Werkzeug zur Begehung einer Straftat bereit (Datenmanipulation, Computersabotage, etc.). Durch eine Veröffentlichung der URL-Listen würde eine geringere kriminelle Energie benötigt, um eine derartige Straftat zu begehen, da der Aufwand für die Suche derartiger Seiten entfiele. Anders, als bei dem beliebten Gegenbeispiel der Haushaltsgegenstände, die zweckentfremdet werden können (Messer,...), besteht für den normalen Anwender keine zwingende Notwendigkeit, die URLs von Malware-Sites zu kennen (zu besitzen).

Die erzeugten Listen sollten deshalb nur einem ausgewählten Kreis von Antivirus - Experten überlassen werden. Diese müssen (sofern sie die Listen in ihren Produkten zum Blockieren derartiger Sites benutzen) die URLs durch Verschlüsselung gegen eine Einsichtnahme durch Dritte schützen.

## 3.6. Datenstrukturen beim Crawlen

Die folgenden Tabellen werden vom MWC beim Crawlen verwendet und gefüllt:

Eintragungen im Feld „Typ“

C = Character (ASCII-Zeichen), gefolgt von der Feldlänge in Zeichen
M = Textfeld beliebiger Länge
D = Datum
L = Logischer (boolean) Wert
N = Gleitkommazahl, gefolgt von der Anzahl der Stellen und ggf. Nachkommastellen

**ENGINES** – Tabelle der Suchengines – Diese Tabelle enthält die Suchengine-Aufrufe, die für Startwerte benutzt werden.

| Feld | Typ | Inhalt |
|---|---|---|
| EN_NAME | C (30) | Name der Engine |
| EN_URL | C (254) | Aufruf der Engine incl. POST-Informationen |
| EN_LANG | M | URL, falls 254 Zeichen nicht ausreichen |

**BADLIST** – Liste bekannt maliziöser Websites. Bereits gespeicherte Sites werden beim Crawlen nicht erneut in die Liste der zu durchsuchenden URLs aufgenommen.

| Feld | Typ | Inhalt |
|---|---|---|
| BL_INFOS | C (30) | Verbale Beschreibung der Site |
| BL_URL | C (254 | URL der Seite |
| BL_VISIT | D | Letzter Besuch |
| BL_CHECKSUM | N(10) | Checksumme über Inhalt der Site |
| BL_CHANGED | L | Flag für Veränderung der Site |

**GOODLIST** – Liste von Websites, die nicht durchsucht werden sollen – zum Beispiel sehr häufig auftretende URLs wie **ad.doubleclick.net** (Werbebanner) werden so von vornherein von der Suche ausgeschlossen. Ebenfalls können Seiten, die bei der Heuristik für „Hits“ sorgen könnten, wie zum Beispiel die Seiten der Antivirenhersteller oder auch Seiten von Forschungseinrichtungen zu biologischen Viren, hier von der weiteren Verarbeitung ausgeschlossen werden.

| Feld | Typ | Inhalt |
|---|---|---|
| GL_NAME | C (30) | Verbale Bezeichnung |
| GL_URL | C (254) | URL |
| GL_URLKURZ | C (40) | Nicht verwendet (vormals Indexkey) |

**SCANLIST** – Liste der zu durchsuchenden Dokumente innerhalb derselben Website. Wird „Geladene URL Crawlen“ angewählt, so werden in dieser Tabelle alle lokalen Hyperlinks abgelegt. Gleichzeitig findet im Feld "SC_HEURIST“ die Eintragung des Heuristik-Wertes für das entsprechende Dokument statt.

| Feld | Typ | Inhalt |
|---|---|---|
| SC_URL | C(254) | URL |
| SC_HEURIST | N(4) | Heuristik Wert dieser URL |
| SC_CHCKSUM | N(10) | Checksumme |
| SC_ZEIT | C(10) | Zeit des Besuchs |

| | | |
|---|---|---|
| SC_DATUM | D | Datum des Besuchs |
| SC_TYP | C(10) | Typ des Verweises |

**EXTLIST** – Liste der Hyperlinks, die zu externen URLs verweisen. Die während des Crawlens einer URL anfallende Hyperlinks auf andere Domains werden in dieser Tabelle abgelegt. Während der Ausführung des Befehls „Externe Liste Crawlen“ bekommen diese Seiten einen heuristischen Wert zugeordnet.

| Feld | Typ | Inhalt |
|---|---|---|
| SC_URL | C(254) | URL |
| SC_HEURIST | N(4) | Heuristik Wert dieser URL |
| SC_CHCKSUM | N(10) | Checksumme |
| SC_ZEIT | C(10) | Zeit des Besuches |
| SC_DATUM | D | Datum des Besuches |
| SC_TYP | C(10) | Typ des Verweises |

**FILELIST** – Liste der Datei-Links. Diese Liste enthält alle Dateiverweise der geladenen URL (beziehungsweise der gecrawlten Website). Der Heuristik-Wert legt für das Programm, welches die Dateien überträgt und scannt, fest, welche Dateien geprüft werden.

| Feld | Typ | Inhalt |
|---|---|---|
| SC_URL | C(254) | URL-Pfad |
| SC_HEURIST | N(4) | Heuristik Wert |
| SC_CHCKSUM | N(10) | Checksumme |
| SC_ZEIT | C(10) | Zeit des Besuches |
| SC_DATUM | D | Datum des Besuches |
| SC_TYP | C(10) | Typ des Verweises |

## 3.7. Vorgehensweise beim Laden jeder einzelnen URL

Beim Laden jeder einzelnen URL durchläuft der MWC folgende Schritte:

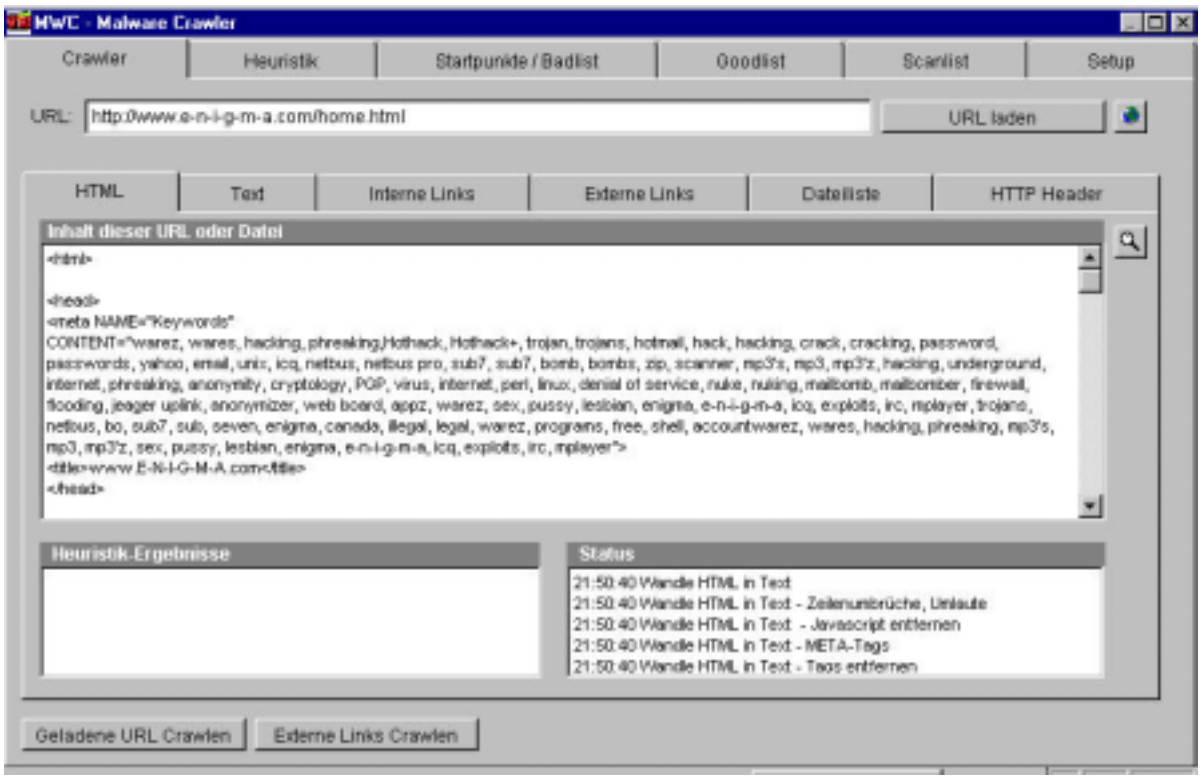



**Abbildung 5 – Unbehandeltes HTML-Dokument**

***1.*** ***Laden der URL:*** Die URL wird mit der in Kapitel 2 beschriebenen Prozedur in den Textbuffer des Malwarecrawlers geladen.

Das unbehandelte HTML-Dokument hat das folgende Aussehen (hier wurde die Seite die in Kapitel 1 in Abbildung 2 gezeigt wurde, analysiert):

```
<html>
<head>
<meta NAME="Keywords"
CONTENT="warez, wares, hacking, phreaking,Hothack, Hothack+, trojan, trojans, hotmail,
hack, hacking, crack, cracking, password, [... usw ...]">
<title>www.E-N-I-G-M-A.com</title>
</head>

<body BGCOLOR="#000000" TEXT="#FFFFFF" VLINK="#5959AB" ALINK="#008080" link="#000000">
<script language="JavaScript"><!--

[... JAVASCRIPT gekürzt ...]

// --></script>

<p align="center"><img src="/images/topbanner.jpg" WIDTH="706" HEIGHT="59"> </p>

<table ALIGN="left" BORDER="0" CELLSPACING="0" CELLPADDING="0" WIDTH="17%">
  <tr BGCOLOR="#000000">
    <td COLSPAN="1" HEIGHT="1" ALIGN="center" WIDTH="13%"></td>
  </tr>
  <tr BGCOLOR="#2F4F4F">
    <th><font SIZE="3" COLOR="#B87333">Menu</font></th>
  </tr>
  <tr BGCOLOR="#000000">
```

```
    <td COLSPAN="1" HEIGHT="1" ALIGN="center" WIDTH="13%"></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font          SIZE="2"><a          HREF="http://www.e-n-i-g-m-a.com/home.html"
TARGET="_parent">Home</a></font></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font       SIZE="2"><a       HREF="http://www.e-n-i-g-m-a.com/disclaimer.html"
TARGET="_parent">Legal
    Disclaimer</a></font></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font         SIZE="2"><a          HREF="http://www.e-n-i-g-m-a.com/vchat.html"
TARGET="_parent">Voice
    Chat</a></font></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font SIZE="2"><a HREF="/search.html" TARGET="_parent">Search</a></font></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><a       HREF="http://members.m4d.com/enigma/topsites/"       TARGET="new"><font
SIZE="2">Top List</font></a></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><a                                HREF="http://e-n-i-g-m-a.com/wwwboard/"><font
SIZE="2">WWWBoard</font></a></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font SIZE="2"><a HREF="/warez.html" TARGET="_parent">Warez</a></font></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font SIZE="2"><a HREF="/contact.html" TARGET="_parent">Contact</a></font></td>
  </tr>
  <tr BGCOLOR="#000000">
    <td COLSPAN="1" HEIGHT="1" ALIGN="center" WIDTH="13%"></td>
  </tr>
<tr BGCOLOR="gray">
    <td><font SIZE="2"><a HREF="/hacks/irc.html" TARGET="_parent">IRC</a></font></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font SIZE="2"><a HREF="/hacks/icq.html" TARGET="_parent">ICQ</a></font></td>
  </tr>
  <tr BGCOLOR="gray">
    <td><font                    SIZE="2"><a                    HREF="/hacks/virii.html"
TARGET="_parent">Virii</a></font></td>
  </tr>

[... usw ...]

</body>
</html>
```

***2.*** ***Wandlung der META-Tag Informationen:*** Die META-Tags werden gesondert extrahiert und an den Anfang des bereinigten Textes gestellt. Dieses hat den Sinnn, daß eventuelle beschreibende Meta-Tags wie zum Beispiel die folgenden:

```
<META name="keywords" content="Keywords für diese Website ">
<META name="description" content="Beschreibung der Website">
<META name="author" content="Autor">
<META name="contacts" content="E-Mail des Autors">
<META name="robots" content="all,follow">
<META name="comment" content="Kommentar">
```

in die heuristische Bewertung mit einbezogen werden. Hierbei sind die META-Tags „keywords“ und „description“ besonders interessant, da die meisten modernen Suchengines ihre Indizes anhand dieser Tags aufbauen, sofern diese im Dokument auftreten. Wenn diese Tags nicht auftreten, werden (zum Beispiel bei Altavista) die ersten **n** Zeichen der Website in dem Suchengine-Katalog gespeichert, wobei **n** für die verschiedenen Suchengines variabel ist.

Der Meta-Tag Teil für diese Website sieht dementsprechend folgendermaßen aus:

```
META-TAG:keywords INHALT:warez, wares, hacking, phreaking,hothack, hothack+, trojan,
trojans, hotmail, hack, hacking, crack, cracking, password, passwords, yahoo, email,
unix, icq, netbus, netbus pro, sub7, sub7, bomb, bombs, zip, scanner, mp3's, mp3,
mp3'z, hacking, underground, internet, phreaking, anonymity, cryptology, pgp, virus,
internet, perl, linux, denial of service, nuke, nuking, mailbomb, mailbomber, firewall,
flooding, jeager uplink, anonymizer, web board, appz, warez, enigma, e-n-i-g-m-a, icq,
exploits, irc, mplayer, trojans, netbus, bo, sub7, sub, seven, enigma, canada, illegal,
legal, warez, programs, free, shell, accountwarez, wares, hacking, phreaking, mp3's,
mp3, mp3'z, enigma, e-n-i-g-m-a, icq, exploits, irc, mplayer (einige Begriffe entfernt).
```


**Abbildung 6 – Extrahierter Text**

### *3. Extraktion der Textinhalte:*

- Jegliche HTML-Tags werden entfernt.
- Java, Javascript, etc. werden entfernt.
- Die HTML-codierten Umlaute (also zum Beispiel ö für ö) werden wieder in ASCII umcodiert.
- Zeilenumbrüche werden entfernt, wenn mehr als zwei aufeinanderfolgende Zeilenumbrüche vorliegen.

Nachstehend der resultierende, automatisch generierte Output (der lediglich nachträglich um einige Zeilenumbrüche gekürzt wurde):

```
www.E-N-I-G-M-A.com

  Menu

  Home
  Legal Disclaimer
  Voice Chat
  Search
  Top List
  WWWBoard
  Warez
  Contact
  Archives
  Unix
  Cracking
  Denial of Service
  IRC
  ICQ
  Virii
  Encryption
  Trojans
  Mailbombers
  Mplayer
  Links

  Cyberarmy
  War Industries
  xpiRox
  Attrition
  HackerNewsNetwork
  Hackers Hideout

  MP3s

  MuchMusic Top 30
  MTV Top 30
  Archive

  Vote for Me

  Vote Page
  Sponsors


Site News-


Mar 24, 2000:  Damn, I really should maintain this site more often...I doubt that
anybody comes here anymore...oh well.  The new forum is up.

Feb 26, 2000:  Added Voicechat.

Feb 24, 2000:  Holy shit i've done alot to the site today...I added a search page,
added some java scripts here and there...Unix, ICQ, Virii, Trojans, and Mplayer now
work (finally).


[... usw ...]
```

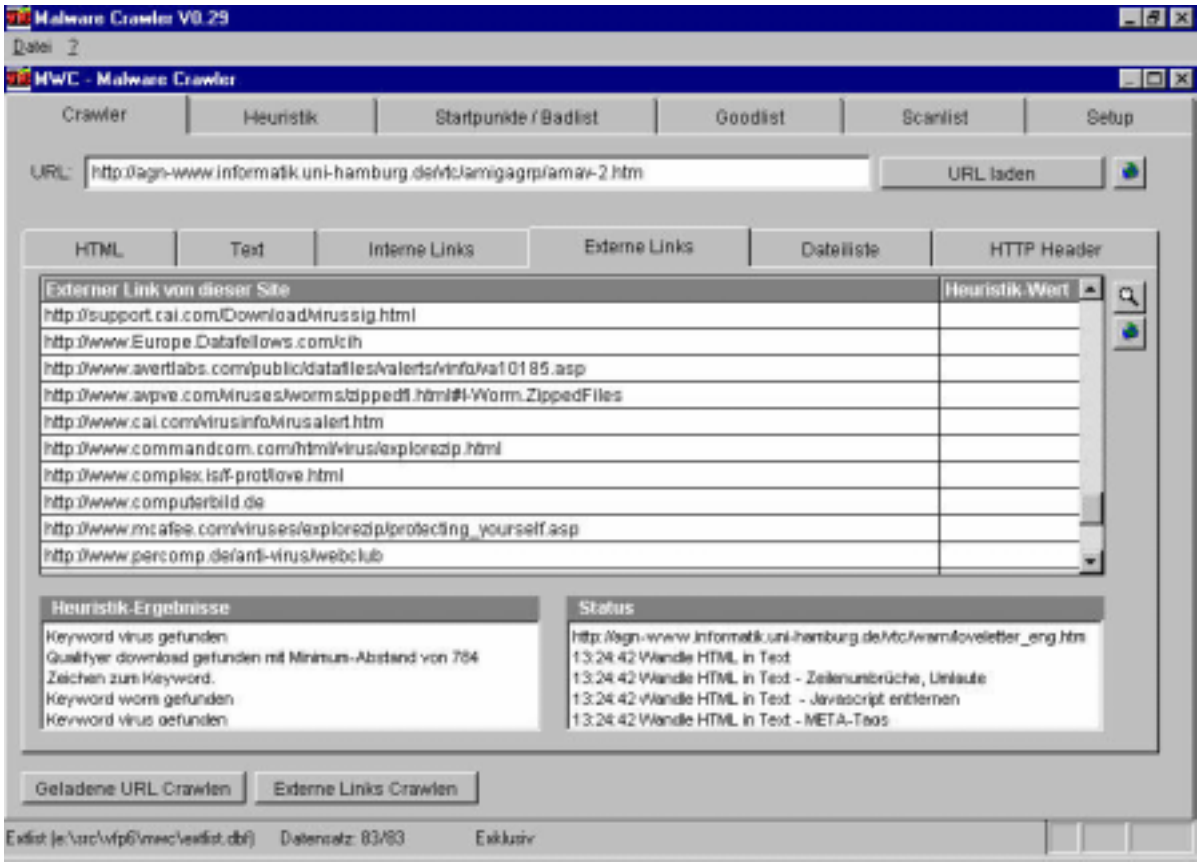



**Abbildung 7 - Extrahierte Linkliste des MWC**

4. ***Extraktion der Hyperlinks innerhalb derselben Site***: Hyperlinks, die innerhalb derselben URL auf Verzeichnisse oder Dokumente verweisen, gelten als interne Hyperlinks. Im Kapitel 3 wurden bereits die verschiedenen Arten der Realisierung eines Hyperlinks eingehend erläutert. Neue Hyperlinks werden vor ihrer Aufnahme auf Duplizität in der Datenbank geprüft und gegebenenfalls eingetragen. Bei der Verarbeitung interner Hyperlinks muß darauf geachtet werden, daß diese sowohl absolut (als komplette URL) oder relativ zum aktuellen Verzeichnis angegeben werden können.

   Der HTML-Parser des Malware Crawlers ersetzt relative Hyperlinks wie zum Beispiel einen Hyperlink **„../index.htm“** von einer geladenen Seite
   mit der eigenen URL **http://www.test.com/verzeichnis/unterverzeichnis/seite.htm**
   durch die absolute URL **http://www.test.com/verzeichnis/index.htm**.

   Ebenfalls werden Verschachtelungen dieser relativen Linkhierarchien wie zum Beispiel die folgende **http://www.test.com/a/b/c/../d/../../e** korrekt aufgelöst.

   Anhand der Endung des jeweiligen Hyperlinks unterscheidet der Malware Crawler, ob es sich um eine HTML (oder ähnliche Datei) handelt, oder ob ein Datei-Link vorliegt.

***5. Extraktion der Externen Hyperlinks***

Bei der Aufnahme externer Hyperlinks wird ein Link nach der Gegenprüfung gegen die Liste der bereits erfaßten externen Hyperlinks in die Datenbank eingetragen. Dieses Verfahren wird analog auch bei den Datei-Links angewendet:

***6. Extraktion der File-Links***

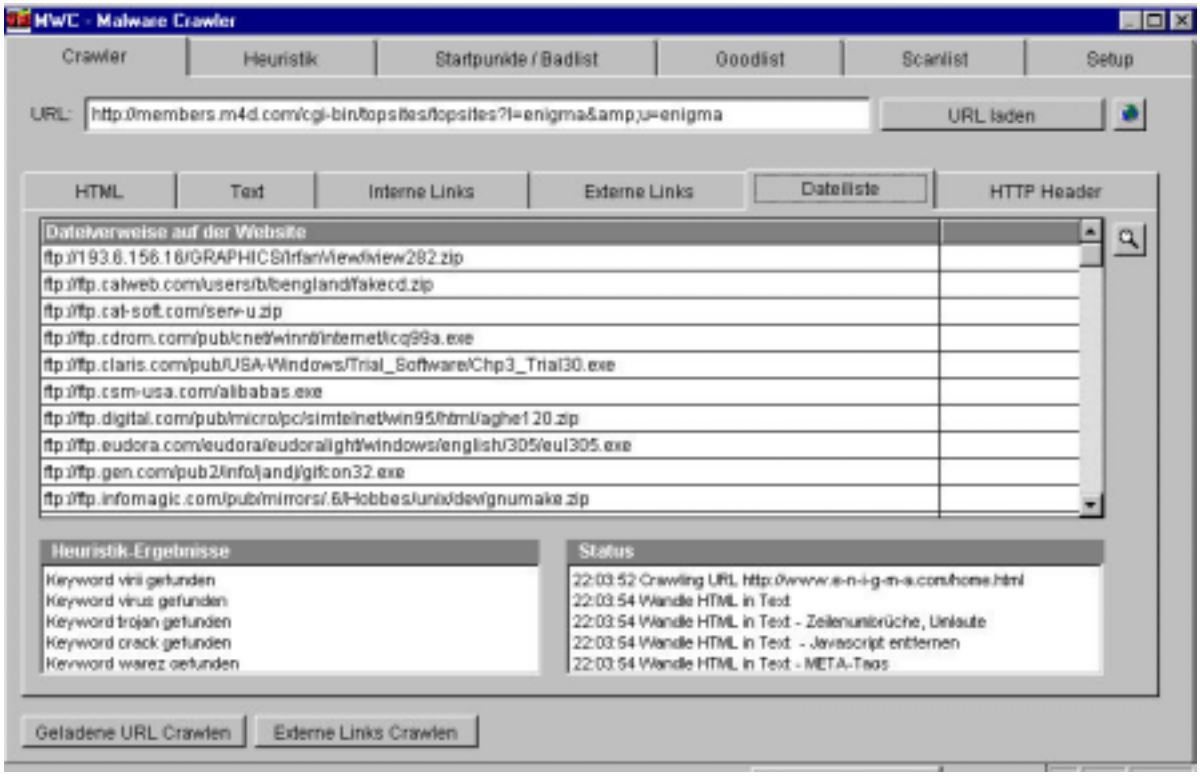



**Abbildung 8 - Extrahierte File-Links**

***7. Holen des HTTP-Headers***

Der HTTP-Header des Servers wird eingetragen

Nachfolgend die entsprechenden Ergebnisse der betrachteten Seite:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Date: Sat, 10 Jan 2000 19:52:14 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Sat, 01 Apr 1999 06:51:37 GMT
ETag: "ee89b05da7b1bf1:26ea5a"
Content-Length: 14734
```

***8. Heuristische Bewertung:*** Die mehrstufige heuristische Bewertung wird in Kapitel 4 näher erläutert.

## 3.8. Der HTTP-Header

Der im vorigen Kapitel abgedruckte HTTP-Header soll hier als Beispiel für die Analyse dieser Headerzeilen dienen:

1. Zeile: **`HTTP/1.1 200 OK`**

**„`HTTP/1.1“ -`** Das HTTP-Protokoll 1.1 wird benutzt (derzeit aktuelles Protokoll). Möglich wären derzeit noch Werte **`HTTP/1.0`** und **`HTTP/0.9`** der Vorgängerversionen. Die ersten Zeichen des HTTP-Headers sind nach [RFC 2068] bzw. in der Neufassung von 1999 [RFC 2616] genormt.

1. Zeile: **`200 OK`**

**„`200 OK“`** - Die Übertragung war erfolgreich. Mögliche Rückgaben in der ersten Zeile nach dem HTTP-Header nach [RFC 2616]:

***Erfolgsmeldungen:***

| | |
|---|---|
| **`200 OK`** | Die Übertragung war erfolgreich |
| **`201 Created`** | Eine neue Resource wurde auf dem Server erzeugt. |
| **`202 Accepted`** | Der Request wurde zur Verarbeitung akzeptiert |
| **`203 Non-Authorative Information`** | Der Header enthält Drittquellendaten. (Ansonsten gleichbedeutend zu 200 OK). |
| **`204 No Content`** | Der Webserver muß keinen neuen Inhalt senden. (Z.B. da keine Veränderung auftrat.) |
| **`204 Reset Content`** | Der User-Agent (z.B. Browser) muß die Seite neu aufbauen. |
| **`206 Partial Content`** | Ein partielles GET (GET mit Range-Angabe) war erfolgreich. |

***Umleitungen:***

| | |
|---|---|
| **`300 Multiple Choices`** | Die angefragte Seite verweist Server-intern auf mehrere verschiedene Dokumente. |
| **`301 Moved Permanently`** | Die angefragte Seite wurde permanent verschoben. Die nachfolgenden Tags geben die neue Position an. |
| **`302 Found (Moved temporarily)`** | Das Dokument wurde bewegt, in den folgenden Tags **`Content-location: <redirect>`** und **`Location: <URL>`** Befindet sich der Verweis auf die neue Speicherstelle des Dokuments. |
| **`303 See Other`** | Der Server liefert einen Verweis auf das Dokument zurück. |
| **`304 Not Modified`** | Rückgabewert für ein konditionales Get. |
| **`305 Use Proxy`** | Die angegebene Resource muß durch einen Proxy angesprochen werden. |
| **`307 Temporary Redirect`** | Der Server liefert einen Verweis auf das Dokument zurück, daß sich temporär an anderer Stelle befindet (für GET und HEAD-Operationen irrelevant). |

***Client Fehlermeldungen:***

| | |
|---|---|
| **400 Bad Request** | Die Anfrage wurde fehlerhaft aufgebaut. |
| **401 Unauthorized** | Der Zugriff ist nicht authorisiert. |
| **402 Payment Required** | - reserviert - |
| **403 Forbidden** | Die Anfrage wurde von dem Webserver abgelehnt. |
| **404 Document not Found** | Das angeforderte Dokument ist nicht vorhanden. |
| **405 Method not Allowed** | Der Request des User-Agent (Browsers) ist nicht gestattet. |
| **406 Not Acceptable** | Die akzeptierten Dokumenttypen des User-Agent können nicht bereitgestellt werden. |
| **407 Proxy Authentication Required** | Der Client muß sich zuerst bei einem Proxy authentisieren. |
| **408 Request Timeout** | Die Verbindung ist aufgrund eines Timeout abgebrochen worden. |
| **409 Conflict** | Die Resource verursachte einen Konflikt (z.B. durch ein R/W Lock der Datei auf dem Webserver) |
| **410 Gone** | Die angefragte Resource ist auf dem Webserver nicht mehr verfügbar, und keine Verweise auf die neue Lokation sind bekannt. (z.B. durch Permanente Löschung) |
| **411 Length Required** | Die Länge des Requests muß angegeben werden. |
| **412 Precondition Failed** | Die Vorbedingungen des vom User-Agent geschickten Request-Headers ergaben logisch „falsch“. |
| **413 Request Entity Too Large** | Die angefragte Länge ist zu groß. |
| **414 Request URI Too Large** | Der angefragte URI (Uniform Resource Identifier) ist länger als es der Server erlaubt. |
| **415 Unsupported Media Type** | Das angefragte Format (Document-Type) ist nicht verfügbar. |
| **416 Requested Range Not Satisfiable** | Der angefragte Bereich ist nicht verfügbar. |
| **417 Expection Failed** | Der im „Request-Header“ angegebene Wert kann durch den Server nicht erfüllt werden. |

***Server Fehlermeldungen:***

| | |
|---|---|
| **500 Internal Server Error** | Unerwarteter Serverfehler |
| **501 Not Implemented** | Diese Funktion ist nicht implementiert |
| **502 Bad Gateway** | Agiert der Server als Gateway oder Proxy, so meldet er so fehlerhafte Antwort des entfernt liegenden Servers. |
| **503 Service Unavailable** | Bei Überlastung des Servers oder temporärer Wartung. |
| **504 Gateway Timeout** | Agiert der Server als Gateway oder Proxy, so meldet er so ein Timeout in der Verbindung mit dem entfernt liegenden Server. |
| **505 HTTP Version Not Supported** | Die HTTP-Protokoll - Version wird nicht unterstützt. |
| **(506 Internal Configuration Error)** | (Serverfehler, noch nicht offiziell definiert) |

Zusätzlich definieren die Hersteller von WWW-Servern weitere, vom W3C Standard abweichende derartige Return-Codes (z.B. für Directory Zugriffsschutz beim Microsoft – Webserver)

Zeile 2: **`Server: Microsoft-IIS/4.0`**

Bei dem Server-System handelt es sich um Windows NT mit der WWW-Server Software „Internet Information Server" Version 4. Teilweise wird hier auch das Betriebssystem des Servers im Klartext mit angegeben (Linux / Solaris, etc.).

Zeile 3: **`Date: Sat, 10 Jan 2000 19:52:14 GMT`**

Datum und Uhrzeit der Übertragung der Datei (nach [RFC 1123] oder [RFC 850] oder asctime-Datum nach [RFC 2068])

Zeile 4: **`Content-Type: text/html`**

Typ der übertragenen Daten im Nachrichtenkörper.

Zeile 5: **`Accept-Ranges: bytes`**

Akzeptierter Zeichenumfang

Zeile 6: **`Last-Modified: Sat, 01 Apr 1999 06:51:37 GMT`**

Letzte Modifikation (nach [RFC 1123] oder [RFC 850] oder asctime-Datum nach [RFC 2068])

Zeile 7: **`ETag: "ee89b05da7b1bf1:26ea5a"`**

Serverspezifische Erkennungsnummer (Entity Tag)
*„Etag is the value of the HTTP Etag header associated wirth the requested entity, and private-key known only to the server."* [RFC 2617]

Zeile 8: **`Content-Length: 14734`**

Länge der übertragenen Datei – dieser Eintrag gehört ebenso wie **`Content-Type`** und **`Last-Modified`** zum Entity-Header. Der Entity-Header hat noch folgende weitere (optionale) Attribute:

| | |
|---|---|
| **`Content-Encoding:`** | Zeichensatzcode |
| **`Content-Language:`** | Sprache (en, de, ...) |
| **`Content-Location:`** | |
| **`Content-MD5:`** | |
| **`Content-Range:`** | |
| **`Expires:`** | Auslaufdatum der Seite |

***Diverse weitere Tags sind optional definiert:***

| | |
|---|---|
| **`Cache-Control:`** | Größe der Proxxy Caches |
| **`Transfer-encoding:`** | Art der Übertragung der Daten (z.B. „**chunked**" – Format, vgl. Kapitel 2 und [RFC 2616 - Kapitel 3.6.1.]) |

Für weitere optionale Tags siehe [RFC 2068] und [RFC 2616].

## 3.9. Crawling-Output des MWC

Nachstehend findet sich die gekürzte Ausgabe des MWC während des Crawlens einer Website[29]. Interessante Stellen wurden gesperrt gedruckt. Der Crawler arbeitet je nach vorliegendem Inhalt des Dokuments bestimmte Programmteile ab (zum Beispiel Javascript-Analyse, Frame-Analyse etc.)

13:24:42 **Crawling URL** http://agn-www.informatik.uni-hamburg.de/vtc/warn/loveletter_eng.htm
13:24:42 **Wandle HTML in Text**
13:24:42 Wandle HTML in Text - Zeilenumbrüche, Umlaute
13:24:42 Wandle HTML in Text - META-Tags
13:24:42 Wandle HTML in Text - Tags entfernen
13:24:42 **Prüfe externen Link** http://www.datafellows.com/v-descs/love.htm
13:24:42 ... bereits in Goodlist
13:24:42 Prüfe externen Link http://www.complex.is/f-prot/love.html
13:24:42 ...eingetragen
13:24:42 Prüfe externen Link http://support.cai.com/Download/virussig.html
13:24:42 ...eingetragen
...
13:24:42 **Heuristik**
13:24:42 Crawling URL http://agn-www.informatik.uni-hamburg.de/vtc/warn/ExploreZip.htm
...
13:24:43 Prüfe externen Link http://www.sophos.com/downloads/ide/index.html#explorez
13:24:43 ... bereits in Goodlist
13:24:43 Prüfe externen Link http://www.symantec.com/press/1999/n990610b.html
13:24:43 ... bereits in Goodlist
13:24:43 Heuristik
13:24:43 Crawling URL http://agn-www.informatik.uni-hamburg.de/vtc/urlsnoop/urlsnoop.htm
13:24:44 Wandle HTML in Text
13:24:44 Wandle HTML in Text - Zeilenumbrüche, Umlaute
13:24:44 Wandle HTML in Text - **Javascript entfernen**
13:24:44 Wandle HTML in Text - META-Tags
13:24:44 Wandle HTML in Text - Tags entfernen
13:24:44 Prüfe externen Link http://www.antivirus.com/vinfo/security/sa011199.htm
13:24:44 ... bereits in Goodlist
13:24:44 **Neue DATEI** http://agn-www.informatik.uni-hamburg.de/vtc/urlsnoop/unsnoop.exe
13:24:44 Neue DATEI http://agn-www.informatik.uni-hamburg.de/vtc/urlsnoop/unsnoop3.exe
13:24:44 Heuristik
...
13:25:35 Crawling URL http://agn-www.informatik.uni-hamburg.de/vtc/amigagrp/amav-2.htm
13:25:35 Wandle HTML in Text
13:25:35 Wandle HTML in Text - Zeilenumbrüche, Umlaute
13:25:36 Wandle HTML in Text - Javascript entfernen
13:25:36 Wandle HTML in Text - META-Tags
13:25:36 Wandle HTML in Text - Tags entfernen
13:27:04 Heuristik

[29] Hier wurde die Website des Fachbereichs AGN – Bereich VTC als Beispiel verwendet , um nicht übermäßig viele URLs von Malware-Seiten in dieser Arbeit zu nennen. (http://agn-www.informatik.uni-hamburg.de/vtc/eng1.htm)

## 3.10. Probleme beim Crawlen

Neben den Problemen der Linkverfolgung stellen triviale Paßworteingabeschirme ,in denen das Kennwort auf derselben Seite im Text angegeben wird, für den MWC eine unüberwindbare Hürde dar. Ebenfalls kann der MWC Viren, die erst in Form einer CGI-Such-Abfrage bereitgestellt werden, nicht übertragen.

Eine weitere Hürde sind unbekannte Dateiendungen von HTML-Dateien. Im Prinzip kann jeder Serveradministrator entscheiden, welche Endung seine HTML-Dateien bekommen. Der MWC kann aber vor der Übertragung einer Datei nur anhand der Endung zwischen HTML und anderen Dateien unterscheiden.

## 3.11. Crawling – Ausblick

Die HTML-Dateien und Textdokumente in gepackten Archivdateien, die nachweislich Malware enthalten, könnten für eine weitere Suche herangezogen werden. Bei einigen Archiven finden sich bis zu 10 verschiedene derartige Attachments von Virii / Malware-Gruppen.

Außerdem ist eine Überprüfung von Newsgroup-Verweisen einschlägiger Newsgroups in den WWW-Raum sowie die Untersuchung von bestimmten IRC-Kanälen erfolgversprechend.

# 4. Heuristische Textanalyse (Heuristik)

Als Verfahren der Relevanz-Bewertung benutzt der Malware-Crawler eine mehrstufige Heuristik. Nachfolgend zunächst die Definition von „Heuristik“[30] sowie die Ansätze anderer Autoren zur Realisierung heuristischer Verfahren im Internet.

***Definitionen Heuristik***

> *Eine Heuristik ist ein auf empirisch gewonnenen Erfahrungen, Hypothesen oder Faustregeln gestütztes Verfahren zur Lösung einer Aufgabenstellung.* [PLN]
>
> *Heuristik – Algorithmen zur Lösung komplexer Probleme verbessert man häufig durch Strategien, die oft auf Hypothesen und Vermutungen aufbauen und die mit hoher Wahrscheinlichkeit (jedoch ohne Garantie) das Auffinden einer Lösung beschleunigen sollen. Solche Strategien heißen Heuristiken. Faustregeln, bereits früher beobachtete Eigenschaften von Lösungen oder die Nachbildung des menschlichen Problemlösungsprozesses sind typische Heuristiken.* [DINF]

Bei der vom MWC eingesetzten Heuristik handelt es sich nicht um die sonst im Zusammenhang mit Viren häufige Bewertung von Codesegmenten, beziehungsweise Codeemulation[31], sondern um eine Heuristik, die auf den Texten der Webseiten und deren Verknüpfungen und Standorten aufsetzt.

Einige Theorien über die Verwendung von Heuristiken zur Suche im Internet sollen im folgenden betrachtet werden:

***Ellen Spertus*** hat in ihrer Arbeit zum Auffinden von Homepages [SPERTUS96] im WWW-Raum einige interessante Ergebnisse gewonnen. Sie unterscheidet bei Verknüpungen in Form von Hyperlinks unter Dokumenten im WWW allgemein zwischen:

- *upward-Links:* Hyperlinks auf Seiten, die eine oder mehrere Directoryebenen höher liegen.
- *crosswise-Links:* Hyperlinks, die auf Parallelverzeichnisse verweisen.
- *downward-Links:* Hyperlinks, die auf Unterverzeichnisse verweisen.
- *outward-Links:* Hyperlinks, die auf andere URLs verweisen.

Hierbei bemißt sie den upward-Links eine für den aktuellen Zusammenhang geringere Bedeutung bei als den anderen Link-Arten.

Im folgenden definiert sie diverse Heuristiken, wobei hier nur die wesentlichen Ergebnisse wiedergegeben werden sollen:

**Externer Link:** Aufgesetzt auf einem Index[32] ist jede Seite, die durch einen einzelnen externen Hyperlink erreichbar ist, mit hoher Wahrscheinlichkeit W demselben Themengebiet zugehörig.

[30] von griechisch heurískein = finden, entdecken

[31] Siehe auch [SCHM98,S.17ff] und http://www.bsi.de/antivir1/virbro/erklaer/heuri.htm

[32] Hauptsächlich geordnete Verzeichnisse und Linklisten

**Wiederholter externer Link:** Aufgesetzt auf einem Index P und folgend zu einem externen Link P‘ wird eine weitere extern referenzierte Seite mit der Wahrscheinlichkeit W‘ zu demselben Themengebiet wie P (oder einer Verfeinerung davon) gehören. (Transitivität der Themenvermutung). W‘ <= W

**Directory/Hyperlink Korrelation:** Wenn eine Seite P oberhalb einer Seite P‘ in der Directorystruktur steht, so existieren sehr wahrscheinlich Hyperlinks zwischen P und P‘ (insbesondere wenn P eine Homepage ist).

**Lokalität der Referenzen:** Wenn zwei URLs $U_1$ und $U_2$ „nah“ zueinander auf einem Webserver stehen, dann haben sie wahrscheinlich ähnliche Themengebiete oder eine andere gemeinsam geteilte Eigenschaft. „Nah“ kann hierbei als Abstand in der strukturellen Hierarchie der Seite definiert werden.

**Zeitliche Gültigkeit:** Wenn eine Seite R eine Seite P in der Vergangenheit referenziert hat, dann existiert eine große Wahrscheinlichkeit, daß R die Seite P referenziert, auch wenn die URL von P sich verändert hat.

Erkenntnisse, die sich für diese Arbeit hieraus ableiten lassen:

> Hyperlinks, die von einer Malware-Seite auf deren Unterverzeichnisse oder Parallelverzeichnisse verweisen, führen sehr wahrscheinlich auch zu Malware-Seiten (beziehungsweise H/P/A/V) Inhalten.[33]
>
> URLs die über (auch mehrfache) externe Hyperlinks des Ursprungs URL erreichbar sind, enthalten ebenfalls sehr wahrscheinlich Malware. Dies gilt auch für strukturell zur Ursprungs-URL ähnliche Sites.
>
> Wenn eine Malwareseite ihre URL ändert, so kann diese weiterhin sehr wahrscheinlich durch Verfolgen der Hyperlinks von anderen Malwareseiten, die zuvor auf diese Seite verwiesen haben, erneut gefunden werden.

Der Malwarecrawler berücksichtigt diese Erkenntnisse in seiner Implementation in der Heuristik Stufe II sowie in der Heuristik Stufe IV (vgl. Kapitel 4.2 und 4.4).

[33] Dies kann am Beispiel www.virusexchange.org nachvollzogen werden.

Ein komplett anderes Verständnis von einer heuristischen Suche im Internet haben ***Jonathan Shakes et. al.*** [SHAKES 97]. Ebenfalls mit dem Ziel, bestimmte Homepages im Internet zu finden wird ein mehrstufiges Verfahren[34] in dem Projekt Ahoy![35] verwendet.

Dieses Verfahren besteht aus den folgenden 6 Komponenten:

| | |
|---|---|
| 1. Referenzquelle | Als Quelle für mögliche Kandidaten im Endergebnis wird eine herkömmliche, indexbasierte Suchengine (vgl. Kapitel 2) wie z.B. Altavista oder Excite verwendet. |
| 2. Filterkomponente | Die Filterkomponente filtert nun mittels einer zuvor angelegten Datenbank oder Liste die sinnvollen Referenzen aus dem Suchergebnis. (Diese Datenbank ist im Fall von Ahoy eine Liste generierter E-Mail Namen, die der Suchanfrage entsprechen könnten.) |
| 3. Heuristik-Filter | Diese Komponente erhöht die Präzision durch die Analyse der Textinhalte auf der Ebene der von der Suchengine zurückgelieferten ersten Zeilen bzw. der zurückgelieferten Meta-Tags. |
| 4. Datenlisten | Diese Komponente kategorisiert und bewertet die Ergebnisse in Form einer Reihenfolge, wobei zwischen „Treffern“ und ähnlichen Ergebnissen unterschieden wird. |
| 5. URL-Generator | Diese Komponente generiert anhand der eingegebenen Suchbegriffe URLs, wenn die Methoden 1 - 4 keine Erfolge erzielten. |
| 6. URL-Mustergenerator | Diese Komponente merkt sich für die erfolgreich besuchten URLs die benutzten Verzeichnisse, in denen Homepages gespeichert werden sowie die Benennung der Verzeichnisse in Abhängigkeit vom gewünschten Namen, der gesucht wurde. Diese Werte werden auch in „URL-Generator“ Komponente weiterverwendet. |

Einige der hier beschriebenen Verfahren wurden bei der Realisierung des Malware-Crawlers verwendet. Die Startliste der URLs, die hier in Phase 1 mittels einer Suchanfrage dynamisch generiert wird, ist ähnlich zu der in dieser Arbeit manuell generierten Start- „Badlist“ – wobei weitere „candidates“ bei dem Malwarecrawler durch Crawling erzeugt werden – beim obigen DRS-Verfahren werden URLs generiert beziehungsweise der Reference-source entnommen.

Die Cross-Filter Komponente von Ahoy prüft textuelle Inhalte der „candidates“ gegen eine eindimensionale Datenbank - allerdings findet sich folgende Einschränkung:

> *Ahoy!'s decision whether to categorize a given reference as a homepage is based entirely on the reference's title, URL, and a short textual extract (if the later is supplied by the search engine). We conjecture that it is unnecessary to download the entire text of a reference or use natural language processing to assist categorization.* [SHAKES 97]

[34] Dieses Verfahren wird dort *DRS – Dynamic Reference Shifting* genannt.

[35] Aufruf der Engine und weitere Informationen bei [SHAKES 99]

Für die Suche nach Webseiten, die von normalen Personen (im Bereich von Ahoy sogar zumeist nur von Wissenschaftlern und Studenten) erstellt wurden, sind die Verfahren von Ahoy adäquat. Die Suche nach Webseiten, die sich bewußt tarnen und häufig ihren Namen und ihre Lokation wechseln, wird so jedoch kaum möglich. Die Seiten, die Malware anbieten, sind nur zu einem geringen Prozentsatz in den Suchengines vertreten und tragen meist keine aussagekräftigen Seitenbeschriftungen in Form von Meta-Tags oder in der Form von durch die Suchengines leicht zu verarbeitendem Text.

Durch den Verzicht auf die Crawling-Komponente ist das DRS-Verfahren auf die Suchengines angewiesen, wobei so beim derzeitigen Stand an WWW-Dokumenten (vgl. Kapitel 2) selbst bei Benutzung des größten Verzeichnisses nur maximal 50% des WWW-Raums durchsucht werden. Für das Auffinden von Homepages im universitären Bereich kann dies dennoch ausreichen, da die Websites der Universitäten regelmäßig indiziert werden und sehr selten die Domainnamen wechseln – für Malwaresites mit einer sehr geringen „UP-Time“ (Lebensdauer) ist dieses Verfahren jedoch nicht geeignet.

Konträr zur Arbeit von Ellen Spertus finden in der Arbeit von Shakes et. al. die Hyperlinks zwischen verschiedenen Dokumenten keinerlei Berücksichtigung.

Dennoch bietet diese Arbeit einen brauchbaren Ansatz für eine heuristische Bewertung von Textinhalten mittels einer Datenbankquelle. Dieser Ansatz wurde bei der Realisierung des MWC konsequent weiterverfolgt, wobei beim MWC keine flache Datenbank, sondern eine mehrstufige Bewertung erfolgt (vgl. Kapitel 4.1.).

Die dynamische Generierung von URLs anhand der Suchanfrage wäre eine interessante Erweiterung der Crawling-Komponente. Es gibt jedoch beim Malwarecrawler keine einzelne Anfrage sondern ein Heuristik-Set. Anhand dieses Heuristik-Sets könnten, ausgehend von einer URL $U_1$ , weitere URLs $U_2$ durch simple Konkatenation der bekannten URL mit dem Keyword des Heuristik-Sets und einem Standard-Dokument (wie zum Beispiel „**virii.html**“ oder „**/virii/index.html**“) erfolgen.

Bei der dynamischen Generierung von URLs kommt es zwingend zu fehlerhaften Seitenanfragen an den entfernten Server. Da eine fehlgeschlagene Anfrage zu einer starken Verzögerung des Crawlens durch Abwarten des „Timeout“ beziehungsweise „404-Dokuments“[36] führt, wurde hier von der Realisierung im MWC Abstand genommen.
Zudem ist die Wahrscheinlichkeit, daß ein Unterverzeichnis nicht durch Hyperlinks erreichbar ist, relativ gering (vgl. Elen Spertus Heuristik).

[36] Server-generiertes Dokument, das aussagt, daß die entsprechende Seite nicht gefunden werden konnte.

Eine Heuristik auf Basis der Suchbegriffe beschreibt ***Udi Manber et.al.*** in seiner Arbeit: *Connecting Diverse Web Search Facilities*. [MANBER_USI].

In der Arbeit von Udi Manber et.al. werden die unterschiedlichen Fragestellungen und verbalen Formulierungen der Suchenden mit Hilfe einer Datenbank vor der eigentlichen Anfragestellung standardisiert. Hierbei wird versucht, die Probleme einfacher Web-Indizes, natürlichsprachliche Anfragen zu verarbeiten, zu umgehen.

Als Beispiele nennt er (unter anderem) folgende typische Fragen von Web-Surfern, die in herkömmlichen Suchengines keine befriedigenden Ergebnisse liefern können:

1. *How much fat is there in a pepperoni Piza ?*
2. *How do you say „search“ in Latin ?*
3. *Howdo you delete a directory in Unix ?*
4. *Give me a list of hotels in Phoenix*

In dem vorgestellten Verfahren des *„two level search“* wird nun in der ersten Phase die richtige Datenbank für die Suche gewählt, in der zweiten Phase wird die aktuelle Information aus dieser Datenbank ermittelt. Interessanter Part ist die Auffächerung der eingegebenen Fragen in mögliche Abwandlungen der ursprünglich gegebenen Schlüsselbegriffe (Subjects) jeweils für die entsprechende Suchengine. So werden die obigen Beispiele wie folgt behandelt:

1. *Subject: nutrition; Query: „pizza“*
2. *Subject: english-to-latin; Query: „search“*
3. *Subject: unix; Query: „delete directory“*
4. *Subject: hotels; Query: „Phoenix“*

Bei der weiteren Suche wird eine von Experten erstellte Datenbank der Kataloge und deren Aufrufschemata und Suchbegriffe verwendet.

Die Technik, anhand eines Suchbegriffs mehrere Ableitungen dieses Begriffs ebenfalls für die Suche zu verwenden, wurde im Malware Crawler für die heuristische Bewertung benutzt. Bei dieser Realisierung werden jedoch keine Suchengines abgefragt, sondern direkt die übertragenen Seiten der besuchten URL betrachtet.

(vgl. auch Kapitel: Heuristik-Ausblick).

Eine Heuristik zum Auffinden von relevanten Änderungen an Webseiten stellt **Brian Starr et.al.** in der Arbeit „Do I Care? Tell me what's changed on The Web" vor. [STARR]

1. Der in dieser Arbeit vorgestellte Agent besucht periodisch eine vom User vorgegebene Anzahl an URLs.
2. Alle Änderungen werden identifiziert.
3. Im Falle von Änderungen wird geprüft, ob diese relevant sind.
4. Der Anwender wird über die Änderung informiert.
5. Anhand des „Feedbacks" des Anwenders wird die Bewertung der Relevanz verfeinert.

Der interessante Ansatzpunkt ist, daß nicht alle Änderungen gemeldet werden – die Relevanz wird durch einen lernenden Agenten[37] bewertet. Anhand der vorgefundenen Wörter in dem HTML-Dokument und der Worte, die mutmaßlich eine „interessante" Veränderung kennzeichnen, wenn diese neu hinzukommen, wird der Output generiert. Hierfür werden Dictonarys relevanter Worte verwendet sowie ein Verfahren, das ggf. den neu hinzugekommenen Text auf Länge, Enthalten von Hyperlinks und Keywords untersucht.

Der Malwarecrawler erkennt Änderungen derzeit nur anhand einer einfachen Checksumme – ein „Feedback" durch einen Experten wäre jedoch ebenso wie ein „Feedback" durch die eingesetzten AV-Produkte denkbar.

Problematisch an diesem Verfahren ist allerdings die Datenmenge, die gespeichert werden muß – die Texte der HTML-Dokumente müssen nahezu komplett gesichert werden, um eine Veränderung zu erkennen. Das vorgestellte Verfahren wird so auch in der Arbeit von Brian Starr nur für einzelne, manuell vom User ausgesuchte URLs, die überwacht werden sollen, verwendet.

In einer leichten Abschwächung der Forderungen von Starr wäre eine Realisation im MWC hingegen denkbar: Die Anzahl der internen und externen Hyperlinks von einem Dokument ließen sich schnell und platzsparend speichern und würden ein relativ aussagekräftiges Kriterium für einen solchen Ansatz bilden.

**Brian T. Bartell et.al.** verfolgen den Ansatz der Experten -Vorauswahl und -Kombination für die Durchführung von Recherchen im WWW-Raum [BARTELL]:

Hierbei werden die zum Erfolg führenden Anfragen (zum Beispiel an Web-Kataloge) durch eine Heuristik vorausgewählt. Dabei wird je nach Art der gestellten Frage der oder die entsprechende(n) Experte(n) zugeordnet.

Als Beispiel nennt Bartell einen „Phrase" – Experten, der eingegebene Suchbedingungen in Form von Phrasen besonders gut verarbeiten kann und einen „Term" – Experten, der Suchanfragen, die aus mehreren Termen bestehen, besonders gut verarbeiten kann. Die eingesetzte Heuristik wählt dann anhand der „Noise"-Wörter (zum Beispiel „and", „or", ...), die in Phrasen vorkommen, zwischen diesen beiden Experten.

In der Kombination dieser beiden Experten werden nun sowohl Anfragen, die aus „Termen" bestehen, als auch Anfragen aus „Phrasen" erfolgreich bearbeitet.

Anhand von 228 Testfragen auf dem Wortraum der „Enzyclopädia Britannica" wurde bewiesen, daß die Kombination zweier Experten bessere Ergebnisse liefert als die Anwendung nur eines der beiden Verfahren.

[37] Agenten als Suchverfahren werden auch in Kapitel 6 näher erläutert

## 4.1. Heuristik Stufe I – Textbewertung

Bei der heuristischen Textbewertung wird der Inhalt der HTML-Seite bewertet, wobei zuvor jegliche Formatierungsanweisung und eventueller Programmcode entfernt wird. Die Textbewertung findet rein auf der Ebene der Textinhalte und der META-Tags (der eventuell vorhandenen Suchbegriffe für Suchengines) statt.

Die Bewertung erfolgt hierbei durch gewichtete Keywords, die aus bekannten Malware-Seiten extrahiert wurden. Bei der Aufstellung der Keywords wurden zuvor durch ein halbautomatisches Differenzverfahren die ungeeigneten Worte, die auch auf normalen Webseiten vermehrt auftreten, aus der Liste der Keywords entfernt.

Bei der Wahl der Keywords muß nach Filippo Mencczer [MENCZER_AD2] beachtet werden, daß alle Beispiele einer Sprache inclusive der Dokumente, die durch Web-Indizes indiziert wurden, sehr stark vom gemeinsamen Kontext abhängen. Für das Verständnis eines Textes macht ein Autor implizite Annahmen über das Auditorium. Für Veröffentlichungen in traditionellen Medien, wie zum Beispiel in Konferenzbänden oder in akademischen Publikationen, trifft diese implizite Annahme im allgemeinen zu, und die Leser verstehen den Kontext der Publikation. Anders hingegen ist dieses im WWW-Raum, da hier virtuell jede Person jedes Dokument einsehen kann – auch wenn diese Personen nicht den Kontext des Autors verstehen.

Das Wort „Virus“ kann so beispielsweise nicht nur auf den vom Malware Crawler gesuchten Seiten vorkommen – ebenso kann es bei den Antivirenherstellern oder in Publikationen über biologische Viren auftreten. Hier wird implizit über den Zusammenhang der Seiten (zum Beispiel medizinisches Journal) die Art des gemeinten Objekts klar – eine automatische Suchengine kann diesen Gesamtzusammenhang jedoch kaum erkennen.

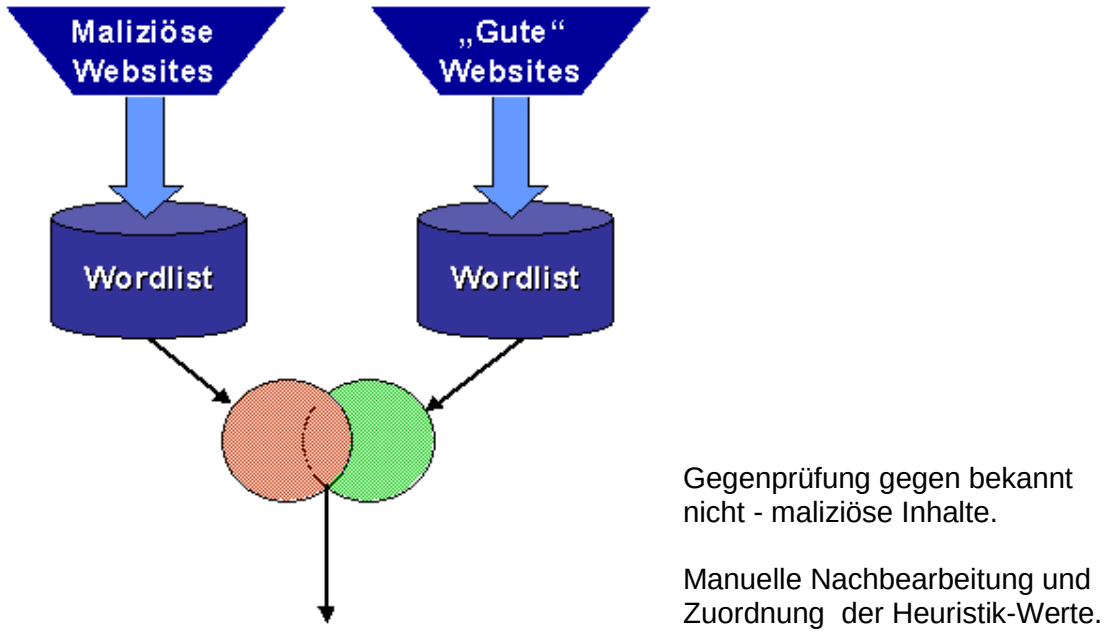



**Abbildung 9 - Differenzverfahren zur Keywordermittlung**

Zusätzlich zu den automatisch extrahierten Keywords wurde eine Expertenbefragung unter den Teilnehmern der einschlägigen AGN-Veranstaltung zum Thema IT-Sicherheit durchgeführt (vgl. Anhang A). Aus dieser Befragung entstanden weitere Anhaltspunkte für die Gewichtung der Keywords und die Erstellung des Heuristik-Sets.

### 4.1.1. Finden von Startwerten für das Differenzverfahren

Das Finden von Start-URLs für das Differenzverfahren beruht auf folgenden Verfahren:

***1. Berücksichtigung von in anderen Arbeiten genannten URLs***

Markus Schmall [SCHM98, S.13] nennt in seiner Diplomarbeit einige Malware-Sites. Leider konnten keine weiteren Arbeiten neueren Datums mit direkten Verweisen auf Malware-Sites gefunden werden. Auch sind die erwähnten Sites von [SCHM98] zum größten Teil nicht mehr erreichbar beziehungsweise enthalten keine Malware mehr, so daß hier keine Startwerte entnommen werden konnten.

***2. Manuelle Recherche***

Folgende Meta-Engines wurden zum manuellen Suchen von Malware-Sites benutzt:
**www.metacrawler.com**, Keyword „virii“
**www.kryltech.com**, Keywords „virii“, „viruz“, „trojan“

***3. Halb-Automatische Recherche mittels des Malwarecrawlers***

Mittels des Query-Strings: „virii“

**http://www.altavista.com/cgi-bin/query?pg=q&sc=on&q=virii&kl=XX&stype=stext&search.x=39&search.y=5**

wurde die Suchmaschine Altavista gecrawlt (also die Suchliste komplett verfolgt, ebenso wie alle Vorschläge der Engine wie zum Beispiel „writing virii“ „virus download“ etc.). Die Ergebnisse dieser Suche wurden manuell besichtigt und ggf. in die Tabelle der maliziösen Seiten eingetragen.

Ein ähnlicher Versuch bei HotBot scheiterte, da HotBot die Verweise auf externe Sites durch ein internes CGI realisiert (vgl. Kapitel 3). Der Aufruf für die HotBot-Recherche lautet:

**http://hotbot.lycos.com/?MT=VIRII&SM=MC&DV=0&LG=any&DC=10&DE=2&BT=H**

Eine Auswertung hätte jedoch unverhältnismäßig viel manuelle Nacharbeit bedeutet.

***4. Newsgroups***

Einschlägige Newsgroups

**alt.2600**
**alt.comp.virus**
**alt.comp.virus.source**

wurden auf URL-Verweise durchsucht

***5. Fragebögen***

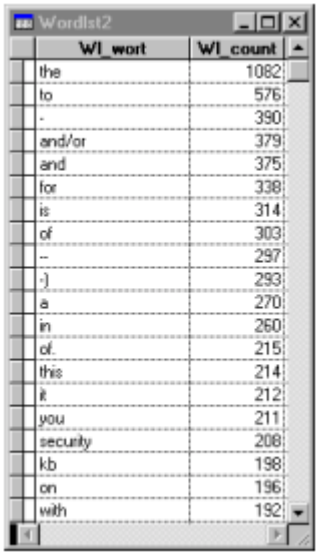
Im Rahmen des in Anhang A abgedruckten Fragebogens wurden auch entsprechende URLs beziehungsweise Top Level Domains genannt. Diese Seiten wurden dann manuell untersucht und in die Liste der Seiten, die maliziöse Inhalte anbieten, aufgenommen.

## 4.1.2. Generierung von Keywords aus bekannten maliziösen Seiten.

Mittels einer automatischen Prozedur wurden alle Worte von den in der Startliste angegebenen URLs in eine Textdatei geschrieben. Diese Textdatei hat das folgende Aussehen:


**Abbildung 10 – Wortliste verschiedener maliziöser Seiten**

Die Worte dieser Textdatei wurden dann gezählt und in eine Tabelle eingeordnet (tokenisiert). Diese Tabelle findet sich nun in der folgenden Abbildung:


Wordlst2

| WI_wort | WI_count |
|---|---|
| the | 1082 |
| to | 576 |
| - | 390 |
| and/or | 379 |
| and | 375 |
| for | 338 |
| is | 314 |
| of | 303 |
| -- | 297 |
| :) | 293 |
| a | 270 |
| in | 260 |
| of. | 215 |
| this | 214 |
| it | 212 |
| you | 211 |
| security | 208 |
| kb | 198 |
| on | 196 |
| with | 192 |

**Abbildung 11 – Wortliste sortiert nach Häufigkeit**

Basierend auf der Wortliste erfolgte dann die Entfernung von nicht als Keyword relevanten Textinhalten. Hierbei wurden die textintensiven Seiten von **www.welt.de**, **www.spiegel.de** und **www.cnn.com** inclusive ihrer Unterseiten (mittels Crawling) für die Generierung der Wortliste „guter" Websites Verwendet.[38]

Dieses Verfahren läßt sich nicht vollständig automatisieren, da zum Beispiel das Wort „Virus" in Pressemeldungen ebenso wie auf Malware-Servern erscheinen kann.

Beispiele:

| | |
|---|---|
| **„Virus"** | kommt sowohl in normalen als auch in Malware-Seiten vor. |
| **„Analysis"** | kann entfallen, da fast nur von AV-Herstellern verwendet. |
| **„Viruz"** | wird fast nur von Virenherstellern verwendet, eignet sich als Keyword. |

In der folgenden Abbildung nun die bereinigte Wortliste nach der Überarbeitung:

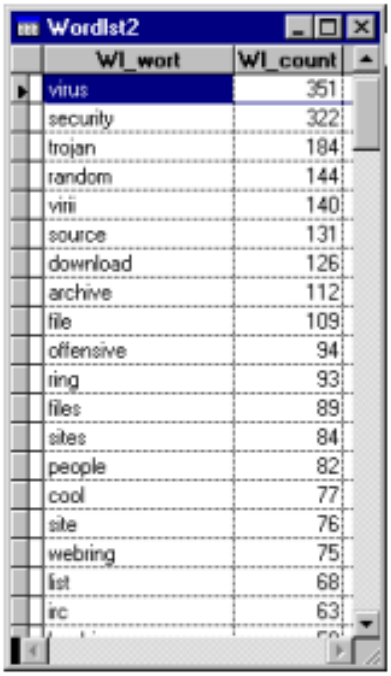



**Abbildung 12 – Halbautomatisch bereinigte Wortliste**

[38] Dieses geschah zu einem Zeitpunkt vor der anhaltenden Berichterstattung über Computerviren aufgrund der aktuellen Vorfälle.

Das vollständige Heuristik-Set des Malware-Crawlers hat das folgende Aussehen:

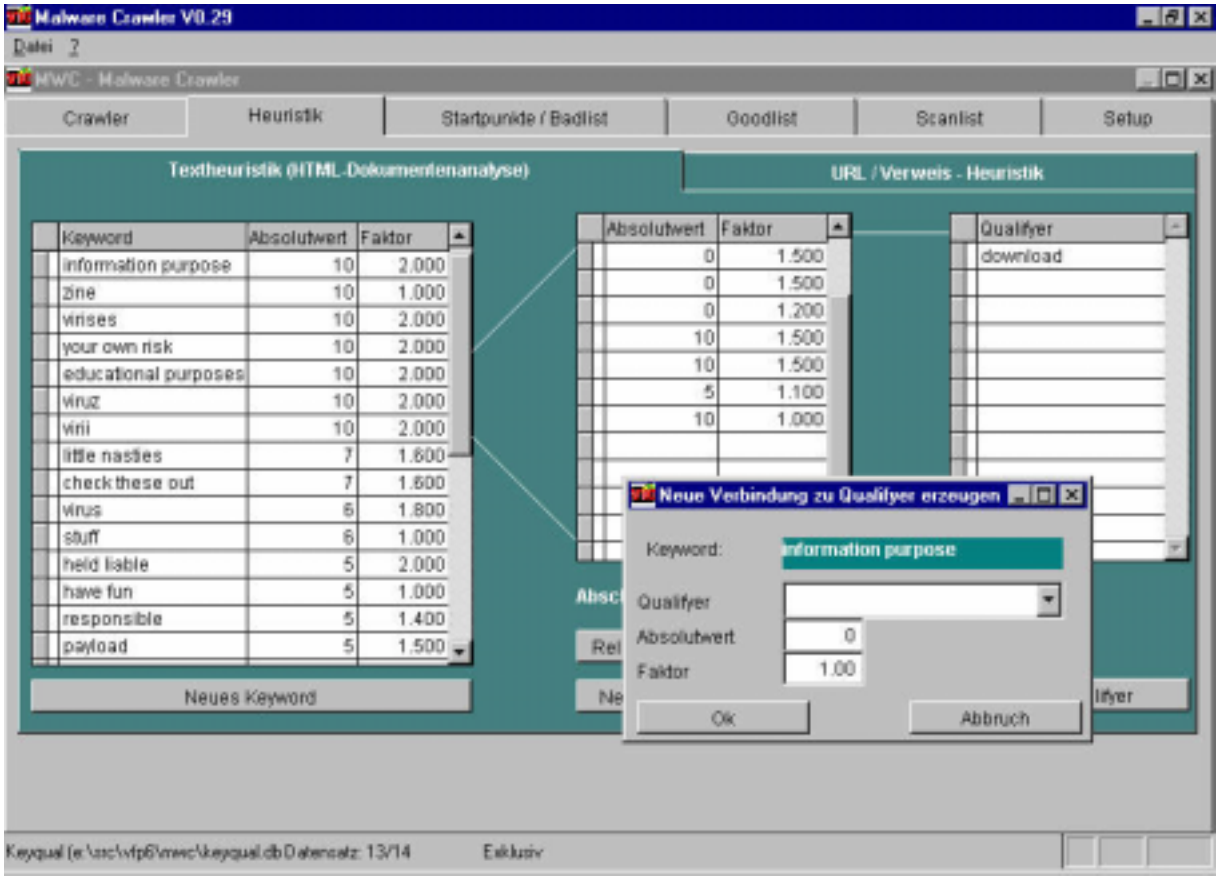



**Abbildung 13 – Vollständiges Heuristik-Set**

## 4.1.3. Generierung von Keywords aus anderen Quellen.

Für die Startwerte erfolgten entsprechende Befragungen von (ehemaligen und aktiven) Studenten am Fachbereich AGN. Von 50 Fragebögen wurden 17 zurückgegeben – dies stellt keine repräsentative Umfrage dar, jedoch ließen sich aus den Antworten weitere Informationen für diese Arbeit ableiten.

> *Retrieval Performance can be greatly improved by using a number of different retrieval algorithms (or experts) and combining their results, in contrast to using just a single retrieval algorithm. Each expert contributes ist estimates of which documents are likely to be relevant to the user's query, and the combined set is typically more valuable than any single expert's estimates.* – Brian Bartell et.al. [BARTELL]

Diese von B. Bartell eigentlich für Algorithmen geschriebene Aussage läßt sich ohne weiteres in die reale Welt übertragen. Bei der Sichtung der im vorigen Kapitel automatisch generierten Keywords kam es bereits zu Häufungen für Wörter, die nicht offensichtlich waren. Die im folgenden vorgenommene Auswertung der Fragebögen trug nun zu einer weiteren Verbesserung des Heuristik-Sets durch dem Autor selbst nicht offensichtliche Keywords bei (wie zum Beispiel das Wort „undetected“, etc.).

***Die im Fragebogen (Anhang A) angegebenen Suchbegriffe für Such-Engines***

| | | |
|---|---|---|
| Backdoor | \|\| | |
| Badexe | \| | |
| COM | \| | ← Manuell entfernt, da als Major-Keyword untauglich |
| Cracked | \| | |
| Damage | \| | ← Manuell entfernt, da als Major-Keyword untauglich |
| Destroy | \| | ← Manuell entfernt, da als Major-Keyword untauglich |
| Download | \|\|\| | ← Besser als Qualifyer geeignet |
| Dropper | \|\| | |
| EXE | \| | ← Manuell entfernt, da als Major-Keyword untauglich |
| Exploit | \| | |
| Gamez | \| | |
| Gain administrator rights | \| | |
| Get administrator rights | \| | |
| Hacked | \| | |
| Hacking | \| | |
| Hostile code | \| | |
| Hostile applet | \| | |
| Hostile source | \| | |
| Hostile program | \| | |
| Irqtrouble | \| | |
| Malware | \|\|\|\|\| | |
| Malicious | \| | |
| Macrovirus | \|\|\| | |
| Nightmare Joker | \| | |
| Resident | \| | ← Besser als Qualifyer geeignet |
| Retro Virus | \| | |
| Remote | \| | ← Besser als Qualifyer geeignet |
| Remote administrator kit | \| | |
| Serialz | \| | |
| Systools | \| | |
| Tojan | \|\|\|\|I\|\|\|\| | |
| Trojans | \|\|\| | |
| Trojanz | \| | |
| Tsr | \| | ← Besser als Qualifyer geeignet |
| Uncqre | \| | |
| Vicodines | \| | |
| Viral | \| | |
| Viren | \|\|\|\| | |
| Virus | \|\|\|\|\|\|\|\|\|\|\|\| | |
| Virus Zines | \| | |
| Virus BBS | \| | |
| Viruses | \|\|\|\|\| | |
| Viruz | \|\|\|\|\|\|\| | |
| VX | \| | ← Besser als Qualifyer geeignet |
| Virii | \|\|\|\|\|\|\|\| | |
| Wirus | \| | |
| Wiruz | \| | |
| Warez | \|\|\| | |
| Worm | \| | |
| Zines | \| | |
| ZIP | \|\| | ← Besser als Qualifyer geeignet |
| .ru | \| | |
| „Namen von Viren“ | \|\| | |
| „ -„- von VX- Gruppen“ | \| | |

### *Empfohlene Suchengines*

| | | |
|---|---|---|
| Alltheweb | \| | |
| Altavista | \|\|\|\|\|\|\|\| | |
| Astalavista.box.sk | \|\|\|\|\| | |
| Fireball | \| | |
| Filez.com | \| | |
| Go.com | \|\| | |
| Google | \|\|\|\|\| | |
| Hotbot | \|\| | |
| Lycos | \| | |
| Metacrawler | \|\|\|\| | |
| Warez.com | \|\| | |
| Yahoo | \|\| | ← Irrelevant, Siehe Kapitel 2 |
| Usenet HTML-Verweise | \|\| | ← Siehe Kapitel 4.1.1. |
| FTP-Search | \| | |

### *Die im Fragebogen angegebenen charakteristischen Keywords*

(Die Keywords sollten nicht unbedingt identisch zu Suchbegriffen sein – hier wurden zum Beispiel Begriffe wie „Download“ für die Qualify-Liste erwartet, die in der eigentlichen Suche keinen Sinn machen würden.)

| | | |
|---|---|---|
| Anti-Heuristic | \| | ← Als Major Keyword geeignet |
| Archive | \|\| | |
| ASM | \| | |
| Backdoor | \|\| | |
| Boot | \|\| | |
| Collection | \|\| | |
| C00l | \| | ← Als Major Keyword geeignet |
| Crack | \|\| | ← Als Major Keyword geeignet |
| Cracker | \| | ← Als Major Keyword geeignet |
| Damage | \| | |
| Disclaimer | \| | |
| Download | \|\|\|\|\|\|\|\|\|\| | |
| Dropper | \| | ← Als Major Keyword geeignet |
| Educational Purpose | \| | |
| Exchange | \| | |
| File | \|\| | |
| Guru | \| | |
| Group | \| | |
| Hack | \|\| | ← Als Major Keyword geeignet |
| Hacker | \| | ← Als Major Keyword geeignet |
| Hacking | \| | ← Als Major Keyword geeignet |
| Hoax | \| | |
| Illegal | \| | |
| Link | \| | |
| Malware | \| | ← Als Major Keyword geeignet |
| Macro | \|\|\|\|\| | |
| Macroviren | \| | ← Als Major Keyword geeignet |
| Micro$oft | \| | ← Als Major Keyword geeignet |
| Monthly Archive | \| | |
| Mpz | \| | |
| New | \| | ← Ungeeignet, entfernt |

| | | |
|---|---|---|
| New Virus | \| | |
| Not Detected By | \| | |
| Polymorphic | \| | |
| Replicate | \| | |
| Resident | \| | |
| Serial | \| | |
| Spread | \| | |
| Script | \| | |
| Source | \|\|\| | |
| Src | \| | |
| Stuff | \| | |
| Toolz | \| | ← Als Major Keyword geeignet |
| Tools | \| | |
| Trojan | \|\|\| | ← Als Major Keyword geeignet |
| Trojanisches Pferd | \| | ← Als Major Keyword geeignet |
| Trojanz | \| | ← Als Major Keyword geeignet |
| Test | \| | ← Entfernt, da ungeeignet |
| Undetected | \| | |
| VBA | \|\| | |
| Viren | \|\| | ← Als Major Keyword geeignet |
| Virii | \|\|\|\|\|\|\|\| | ← Als Major Keyword geeignet |
| Viris | \| | ← Als Major Keyword geeignet |
| Viruz | \|\|\|\|\| | ← Als Major Keyword geeignet |
| Virus | \|\|\|\|\|\| | ← Als Major Keyword geeignet |
| Viruses | \| | ← Als Major Keyword geeignet |
| Warez | \|\|\|\| | ← Als Major Keyword geeignet |
| Word | \|\|\| | |
| ZIP | \|\|\| | |
| Zone | \| | |
| „Namen von Viren“ | \|\| | ← Teilweise als Major Keyword geeignet |

## 4.1.4. Aufteilung in Keyword und Qualifyer

„Download“ würde sicherlich ein schlechtes Keyword abgeben, da es auf diversen Webseiten zu finden ist. Im Zusammenhang mit „Virus“ ist dieses Wort hingegen recht interessant.

Beim Einsatz einer einfachen, flachen Dateistruktur für die heuristische Bewertung, könnte diese Erkenntnis nicht abgebildet werden. Download hätte entweder eine heuristische Relevanz oder nicht. Mittels der Aufteilung in Keywords und Qualifyer kann der Malware-Crawler derartige Zusammenhänge abbilden. Darüber hinausgehend kann der MWC den Heuristik-Wert aufgrund eines Keywords nicht nur erhöhen, sondern auch erniedrigen.

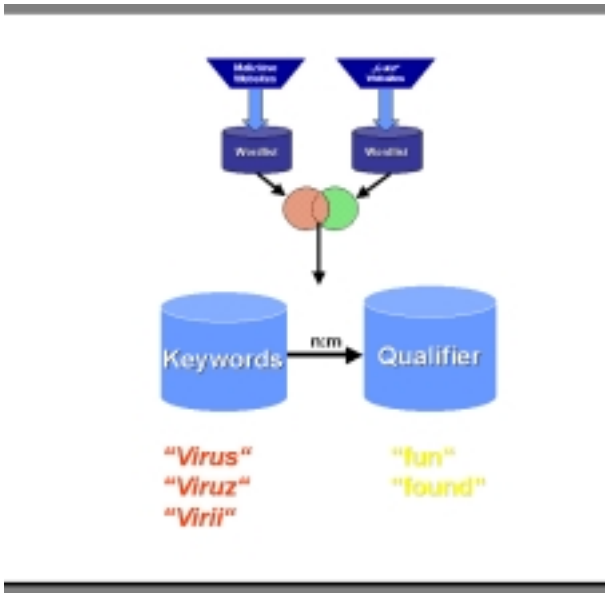



**Abbildung 14 – Keyword – Qualifyer Beziehung**

***Beispielsätze:***

1. Here is our **virus**, have fun with it[39]
2. Alert: New **virus** found on xxx’s Website

Beide Sätze enthalten dasselbe Keyword, nämlich „**Virus**“, wobei die erste Seite im Bereich der zu durchsuchenden Seiten liegt, und die zweite Seite für den Malware-Crawler nicht interessant ist.

Wenn man nun davon ausgeht, daß das Wort „**fun**“ bei seriösen Webseitenanbietern nicht in direkter Nähe zu dem Wort „**Virus**“ gebraucht wird, so kann man darauf aufbauend den Heuristik-Wert entsprechend erhöhen, wenn eine solche Konstellation gefunden wird.

Analog dazu wird ein Betreiber einer Malware-Seite höchst selten Viren-Alerts publizieren. In diesem Zusammenhang kann man also im Falle des Auffindens der Wörter „found“ oder „alert“ in der Nähe des Keywords „virus“ den Heuristik-Wert entsprechend erniedrigen.

[39] Vgl. auch Abbildung 1 in der Einleitung.

### 4.1.5. Verwendete Datenbankstrukturen

**KEYWORDS** – Tabelle der Keywords für die Heuristik

| Feld | Typ | Inhalt |
|---|---|---|
| KEYWORD | C(30) | Matchcode des Keywords für die Heuristik |
| HEUR_F | N(5,3) | Heuristik-Faktor |
| HEUR_A | N(5) | Heuristik-Absolutwert |
| OBJ_ID | N(10) | Verknüpfungs-ID für Relation |

**QUALIFY** – Tabelle der Qualifyer für die Heuristik

| Feld | Typ | Inhalt |
|---|---|---|
| QUALIFYER | C(30) | Matchcode für das Qualify-Wort |
| OBJ_ID | N(10) | Verknüpfungs-ID für Relation |

**KEYQUAL** – Verknüpfungstabelle der folgenden Relationen (jeweils 1:n)

```
KEYWORDS.OBJ_ID   -> KEYQUAL.ID_K
KEYQUAL.ID_Q      -> QUALIFY.OBJ_ID
```

| Feld | Typ | Inhalt |
|---|---|---|
| ID_K | C(30) | Matchcode für das Qualify-Wort |
| ID_Q | N(10) | Verknüpfungs-ID für Relation |
| QUAL_F | N(5,3) | Heuristik-Faktor des referenzierten Qualifyers |
| QUAL_A | N(10) | Heuristik-Absolutwert |

## 4.1.6. Algorithmen

1. Wenn ein Keyword gefunden wurde, erfolgt die Berechnung des Heuristik - Wertes wie folgt:

für alle zugeordneten Qualifyer zu dem Keyword:[40]

**heur_a = heur_a + qual_a´**

Alle Heuristik-Werte der Qualifyer werden zu dem Heuristik-Wert des Keywords addiert, dabei wird zuvor in Abhängigkeit des Parameters $\alpha$ eine Abschwächung des Qualifyer-Werts in Abhängigkeit seines Abstands vom Keyword vorgenommen (zum Beispiel $\alpha$ = 200). Dieses ist notwendig, um auf die Dokumentstruktur einzugehen und entferntere Qualifyer weniger zu beachten, als direkt in der Nähe des Keywords stehende.

**qual_a´ = MIN ($\alpha$ / MAX(abst,1), 1) * qual_a**

```
abst = Abstand zw. Keyword und Qualifyer in Ascii-Zeichen)

MIN() = Minimum zweier Zahlen
MAX() = Maximum zweier Zahlen
```

2. Danach folgt die Berechnung für alle Qualifyer zu diesem Keyword:

```
heur_a = heur_a * qual_f
heur_f = heur_f * qual_f
```

Der Heuristik-Wert wird bei dieser Berechnung mit Hilfe der Qualifyer-Faktoren berechnet.

3. Darauffolgend wird der Gesamt-Heuristikwert berechnet:

**heur = heur * heur_f + heur_a**

Der Gesamt-Heuristikwert wird mit dem Faktor des aktuellen Keyword-Qualifyer- Sets multipliziert und der Absolutwert dieses Sets wird dazuaddiert.

Es handelt sich bei dieser Heuristik nicht um eine einfache Addition von Heuristikwerten, wie es bei anderen Heuristiken üblich ist. Die Reihenfolge der Abarbeitung erfolgt stets von dem größten heur_a zum kleinsten. Durch die Staffelung mehrerer Multiplikationen und Additionen ist die Reihenfolge der Abarbeitung für ein konsistentes und eindeutiges Ergebnis wichtig.

[40] Für die Absolutwerte der Heuristik sind Werte zwischen –10 und 10 vorgesehen

## 4.1.7. Implementation in Visual Foxpro

```
                                                    && === Heuristik Teil 1:
                                                    && Keyword/Qualifyer-
                                                    && Heuristik mit
                                                    && Abschwächung in
                                                    && Abhängigkeit deren
                                                    && Entfernung

SELECT keywords                                     && Keyword - Tabelle
GO TOP

DO WHILE .NOT. EOF()  .AND. glAbort#.T.
   lcTemp=LOWER(ALLTRIM(keywords->keyword))
   IF AT(lcTemp,lcLower)>0                          && Keyword gefunden ?

      DO pHrep WITH "Keyword "+lcTemp+" gefunden"   && Ja !

      lnHeur_a=keywords->heur_a
      lnHeur_f=keywords->heur_f
```

Im folgenden Codesegment findet die Addition der Heuristik-Werte für die jeweiligen Keywords und Qualifyer statt.

```
                                                    && === LOOP 1 - Absolutwerte
                                                    && addieren
      SELECT keyqual
      GO TOP
      SEEK keywords->obj_id                         && Überhaupt Qualifyer
                                                    && Relationen vorhanden ?
      IF FOUND()
         lnRecnoKeyqual=RECNO()                     && (Aus Performance-Gründen)
         DO WHILE keyqual->id_k=keywords->obj_id    && Ja, dann Loop über alle
            SELECT qualify
            SEEK keyqual->id_q                      && Prüfen, ob Qualify-Wort
                                                    && noch vorhanden
            IF FOUND()
               lcTemp1=LOWER(ALLTRIM(qualify->qualifyer))
               IF AT(lcTemp1,lcLower)>0             && Prüfen, ob Wort auch im
                                                    && Text
```

In Abhängigkeit des Abstandes zwischen Keyword und Qualifyer wird das Gewicht des Qualifyers berechnet. Hierbei spielt der Parameter „Abschwächungsfaktor“ der ersten Seite des Heuristik-Sets eine entscheidende Rolle.

```
                                                   && === Abschwächungsfaktor
                                                   && anhand der kürzeseten
                                                   && Entfernung der
                                                   && Teilstrings berechnen

                  lnLoop=1
                  lnMinAbstand=1000                && Immaginäre Maximum-
                                                   && Entfernung
                  DO WHILE AT(lcTemp,lcLower,lnLoop)>0 ;
                        .and. lnLoop<11            && nach 10 Fundstellen
                                                   && aufhören, um Denial of
                                                   && Service Attacken zu
                                                   && vermeiden
                     lnAbstand=ABS(AT(lcTemp,lcLower,lnLoop);
                        -AT(lcTemp1,lcLower,lnLoop))
                     lnMinAbstand=MIN(lnAbstand,;
                         lnMinAbstand)             && Minimum bilden
                     lnLoop=lnLoop+1
                  ENDDO

                  DO pHrep WITH "Qualifyer "+lcTemp1+;
                       " gefunden mit Minimum-Abstand von "+;
                       ALLTRIM(STR(lnMinAbstand))+;
                       " Zeichen zum Keyword."     && Report ausgeben

                  IF lnMinAbstand=0
                     lnMinAbstand=1                && Keine Teilung durch 0,
```

```
                                                    && falls jemand
                                                    && Keyword=Qualifyer eingibt
                  ENDIF
                  lnMinAbstand=gnAbschwaech/lnMinAbstand    && Richtwert der
                                                    && Abstandsmessung (Ab 200
                                                    && Zeichen z.B. beginnt
                                                    && Abschwächung)
                  IF lnMinAbstand>1
                     lnMinAbstand=1                 && 1 ist Maximal-Faktor
                                                    && (=direkte Nähe der
                                                    && Strings)
                  ENDIF
                                                    && ===
                  lnQual_a=keyqual->qual_a * lnMinAbstand   && Wert aus Datenbank holen
                  lnHeur_a=lnHeur_a+lnQual_a        && Absolutwert(e) addieren
                                                    && pro Qualifyer
               ENDIF
            ENDIF
            SELECT keyqual
            SKIP
         ENDDO
```

In der folgenden Routione werden die Heuristik - Faktoren berechnet. Wichtig hierbei ist, daß die Tabelle der Keywords nach Relevanz sortiert ist.

```
                                                    && === LOOP 2 - Faktoren
                                                    && berechnen
         SELECT keyqual
         GO lnRecnoKeyqual                          && Speedup (der Loop muß 2x
                                                    && durchlaufen werden,
                                                    && da sonst ein Array die
                                                    && Werte halten müßte,
                                                    && was genauso lange dauern
                                                    && würde)
         SEEK keywords->obj_id                      && Überhaupt Qualifyer
                                                    && Relationen vorhanden ?
         IF FOUND()
            DO WHILE keyqual->id_k=keywords->obj_id && Ja, dann Loop über alle
               SELECT qualify
               SEEK keyqual->id_q                   && Prüfen, ob Qualify-Wort
                                                    && noch vorhanden
               IF FOUND()
                  IF AT(LOWER(qualify->qualifyer),;
                     lcLower)>0                     && Prüfen, ob Wort auch im
                                                    && Text
                     lnQual_f=keyqual->qual_f       && Wert aus Datenbank holen
                     lnHeur_a=lnHeur_a*lnQual_f     && Qualifyer-Faktor für
                                                    && Absolutwert
                                                    && berücksichtigen
                     lnHeur_f=lnHeur_f*lnQual_f     && Qualifyer-Faktor
                                                    && berücksichtigen
                  ENDIF
               ENDIF
               SELECT keyqual
               SKIP
            ENDDO
         ENDIF
      ENDIF
   ENDIF
   lnRetval=lnRetval*lnHeur_f+lnHeur_a
   lnHeur_a=0                                       && Absolutwerte mit 0
                                                    && initialisieren
   lnHeur_f=1                                       && Faktoren mit 1
                                                    && initialisieren
   lnQual_a=0                                       && Absolutwerte mit 0
                                                    && initialisieren
   lnQual_f=1                                       && Faktoren mit 1
                                                    && initialisieren
   SELECT keywords
   SKIP
ENDDO
```

### 4.1.8. Probleme dieser Heuristikart

Eine rein auf Text aufsetzende Heuristik kann keine Grafiken (wie sie zum Beispiel auf Buttonbeschriftungen vorkommen) bewerten. Einige Seiten benutzen grafischen Text als gestalterisches Element und entziehen sich somit der Textanalyse. Denkbare Verfahren des OCR[41] würden an den für OCR-Methoden ungeeigneten Möglichkeiten der Webautoren scheitern, Texte verschiedenfarbig, schattiert, animiert etc. darzustellen.

Plug-Ins wie zum Beispiel Macromedia-Shockwave und Macromedia-Flash bieten die Möglichkeit, Text auf dem Bildschirm darzustellen – Dieser Text ist jedoch in den jeweiligen Formaten (meist gepackt) in für den MWC unlesbarer Form enthalten und kann nur von den jeweiligen Plug-Ins gelesen und dargestellt werden. Derzeit sind jedoch noch keine Malware-Seiten bekannt, die diese Plug-Ins einsetzen (nicht zuletzt wohl wegen Sicherheitsproblemen mit Plug-Ins im allgemeinen).

Java-Applets können ebenso wie Daten von Plug-Ins nicht automatisch analysiert werden. Hier gelten im wesentlichen die Einschränkungen, die schon bei der Linkverfolgung im Kapitel 3 erwähnt wurden.

Da der Autor nur die Sprachen Englisch und Deutsch spricht und auch die befragten Personen, die den Fragebogen (vgl. Anhang A) ausgefüllt haben, hier nur deutsche und englische Begriffe eingetragen haben, ist das Heuristik-Set nur bedingt zur allgemeingültigen Suche geeignet. Zwar hat sich in der Malware-„Szene“ die englische Sprache fast überall durchgesetzt – eine rein russische Viren-Site bleibt dem Programm aber mit diesem Heuristik-Set verschlossen. Durch die Änderungsmöglichkeiten des Heuristik-Sets kann dieser Mangel aber leicht ausgeglichen werden, wenn die entsprechenden Experten Begriffe hierzu beisteuern.

Frames teilen den Bildschirm in zwei oder mehr Dokumentbereiche. Es ist mit Frames möglich, zum Beispiel auf der einen Seite eine Navigation zu realisieren und auf der anderen Seite den Inhalt darzustellen. Befinden sich nun in der Navigationsebene die relevanten Keywords wie zum Beispiel „Virii, Trojan, Bombs,...“ und in der Inhaltsebene die Qualifyer wie zum Beispiel „download“, so kann die Heuristik keine Nähe dieser Dokumente bewerten. Ebenso wird der Navigations-Frame einen sehr hohen Heuristik-Wert aufgrund des Vorliegens diverser Keywords erhalten – der eigentlich relevante Inhalt-Frame wird aber heuristisch unterbewertet. Wegen der verschiedenartigen Verschachtelungsmöglichkeiten von Frames ist hier eine automatische Lösung höchst kompliziert – das Dokument müßte in Form eines Parsers im Crawler komplett zusammengesetzt werden.[42]

Das Differenzverfahren muß manuell aufgerufen und gewartet werden – würde die Liste der Keywords anhand der gefundenen Malware-Sites ergänzt beziehungsweise bewertet, so würde der Malware Crawler sich in eine bestimmte Richtung von Websites bewegen und nicht allgemeingültig arbeiten können. Eine Realisation in Form eines lernenden Agenten würde dieses Problem des „Im Kreis Laufens“ berücksichtigen müssen.

[41] Optical Character Recognition

[42] Einige Suchengines verzichten sogar gänzlich auf die Verfolgung von Frame-Inhalten, vgl. Kapitel 3

## 4.2. Heuristik Stufe II – Verweis-Heuristik

Ellen Spertus [SPERTUS96] schreibt über Web-Informations-Tools, daß diese lediglich den Text der Seiten berücksichtigen und die wertvollen Informationen, die in den Hyperlinks enthalten seien, ignorieren würden. Das Web könne zwar wegen seiner Heterogenität, dynamischen Struktur und grenzenüberschreitender Links nicht als traditionelles Hypertextsystem betrachtet werden; dennoch sei menschliche Intelligenz in die Gestaltung, Setzung und Benennung eines jeden Hyperlinks eingeflossen. Diese wertvolle Information, die bereits von normalen Besuchern von Websites genutzt würde, solle auch in automatischen Katalogen nicht unberücksichtigt bleiben.

Seit Ellen Spertus vom MIT diese Aussage 1996 getroffen hat, bewegte sich bei den großen Suchengines wenig in dieser Richtung. Lediglich die seit 1999 verfügbare Suchengine „Google“ unterstützt Ansätze dieser Theorie, benutzt die Ergebnisse allerdings nur für die Erstellung des „Rankings“ – also der Reihenfolge der Anzeige der Suchergebnisse. Dies liegt nicht zuletzt an der Schwierigkeit, die Netzstruktur zwischen den einzelnen WWW-Dokumenten in einer schnell durchsuchbaren und reproduzierbaren Form auf den Rechnern der Suchengines abzulegen.

Der Malware-Crawler bewertet ebenfalls die Verweise auf andere Seiten. In der sogenannten „Badlist“ (Siehe Kapitel 3) befinden sich Links auf bekannt maliziöse Seiten. Findet der Malwarecrawler nun einen Hyperlink auf eine solche Seite, so vergrößert er den Heuristik-Wert entsprechend seiner Parameter (analog zur Verbesserung des „Rankings“ bei der „Google“-Engine). Hierbei kommt dem Malwarecrawler zugute, daß er - anders als Suchengines - nicht jegliche Suchanfrage beantworten soll und somit nicht vielfältige Netzstrukturen beachten muß, sondern sich in nur einem Zusammenhang bewegt.

### 4.2.1. Implementation in Visual Foxpro

```
                                                                    && === Heuristik
                                                                    && Teil 2: URL-
                                                                    && Verweis –
                                                                    && Heuristik
                                                                    && (Verweise auf
                                                                    && bekannte Malware-
                                                                    && Sites)

lcTemp=oFormset.mwc_form1.pg_frames.pg_crawler.Outputframe.Page4.Edit1.VALUE
                                                                    && Externe Links
                                                                    && holen
SELECT badlist
SET ORDER TO TAG bl_url
DO WHILE .NOT. EOF() .AND. glAbort#.T.
   IF ALLTRIM(badlist->bl_url) $ lcTemp
      DO pHrep WITH "Link zu bekannter Site "+;
          ALLTRIM(badlist->bl_url)+" gefunden."                     && Report schreiben
      lnRetval=lnRetval*gnFBadlist+gnABadlist                       && Faktor+
                                                                    && Absolutwert für
                                                                    && Link eintragen
   ENDIF
   SKIP
ENDDO
```

## 4.3. Heuristik Stufe III – URL-Heuristik

Da in der „Badlist“ aus der Heuristik Stufe II nicht alle Seiten, die maliziöse Inhalte anbieten, enthalten sein können, findet in der Heuristik der Stufe III aufgrund von Teilen kompletter URLs eine weitere Bewertung statt.

Während der Anfertigung des Heuristik-Sets wurden die Erfahrungen bei der Erstellung der URL-Startliste sowie die Ergebnisse der Fragebögen im Heuristik-Set verarbeitet.

Die Ergebnisse des Fragebogens (Anlage A) im einzelnen:

***Domains mit überproportional viel Malware***

| | | |
|---|---|---|
| .altern.org | \| | |
| .angelfire.com | \| | |
| .aol.com | \| | |
| .astalavista.com | \| | |
| .box.sk | \|\|\| | |
| .coderz.net | \| | |
| .com | \|\|\| | ← Entfernt, da zu allgemein |
| .geocities.com | \|\|\|\|\| | |
| .msn.com | \| | |
| .net | \| | ← Entfernt, da zu allgemein |
| .pl | \| | |
| .ru | \|\|\|\|\| | |
| .ro | \| | |
| .sk | \| | |
| .sok4ever | \| | |
| .to | \| | |
| .tripod.com | \| | |
| .tw | \| | |
| .t-online.de | \| | |
| .xs4all.nl | \| | |
| .xoom.com | \| | |
| .yahoo.com | \| | |

### 4.3.1. Verwendete Datenbankstrukturen

**TLDLIST** – Liste der Top-Level Domains, die überproportional viel Malware enthalten.

| Feld | Typ | Inhalt |
|---|---|---|
| TLD | C(80) | Zeichenkette die Top-Level Domain enthält |
| TLD_F | N(5,3) | Numerischer Faktor für die Heuristik |

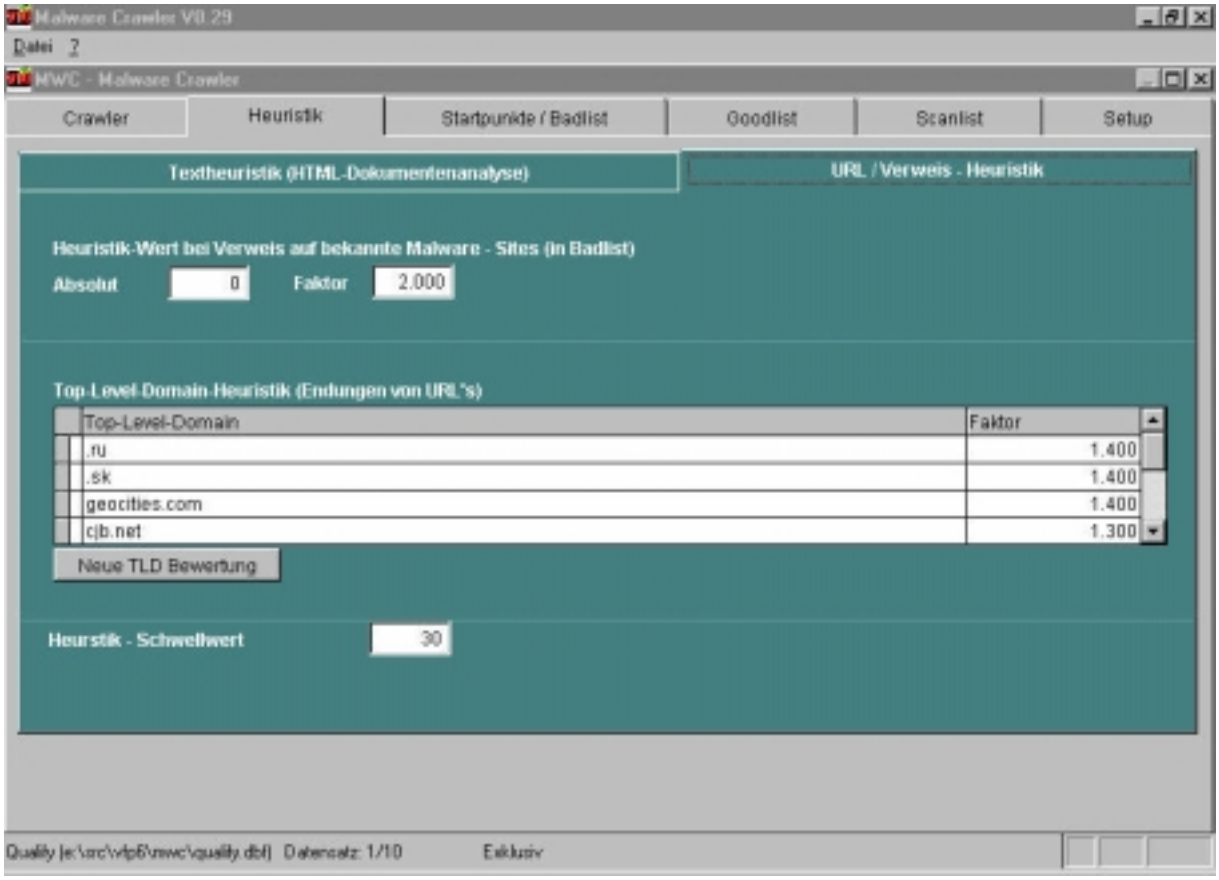



**Abbildung 15 – URL-Heuristik, Verweis-Heuristik und Schwellwert**

## 4.3.2. Implementation in Visual Foxpro

```
                                                        && === Heuristik
                                                        && Teil 3: URL-TLD-
                                                        && Heuristik Top
                                                        && Level Domain-
                                                        && Heuristik

lcTemp=oFormset.mwc_form1.pg_frames.pg_crawler.tx_starturl.VALUE    && Eigene URL holen
SELECT tldlist                                          && TLD-Liste
                                                        && durchscannen
                                                        && (Eine Suche geht
                                                        && hier nicht, da
                                                        && auch Teil-URLs
                                                        && eingetragen
                                                        && werden können.
DO WHILE .NOT. EOF() .AND. glAbort#.T.
   IF ALLTRIM(tldlist->tld) $ lcTemp
      DO pHrep WITH "Domain "+ALLTRIM(tldlist->tld)+;
         "  aus TLD-Liste gefunden."
      lnRetval=lnRetval*tldlist->tld_f                  && Faktor für Domain
                                                        && in Rückgabewert
                                                        && einrechnen
   ENDIF
   SKIP
ENDDO
```

### 4.3.3. Probleme dieser Heuristik-Art

Auch hier bestünde bei einer automatisch höheren Gewichtung einer bestimmten Top-Level-Domain beim Auffinden von Viren das Problem der „Fokussierung“ – d.h. daß der MWC sich zu sehr auf diese Domaingruppe konzentriert und seinen allgemeinen Auftrag verwässert.

## 4.4. Heuristik Stufe IV – Pfad-Heuristik

Betrachtungen bei der Zusammenstellung der Startwerte haben erbracht, daß viele Viren-Sites sich auf bestimmten Domains zusammenfinden. Nicht selten finden sich 10 und mehr Seiten auf ein und derselben Domain. Dementsprechend ist es sinnvoll, in Abhängigkeit der Entfernung zwischen zwei Webseiten die entsprechenden Heuristikwerte anzupassen. Die Entfernung wird in diesem Falle anhand der Verzeichnistiefe innerhalb der Domain (gekennzeichnet durch „/“ im Pfad) gemessen.

Findet sich eine Seite in einem Parallel-Pfad, so erhält sie den folgenden Heuristik-Wert:

$$\mathbf{HEUR_{URL2}' = MAX( HEUR_{URL2},\ HEUR_{URL1} / MAX( DIST( URL1 , URL2 ) , 1 ) )}$$

mit:

| | |
|---|---|
| URL1 | Aktuelle URL |
| URL2 | URL auf derselben Domain |
| $HEUR_{URL}$ | Heuristik-Wert einer URL |

**DIST( URL1, URL2 )** = Abstand der URL1 zur URL2 gemessen in nötigen Additionen von „/“ und „..“, um von der URL1 zur URL2 zu gelangen

***Beispiel:***

URL1= **www.domain.com/homepages/codergruppe**
URL2= **www.domain.com/homepages/hackergruppe**

Also erreicht man URL2 von URL 1 aus folgendermaßen:
www.domain.com/homepages/codergruppe/ **../** hackergruppe

Es wurden ein „/“ und ein „..“ eingefügt – DIST(URL1,URL2) ist also = 2

für angenommene Werte:

$HEUR_{URL1} = 2$ und $HEUR_{URL2} = 10$ ergibt sich also zum Beispiel

$HEUR_{URL2}'$ = MAX(2, 10 / MAX(2,1)
= 5

Die URL2 wurde aufgrund des hohen Heuristik-Wertes ihres Parallelverzeichnisses von Heuristikwert 2 auf den Wert 5 hochgestuft.

## 4.5. Heuristik – Report-Ausgabe des MWC

Hier einige Beispiele für die vom MWC ausgegebenen Informationen im „Heuristik - Ergebnisse“ Fenster (unten links) während des Crawlens:

**Keyword** virus gefunden
**Qualifyer** download gefunden mit Minimum-Abstand von 784 Zeichen zum Keyword.
Keyword worm gefunden
Keyword virus gefunden
Keyword payload gefunden
...
Keyword group gefunden
Keyword hack gefunden
Keyword stuff gefunden
Keyword trojan gefunden
Qualifyer download gefunden mit Minimum-Abstand von 54 Zeichen zum Keyword.
...
Keyword infect gefunden
Keyword trojan gefunden
Keyword hack gefunden
Keyword code gefunden
Keyword hack gefunden
Keyword warez gefunden
... usw ...

Zusätzlich zu dem Report-File werden die Dokumente mit hohen Heuristik-Werten im Bildschirm „Interne Links“ geordnet dargestellt und nach Relevanz farbig hinterlegt:

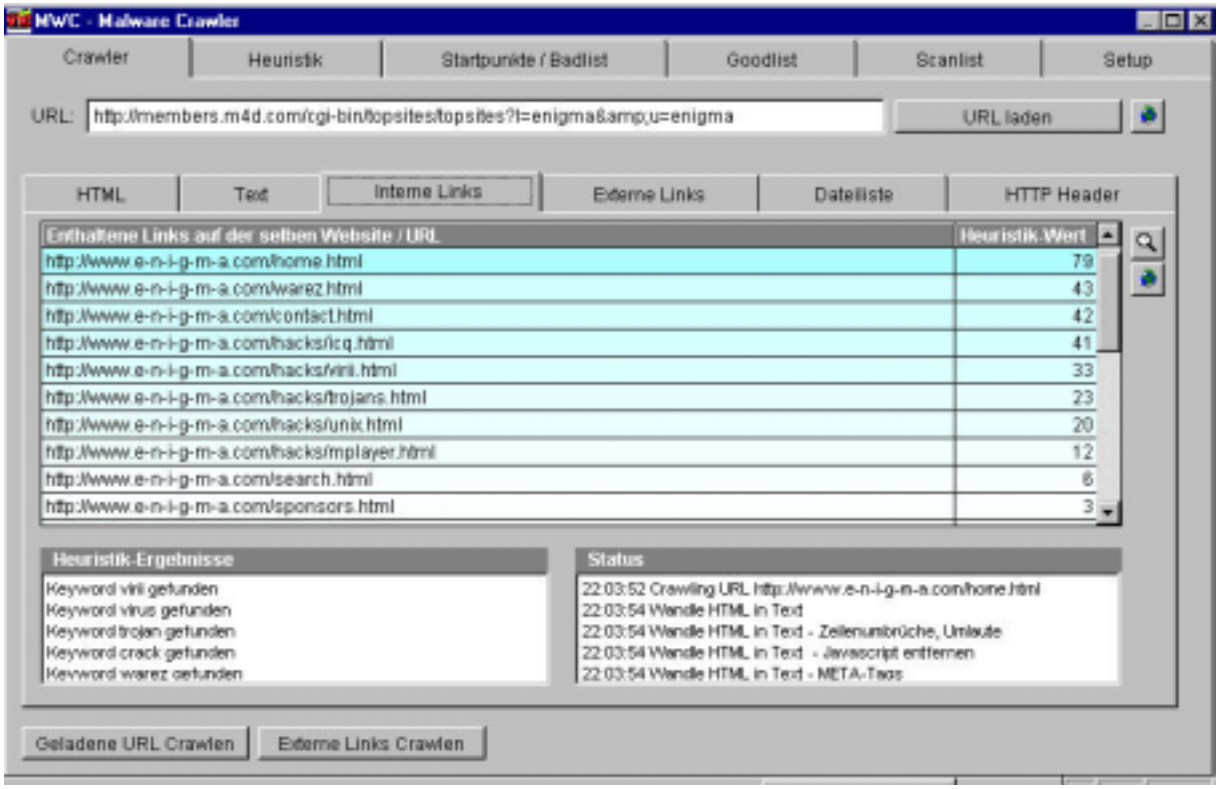



**Abbildung 16 – Heuristisch bewertete Website**

## 4.6. Heuristik – Ausblick

### *Codierungsspezifische Heuristiken*

Zukünftige Heuristiken könnten sich zusätzlich an der Größe des Zeichensatzes (zu erkennen am **<H1>...** Header Tag oder am **<FONT SIZE..>** Tag) orientieren.

Weiterhin ist es sehr auffällig, daß Malware-Sites meist einen schwarzen Hintergrund bevorzugen. Auch hier könnte man bei dem **<BODY BGCOLOR="#00000"** – Tag einen entsprechenden Heuristik-Wert zuordnen.

Noch allgemeiner wäre eine Erweiterung des MWC um eine Funktion des Pattern-Matchings im HTML-Code, um bestimmte Codierungsmethoden in HTML oder JavaScript heuristisch zu bewerten.

### *Linkspezifische Heuristiken*

Die Anzahl der Hyperlinks, die auf die Seite verweisen, sorgen heutzutage bei einigen wenigen Suchengines für eine Verbesserung des „Rankings", also der Reihenfolge der Anzeige. Inwieweit dies für die heuristische Bewertung von Malware-Sites relevant ist, bleibt zu prüfen (Siehe auch [CT 99/23]).

Eine interessante Möglichkeit bietet die Altavista-Engine durch ihre Fähigkeit, Seiten, die auf eine Malware-Site verweisen, zu finden (link:URL – Abfrage). Hierbei ist allerdings die Aktualität des Verzeichnisses ein Problem.

### *HTTP- Headerspezifische Heuristiken*

Das Änderungsdatum der Viren-Site könnte einen Aufschluß darüber geben, wie interessant die vorliegenden Informationen sind. Leider ist dieses Datum in HTML-Dokumenten nicht standardisiert zu finden. Der HTTP-Header kann hier unter Umständen Auskunft geben.

Das Vorhandensein von Robots-Exclusion-Maßnahmen (vgl. Kapitel 2) könnte als weiteres heuristisch relevantes Kennzeichen für Malware – Seiten herangezogen werden.

### *Terminisierung*

Eine weitreichendere Ergänzung wäre die bisher unter anderem aus der Medizin bekannte „Terminologisierung" der Begriffe: Mehrere Ausprägungen einer Bezeichnung, die ein und dasselbe aussagen oder bedeuten (entweder in fremdsprachlicher Abwandlung oder durch Vorliegen zweier Bezeichnungen für dasselbe Objekt) werden unter einem „Ober-Term" zusammengefaßt.

Anstelle der medizinischen Beispiele[43] zwei Beispiele aus dieser Arbeit:

**Oberbegriff „Viren in Hackersprache“**
**Unterbegriffe „Virii,Viruz,...“**

**Oberbegriff „Download“**
**Unterbegriffe „Download, Downloadeur, ...“**

Das Heuristik-Set hätte dementsprechend an jedem jetztigen Endpunkt einen entsprechenden Unter-Baum der realen Begriffe für eine Wurzel.

In einer Arbeit zum Thema *UMLS-Unified Medical Language System* des Klinikums Aachen wird ein solcher Ansatz zur Volltextrecherche in medizinischer Literatur präsentiert. [RTWHAC] Gleichzeitig wird jedoch auch auf den damit verbundenen manuellen Aufwand der Erstellung einer solchen Struktur hingewiesen.

### *Semantische Satzerkennung*

Einen noch größeren Fortschritt würde es bedeuten, wenn sich ein System der semantischen Satzerkennung für die Sprachen Deutsch und Englisch an den Malware-Crawler anschließen und sich darauf eine Heuristik aufbauen ließe. Der Abrbeitsbereich NATS des Fachbereichs Informatik an der Universität Hamburg beschäftigt sich im Rahmen von Robotersteuerungen mit derartigen Satzerkennungen – eventuell können die dort eingesetzten Verfahren mit geringem Aufwand übertragen werden.

[43] (Pilus, Haar, Behaarung...)

# 5. Malware-Scan Modul

Das Malware – Scan Modul ist nachträglich zur Grundkonzeption des Malware Crawlers hinzugekommen (zuvor sollte der MWC gefundene URLs nur anzeigen) – es soll die manuelle Sichtung der gefundenen Websites ersetzen durch ein automatisiertes Verfahren.

Das AV-Modul dient zum Aufruf der Antivirenprodukte. Hierbei handelt es sich um ein Modul, das ausgehend von der generierten Dateiliste (FILELIST, vgl. Kapitel 3) des Malwarecrawlers die entsprechenden Dateien überträgt und die vorhandenen Malware-Scanner mit ihren Parametern aufruft. Die von den Scannern gelieferten Ergebnisse werden dann zur Bewertung der jeweiligen Datei herangezogen.

Als Scanner werden folgende Produkte verwendet:

| | |
|---|---|
| AVP | von Eugene Kaspersky |
| SCAN | von Network Associates (McAfee) |
| FWIN | von Stefan Kurzhals |
| HEUR.EXE | von Markus Schmall wurde nicht benutzt, da der Autor dieses Progamm für derzeitige Macro-Viren nicht mehr als aktuell bezeichnete. |

Bei einem Positiv-Befund für ein Sample wird dann eine E-Mail an eine Mailingliste generiert. Ebenfalls wird die Datei in diesem Fall auf dem Crawler-Rechner gespeichert.

Im Falle eines Negativ-Befundes wird die Datei gelöscht und ein entsprechender Eintrag einer Checksumme in der Log-Datei verhindert eine erneute Übertragung dieser Datei. Da keine Angriffe auf die Checksumme vermutet werden, wird hier kein besonders sicherer Algorithmus (wie zum Beispiel [RIVEST] MD5) verwendet, sondern ein in der Programmiersprache vorhandener Checksummenbefehl, der eine 10-Stellige Zahl als Ergebnis liefert.

Der benutzte Aufruf von AVP im Batch-Betrieb:

**AVPDOS32 %1 /m /p /b /t=d:\ /u=d:\avprep.txt /o /y /s /* /v**

| | |
|---|---|
| /m | - Kein Speichercheck |
| /p | - Kein MBX - Check |
| /b | - Kein Bootrecord - Check |
| /t | - Temp Directory |
| /u | - Reportdatei |
| /o | - Auch uninfizierte Dateien in Report anzeigen |
| /y | - Alle Antworten mit „Ja“ bestätigen |
| /s | - Keine Tonausgabe |
| /* | - Alle Dateien |
| /v | - Redundant |

Ein so erzeugtes Report-File sieht dann wie folgt aus:

```
AVPDOS32 Start  26-01-100 15:23:46

[...]

e:\suspect\MUCHOV~1.ZIP    archive: ZIP
e:\suspect\MUCHOV~1.ZIP/PKUNZJR.COM   ok.
e:\suspect\MUCHOV~1.ZIP/KINISON.COM   packed: CryptCOM.b
e:\suspect\MUCHOV~1.ZIP/KINISON.COM   infected: VCL-based
e:\suspect\MUCHOV~1.ZIP/RICHARDS.COM  infected: VCL-based.trojan
e:\suspect\MUCHOV~1.ZIP/CODEZERO.COM  packed: CryptCOM.b

[... usw ...]

Current object: e:\suspect

         Sector Objects :      0              Known viruses :     93
                  Files :    140               Virus bodies :    117
                Folders :      2                Disinfected :      0
               Archives :      2                    Deleted :      0
                 Packed :    127                   Warnings :      0
                                                 Suspicious :      2
    Scan speed (Kb/sec) :    536                  Corrupted :      0
              Scan time :  00:00:32               I/O Errors :      0


Scan process completed.

Result for all objects:

         Sector Objects :      0              Known viruses :     93
                  Files :    140               Virus bodies :    117
                Folders :      2                Disinfected :      0
               Archives :      2                    Deleted :      0
                 Packed :    127                   Warnings :      0
                                                 Suspicious :      2
    Scan speed (Kb/sec) :    536                  Corrupted :      0
              Scan time :  00:00:32               I/O Errors :      0
```

Für den Malwarecrawler sind die gesperrt gedruckten Zeilen relevant. Für jede einzelne Datei die geladen wird, werden die Virenscanner einzeln gestartet.
Für AVP gilt folgende Bedingung für Meldungen.

**Known viruses :** > 0 → Meldung
**Virus bodies :** > 0 → Meldung
**Suspicious:** > 0 → Meldung

Andere Ausgaben werden nicht beachtet.

# 6. Ausblick

In einer weiterführenden Arbeit könnten zum Beispiel die grafische Darstellung der Hyperlinks der einzelnen Malware-Sites stehen. Interessanterweise handelt es sich dabei meist um enge Gruppen von 5-20 Sites, die sich gegenseitig referenzieren, wobei dann dort jeweils 1-2 Hyperlinks auf eine andere derartige Struktur verweisen. Grafisch könnte man sich dies als Kugelstrukturen (die jeweilige homogene Gruppe), die durch Stabverbindungen (die Außenlinks) auf andere solche Strukturen verweisen, vorstellen. Zusätzlich gibt es dann noch gänzlich außenstehende Sites, die zwar auf diverse Seiten verweisen, selbst aber nicht (oder kaum) referenziert werden.

Eine weitere interessante Betrachtung wäre die Autorenverfolgung (Autorengruppenverfolgung) beziehungsweise die Verfolgung von Erscheinungsterminen von Malware auf bestimmten Websites. Ebenso sind Aktivitätszyklen der Malwareanbieter (zum Beispiel korrespondierende Aktivitäten zu Ferienterminen etc.) darstellbar.

Für Sprachforscher lohnend dürfte eine Betrachtung der eingebürgerten „Hacker / Phreaker / Anarchy / Virii“ – Sprache auf den entsprechenden Seiten in ihren verschiedensten Ausprägungen sein.

Des weiteren lassen sich aus den erkannten Viren Zuwachsstatistiken für Malware allgemein und für Malware im Internet im speziellen hochrechnen.

Der Malware Crawler kann aufgrund der Anzahl der Websites nur bedingt effektiv arbeiten. Ein Einsatz zum Beispiel in Proxxys, Firewalls, bei Providern und Webknoten ist jedoch durchaus denkbar. Im AGN-eigenen Labornetz empfiehlt sich so zum Beispiel ein Einsatz im Proxxy des Firewalls.

Die Zielsuche nach einem bestimmten aktuellen Vorfall auf den bereits als relevant bewerteten Seiten stellt eine weitere Aufgabe für die zukünftige Programmierung des Malware Crawlers dar.

Bei der Umsetzung zeitkritischer Routinen in C++ könnten fortschrittlichere Verfahren des String-Matchings, die für große Textmengen entwickelt wurden, eingesetzt werden.[44]

[44] Zum Beispiel das von Robert Muth et.al. vorgestellte Verfahren „Approximate Multiple String Search“ [MUTH]

## 6.1. Ökonomische Betrachtungen

> *Robots require considerable bandwidth. Firstly robots operate continually over prolonged periods of time, often months. To speed up operations many robots feature parallel retrieval, resulting in a consistently high use of bandwidth in the immediate proximity. Even remote parts of the network can feel the network resource strain if the robot makes a large number of retrievals in a short time ("rapid fire"). This can result in a temporary shortage of bandwidth for other uses, especially on low-bandwidth links, as the Internet has no facility for protocol-dependent load balancing.* – Martijn Koster [KOSTER95]

Folgende Probleme können entstehen bei der Benutzung von Crawler / Robot-Verfahren, die bei normalen Browsern im allgemeinen nicht entstehen:

1. **Erzeugung von hoher Netzwerklast durch kontinuierliche Zugriffe mit voller Bandbreite (Network load)**

2. **Download von irrelevanten beziehungsweise allen Dateien, Logfiles, Temporärdateien, Images etc. (Quota load)**

3. **Erzeugung von CPU-Last durch mehrere gleichzeitige Verbindungen (CPU load)**

4. **Anfrage von Dokumenten in hoher Frequenz sorgt für hohe Last des Dateisystems (Rapid fire)**

Insbesondere sehr einfach konzipierte Crawler oder solche, die aufgrund eines Fehlers im „Kreis“ laufen, sorgen für derartige Probleme. Moderne Crawler wechseln hingegen bei der Anforderung der Dokumente nach jeder Anfrage (oder nach einer gewissen Anzahl an Anfragen) den Server. Da der Malware Crawler für die Heuristik ca. gleich viel Zeit benötigt wie zum Download der Datei, und der Crawler nicht mit mehreren gleichzeitigen Tasks arbeitet, ist hier derzeit kein Handlungsbedarf gegeben.

Das erzeugte Dateivolumen des Malwarecrawlers ist immens. Innerhalb von zwei Tagen „Crawlens“ belegte der Malwarecrawler 14 MB an URL-Liste und ein nahezu ebensolanges Report-File.

Es bleibt zu überlegen, ob das Report-File eventuell ab einem bestimmten Datum gelöscht werden muß, um dem vorhandenen Festplattenvolumen der Testinstallation Sorge zu tragen.

Die Netzbelastung des Crawlers ist eher gering, da das System durch die Zwangspausen während der Analyse der übertragenen HTML-Dokumente keine Netzlast erzeugt. Anders als beim normalen Browsen werden beim MWC keine Bilder, Animationen und Audiodateien übertragen – auch hier wird wesentlich an Bandbreite für die Übertragung der HTML-Dateien gespart.

Die Netzbelastung des Download-Moduls ist jedoch nahezu konstant auf der maximalen Bandbreite des Systems. Aus diesem Grunde wurde diese Komponente für nachts (22:30h – 6h) geplant.

## 6.2. Agentensysteme

***Filippo Menczer et.al.*** haben in ihrer Arbeit „Artifical Life Applied to Adaptive Information Agents“ einen theoretischen Entwurf für Such-Agentensysteme im Internet veröffentlicht. [MENCZER_ADP] Bei der Suche mit Agentensystemen wird die Suchmethode nicht vordefiniert – vielmehr werden sich selbst in ihrer Arbeitsweise verändernden Programmen einige wenige Eingabewerte gegeben, die diese dann verarbeiten. Die Agentensoftware hat nun die Möglichkeit quasi selbständig zu entscheiden, wie sie die Aufgabe vollführt und welche Hyperlinks sie verfolgt beziehungsweise welche sie nicht beachtet. Hierbei bekommt sie als „Feed-Back“ in diesem Modell für Erfolge „Energie“ zugeführt – bei erfolglosen Versuchen verbraucht diese Software diese „Energie“. Als Startpunkt für die Agenten wird in dieser Arbeit neben den Suchbegriffen auch das entsprechende Suchergebnis einer traditionellen Suchmaschine übergeben.

Hierbei gelten folgende Ausgangsbedingungen für die Agenten:

1. Jeder Agent bekommt das(die) zu suchende(n) Wort (Wörter) als Eingabe. **(INPUT)**

2. Die Agenten werden mit einer bestimmten Anzahl an „Energie“ ausgestattet, mit den zu folgenden Hyperlinks β und der zu berichtenden Fundstellen-Treshhold γ. **(1.)**

3. Die Agenten können Hyperlinks folgen. **(2.1.)**

4. Jede Aktion, die ein solcher Agent ausführt (das Einladen eines HTML-Dokuments), verbraucht eine gewisse Menge an Energie. **(2.2)**

5. Erfolgreiche Ergebnisse werden durch Zuführung von Energie belohnt. **(2.2)**

6. Erfolglose Agenten „sterben“ aufgrund fehlender „Energie“. **(2.3)**

7. Erfolgreiche Agenten können sich ab einer bestimmten Energiemenge reproduzieren **(2.3.)**

INPUT: $n>0$, $e>0$, $t>0$; query word(s)

1 initialize population of $n$ agents with random $\beta$, $\gamma$, $E$

2 while *number of agents* $> 0$ do for all agents:

2.1 compute for each link $i$ in current document the estimate $L_i$ = *sum of occurrences of each query word in document, weighted inversely to the number of headers between the positions of i and the word in the document*

pick link $i$ to follow according to the probability distribution $P_i = \exp(\beta L_i) \Big/ \sum_j \exp(\beta L_j)$

2.2 $E \leftarrow E - N + eR$ where $N$ = *server access cost measure* and current document's relevance $R$ = *number of occurrences of query words in document*

if $(R+1)\gamma > \frac{1}{2} \Rightarrow E \leftarrow E + F$ where $F$ = *user feedback energy (optional)*

2.3 if $E > t$ clone agent, split parent's energy with offspring, mutate offspring's $\beta$, $\gamma$

else if $E < 0$ destroy agent

**Abbildung 17 – Agentenschema**

Bei dieser Arbeit wurde als ein Ergebnis festgestellt, daß die Agenten, die zum Erfolg führen, je nach Anfrageworten anders geartet sind.

Die Netzwerkauslastung, die dieses System erzeugt, läßt sich durch die „Energie“ – Zuteilung dynamisch steuern. Die Netzwerkauslastung von Agentensystemen ist hier bei gleicher Erfolgsquote meist geringer als bei herkömmlichen Crawler-Verfahren.

Die Arbeit von Menczer et.al. beschränkt sich bei der Agenten-Software selbst auf ein simples Pattern-Matching Verfahren. Bei dieser Methode sind die freien Parameter für den Agenten also nur die zu verfolgenden Hyperlinks $\beta$ und der zu berichtenden Fundstellen-Relevanz-Treshhold $\gamma$. Ein Agent, der anhand bereits erfolgreich identifizierter Fundstellen sein Pattern-Matching anpaßt beziehungsweise neue „Keywords" hinzufügt oder sogar die Art des Pattern-Matchings in Form von gezielter Unschärfe verändert, wäre eine sinnvolle Erweiterung. Ebenso ist eine Kommunikation unter den Agenten in Form einer Liste der besuchten und der erfolgreich besuchten Websites wünschenswert. Eine weitere Möglichkeit der Agenten – das Verschmelzen zweier Agenten - wurde hier ebenfalls außer acht gelassen und ist in diesem Modell nicht durchführbar.

***Melania Degeratu*** nennt in ihrer Arbeit über die reale Implementation einer „Info-Spider"[45], der bisher nach ihrer eigenen Aussage (1999) einzigen[46] derartigen Implementation einer agentenbasierten Suchengine einige Vorteile und Nachteile mobiler Agenten gegenüber herkömmlichen Suchverfahren [DEGERATU]:

***Vorteile:***
- Keine „outdated Hyperlinks" wie bei Katalogen
- Geringere Netzwerkbelastung
- Höhere Präzision der Fundstellen

***Nachteile:***
- Langsamere Antwortzeit als Suchmaschinen
- Werden Agenten weit entfernt von dem gesuchten Ziel aufgesetzt, so sind diese nicht sehr erfolgreich.

[45] Die Test-Engine ist unter http://dollar.biz.uiowa.edu/infospiders aufrufbar.

[46] Sie nennt hier einige andere Systeme wie zum Beispiel „Fish-Search", die jedoch nicht so fortschrittlich arbeiten und sich auf Kataloge oder größere User-Eingaben stützen.

Eine detailierte Beschreibung der Funktion dieses Agenten befindet sich bei F.Manczer [MANCZER_AD2]:

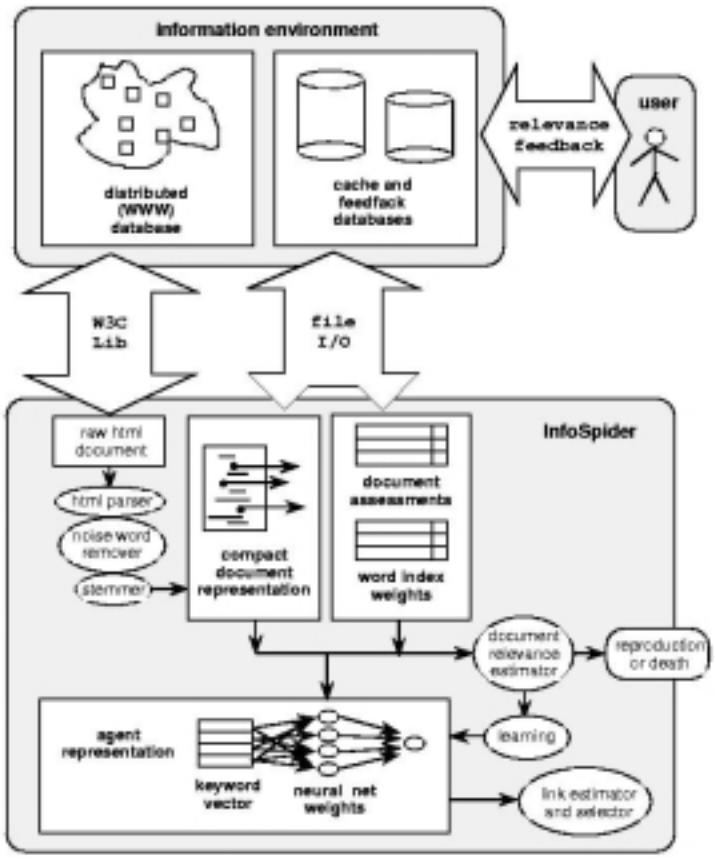



**Abbildung 18 – Info Spider Struktur**

Ähnliche Funktionen des Malware Crawlers finden sich naturgemäß beim Zugriff auf WWW-Resourcen, der HTML-Wandlung, dem Filtern von „Rauschdaten“, der Tokenisierung etc. Interessant ist, daß auch hier Datenbanken der besuchten Websites in ähnlicher Art geführt werden.

Die Agentenkomponente (neuronales Netz, Lernkomponente, Linkgenerierung, Relevanzbewertung) ist jedoch ein komplett anderer Ansatz als er im Malware Crawler realisiert wurde.

Mögliche Erweiterung des Malware Crawlers in Richtung Agentensystem wären:

- Automatische Aufnahme und Generierung von neuen Keywords in das Heuristik-Set (und damit Veränderung der zu verfolgenden Hyperlinks)
- Instanziierung und selektive Löschung mehrerer konkurrierender Malware-Crawler
- Erfolgsbewertung der Instanzen

## 6.3. Mobile Agenten

Die Netzlast stellt ein großes Problem für jegliche Suchanfragen im WWW dar, wie Filippo Menczer in seinem Artikel über adaptive Website Agenten schreibt: [MENCZER_ADP]:

> *However, in the long run we believe that network access will become a serious bottleneck for any client-based distributed search. The answer, of course, is to transfer agents from clients to servers. While many well-founded concerns make this solution unfeasible at present, we imagine that in the very busy network of a near future, the owner of a server might become willing to give up some CPU cycles on ist machine in exchange for improved bandwidth. –*

Bisher gibt es kein allgemein akzeptiertes Format zur Aufnahme und Ausführung von fremden Agenten auf dem eigenen Server. Dies mag nicht zuletzt auch an den Möglichkeiten des Mißbrauchs solcher Systeme liegen (zum Beispiel Überlastung des Servers, Zugriff auf andere Rechner von dem Agentenprogramm aus).

Ein allgemeingültiger Agentenstandard müßte nach Auffasung des Autors zumindest folgende Bedingungen für alle Instanzen des Agenten auf dem jeweiligen Server statuieren:

- ***Netzbelastung***: Die Agenten dürfen nur eine maximal vorgegebene Netzbelastung erzeugen.
- ***Serverbelastung***: Die Agenten dürfen nur eine maximal vorgegebene CPU-Last erzeugen.
- ***Speicherbelastung***:Die Agenten dürfen nur eine maximal vorgegebene Speichermenge verbrauchen.
- ***Lokalität***: Die Agenten dürfen nicht von dem Ursprungs-Server auf andere Systeme zugreifen.
- ***Lebensdauer:*** Die Agenten müssen eine ihrer Aufgabe angemessene Lebensdauer einhalten, ferner müssen diese explizit auf die Seite eingeladen werden und vom Betreiber des Servers sofort und jederzeit beendet werden können.
- ***Ausgaben:*** Die Agenten dürfen nur an den aussendenden Server Rückmeldungen geben.
- ***Selbstkontrolle:*** Die Agenten müssen bei drohenden Problemen (Speicher, Netzlast, CPU-Last) des Host-Systems diese erkennen können und sich selbst terminieren.
- ***Report-Pflicht:*** Die Agenten müssen auf Anfrage ihre Aufgabe, Herkunft und Systembenutzung dem Serverbetreiber anzeigen.

Selbst wenn ein solcher Standard eines Tages implementiert werden sollte, so werden die Autoren der Malware-Websites sicherlich diesen Standard auf Sicherheitslücken überprüfen – auf den eigenen Systemen derartige Agenten jedoch wahrscheinlich aussperren.

## 6.4. Probleme

Neben den in Kapitel 2 genannten Problemen der „HTTP-Get“ Implementation bestehen folgende dem Verfahren inhärente Probleme:

Unbekannte Packerformate können nicht erkannt werden – Da der Packer nicht vorliegt und die gegenwärtigen Virenscanner im besten Falle 4 Packerformate kennen (ZIP,ARJ,LHA,RAR)[47], bleiben Virenarchive selbst, wenn sie aufgrund hoher Heuristikwerte übertragen wurden, unentdeckt.

Verschlüsselte Dateien: Wenn die zu übertragenden Dateien verschlüsselt sind (hierbei spielt es keine wesentliche Rolle, ob die in den Packern implementierten Verschlüsselungsverfahren verwendet werden oder externe Verfahren ihre Anwendung finden), so werden diese zwar auf der Festplatte gespeichert – der Virenscanner erkennt jedoch keine zu untersuchende Datei. Auf den Einsatz von „Brute-Force“ Entschlüsselungsverfahren und paßwortbrechenden Tools wurde verzichtet, da unter Umständen auch nicht-maliziöse Inhalte übertragen werden, deren Entschlüsselung rechtlich bedenklich ist.

Zugangskontrollen zu Websites: Zugangskontrollen (seien sie trivial gestaltet, indem das Paßwort direkt neben dem Login-Schirm steht oder kompliziert, indem erst eine E-Mail Adresse zum Erhalt des Paßworts erforderlich ist) können nicht automatisch überwunden werden. Der Malwarecrawler ist nicht in der Lage, Formulareingaben im GET oder POST-Verfahren vorzunehmen. (Zusätzlich wäre der zu sendende String dem MWC-Programm auch nicht bekannt.)

Cookies: Wenn eine Website Cookies einsetzt, um den User von Seite zu Seite zu geleiten, so wird der Malwarecrawler bei Seitenanfragen jenseits der Eingangsseite mit „Bitte Einloggen“ – Seiten konfrontiert. Der Malwarecrawler kann keine Cookies unterstützen. (Dies trifft auch auf alle anderen bekannten Crawler zu.)

Bewußt falsch gesetzte Endungen: Findet auf einer Website ein Verweis auf eine .JPG Grafik statt, so ignoriert dies der Malwarecrawler, da Grafiken dieses Typs keine Viren enthalten können. Findige Anbieter von Malware könnten nun hinter einem solchen Link auch ein ZIP-Archiv oder ähnliches verstecken.

## 6.5. Zweckentfremdung dieser Arbeit

Durch das dynamische Heuristik-Set kann diese Arbeit leicht zweckentfremdet werden zum Beispiel zur Suche von MP3-Musikstücken oder zum Auffinden anderer illegaler Inhalte im WWW. Aber auch ein Einsatz der Heuristik-Komponente in Netzwerkknoten zum Beispiel totalitärer Systeme mit einem entsprechenden Heuristik-Set wäre denkbar und nicht im Sinne des Autors. Natürlich kann der MWC auch für sinnvolle Recherchen eingesetzt werden (So hat der Autor zum Beispiel einige Literaturquellen dieser Arbeit mit den Verfahren des MWC selbst gefunden und erschlossen).

[47] Vgl. VTC-Test 2000 [BRU2000]

# Anhang A: Fragebogen

**Fragebogen zur WWW - basierten Malwaresuche**

***Mit welchen Suchbegriffen würden Sie im WWW nach Malware (zum Beispiel Viren) suchen ?***

***Mit welchen Suchmaschinen würden Sie diese Suche durchführen ?***

***Welche anderen Verfahren würden Sie verwenden ?***

o Meta-Suchmaschinen
o Spezielle Link-Sites verfolgen
o Crawling-Verfahren
o Andere, und zwar:

***Was erwarten Sie (in %) zu finden bei einer derartigen Suche ?***

Bekannte Malware: %
Neue Malware: %

***Welche 10 Worte im Text der Website sind für Sie charakteristisch für Malware - Websites ?***

***Auf welchen (Top Level-) Domains (zum Beispiel*** `.com` ***oder*** `.provider.de`***) befindet sich Ihrer Meinung nach im Verhältnis überproportional viel Malware ?***

***Wie schätzen Sie auf einer Skala von 0-7 Ihre Erfahrung im Umgang mit dem Internet ein ?***
0 1 2 3 4 5 6 7 (0=keine Erfahrung, 7=ausgiebige Erfahrung, Zutreffendes bitte ankreuzen)

***Wie schätzen Sie auf einer Skala von 0-7 Ihre Erfahrung im Umgang mit Suchmaschinen ein ?***
0 1 2 3 4 5 6 7

***Wie schätzen Sie auf einer Skala von 0-7 Ihre Erfahrung mit Malware ein ?***
0 1 2 3 4 5 6 7

***Wieviele Malware / Viren - Websites haben Sie bisher ca. schon gesehen ?***
o bis 10 o 10-50 o mehr als 50 o mehr als 100

# Anhang B: Quellenverzeichnis

### Quellen zum Thema Malware / Viren


[BSI] Bundesamt für Sicherheit in der Informationstechnik
http://www.bsi.de

[BRU2000] Prof. Dr. Klaus Brunnstein et.al.: AVTC- Anti-Malware Test 2000
http://agn-www.informatik.uni-hamburg.de

[COHEN 94] Fred Cohen: „Computer Viruses – Theory and Experiments“
1984

[FREITAG-WP] Sönke Freitag: Whitepaper – Mögliche Weiterentwicklungen von (Amiga) -Viren, Anti-Debugging Techniken, Stealth-Techniken 1995

[FREITAG-CR] Sönke Freitag: Analyse des ersten polymorphen Amiga-Viruses Crime92
http://agn-www.informatik.uni-hamburg.de

[FREITAG-AV] Sönke Freitag: Amiga AV-Test
http://agn-www.informatik.uni-hamburg.de/vtc/amiga

[GORDON 94] Sarah Gordon „Faces behind the Masks“
In: Secure Computing“ , August 1994, (S. 40-43),
September 1994 (S. 41,42), Oktober 1994

[IW040500] Internet World Newsletter
http://www.internetworld.de/index_3175.html
http://www.internetworld.de/index_3165.html

[NAI] Network Associates Website
http://www.nai.com

[SOLOMON90] Epidemiology and computer viruses – Whitepaper 1990
http:// agn-www.informatik.uni-hamburg.de
(inzwischen dort nicht mehr verfügbar)

[VTC] Virus Test Center, Arbeitsbereich AGN, Informatik,
Universität Hamburg
http://agn-www.informatik.uni-hamburg.de/vtc

**Quellen zum Thema Crawling / Internetrecherche**


[BARABASI] Albert-László Barabási
Homepage: http://www.nd.edu/~alb/
Publikationen: http://www.nd.edu/~networks/

[BOTSPOT] BotSpot – Website die sich hauptsächlich mit Web-Robotern beschäftigt.
http://www.botspot.com

[CT 98/13] C't „Schatzsucher – Die Internet-Suchmaschinen der Zukunft"
Heise Verlag 1998, Heft 13, (S.178 ff)

[CT 99/23] C't „Orientierungslose Infosammler"
Heise Verlag 1999, Heft 23, (S.158 ff)

[EICHMANN94] Eichmann, D., Ethical Web Agents
Proceedings of the 2. WWW Conference, Chicago 1994.

[EXCL] Robots Exclusion Standards
www.tardis.ed.ac.uk/~Esxw/robots/

[KOCH96] T.Koch, A.Ardö, A.Brümmer, S.Lundberg: The building and maintenance of robot based internet search services:
A review of current indexing and data collection methods,
Lund University Library, NetLab
http://www.lub.lu.se/desire/radar/reports/D3.11/tot.html

[KOCH96_2] T. Koch: Suchmaschinen im Internet
http://www.lub.lu.se/tk/demos/DO9603-manus.html

[KOCH99] T. Koch: Browsing and Searching Internet Resources
http://www.lub.lu.se/netlab/documents/nav_menu.html

[KOSTER94] Martijn Koster: Aliweb - Archie-Like Indexing the Web
http://info.webcrawler.com/mak/projects/aliweb/paper-www94/paper.html

[KOSTER95] Martijn Koster: Robots in the Web: threat or treat?
http://info.webcrawler.com/mak/projects/robots/threat-or-treat.html

[KOSTER_ROB] Martijn Koster: The Web Robots Page (1994-2000)
http://info.webcrawler.com/mak/projects/robots/robots.html

[MARB] Marc Bauer's Search Code Page 2001
http://marcbauer.purespace.de

[ND] Universität Notre Dame
http://www.nd.edu

[SEW] Search Engine Watch Website
http://www.searchenginewatch.com

[W3C] W3 Consortium – HTML Definition
http://www.w3.org/MarkUp

[W3C-VAL] W3 Consortium Validator
http://validator.w3.org

[WHATIS] Whatis.com – Online Lexikon
http://www.whatis.com/crawler.htm

**Quellen zur Textanalyse / Text - Heuristik / Link – Heuristik / Agenten**

[BARTELL] Brian T. Bartell, G.W.Cottrell, Richard K. Belew
Automatic Combination of Multiple Ranked Retrieval Systems
http://www.cs.ucsd.edu

[DEGERATU] Melania Degeratu, F.Menczer: Info Spiders: Complementing search Engines with Online Browsing Agents
AMCS, University of Iowa
http://www.cs.ucsd.edu beziehungsweise http://www.math.uiowa.edu

[MANBER_USI] Udi Manber, Peter A. Bigot:Connecting Diverse Web Search Facilities
http://glimpse.cs.arizona.edu/publications.html

[MENCZER_ADP] F. Menczer, W. Willuhn, R.K.Belew :
Artifical Life Applied to Information Agents
Management Science Department, University of Iowa
http://www.cs.ucsd.edu

[MENCZER_AD2] F. Menczer, R.K.Belew:Adaptive Retrieval Agents:
Internalizing Local Context and Scalingup to the Web
http://www.cs.ucsd.edu

[RTWHAC] Rtwh-Aachen: Computerunterstützte Terminologiearbeit,
Verfahren UMLS – (Textanalyse)
http://www.klinikum-aachen.de/cbt/ok3/mtc/Kapitel6/index.html

[SHAKES97] Jonathan Shakes, Marc Langheinrich, Oren Etzioni
Dynamic Reference Shifting: A Case Study in the Homepage Domain (Ahoy Paper), Department of Computer Science and Engineering, University of Washington (Proceedings of the Sixth international World Wide Web Conference, p 189-200, 1997)
http://ahoy.cs.washington.edu:6060/doc/paper.html

[SHAKES99] Jonathan Shakes: Ahoy Homepage
http://ahoy.cs.washington.edu:6060/doc/home.html

[SPERTUS96] Ellen Spertus: ParaSite - Mining Structural Information on the Web
MIT Artificial Intelligence-Lab
and University of Washington Dept. Of CSE
http://www.scope.gmd.de/info/www6/technical/paper206/paper206.html

[STARR] Brian Starr, Marc S. Ackermann, Michael J. Pazzani:
Do I Care? Tell me what's changed on The Web"
http://www.ics.edu/~pazzani/Publications/OldPublications.html

**Quellen zur Programmiersprache (Visual) Foxpro**


[ANTONOVICH] Michael P. Antonovich: Using Visual Foxpro 3, Most Complete Reference
QUE 1995, ISBN 0-7897-0076-X, 4.Auflage 1997

[CIS-VFP] dFPUG Newsgroup auf CompuServe (GO DFPUG)

[DFPUG] Deutsche Foxpro User Group
http://www.dfpug.de
http://www.dfpug.de/forum

[GRIVER] Yair Alan Griver: Foxpro 2.6 Codebook
SYBEX 1994, ISBN: 0-7821-1551-9, 10. Auflage

[MSDN] Microsoft Developer Network
http://www.msdn.com

[MS-FP26] Microsoft – Originaldokumentation zur Programmiersprache Foxpro 2.6

[MS-FPG] Microsoft (Visual) Foxpro Newsgroups
news://microsoft/public/de/fox
news://microsoft/public/fox/internet
news://microsoft/public/fox/vfp/web

[MS-MAPI] Microsoft Mapi Newsgroup
news://microsoft/public/win32/programmer/mapi

[MS-PRESS] Microsoft Visual Foxpro Programmierhandbuch
(Visual Studio Dokumentation)
MICROSOFT-PRESS 1998, ISBN 3-86063-057-1, 15. Auflage 1999

[MS-VFP3] Microsoft – Originaldokumentation zur Programmiersprache
Visual Foxpro 3

[MS-VFP5] Microsoft – Originaldokumentation zur Programmiersprache
Visual Foxpro 5

[MS-VFP6] Microsoft – Originaldokumentation zur Programmiersprache
Visual Foxpro 6

[UT] The Universal Thread
Online-Newsgroup für VFP-Entwickler
www.universalthread.com

[WWIND] West Wind Web Connection (HTTP-OCX und DLL)
http://www.west-wind.com

**Übergreifende Quellen**


[DINF] Duden der Informatik, 1988, S. 264
ISBN 3-411-02421-6

[INSO] Inso Corp. – Hersteller von Quickview
http://www.inso.com

[MUTH] Robert Muth, Udi Manber
Approximate Multiple String Search
http://glimpse.cs.arizona.edu/publications.html

[PLN] Online Lexikon der Planar GmbH
http://www.planar.de/high/buc/lexikon/seiteH.html

[RFC 850] Mark R. Horton - Standard for Interchange of USENET Messages
Juni 1983 http://www.faqs.org/rfcs/rfc850.html

[RFC 1123] R. Braden – Requirements for Internet Hosts – Application and Support
Oktober 1989 http://www.faqs.org/rfcs/rfc1123.html

[RFC 2068] R. Fielding et. al. - Hyper Text Transfer Protocol 1.1,
Januar 1997 http://www.faqs.org/rfcs/rfc2068.html

[RFC 2616] R. Fielding et. al. - Hyper Text Transfer Protocol – HTTP/1.1
Juni 1999 http://www.faqs.org/rfcs/rfc2616.html

[RFC 2617] J. Franks et. al. – HTTP Authentication – Basic and Digest Access Authentication, Juni 1999
http://www.faqs.org/rfcs/rfc2617.html

[RIVEST92] R.Rivest: RFC-1321, MD5 Message Digest Algorithm,
MIT Laboratory for Computer Science and RSA Data Security, Inc., 1992
zum Beispiel: http://info.internet.isi.edu/in-notes/rfc/files/rfc1321.txt

[SCHM98] Markus Schmall:Heuristische Viruserkennung,
Diplomarbeit Universität Hamburg, Fachbereich AGN / Informatik

# Anhang C: Anleitung zum Programm „Malware Crawler“

***Installation:***

Die Installation des Malware-Crawlers erfolgt durch ausführen der Installationsdatei. Während der Installation sind die Anweisungen des Installationstools zu beachten.
Nach erfolgreicher Installation befindet sich eine weitere Programmgruppe (AGN-Malware Crawler) im Startmenü.

Nach dem Start des Malware-Crawlers kann im Menü „Datei“ der Hauptbildschirm des Malware-Crawlers aufgerufen werden.

In der oberen Bildschirmleiste ist nun die Navigation zwischen den verschiedenen Bildschirmen des Malware-Crawlers möglich.


***Crawler*** – Dieser (per Default) zuerst angezeigte Bildschirm beinhaltet die Crawling Ausgaben und die Eingabemöglichkeit für die Start-URL.

***Heuristik*** – In diesem Schirm können die Parameter der Heuristik eingestellt werden.

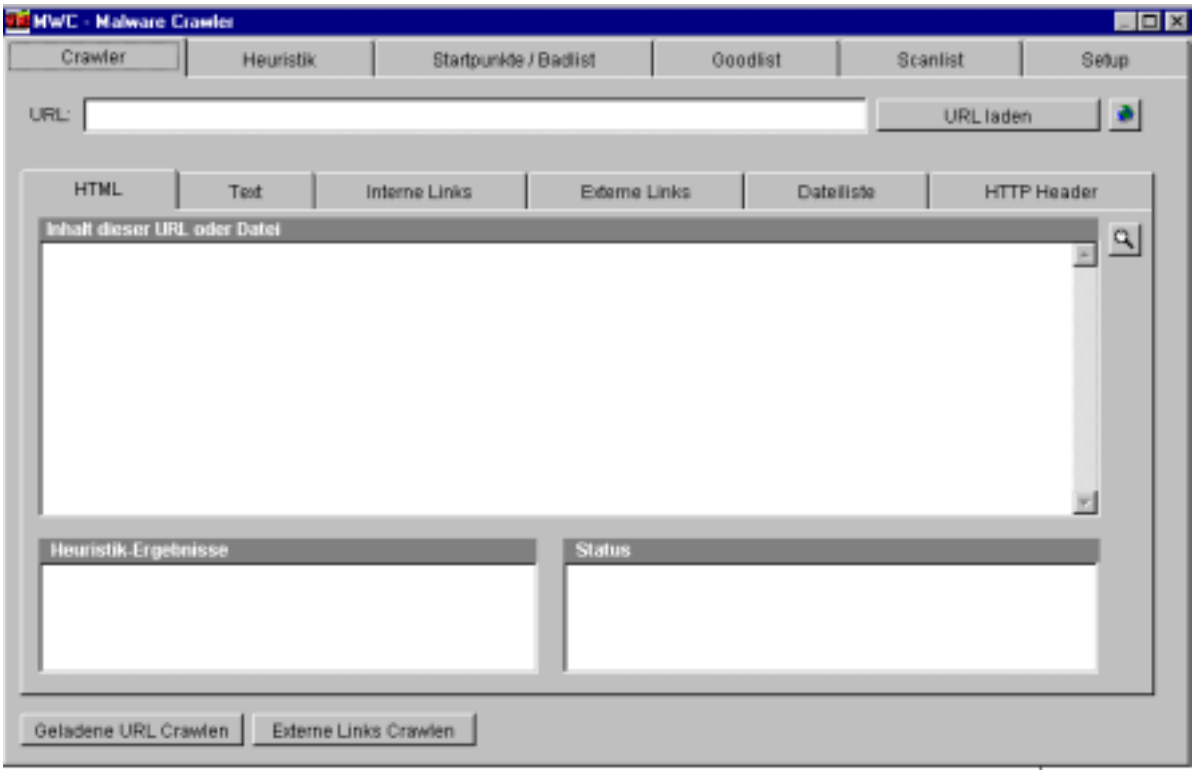
***Startpunkte / Badlist*** – Dieser Bildschirm enthält vordefinierte Suchbedingungen für bekannte Suchengines sowie bekannte Malware-Sites.

***Goodlist*** – Auf dieser Seite sind alle URLs vermerkt, die vom Crawling ausgenommen werden sollen.

***Scanlist*** – Wurde eine Seite geladen und durchsucht, so finden sich hier die URLs.

***Setup*** – Geplant für Einstellungen des MWC (derzeit erfolgen die Einstellungen über die Datei MWC.ini, und sind vordefiniert)

### *Crawler*




### *Der Crawler – Schirm*

In dem Crawler-Schirm werden folgende Informationen angezeigt:

### *URL-Zeile*

**`URL:`** In diesem Feld kann die zu ladende URL eingetragen werden, und mit dem Button **[URL laden]** analysiert werden. Nach der Eingabe der URL reicht auch ein einfaches **`<RETURN>`**, um den selben Effekt zu erzielen.

**`Weltkugelsymbol:`** Wird dieses Symbol angewählt, so wird der Standardbrowser mit der eingetragenen URL gestartet.

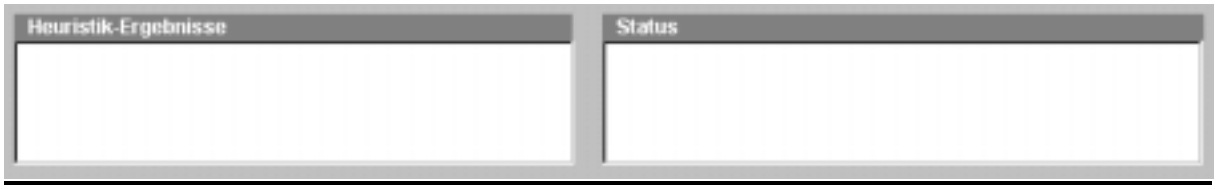
### *Ansichtsleiste:*

Je nach Wahl der Ansichtsleiste (HTML / Text / Interne Links / Externe Links / ...) wird folgendes im oberen Ausgabefenster angezeigt:

- HTML (Code der geladenen Seite)
- Text (Bereinigt von allen HTML- und anderen Programmsegmenten)
- Interne Links (Links innerhalb der Verzeichnisstruktur der aktuellen URL)
- Externe Links (Links auf andere URLs)
- Dateiliste (Liste der Dateien der URL)
- HTTP Header (HTTP-Headerinformationen)

In allen Fenstern steht die mit dem Symbol der **Lupe** versehene Schaltfläche zur Verfügung. Diese Schaltfläche stellt die Informationen des Ausgabefensters in einem größeren Fenster dar. Der so dargestellte Text kann dann z.B. nach Markieren mit der Maus mittels der Tastenkombination **<STRG> + <C>** in die Zwischenablage kopiert werden (dies gilt ebenfalls für die Liste der Links). In Textfeldern reicht auch ein Doppelter Mausclick, um den Effekt der „**Lupe**"-Funktion zu erzielen.

In den Fenstern mit URL-Angabe befindet sich auf der rechten Seite unter dem „**Lupe**" – Symbol noch ein weiteres Symbol zum Aufruf der aktuell selektierten **URL**. Die selektierte URL ist hierbei diejenige in der Tabelle, in der der Mauscursor zuvor mit der linken Taste betätigt wurde.



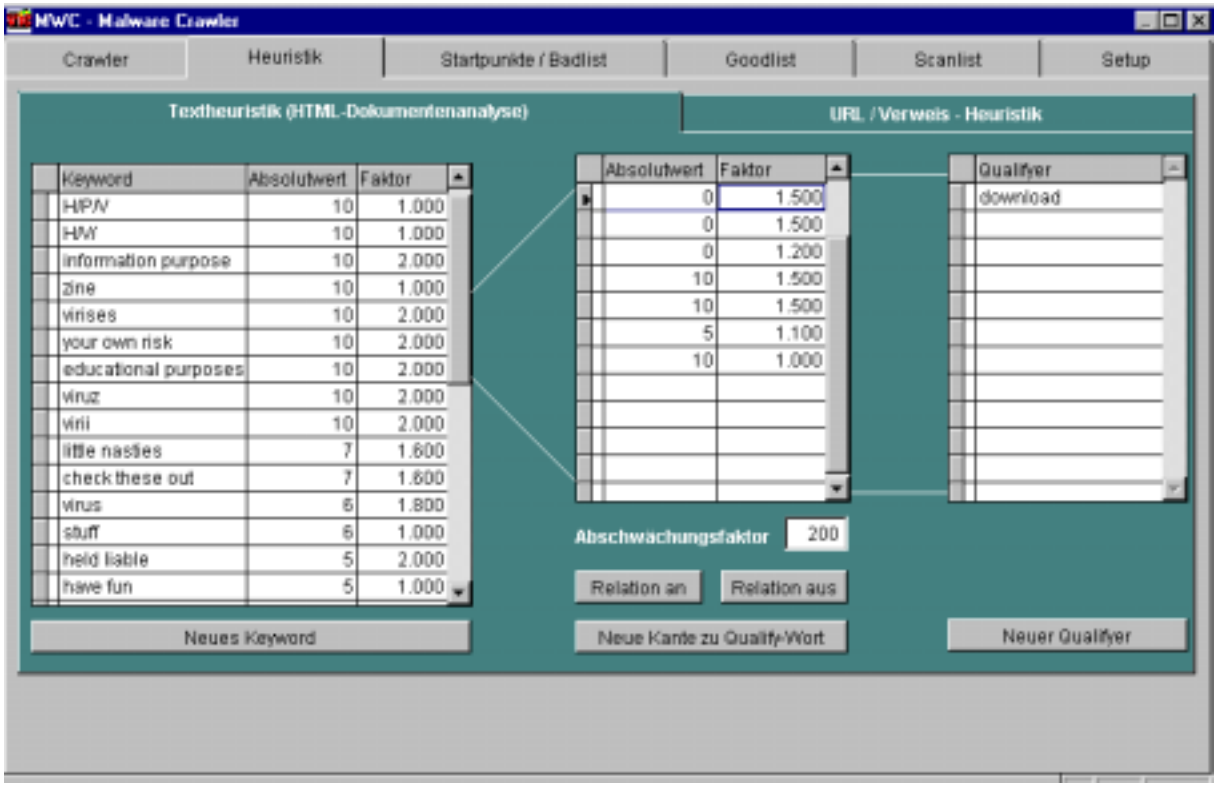

In dem linken unteren Fenster erfolgen während des Crawlens die Ausgaben der Heuristik. Dieses Fenster kann mit einem Doppelclick auf den Inhalt ebenfalls vergrößert werden.

Im rechten Fenster erfolgt die Ausgabe von allgemeinen Statusmeldungen. Dieses Fenster kann mit einem Doppelclick auf den Inhalt ebenfalls vergrößert werden.

**`Geladene URL Crawlen:`** Mit dieser Schaltfläche wird das Crawlen innerhalb einer URL veranlaßt. Der Crawler verläßt hierbei diese URL nicht.

**`Externe Links Crawlen:`** Wird diese Schaltfläche betätigt, so erfolgt eine rekursive Linkverfolgung aller Links der Externen Linkliste. Diese Linkliste muß zunächst ausgehend von einer URL mit der Funktion **`Geladene URL Crawlen`** gefüllt worden sein (Startwerte).

### *Heuristik*




***Heuristik – Schirm 1***

Im Heuristik-Schirm befinden sich auf der linken Seite die Keywords. Ein neues Keyword kann durch anwählen der Schaltfläche **`Neues Keyword`** erstellt werden. Daß neue Keyword befindet sich dann am Ende der Liste mit dem Namen <NEU>, und muß entsprechend gefüllt werden. Der erste numerische Wert gibt jeweils den absoluten Heuristikwert an, der in die Berechnung mit einbezogen wird. Je höher dieser Wert gewählt wird, desto wichtiger ist das Keyword für die Suche. Der zweite numerische Wert stellt den heuristischen Faktor dar.

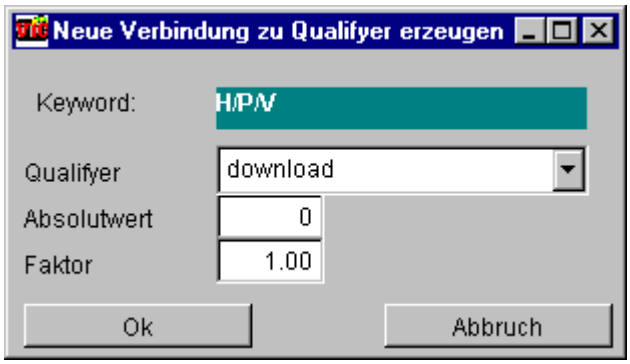
Analog kann mit der Schaltfläche **`Neuer Qualifyer`** ein neues Qualify-Wort erstellt werden, das jedoch nicht automatisch dem aktuellen Keyword zugeordnet wird.

In der Mitte der Bildschirmanzeige befindet sich die Verknüpfungstabelle zwischen Keywords und Qualifyern. Durch die Schaltfläche „**`Relation an`**“ kann die Relation angeschaltet werden – es werden dann nur noch die zu der in der Keywordtabelle gewählten zugeordneten Daten angezeigt. Die numerischen Werte stellen hier wieder den Absolutwert und den Faktor für die Heuristik der Qualifyer bereit. Da für verschiedene Keywords die Gewichtung der Qualifyer unterschiedlich seien kann, konnte diese Information nicht direkt in der Tabelle der Qualifyer gespeichert werden.

Mit der Schaltfläche „**`Relation aus`**“ wird die Relation aufgehoben. Nach Verlassen des MWC-Schirms und Neuaufruf werden wieder alle Daten angezeigt.

Der Abschwächungsfaktor ist der (in Kapitel 4.1 beschriebene) Wert $\alpha$, der für eine Verwässerung der Relevanz der Qualifyer in Abhängigkeit vom Abstand vom Keyword sorgt.

Mit Betätigung der Schaltfläche „**`Neue Kante zu Qualify-Wort`**“ erscheint folgende Eingabemaske:



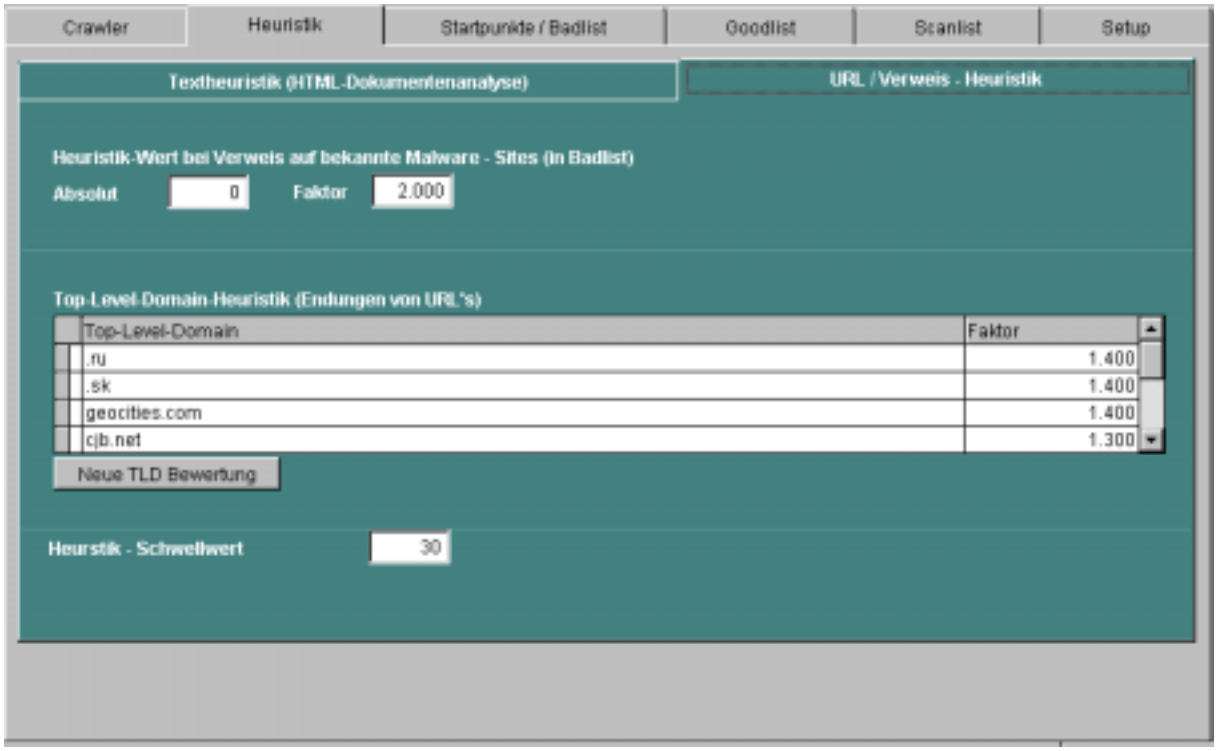

**Neue Verbindung zwischen Keyword und Qualifyer erzeugen**

In dieser Eingabemaske kann in der Qualifyer – Drop-Down Liste aus den bestehenden Qualifyern ausgewählt werden. Im Absolutwert wird der Heuristik-Absolutwert für den Qualifyer eingetragen, im Faktor der entsprechende Faktor für diesen Qualifyer.

Mit **`OK`** wird die neue Verknüpfung gespeichert, **`Abbruch`** beendet die Eingabe.

Eintragungen lassen sich auch direkt in den Tabellen vornehmen, nachdem mit der Maus die entsprechende Stelle angewählt wurde.

In der oberen Leiste des Heuristik-Schirms kann die URL/Verweis-Heuristik aufgerufen werden:




***Heuristik – Schirm 2***

In diesem Bildschirm erfolgen die weiteren Einstellungen der Heuristik.

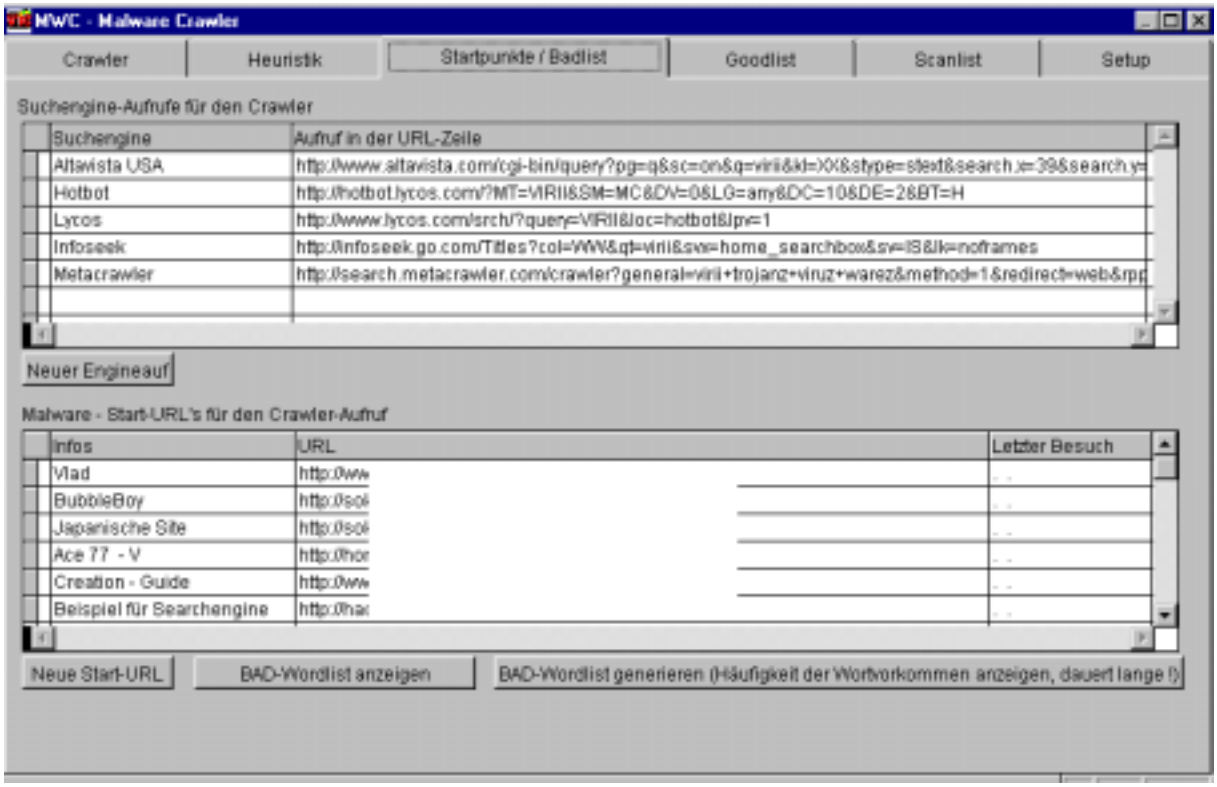
Mit den ersten beiden Werten kann die Gewichtung bei einem vorgefundenen Verweis auf eine bereits als maliziös identifizierte Seite erfolgen. Der erste Wert ist der absolute Heuristik-Wert. Der zweite Wert der Faktor für die Multiplikation.

In der Tabelle der Top-Level Domains kann für jede einzelne Domain (oder für Teile derselben) ein Faktor für die Heuristik festgelegt werden.

Mit der Schaltfläche „Neue TLD-Bewertung“ wird ein neuer, leerer Eintrag erzeugt, der dann mit den gewünschten Daten gefüllt werden kann.

Der Heuristik-Schwellwert stellt den Wert dar, unter dem eine URL nicht als heuristisch interessant zu bewerten ist.

### *Startpunkte / Badlist*



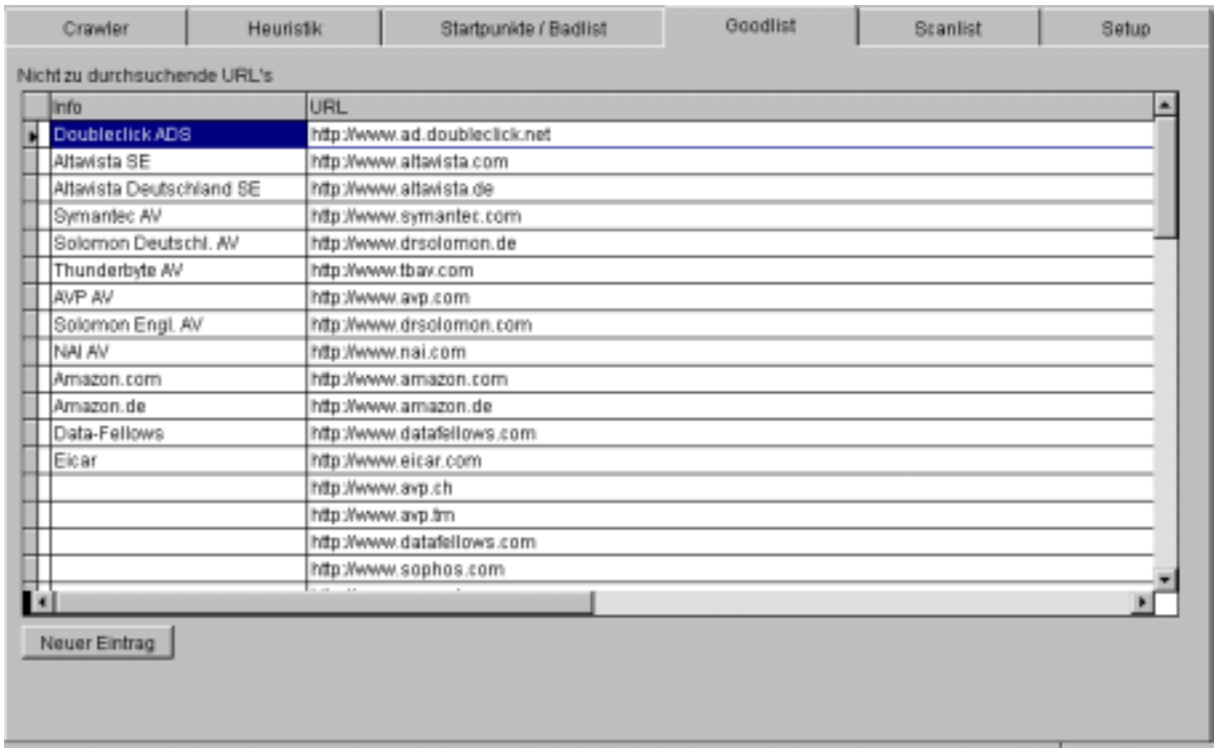

In der Liste der Startpunkte können URL-Aufrufe für Suchengines vermerkt werden. Mit der Schaltfläche **`Neuer Engineaufruf`** kann eine Leerzeile am Ende der Liste erzeugt werden, in die der entsprechende Engineaufruf eingetragen werden kann.

In der „Badlist“ werden die als bereits Maliziöse Inhalte enthaltenen Websites geführt. Mit der Schaltfläche **`Neue Start-URL`** kann eine Leerzeile am Ende der Liste erzeugt werden, in die eine neue Site eingetragen werden kann.

Die Schaltfläche **`BAD-Wordlist anzeigen`** zeigt die vorberechnete Liste der verwendeten Wörter der Malware-Websites.

Die Schaltfläche **`BAD-Wordlist generieren`** berechnet anhand der in der Liste eingetragenen URLs die Liste der verwendeten Wörter automatisch neu. (Achtung: Dieses dauert relativ lange !)

### *Goodlist*




In der Goodlist sind alle Links vermerkt, die nicht durch den Crawler besucht werden sollen. Hierbei handelt es sich um Seiten von Antivirus-Herstellern, Suchengines und Werbebannerfirmen. Diese drei Typen von Seiten werden sehr häufig von Malware-Seiten referenziert, und würden – wenn diese nicht von vornherein ausgeschlossen würden – den Malwarecrawler verlangsamen.

### *Scanlist*

Die zwei Listen entsprechen der externen und internen Dateiliste. Die Anzeige dient rein informativen Zwecken.

– Last but not least –

Das About – Fenster. Am unteren Bildschirmrand wird die Version und das Erstellungsdatum dieser angezeigt.

## Eidesstattliche Erklärung

Ich versichere, die vorliegende Arbeit selbständig und nur unter Benutzung der angegebenen Hilfsmittel angefertigt zu haben.

Hamburg, den

Sönke Freitag