# Analyse von Webcrawlern mithilfe verschiedener Tarpits

Thomas Depian
5B der Technologischen Fachoberschule
Oberschulzentrum J. Ph. Fallmerayer
Brixen, Italien


**Tutor:** Professor Andreas Villscheider, Systeme und Netze

6. Juni 2018

# Abstract

Webcrawler sind in der heutigen Zeit nicht mehr wegzudenken. Sie durchforschen und analysieren für große Internetdienstanbieter wie Google oder Amazon das Internet und liefern ihnen brauchbare Informationen. Einige von ihnen haben jedoch auch das Ziel, E-Mail-Adressen aus Webseiten zu filtern, Passwörter zu hacken und Schaden anzurichten. Diese Arbeit hat sich das Ziel gesetzt, Webcrawler mithilfe einer im Internet weit verbreiteten Technologie, der Tarpit, zu fangen und ihr Verhalten zu dokumentieren. Dieses Verhalten wird auf crawlertypische Muster, wie die Suche nach E-Mail-Adressen oder das Indexieren von Webseiten, hin untersucht und offenbart somit die Absichten der zuvor gefangenen Webcrawler. Der erste Teil dieser Arbeit beschäftigt sich hierbei mit den theoretischen Aspekten eines Webcrawlers. Hier werden anhand von einschlägigen Beispielen die Funktionsweise von Webcrawlern beleuchtet und ihre Aufgaben, welchen sie im Internet nachgehen, aufgezeigt. Des Weiteren wird auch auf die Praxis des Tarpittings näher eingegangen und ihre Arbeitsweise dem Leser nähergeführt. Im zweiten Teil dieser Arbeit werden die zuvor erläuterten Konzepte in einem Experiment umgesetzt, in welchem eine HTTP-Tarpit realisiert wird. Mithilfe dieser Tarpit soll die Existenz von Webcrawlern bewiesen werden, indem sie auf einer speziell für diese Aufgabe präparierten Webseite Hyperlinks aufrufen, welche auf nicht existierende Ziele verweisen. Anstelle eines 404-Errorcodes meldet der Webserver dann eine zufällig generierte Webseite, welche weitere Hyperlinks enthält. So wird versucht die Webcrawler für eine möglichst große Zeitspanne auf dem System zu halten, damit ihr Verhalten anhand von den generierten Logfiles analysiert werden kann. Hierbei geht diese Arbeit auch auf den Aufbau der zum Experiment gehörenden Webelemente, sprich den Webserver und die dazugehörigen Webseiten, ein. Abschließend werden die Logfiles ausgewertet, ein Resümee gezogen und die Ergebnisse der Tarpit präsentiert.

# Inhaltsverzeichnis

# 1 Einführung

## 1.1 Motivation

Heutzutage ist unser alltägliches Leben vom Internet und dessen Erscheinungen stark geprägt. Ob wir nur schnell etwas „googeln“ oder unsere Mails checken und zum wiederholten Male unerwünschte Nachrichten in unserem Posteingang vorfinden, all dies sind das Ergebnis einer Erfindung, welche das Internet revolutionierte. Die Rede ist von Webcrawlern, kleine Programme, welche das Internet nach brauchbaren Informationen durchforsten und diese uns zur Verfügung stellen. Laut einer von Imperva Incapsula durchgeführten Untersuchung im Jahre 2016 stammen lediglich 42,8% [1] des Internet traffics[1] von Menschen. Die restlichen 57,2% [1] stammen von Webcrawlern und Webspidern, sprich Bots. Webcrawler, oder Bots im Allgemeinen, spielen somit eine wesentliche Rolle in unserem täglichen Onlineleben. Ihre Verhaltensmuster und Absichten zu erkennen und zu verstehen ist somit zu einem wichtigen Gebiet in der informationstechnischen Forschung geworden.

## 1.2 Aufbau

Diese Arbeit beschäftigt sich zunächst mit der zentralen Frage „Was ist ein Webcrawler (vgl. Abschnitt 2.1.1) und wie funktioniert er (vgl. Abschnitt 2.1.2)?“ In der Arbeit wird zudem eine der möglichen Absichten eines Webcrawlers anhand von GoogleBot (vgl. Abschnitt 2.1.4) erklärt.
Des Weiteren beschäftigt sich diese Arbeit mit dem Prinzip des Tarpittings (vgl. Abschnitt 2.2.1), welche in einem zweiten Moment im Hauptteil, praktisch umgesetzt wird.

[1] Als *Internet traffic* bezeichnet man den Datenfluss im Internet, welcher durch das Aufrufen von Webseiten, Herunterladen von Medieninhalten oder ähnlichem entsteht.

# 2 Einleitung

## 2.1 Webcrawler

### 2.1.1 Definition

In der heutigen Zeit fällt oft der Begriff *Bot*. Seltener werden die Begriffe *Webcrawler* oder gar *Webspider* und *Harvester* verwendet, obwohl diese meistens zutreffender wären. Die oben angeführten Bezeichnungen sollen hier nun nochmals genauer definiert werden:

**Bot:** Ein Bot ist ein Programm, welches autonom oder nach Erhalt von Eingaben Aufgaben und Aktionen im Internet ausführt. Bot ist hierbei die Kurzform des Begriffs Robot[2].

**Webcrawler:** Ein Webcrawler ist ein gängiges Beispiel eines Bots[2]. Hyperlinks[2] folgend, speichert ein Webcrawler den Inhalt von Webseiten, um diese dann in einem zweiten Moment von Suchmaschinenbetreibern mit Algorithmen auswerten und bewerten zu lassen[2]. Webcrawler können jedoch auch für schadhafte Zwecke missbraucht werden und beispielsweise alle E-Mail-Adressen aus einer Webseite herausfiltern.

**Webspider:** Ein Webspider ist ein Bot, welcher sich von Hyperlink zu Hyperlink durch das Internet bewegt und hierbei Webseiten ausfindig macht und die Ergebnisse wiederum meist an einen Suchmaschinenbetreiber meldet[3]. Seine Funktion entspricht somit der eines gängigen Webcrawlers.

**Harvester:** Ein Harvester ist eine spezielle Form eines Webcrawlers. Ein Harvester durchforstet dabei gezielt das Internet nach E-Mail-Adressen, um in einem zweiten Moment an diese Adressen Spamnachrichten[3] verschicken zu können.

Wie aus den oben aufgeführten Definitionen hervorgeht, beschreiben die Be-

[2]Ein Hyperlink ist ein Element in einer Webseite (Wort, Text oder auch Bild), mit welchem man mittels eines Mausklicks auf eine andere Webseite gelangt.
[3]Spamnachrichten sind Nachrichten, welche vom Empfänger nicht erwünscht sind.

griffe stets ein ähnliches Prinzip. Vor allem die Ausdrücke Webcrawler, Webspider und Harvester sind in ihrer Bedeutung nahezu ident. Deshalb werden diese Begriffe oft als Synonyme verwendet und unter dem Oberbegriff Bot zusammengeführt.

### 2.1.2 Funktionsweise

Wie bereits aus den Definitionen unter Punkt 2.1.1 hervorgeht, hat ein Webcrawler stets ein bestimmtes Ziel, auf welches er hinarbeitet. Um dieses Ziel zu erreichen, durchforstet ein Webcrawler das Internet, indem er von einem oder mehreren Startadressen, sog. *Seeds*, ausgehend, Webseiten aufruft, deren Inhalt herunterlädt, ihn mithilfe von diversen Algorithmen analysiert und dann die Webseite wieder verlässt. Während des Aufrufs der Webseite, filtert sich ein Webcrawler alle weiterführenden Hyperlinks heraus, um sie dann später in einem zweiten Moment aufrufen zu können. Diese Hyperlinks speichert er zusammen mit den Seeds in einer Liste, einer sog. *Queue*[4]. Die Einträge in dieser Liste werden meist als URLs[5] gespeichert. Diese Liste wird Schritt für Schritt, meist nach dem FIFO-Prinzip[6] abgearbeitet. So besucht er rekursiv[7] Webseite um Webseite und übermittelt die gewonnenen Informationen an ein zentrales System, beispielsweise einem Server. Alternativ können die gewonnenen Informationen auch direkt im Webcrawler zwischengespeichert und später von Hand ausgelesen werden. Diese Arbeitsweise wird im Flussdiagramm aus Abbildung 1 nochmals grafisch verdeutlicht.

[4] Das englische Wort „Queue“ kann ins Deutsche mit dem Wort *Warteschlange* übersetzt werden.

[5] URL steht für *Uniform Resource - Locator* und ist ein eindeutiger Zeiger auf ein Objekt im Internet. https://www.example.org wäre z.B. eine URL.

[6] FIFO steht für *First In - First Out* und beschreibt hierbei ein Verfahren, bei welchem das Objekt (in diesem Fall die URL), welches als erstes gespeichert wird (IN), auch als erstes verwendet wird(OUT).

[7] *Rekursiv* bedeutet sich selbst definierend und beschreibt in der Informatik einen Sachverhalt, in welchem sich ein Programm, eine Funktion wiederholend selbst aufruft, um eine komplexes Problem auf einfache Operationen herunterzubrechen.

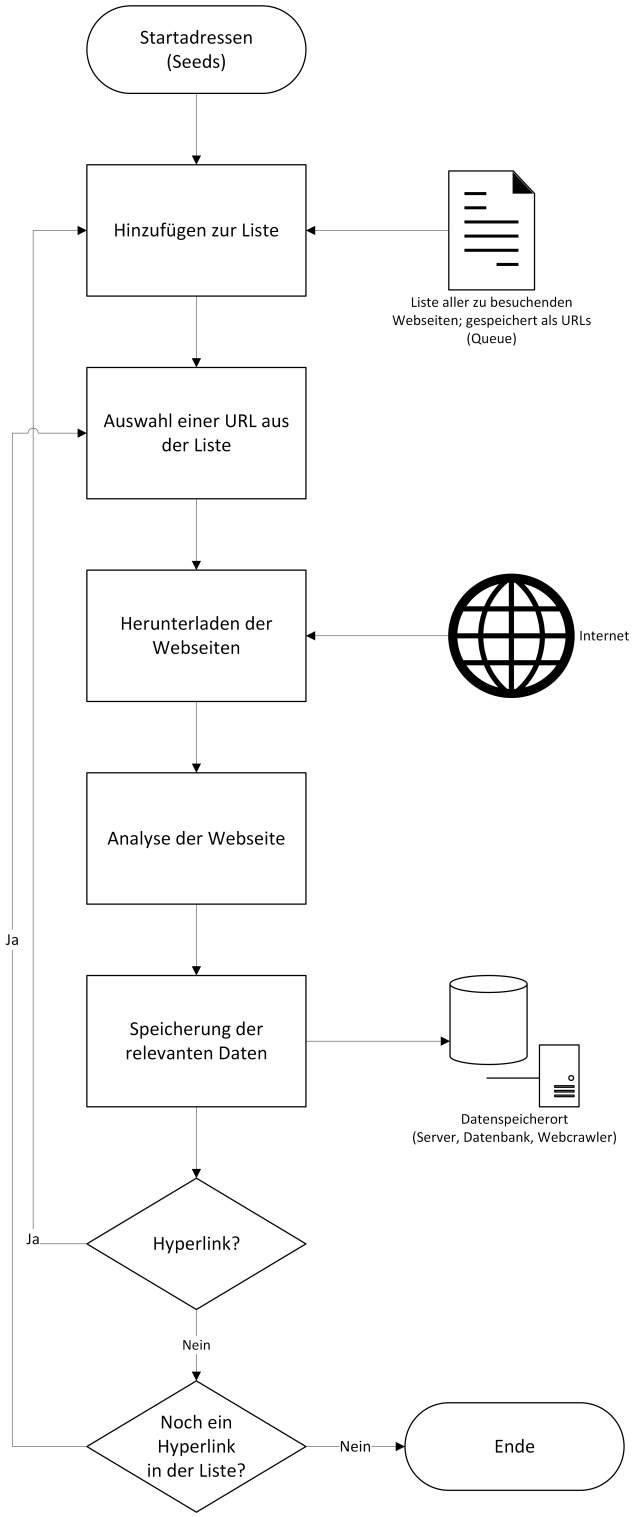


Abbildung 1: Arbeitsweise eines Webcrawlers

Da das Internet ein sich stets weiterentwickelndes und wandelndes Medium ist, in welchem sich Webseiten teils im Stundentakt ändern, muss ein Webcrawler gewissen Prinzipien folgen, welche in den folgenden vier Richtlinien aufgelistet sind[4, S. 265-267]. Somit kann gewährleistet werden, dass ein Webcrawler ein akzeptables Ergebnis erzielt.

1. **Selection Policy *(Auswahlrichtlinie)***
   Im Durchschnitt kommen alle zwei Sekunden rund zehn neue Webseiten hinzu[8]. Dass ein Webcrawler nicht alle Webseiten besuchen kann ist somit selbsterklärend. Deshalb liegt es im Interesse des Betreibers eines Webcrawlers, dass dieser gezielt jene Webseiten untersucht, welche eine hohe Chance auf einen für den Betreiber relevanten Inhalt haben. Jeder unnötige Aufruf und jede unnötige Analyse einer Webseite verbraucht Ressourcen. Vor allem die Ressource *Zeit* ist für Webcrawler besonders wertvoll, denn mit jeder vergangenen Sekunde werden wieder neue Webseiten online gestellt.

2. **Revisit Policy *(Wiederbesuchsrichtlinie)***
   Neben den neu ins Netz gestellten Internetseiten, werden die bestehenden Webseiten auch stets aktualisiert. Neue Einträge kommen hinzu, alte werden gelöscht. Eines der besten Beispiele hierfür sind die unzähligen Nachrichtenportale im Internet: Dort werden oft im Minutentakt neue Meldungen, was im Grunde nichts anderes als neuer Inhalt ist, hinzugefügt. All dies sind Einträge, welche sich ein potentiell interessierter Webcrawler erneut holen und analysieren muss. Deshalb müssen Webcrawler abwägen, ob, wann und wie oft sie eine Seite erneut besuchen möchten oder auch müssen.

3. **Politeness Policy *(Höflichkeitsrichtlinie)***
   Eine Webseite ist heutzutage in wenigen Sekunden komplett geladen, Bilder, Videos oder ähnliche Medien inklusive. Bedenkt man nun, dass GoogleBot[9], ein Webcrawler des Internetgiganten Google, an die zehn Webseiten pro Sekunde[6] anfragt und man mit einer durchschnittlichen Seitengröße von 3 Megabyte[7] rechnen kann, so kommt man auf folgende

[8] Messung selbst durchgeführt mit den Daten von internetlivestats.com[5].
[9] Mehr zu GoogleBot in Kapitel 2.1.4.

einfache Gleichung:

$$\begin{aligned} 10\ ^{Webseiten}/_{Sekunde} * 60\ Sekunden &= 600\ ^{Webseiten}/_{Minute} \\ 600\ ^{Webseiten}/_{Minute} * 60\ Minuten &= 36.000\ ^{Webseiten}/_{Stunde} \\ 36.000\ ^{Webseiten}/_{Stunde} * 3\ ^{Megabyte}/_{Webseite} &= 108.000\ ^{Megabyte}/_{Stunde} \end{aligned}$$

Wie man hier erkennen kann, ist ein Webcrawler in der Lage, in kürzester Zeit eine Unmenge an Datenverkehr zu produzieren und somit wertvolle Bandbreite eines Webservers zu verschwenden. Da die Performance einer Webseite zu einem großen Teil auch von der Antwortzeit und der Auslastung des Webservers abhängt, kann durch einen Missbrauch eines Webcrawlers die Performance einer Webseite stark leiden. Während Webcrawler mit einem guten Zweck versuchen die Ressourcen des Webservers so wenig wie möglich in Anspruch zu nehmen, sind schadhafte, oder einfach nur schlecht geschriebene Webcrawler in der Lage ganze Server lahm zu legen und somit einen großen Schaden anzurichten.

4. **Parallelization Policy** ***(Parallelisierungsrichtlinie)***
   Da ein Webcrawler unter Umständen eine große Zeitspanne damit beschäftigt ist, auf die Antwort des Webservers zu warten, ist es natürlich selbstverständlich, dass man versucht die einem zur Verfügung stehenden Ressourcen bestmöglich auszunutzen. Ein Ansatz hierfür ist es, einen Webcrawler parallel ablaufen zu lassen. Hierfür müssen sich diese jedoch untereinander absprechen, damit zwei parallelisierte Webcrawler nicht dieselbe Webseite analysieren und somit erneut wertvolle Ressourcen verschwenden.

Das Ziel eines jeden Entwicklers und Betreibers von Webcrawlern sollte es sein, diese vier Richtlinien zu respektieren und zu gewährleisten.
Diese Richtlinien werden jedoch des öfteren verletzt. Unerfahrene Programmierer können durch Programmierfehler diese Richtlinien unbewusst missachten und damit einen großen Schaden anrichten. Meistens werden diese Richtlinien jedoch bewusst verletzt, um in möglichst kurzer Zeit eine große Anzahl an Informationen zu gewinnen. Webcrawler, die diese Richtlinien bewusst verletzen, haben meist böse Absichten und sollten von den Webseiten ferngehalten werden.

### 2.1.3 Interaktionsmöglichkeiten

Das Internet ist ein öffentliche Medium und somit sind auch die Webseiten für jedermann zugänglich. Diese Tatsache hat maßgeblich zur Beliebtheit und zum Erfolg des Internets beigetragen und genau deshalb setzen so viele Unternehmen auf das Internet als Werbefläche. Haben Webseitenbetreiber also überhaupt eine Möglichkeit bösartige oder unerwünschte Webcrawler von ihren Webseiten fernzuhalten? In den vergangen Jahren wurden jedenfalls vermehrt Mittel geschaffen, um Betreibern von Webseiten die Möglichkeit zu geben mit Webcrawlern zu interagieren. Auf die bekanntesten dieser Mittel soll nun kurz eingegangen werden.

**robots.txt** Das Robots Exclusion Protocol, besser bekannt als *robots.txt* wurde 1994[8] vom Holländer Martijn Koster[9] erfunden und hat sich zum de-facto-Standard in Sachen Interaktion mit Webcrawlern entwickelt, obwohl es nie ein offizieller Standard wurde.[8]
Um das Robots Exclusion Protocol zu verwenden, muss eine Datei mit dem Namen *robots.txt* im Wurzelverzeichnis der Webseite abgespeichert werden. In dieser Datei können dann Regeln festgelegt werden, die besagen, welcher Webcrawler welche Dateien und Ordner besuchen darf bzw. welche nicht. Das einhalten dieser Regeln ist jedoch fakultativ, weshalb Webcrawler die Regeln der robots.txt auch einfach ignorieren können.

**.htaccess** *.htaccess* ist der Name einer Textdatei, die für die Konfiguration eines Apache Webservers verwendet wird. Da der Webserver von Apache die im Internet mit einem Marktanteil von 46,9%[Stand 24.03.2018: 10] am weitesten verbreitete Serversoftware ist, ist auch die Verbreitung von .htaccess dementsprechend hoch. Apache akzeptiert hierbei eine Vielzahl an Regeln in der .htaccess-Datei; darunter auch Regeln, welche Anfragen anhand der IP-Adresse und/oder des User Agent[10] blocken. Im Gegensatz zu robots.txt sind die Regeln in .htaccess verpflichtend.

**<meta />** Der <meta />-Tag ist ein Tag im HTML-Header in welchem verschiedene Zusatzinformationen, wie Autor, Schlagwörter oder ähnliches dem aufrufenden Browser übermittelt werden können. Mit einer dieser Zusatzinformationen des <meta />-Tags kann der Betreiber einer Web-

[10] Ein *User Agent* ist eine Zeichenfolge, welche bei jedem Aufruf einer Seite an den Webserver übermittelt wird. In ihr stehen verschiedenste Informationen zu verwendetem Browser, Betriebssystem etc.[11].

seite auch mit den Webcrawlern von Suchmaschinenbetreibern interagieren. Mit dem Tag *<meta name="robots"content="noindex, nofollow"/>* kann der Betreiber beispielsweise den Anbietern von Suchmaschinen mitteilen, dass er diese HTML-Seite nicht in die möglichen Suchresultate aufnehmen möchte und der Webcrawler keinen weiteren Hyperlinks auf dieser Seite folgen sollte[12]. Wie bei der Verwendung von robots.txt ist auch diese Einhaltung fakultativ.[13]

**Automated Content Access Protocol** Das Automated Content Access Protocol (kurz ACAP) ist ein 2007 gegründetes Projekt, welches die Copyright-Ansprüche von Inhaltserstellern im Internet besser schützen soll.[14] Es will hierbei das weitverbreitete robots.txt nicht verdrängen, sondern eine Alternative bieten. ACAP wird heutzutage kaum verwendet, da es namhafte Internetkonzerne, wie etwa Google, nicht nutzen[15].

### 2.1.4 GoogleBot

Einer der wohl bekanntesten Webcrawler ist der *GoogleBot*. GoogleBot ist der Webcrawler von Google, der größte Suchmaschinenanbieter weltweit mit über 90% Marktanteil[16]. Google setzt hierbei Algorithmen ein, um die GoogleBots zu steuern und festzulegen, welche Webseite wann und wie oft abgerufen wird[17]. Die gecrawlten Seiten werden dann von Google indexiert[11] und können im Bedarfsfall abgerufen und als Suchergebnisse in Bruchteilen einer Sekunde ausgegeben werden.
Da GoogleBot der weltweit größte operierende Webcrawler ist, ist er auch einer der am besten dokumentierten Webcrawler. Hier ein paar Fakten über den GoogleBot:

- Im Jahre 2012 hatte Google mit seinem GoogleBot bereits 96%[19] aller Webseiten weltweit gecrawlt, Tendenz steigend.

- Der User Agent von GoogleBot lautet `Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)`[20].

[11] Die Indexierung einer Webseite durch einen Suchmaschinenbetreiber bedeutet nichts anderes, als dass die Webseite von einem Suchmaschinenbetreiber gefunden wurde. Indexierte Webseiten werden dann in einem zweiten Moment als Vorschlag auf eine Suchanfrage ausgegeben[18].

- Im Durchschnitt besuchen GoogleBots eine Webseite 187mal am Tag[21][12].

- Durchschnittlich stammt jeder fünfundzwanzigste Besuch von einem GoogleBot *nicht* von GoogleBot. Dies bedeutet dass bösartige Bots den User Agent vom GoogleBot missbrauchen, um sich als dieser auszugeben und so unerkannt zu bleiben[21][12].

- Google verwendet mehrere Crawler, welche für unterschiedliche Zwecke entworfen wurden[20].

## 2.2 Tarpitting

### 2.2.1 Definition

Unter Tarpitting versteht man den Vorgang, bei welchem ein Webcrawler mithilfe einer Tarpit durch verschiedenste Maßnahmen angelockt und später ausgebremst bzw. ausgeschaltet wird. Tarpits können dabei auf allen Layern des ISO/OSI-Modells[13] implementiert werden[23]. Das Prinzip einer Tarpit ist dabei analog zu ihrem Namensgeber, einer Teergrube (engl. Tarpit), in welcher beispielsweise Tiere steckenbleiben. Tarpits arbeiten heutzutage oft in Kombination mit einem Honeypot[14], welcher die Bots anlocken soll. Die Ausdrücke Honeypot und Tarpit werden deshalb heutzutage oft als Synonym verwendet[25]. Da tarpitting ein Mittel zur Bekämpfung von Bots ist, wird versucht es weitestgehend geheim zuhalten um den Entwicklern von Bots keine möglichen Schwachstellen aufzuzeigen. Deshalb ist, wenn überhaupt, nur eine unvollständige Dokumentation öffentlich einsehbar. Auch die IP-Adressen und URLs von Tarpits werden geheimgehalten.

[12] Wert aus einer 2014 von Incapsula (amerikanisches Softwareunternehmen) durchgeführten Studie, in der sie 10.000 Webseiten ihrer Kunden über 30 Tage lang beobachteten.

[13] Das ISO/OSI-Modell ist ein Schichtenmodell, welches heutzutage das theoretische Modell für die Datenkommunikation zwischen Computern in einem Netzwerk und im Internet bildet[22, S. 430].

[14] Ein *Honeybot* ist ein System, welches versucht, Angreifer anzulocken, mit dem Ziel, Informationen wie die IP-Adresse oder den Standpunkt dieser herauszufinden oder neue Angriffsmethoden zu erkennen. Hierbei ist ein Honeypot ein „normales“ System, welches jedoch bewusst bekannte Sicherheitsschwachstellen und -lücken aufweist[24].

### 2.2.2 Formen

Wie bereits erwähnt, können Tarpits theoretisch in jedem Layer des ISO/OSI-Modells implementiert werden. Doch vor allem die Implementation und der Betrieb auf den unteren Schichten des ISO/OSI-Modells ist aufwendig, mühsam und fehleranfällig, da es dort keine Mittel zur Identifikation der Pakete und Einteilung derselbigen in „Gut“ und „Böse“ gibt. Deshalb werden heutzutage Tarpits hauptsächlich im IP-, TCP- und Application-Layer eingesetzt. Während Tarpits auf dem IP-Layer versuchen durch minimieren der versendeten Daten die Verbindung künstlich in die Länge zu ziehen, nutzen TCP-Tarpits den Three-Way-Handshake[15] aus, um potentielle Angreifer in die Irre zu führen. Tarpits auf dem Application-Layer, lassen sich in zwei große Gruppen unterteilen: SMTP-Tarpits und HTTP-Tarpits. SMTP-Tarpits behindern Mailserver, welche unerwünschte E-Mails, vielfach als Spam bezeichnet, versenden, in dem sie beispielsweise den Verbindungsaufbau, welcher vor dem versenden stattfinden muss, verlangsamen[26]. Die Funktionsweise von HTTP-Tarpits wird im folgenden Abschnitt 2.2.3 genauer beschrieben.

### 2.2.3 HTTP-Tarpit

HTTP-Tarpits sind neben SMTP-Tarpits eine der am häufigsten eingesetzten Tarpits. Sie haben aber weniger das Ziel den Gegenüber auszubremsen, sondern ihn eine möglichst lange Zeit zu „beschäftigen“, indem sie ihm stets neue unsinnige Informationen liefern.
Wie aus Abbildung 1, Abschnitt 2.1.2, erkennbar ist, besitzen Webcrawler eine Queue, in welcher die Adressen aller noch zu besuchenden Webseiten vermerkt sind. Ziel einer HTTP-Tarpit ist es nun, diese Queue kontinuierlich mit sinnlosen Adressen zu befüllen, damit ein gefangener Webcrawler möglichst viel Zeit damit verbringt, diese Adressen zu besuchen. Die aufgebrachte Zeit ist für den Webcrawler verschwendete Zeit. Wie bereits vorhin erwähnt ist Zeit jedoch eine der wichtigsten Ressourcen eines Webcrawlers. Meist zeigen die generierten Adressen auf die Tarpit selber und somit beginnt der Prozess von neuem. Ausgereifte Tarpits sind hierbei in der Lage Webcrawler für mehrere Tage zu beschäftigen[27].
Eine HTTP-Tarpit ist auch in der Lage Harvester in ihrer Arbeit zu behin-

[15] Der Three-Way-Handshake wird vom Transmission Control Protocol (TCP) verwendet um eine Verbindung aufzubauen.

dern. Hierzu werden einige der generierten Adressen durch E-Mail-Adressen ausgetauscht. Um die E-Mail-Adressen für den Harvester sichtbar zu machen, muss im HTML-Quelltext beim Hyperlinktag *<a>* bei der Zieladresse der entsprechende Präfix *mailto:* gesetzt werden. Der Tag sieht dann folgendermaßen aus: *<a href:"mailto:jemand@example.com»jemand@example.com<a/>*.
Diese verbreiteten E-Mail-Adressen sind entweder „tot“, sprich sie führen zu keinem E-Mail-Server oder „infiziert“, sprich sie zeigen auf einen präparierten E-Mail-Server der beispielsweise SMTP-Tarpitting betreibt.
Obwohl die Dokumentation solcher Tarpits bewusst geheim gehalten wird, rüsten Entwickler von Webcrawlern ihre Programme kontinuierlich gegen solche Fallen. Sie limitieren beispielsweise die Anzahl der auf einer Domaine aufgerufenen Seiten oder bauen andere Abwehrmaßnahmen gegen Tarpits ein.
Zu bedenken ist auch, dass der Besitzer einer Tarpit hier absichtlich Ressourcen verschwendet, indem er bewusst bösartige Webcrawler auf dem Webserver hält. Mit jedem neuen Webcrawler steigt des Weiteren die Gefahr eines Denial of Service, sprich ein Zusammenbruch des Servers aufgrund von Überlastung, drastisch[28].
Auch ist die Frage nach der Legalität nicht vollständig geklärt. Rechtlich bewegt man sich mit dem Betreiben einer Tarpit in einer Grauzone. Um jedoch sich selbst abzusichern, sollte man mithilfe von robots.txt[16] die Webcrawler „warnen“, indem man sie ausschließt. Sollten Webcrawler dann trotzdem in die Tarpit geraten wurden sie vorher gewarnt und sind demnach „selbst schuld“ [25]. Somit ist auch gewährleistet, dass keine gutwilligen Webcrawler, wie etwa GoogleBot[17], welche sich an die Regeln von robots.txt halten, von der Tarpit betroffen sind[28].

# 3 Hauptteil

## 3.1 Ziel des Projektes

Ziel dieses Projektes ist, die vorher eingeführten Konzepte der Webcrawler mithilfe verschiedenster Arten von HTTP-Tarpits zu beweisen. Hierbei werden, wie unter Punkt 3.2.1 beschrieben, drei verschiedene Formen einer HTTP-Tarpit umgesetzt. Webcrawler sollen dann in diese Tarpit geraten. Ihr Ver-

[16]Mehr zum Thema *robots.txt* ist im Kapitel 2.1.2 auf Seite 6 aufgezeigt.
[17]Der Webcrawler GoogleBot wird in Kapitel 2.1.4 auf Seite 9 genauer beschrieben.

halten innerhalb dieser Tarpit wird dann mitgeloggt und anschließend mithilfe von Drittanbietersoftware auf crawlertypische Verhaltensmuster hin analysiert. Des Weiteren wird der Zusammenhang zwischen Webcrawlern und allgemein bekannten Gefahren bei der Benutzung des Internets, wie beispielsweise das Veröffentlichen der E-Mail-Adresse auf einer Webseite oder das Benutzen von schwachen Passwörtern, untersucht und die Risiken und Auswirkungen dieser Gefahren aufgezeigt.

## 3.2 Umsetzung

### 3.2.1 Allgemein

Um Webcrawler analysieren zu können wurde eine kleine Webseite erstellt und auf einem Raspberry Pi[18] betrieben. Als Webserver wurde hierbei Apache verwendet, da es sich durch die weite Verbreitung und eine Vielzahl an hinzufügbaren Erweiterungen bestens eignet. Da sowohl Apache als auch dessen Erweiterungsmodule Open Source sind, sprich der Quelltext ist öffentlich zugänglich, stehen sie kostenlos zum Download bereit.
Die Webseite wurde größtenteils in PHP verfasst. PHP ist eine relativ alte, aber dennoch weit verbreitete Skript-Sprache mit der sich dynamische Webseiten generieren lassen.

[18] Ein Raspberry Pi ist ein sogenannter Einplatinencomputer, sprich ein Computer dessen gesamte Komponenten auf einer einzigen Platine zusammengelötet wurden. Ein Raspberry Pi zeichnet sich durch ein ausgezeichnetes Preis/Leistungsverhältnis mit einer für die meisten Anforderungen ausreichende Leistung bei einem moderaten Preis von ca. 50€ aus. Er wurde für den Dauerbetrieb entwickelt und eignet sich deshalb bestens für einen Webserver.


Abbildung 2: Das Logo der Webseite

Die Webseite trägt den Namen *VEVETA* und ist öffentlich unter der Adresse `http://maturaprojekt.ddns.net` erreichbar. In Abbildung 2 ist das Logo der Webseite abgebildet.[19] Es zeigt neben dem Namen der Webseite auch eine Spinne, welche an einem Browserfenster herunterhängt. Die Spinne steht hierbei symbolisch für eine Webspider, die gerade dabei ist eine Webseite zu analysieren. Die Abkürzung *VEVETA* steht für VErbund VErschiedener TArpits und vermittelt somit sofort den Zweck dieser Webseite: Drei verschiedene Tarpitformen werden implementiert, um gleichzeitig drei verschiedene Arten von Webcrawlern anlocken und fangen zu können:

**Hyperlink-Tarpit** Hierbei werden verschiedene Hyperlinks, wie unter Punkt 3.2.2 beschrieben, ausgegeben, um somit jegliche Art von Webcrawlern anzulocken.

**Harvester-Tarpit** Hierbei werden verschiedene zufällig generierte E-Mail-Adressen ausgegeben um Harvester anzulocken. Nähere Details hierzu sind unter Punkt 3.2.3 aufgeführt.

**Brute-Force-Tarpit** Hier wird dem Aufrufer eine Log-In-Maske dargestellt. Jede Eingabe wird hierbei als „falsch“ gewertet und mitgeloggt. Einzelheiten dazu sind unter Punkt 3.2.4 einsehbar.

[19] Das Logo wurde auf der Internetseite `http://logomakr.com` erstellt.

Um den Verlauf der Tarpit besser beobachten zu können wurden kontinuierlich Logfiles[20] geschrieben. Apache schreibt hierbei eigenständig einen Accesslog in welchem jeder Aufruf der Webseite mit IP-Adresse, Zeit, Adresse der aufgerufenen Seite, Referring Site[21] und User-Agent notiert wird. Um diese Logfile kontinuierlich auswerten zu können, wurde die Open Source Drittanbieter-Software *GoAcces*[22] verwendet, welche Accesslogs auswertet und, unter anderem live, Statistiken über den Traffic[23] der Webseite ausgibt.
Um alle Webcrawler vor den Gefahren zu warnen, muss die Datei robots.txt um die folgenden Zeilen entsprechend angepasst werden.

```
User-agent: *
        Disallow: /
```

Hiermit wird allen Webcrawlern empfohlen, die komplette Webseite zu meiden. Alle Dateien, welche im Verlauf dieser Arbeit entstanden sind, können online auf GitHub eingesehen werden:
`https://github.com/Analyse-von-Webcrawlern`.

### 3.2.2 Hyperlink-Tarpit

Die erste der drei implementierten Tarpitformen ist die klassische HTTP-Tarpit, wie sie auch unter Punkt 2.2.3 beschrieben ist. Hierbei werden fortlaufend neue Hyperlinks generiert, welche allesamt „ins Leere“, sprich auf keine weitere Webseite, zeigen. Dies würde bei einem herkömmlichen Webserver einen *404 Not Found* HTTP-Statuscode hervorrufen, welcher dem Aufrufer mitteilt, dass die angeforderte Ressource, sprich die Webseite, auf dem Webserver nicht existiert. Ein vermehrtes auftreten von solchen 404-Codes würde einen Webcrawler alarmieren und er würde den Crawl-Vorgang abbrechen, da er der Meinung ist, auf diesem Webserver gäbe es nichts zu holen. Um dies

[20] Ein Logfile ist eine Datei in der eine Software, je nach Konfiguration, verschiedene Informationen, speichert, um diese zu einem späteren Zeitpunkt auswerten zu können. Somit können Rückschlüsse auf Fehler, Auslastung und der gleichen gezogen werden.

[21] Eine Referring Site ist eine Webseite welche vor dem Aufruf der eigentlichen Webseite aufgerufen wurde, sprich es ist diese Webseite von der aus ein Aufrufer über einen Link auf eine andere, „meine“, Webseite gelangt ist.

[22] Nähere Informationen, Downloadlinks, Live-Demo, Dokumentation, etc. sind auf der *Homepage* von GoAccess (`https://goaccess.io/`) angeführt.

[23] Als *Traffic* einer Webseite bezeichnet man den anfallenden Datenverkehr, welcher beispielsweise durch einen Aufruf einer Seite entsteht.

zu verhindern muss dem Webserver mitgeteilt werden, dass wir bei einem 404-Code gerne eine eigene Seite anzeigen möchten. Dies kann bei Apache durch den Befehl

```
ErrorDocument 404 [ADRESSE]
```

erledigt werden. Diesen Befehl kann man dann sowohl in der Datei *.htaccess* festlegen als auch in der Datei *000-default.conf*, welche sich in Linux-Systemen unter dem Pfad */etc/apache2/sites-available/000-default.conf* befindet. Bei der Änderung der .conf-Datei ist jedoch Vorsicht geboten und es empfiehlt sich vorher ein Backup zu machen. Für den weiteren Verlauf dieses Projektes sollen alle 404-Codes auf die Datei *reply.php* umgeleitet werden:

```
ErrorDocument 404 /reply.php
```

Das PHP-Skript, welches in der Datei *reply.php* enhalten ist, generiert dann eine Webseite mit zufälligem Text, welcher wiederum zufällige Hyperlinks enthält. Als aller erstes muss jedoch der 404-Statuscode vom Webserver unterbunden werden. Denn der zuvor aufgeführte Code leitet nur die Anfrage an die Datei reply.php weiter, behält jedoch den Statuscode, in diesem Fall 404, bei. Um den Statuscode zu ändern muss das PHP-Skript den Statuscode überschreiben und dem Aufrufer den HTTP-Statuscode *200 OK* senden. Dies passiert in den folgenden Zeilen, welche am Beginn des PHP-Skriptes stehen und den HTTP-Header manipulieren:

```
<?php
  header("HTTP/1.1␣200␣OK");
  header("Status:␣200␣OK");
  header("Expires:␣".gmdate("D,␣d␣M␣Y␣H:i:s")."␣GMT");
  header("Last-Modified:␣".gmdate("D,␣d␣M␣Y␣H:i:s")."␣GMT
     ↪ ");
  header("Cache-Control:␣no-store,␣no-cache,␣must-
     ↪ revalidate");
  header("Cache-Control:␣post-check=0,␣pre-check=0",
     ↪ false);
  header("Pragma:␣no-cache");
  header("Content-Encoding:␣iso-8859-1");
?>
```

Da sich der Inhalt der durch das Skript generierten Webseite bei jedem Aufruf ändert, darf es nicht im Cache gespeichert werden.[24] Das Cachen der Webseite ist in diesem Fall schlecht, da es einen Webcrawler am Generieren von neuem Inhalt hindern könnte. Dies wird ebenfalls durch den zuvor aufgeführten Codeteil gewährleistet, da er sowohl das Änderungs-, als auch das Verfallsdatum für die Speicherung im Cache auf das aktuelle Datum setzt. Diese Zeitangaben muss in der Greenwich Mean Time[25] angegeben werden[29]. Des Weiteren schreibt er ein *no-store*, *no-cache* und *must-revalidate* vor, welche allesamt dem Aufrufer das Cachen der Webseite untersagen.
Das Skript generiert anschließend eine Webseite, welche aus drei unterschiedlich langen Spalten besteht. Die Spalten werden anschließend mit Buchstaben, Wörtern und Zeichen befüllt. Die Verteilung der Buchstaben entspricht dabei ungefähr den realen Werten der englischen Sprache. Hierfür wurde ein Array erstellt, in welchem die Buchstabe anhand der Verteilung der Buchstaben in der englischen Sprache aufgelistet sind.[26] Dies bedeutet, wenn ein Buchstabe beispielsweise eine sechs-prozentige Verteilung in der englischen Sprache aufweist, so belegt er auch sechs Prozent der verfügbaren Arrayplätze. Um die generierte Webseite noch realistischer wirken zu lassen und somit keinen Anschein einer Tarpit zu erwecken, wurden folgende Konzepte umgesetzt:

**Generieren von "echten" Wörter** Es werden neben den zufällig generierten Buchstaben auch noch vordefinierte englische Wörter[27], Satzzeichen und Leerzeichen mit unterschiedlicher Wahrscheinlichkeit zum Text hinzugefügt. Durch das Einfügen von Leerzeichen entstehen weitere "wörterähnliche" Strukturen.

**Variieren der Spaltenlänge** Die generierte Webseite besitzt drei Spalten, welche jeweils eine unterschiedliche Länge aufweisen. Eine Längeneinheit ist hierbei ein Zeichen oder ein vordefiniertes Wort. Die Länge der Spalten bewegt sich hierbei zwischen 3.000 und 4.000 Zeichen bzw. vordefinierte Wörter.

[24] Moderne Browser speichern, man spricht hier von cachen, Dateien für einen gewissen Zeitraum, damit sie bei einer erneuten Anfrage nicht erneut vom Webserver heruntergeladen werden müssen und somit Zeit und Traffic gespart wird.
[25] Die Greenwich Mean Time (GMT) ist die Zeitzone, welche in Greenwich/London gilt.
[26] Eine Tabelle der Verteilung der Buchstaben kann unter `http://kryptografie.de/kryptografie/kryptoanalyse/haeufigkeitsverteilung.htm` eingesehen werden.
[27] Beispiele für vordefinierte englische Wörter wären *the*, *be*, *and* usw..

**Generieren von Hyperlinks** Auf einer Webseite befinden sich durchschnittlich 6,4 Hyperlinks, welche auf einen anderen Bereich der Seite, eine komplett andere Webseite oder auf eine andere Domain zeigen[27]. Je mehr Hyperlinks auf einer Webseite sind, umso effizienter kann man einen Webcrawler für eine Zeit lang „beschäftigen“. Damit die generierte Webseite jedoch trotzdem eine realistische Anzahl an Hyperlinks aufweist, generiert das PHP-Skript mit einer Wahrscheinlichkeit von einem Prozent eine Hyperlink auf einem zufällig generierten Ziel mit einer Länge zwischen 5 und 25 Zeichen. Dies ergibt bei einer Spaltenlänge von 3.000 bis 4.000 Zeichen eine durchschnittliche Hyperlink-Anzahl von 90 bis 120. Um die Struktur der Hyperlinks zu variieren zeigen sie jeweils mit einer Wahrscheinlichkeit von je 25% auf eine Datei mit der Endung .htm oder .php und mit einer Wahrscheinlichkeit von 50% auf eine .html-Datei. Die Verteilung dieser Wahrscheinlichkeiten ist nochmals in Abbildung 4 auf Seite 20 aufgezeigt.

Somit generiert ein Webcrawler bei jedem Aufruf des Skriptes um die 100 neue Hyperlinks. Ruft er nun $n$-mal dieses Skript auf, so generiert er $100^n$ virtuelle, nicht existente, Dateien. Dies ist eine exponentielle Funktion, welche enorm schnell wächst. Bei sechs-maligem Aufrufen der Seite, was für einen Webcrawler de facto kein Aufwand ist, hat er somit bereits $100^6 = 1.000.000.000.000$ (Eine Billion) Hyperlinks gezeugt, welche in seiner Queue gespeichert sind. Wenn ein Webcrawler theoretisch ein Zehntel einer Sekunde für das Abarbeiten einer Webseite brauchen würde, wäre er hiermit schon über 31.000 Jahre beschäftigt.
Ein Webcrawler ist somit schnell gefangen. Sollte ein Webcrawler die Falle bemerken, so wurde zumindest seine Queue unbrauchbar gemacht. Durch ein Leeren dieser Queue würde der Webcrawler zwar wieder diese sinnlosen Hyperlinks los werden, jedoch würde er auch alle anderen darin gespeicherten Hyperlinks verlieren.
Anzumerken ist jedoch, dass die oben angegebenen Zahlen nur einem theoretischen Wert entsprechen und das Skript in der Praxis mit einer Wahrscheinlichkeit von 10% anstelle der „normalen“ Hyperlinks eine E-Mail-Adresse geniert, siehe hierfür Punkt 3.2.3.

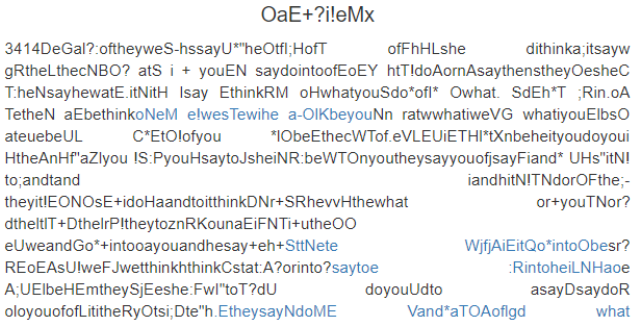


Abbildung 3: Beispielausgabe des Skriptes *reply.php*. Zur besseren Übersicht wird nur eine der drei Spalten angezeigt

Das gesamte reply.php-Skript ist mit allen weiteren Dateien online auf GitHub zu finden: `https://github.com/Analyse-von-Webcrawlern`.

### 3.2.3 Harvester-Tarpit

Um Harvester in die Falle zu locken besitzt VEVETA zwei Methoden:

1. Generieren einer E-Mail-Adresse anhand der IP-Adresse

2. Generieren verschiedener E-Mail-Adressen

#### Generieren einer E-Mail-Adresse anhand der IP-Adresse

Dem Aufrufer wird auf der Landingpage[28] neben einem Kontaktformular eine E-Mail-Adresse angezeigt, welche folgende Form aufweist: *[IP-ADRESSE]-[DATUM]-[UHRZEIT]@maturaprojekt.ddns.net.*
Damit Harvester diese E-Mail-Adresse auch schnell finden, wurde sie mit dem Schlüsselwort *mailto:* gekennzeichnet, das in HTML als Kennzeichen für eine E-Mail-Adresse verwendet wird. Der Code zum generieren dieser E-Mail-Adresse ist in der Datei *index.php* inkludiert:

[28] Als Landingpage wird in der Regel jene Webseite bezeichnet, welche beim Aufruf einer URL als erstes angezeigt wird

```php
<?php
  function generateMailAdress(){
    //IP-Adresse
    $adress = getenv('REMOTE_ADDR');
    //Datum und Zeit
    $currentdate = date("d.m.Y-H.i.s");
    //Domain
    $domain = "@maturaprojekt.ddns.net";

    return $adress."-".$currentdate.$domain;
  }
?>
```

Ein Aufruf der Datei index.php würde beispielsweise folgende E-Mail-Adresse generieren: *82.52.47.189-02.04.2018-16.20.58@maturaprojekt.ddns.net*, welche aus den Augen eines Harvesters folgende HTML-Struktur aufweist:

```html
<a href="mailto:82.52.47.189-02.04.2018-16.20.58
   ↪ @maturaprojekt.ddns.net">
        82.52.47.189-02.04.2018-16.20.58@maturaprojekt.
           ↪ ddns.net
</a>
```

#### Generieren verschiedener E-Mail-Adressen

Des Weiteren generiert das Skript *reply.php*, wie unter Punkt 3.2.2 beschrieben, auch E-Mail-Adressen. Wie unter demselben Punkt erwähnt, besteht die einprozentige Chance, dass ein Hyperlink generiert wird. Wenn dieser Fall eintritt, besteht des Weiteren eine zehn-prozentige Chance, dass anstelle eines Hyperlinks eine E-Mail-Adresse generiert wird. Somit wird mit einer Wahrscheinlichkeit von 0,1% eine E-Mail-Adresse vom Skript generiert. Dies entspricht in etwa fünf bis zehn E-Mail-Adressen pro generierter Webseite. Um diese E-Mail-Adressen so realistisch als möglich aussehen zu lassen, wurden online auf der Webseite `http://www.freedatagenerator.com/csv-data-generator` E-Mail-Adressen generiert, von denen eine mit einer Wahrscheinlichkeit 50% ausgewählt wird; ansonsten wird eine E-Mail-Adresse bestehend aus zufälligen Zeichen generiert. Bei beiden Varianten wird vorne an der E-Mail-Adresse die

Zeit zum Zeitpunkt des Generierens in Form eines Unix-Timestamps[29] angefügt.
Die Verteilung der Ausgabe des Skriptes reply.php ist in der Abbildung 4 aufgeführt.
Alle E-Mail-Adressen verweisen auf die Domain *@maturaprojekt.ddns.net* und landen somit auf einem Mail-Server, welcher neben dem Apache-Webserver auf dem Raspberry Pi läuft.
Diese Form der Tarpit dient in erster Linie dazu, Harvester zu entlarven und ihre Mailinglisten mit unsinnigen E-Mail-Adressen zu verstopfen. Durch das Anhängen des Timestamps kann man im Nachhinein herausfinden, welche IP-Adresse und welcher User-Agent zu einem Harvester gehörte, da man diese in den Logfiles beim Eintrag, dessen Zeit zum Timestamp korrespondiert, ablesen kann. Da der Timestamp jedoch erst nachträglich eingefügt wurde, können nicht mehr alle E-Mail-Adressen einem Harvester zugeordnet werden.

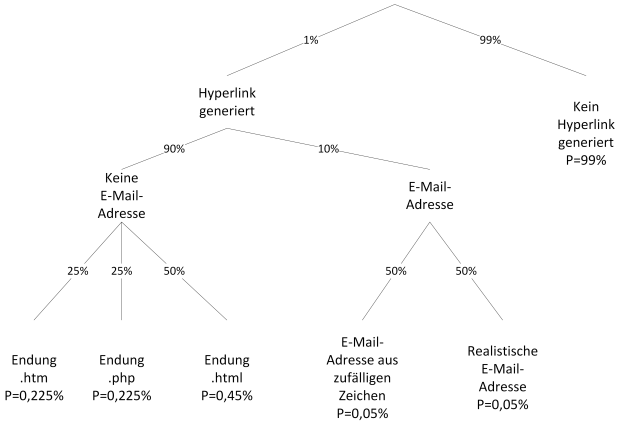


Abbildung 4: Wahrscheinlichkeitsbaum der Ausgabe des Skriptes *reply.php*. Die Wahrscheinlichkeit ist am jeweils letzten Blatt eines Astes angeführt. $P$ entspricht hier der Wahrscheinlichkeit

[29] Der Unix-Timestamp ist eine einfache, aber dennoch exakte Art Zeit zu messen. Der Unix-Timestamp besteht aus einer Zahl, welche die Anzahl an Sekunden, die seit dem 1. Januar 1970 vergangen sind, angibt.

### 3.2.4 Brute-Force-Tarpit

Eine der gängigsten Arten um Passwörter zu knacken ist die sogenannte *Brute-Force*-Methode, zu Deutsch „rohe Gewalt“-Methode. Hierbei werden gängige Kombinationen aus Benutzername und Passwort in einem Anmeldeformular ausprobiert, mit der Hoffnung, dass eine der Kombinationen richtig ist.
Die erstellte Tarpit bietet auch eine Anlaufstelle für solche Webcrawler. Unter der URL `http://maturaprojekt.ddns.net/wp-login.php` ist ein WordPress-Login simuliert. WordPress ist mit 60% Marktanteil[30] das am weiten Verbreiteste Content-Management-System[30] und erfreut sich großer Beliebtheit. Durch die einfache und schnelle Installation bzw. Einrichtung ist Wordpress auch bei Anfängern eine beliebte Software für das Erstellen einer Webseite. Um Änderungen an der Webseite vornehmen zu können, müssen sich die Anwender über eine Log-In-Form anmelden, welche standardmäßig über *[DOMAINE]/wp-login.php* erreichbar ist. Durch die weite Verbreitung und den allgemein bekannten Standartpfad ist diese Log-In-Form ein häufiges Ziel von Brute-Force-Angriffen. Webcrawler überprüfen, ob sie die zuvor erwähnte Log-In-Form unter der gewohnten Adresse finden und starten dann mit einem Brute-Force-Angriff auf diese Log-In-Maske.
VEVETA besitzt ebenfalls eine WordPress-Log-In-Form. Die Struktur dieser stammt aus einer öffentlich zugänglichen Demo von *Softaculous*[31], welche so modifiziert wurde, dass sie den Anschein einer gängigen WordPress-Log-In-Form der Webseite *maturaprojekt.ddns.net* erweckt. Die darin enthaltene Eingabeform ist so modifiziert, dass nach dem Absenden der Eingabedaten diese in einem Log zusammen mit IP-Adresse und Datum gespeichert werden. Der hier aufgezeigte Codeabschnitt ist für dies verantwortlich:

```
<?php
  if (isset($_POST["log"])) {
        //Pfad zum Logfile
    $file = fopen("/home/thomas/Schreibtisch/failed-login
       ↪ -attemps.log", "a");
        //Holen der benoetigten Informationen
    $txt = getenv('REMOTE_ADDR')."␣[".date('d.m.Y-H.i.s')
       ↪ ."]␣".$_POST['log']."␣".$_POST['pwd']."␣-␣wp-
       ↪ login.php\n";
```

[30] Ein Content-Management-System (CMS) ist eine Software, welche einen oder mehrere Anwender bei der Erstellung, Barbeitung und Verwaltung von Inhalten, meist Webseiten, unterstützt.

[31] Die Demo ist Einsehbar unter der Adresse `https://demos1.softaculous.com/WordPress/wp-login.php`.

```
        //Schreiben der Informationen
    fwrite($file, $txt);
        fclose($file);
}
?>
```

Nach dem Abschicken der Log-In-Daten wird der Aufrufer wieder auf die Seite */wp-login.php* zurück geleitet, dies erzeugt den Eindruck eines falsch eingegebenen Passworts und der Aufrufer kann seine nächste Kombination aus Benutzername und Passwort ausprobieren. Diese Art der Tarpit dient somit in erster Linie der Analyse von Webcrawlern, welche die Absicht haben, Brute-Force-Angriffe auf einer Webseite auszuüben.

## 3.3 Auswertung

### 3.3.1 Hyperlink-Tarpit

Die erste der drei Tarpitformen, die VEVETA implementiert, ist, wie bereits zuvor erwähnt, die Hyperlink-Tarpit, dessen Ziel es ist, die Queue von klassischen Webcrawlern mit sinnlosen Hyperlinks zu verstopfen und sie so im besten Fall zu fangen. Hierbei lieferte die Analyse mit *GoAccess* folgende Resultate[32]: In den 247 Tagen, in welchen die Tarpit online war, wurden 342.020 Anfragen von 20.232 verschiedenen Besuchern[33] gezählt, dies entspricht einem Mittel von 1.384,69 Anfragen pro Tag. Wenn man bedenkt, dass die Seite nur für das Fangen von Webcrawlern online gestellt wurde, ist dies ein akzeptabler Wert. Im Durchschnitt hat hierbei jeder Besucher 16,9 Anfragen getätigt. Es wurden hierbei 185.300 verschiedene Webseiten bzw. Dateien angefragt, einige davon natürlich auch mehrfach. Ein Logfile, in welchem der Name und das Datum der generierten virtuellen Dateien gespeichert wurde, hatte am 04. April über 28.510.000 Einträge. Das Skript in reply.php zeigte somit sein volles Potential. Aus Tabelle 1 lässt sich ganz klar erkennen, dass Ende Dezember ein Webcrawler in die Hyperlink-Tarpit geraten sein muss. Durch die Analyse der Logfiles ist ersichtlich, dass der Webcrawler mit dem User Agent

[32] Die Resultate basieren auf den Logfiles, welche zwischen dem 01. August 2017 und 04. April 2018, sprich innerhalb von ca. acht Monaten, erstellt wurden.

[33] GoAcces zählt alle Aufrufe, welche an einem Tag von der gleichen IP-Adresse aus mit dem gleichen User-Agent getätigt wurden, als einen Besucher.

| Anzahl der Anfragen | Datum |
|---|---|
| 17.687 | 20. Dezember 2017 |
| 17.117 | 22. Dezember 2017 |
| 16.925 | 21. Dezember 2017 |
| 16.853 | 19. Dezember 2017 |
| 15.299 | 24. Dezember 2017 |
| 15.076 | 18. Dezember 2017 |
| 14.460 | 26. Dezember 2017 |
| 13.251 | 25. Dezember 2017 |
| 12.647 | 17. Dezember 2017 |
| 12.074 | 27. Dezember 2017 |
| 12.050 | 23. Dezember 2017 |
| 11.809 | 15. Dezember 2017 |
| 10.650 | 28. Dezember 2017 |
| 9.388 | 26. Dezember 2017 |
| 6.554 | 26. Februar 2018 |
| 6.325 | 27. Februar 2018 |
| 4.383 | 29. Dezember 2017 |
| 4.187 | 29. Januar 2018 |
| 3.892 | 30. März 2018 |
| 3.848 | 8. Februar 2018 |

Tabelle 1: Tage mit den meisten Anfragen

*Mozilla/5.0 (compatible; DotBot/1.1; http://www.opensiteexplorer.org/dotbot, help@moz.com)* für diese Aufrufe verantwortlich ist. DotBot ist ein Webcrawler, welcher von dem Suchmaschinenbetreiber Dotmic betrieben wurde. Dotmic ist eine sogenannte e-Commerce-Suchmaschine, dies bedeutet sie ist vor allem auf Onlineprodukte spezialisiert[31]. Nach der Schließung von Dotmic hat die amerikanische Firma SEOmoz, ein Unternehmen die sich auf SEO[34] spezialisiert hat, die Betreibung des DotBot übernommen. DotBot wird von vielen als ein bösartiger Bot bezeichnet und gehandhabt, da er in der Vergangenheit öfters die robots.txt-Regeln missachtet hat und laut Distil Networks, dem weltweit führenden Unternehmen in der Entdeckung und Beseitigung von Bots, Inhalte von Webseiten ohne die Erlaubnis und namentliche Erwähnung

[34]Search Engine Optimization (SEO), zu Deutsch Suchmaschinenoptimierung, ist der Prozess der Anpassung der Struktur und des Inhaltes einer Webseite, mit dem Ziel, dass die Webseite von Suchmaschinen besser gefunden wird und bei Anfragen weiter oben in der Ergebnisliste aufscheint. Dies hat sich heutzutage zu einem wichtigen Bestandteil im Sektor Onlinemarketing hervorgehoben.

des Urhebers stiehlt[32]. DotBot hat über den Verlauf der acht Monate mit 195.331 Anfragen weitaus am meisten Aufrufe getätigt. Zum Vergleich: Der „zweitplatzierte“, MegaIndex, hat die Webseite nur 6.398mal aufgerufen, über 30mal weniger. Durch die Missachtung der robots.txt und einem aggressiven Auftreten, welches auch in der Tarpit ersichtlich war, erzeugt DotBot somit als Nebeneffekt eine große Menge an Traffic und kann somit den Webserver unter Umständen so dermaßen belasten, dass er auf Anfragen von „normalen“ Internetanwendern nicht mehr in der Lage ist zu antworten. Hiermit sieht man eindeutig, dass Webcrawler mit bösen Absichten die Richtlinien unter Punkt 2.1.2 missachten. DotBot hat hierbei beispielsweise eindeutig die Höflichkeitsrichtlinie missachtet.

### 3.3.2 Harvester-Tarpit

Das Skript *reply.php*generiert, wie uter Punkt 3.2.3 beschrieben, neben neuen Hyperlinks auch durchschnittlich fünf bis zehn E-Mail-Adressen, welche von einem potentiellen Harvester eingesammelt werden können. Hierbei wurden innerhalb dieses achtmonatigen Betriebes über 300 E-Mails empfangen.
Vor allem eine E-Mail, besser gesagt eine regelrechte Spamwelle fiel hier besonders auf. Am 16. März 2018 kamen innerhalb von zwei Stunden rund 225 E-Mails an; sechs Tage später, am 22. März, folgten weitere 68 Nachrichten. Diese 293 E-Mails hatten alle den selben Aufbau: Adressiert an verschiedene Adressen, welche allesamt vom PHP-Skript generiert wurden, stammten sie von der *Wolseley Industrial Group* und forderten den Empfänger auf eine Rechnung (Purchasing Order, PO) nochmals zu überarbeiten, zu bezahlen und an eine gewisse Helen Costanzo zurückzuschicken. Der E-Mail sind auch noch zwei Dokumente beigefügt.

**From:** "Wolseley Industrial Group" <admin@dealer.com>
**Subject:** PO515986
**Date:** Fri, 16 Mar 2018 11:25:20 -0000
**Reply-To:** costanzo.helen@mail.com
**User-Agent:** SquirrelMail/1.4.22

Dear Sir,

As per our last phone call, here is the attached PO copy for your reference & submit your revised invoice before the end of this month.

Thank you & best regards,

Helen Costanzo | Purchasing Clerk
Wolseley Industrial Group
Direct No: +1 (757) 874-7795 | Email: sales@wolseleyind.com
12500 Jefferson Ave | Newport News, VA 23602 - USA.
Website: http://wolseleyindustrialgroup.com/

**[PO515986.doc application/octet-stream (90608 Bytes)]**

**[PO-515986.xls application/octet-stream (84383 Bytes)]**

Abbildung 5: E-Mail von der Wolseley Industrial Group

Die E-Mail, ersichtlich in Abbildung 5, erscheint auf den ersten Moment glaubwürdig, denn die Wolseley Industrial Group ist ein in Amerika angesiedeltes Unternehmen und Helen Costanzo ist ein glaubwürdiger Name, vor allem für eine Angestellte eines amerikanischen Unternehmens. Vor allem die Signatur der E-Mail wirkt durch die echten Daten des Unternehmens authentisch. Die Signatur einer E-Mail kann man jedoch heutzutage mit nahezu jedem E-Mail-Programm einfach bearbeiten und somit fälschen. Bei genauerem Hinsehen erkennt man jedoch schnell, dass diese E-Mail gefälscht ist. Als Absender versteckt sich hinter den Namen Wolseley Industrial Group die E-Mail-Adresse *admin@dealer.com*. Diese E-Mail-Adresse taucht auch als Absender anderer dubioser E-Mails auf, unter anderem auch bei einem „Offiziellen Gewinnerbrief“ einer Lotterie[33]. Am 22. März 2018 warnte auch die Wolseley Industrial Group selbst vor gefälschten E-Mails in ihrem Namen[34]. Das Unternehmen warnt hierbei ausdrücklich davor, die angefügten Dateien zu öffnen, da sie mit Viren belastet sein könnten. Das Unternehmen bezeichnet diese Nachricht hierbei als eine Scam-Mail, also eine E-Mail, in welcher aufgefordert wird einen gewissen Geldbetrag zu bezahlen, um dann zu einem späteren Zeitpunkt daraus irgendeinen materiellen oder finanziellen Gewinn zu erzielen. Der deutsche Fachausdruck hierfür ist *Vorschussbetrug* und ist strafbar[35].
Welcher Harvester für das Sammeln der E-Mail-Adressen, an welche diese Spamnachrichten geschickt wurden, verantwortlich ist, lässt sich nicht mehr zurückverfolgen. Da zu diesem Zeitpunkt der E-Mail-Adressen-Präfix mit dem Unix-Timestamp, wie unter Punkt 3.2.3 erwähnt, noch nicht implementiert wurde und ein Harvester diese Adressen zu jedem beliebigen Zeitpunkt X oder

gar über mehrere Wochen verteilt gesammelt haben könnte, kann man im Nachhinein nicht mehr zweifelsfrei feststellen, welcher Harvester die betroffenen Adressen eingesammelt hat. Auch das betroffene Unternehmen gibt hierzu keine näheren Auskünfte.
Bereits wenige Wochen später, am 12. April 2018, wiederholte sich der zuvor genannte Vorfall: Eine E-Mail, diesmal vom Unternehmen *Bearing Distributors Inc*, fordert den Empfänger auf, die mitgeschickte Rechnung zu überprüfen und an eine Costanzo Helen zurückzuschicken. Das betroffene Unternehmen hat sich bislang (14. April 2018) noch nicht zum Vorfall geäußert.

### 3.3.3 Brute-Force-Tarpit

Neben der Harvester-Tarpit war auch die Brute-Force-Tarpit erfolgreich. Insgesamt wurden 7.051 Loginversuche getätigt. Hierbei wurden aus zehn verschiedene Benutzernamen und 1.107 verschiedenen Passwörtern 3.793 verschiedene Kombinationen gewählt, somit wurde jede Kombination im Schnitt mehrfach probiert. In den Tabellen 2 und 3 sind die häufigsten Benutzernamen und Passwörter aufgezeigt, welche in der simulierten Wordpressloginform eingegeben wurden.

| **Benutzername** | **Anzahl an Versuchen** |
|---|---|
| admin | 2340 |
| maturaprojekt | 1308 |
| test | 1246 |
| webmaster | 1052 |
| root | 1052 |
| administrator | 46 |

Tabelle 2: Die am häufigsten eingegebenen Benutzernamen

Analog hierzu sind in Tabelle 4 die zehn häufigsten Passwörter von 2017 aufgezeigt.[35] Vergleicht man nun Tabelle 3 mit Tabelle 4, so kann man deutlich erkennen, dass unter anderem die Standardpasswörter aus Tabelle 4 häufig probiert wurden. Dies lässt sich vor allem dadurch erklären, dass die Betreiber solcher Webcrawler bewusst ihre Liste der zu probierenden Passwörtern

[35]Die zehn häufigsten Passwörter laut SplashData, hier geht es zur kompletten Liste: (`https://13639-presscdn-0-80-pagely.netdna-ssl.com/wp-content/uploads/2017/12/Top-100-Worst-Passwords-of-2017a.pdf`).

| Passwort | Anzahl an Versuchen |
|---|---|
| admin | 24 |
| PASSWORD | 24 |
| nicole | 19 |
| michael | 19 |
| daniel | 18 |
| jessica | 17 |
| 111111 | 15 |
| lovely | 15 |
| ashley | 15 |
| iloveyou | 14 |
| ... | ... |
| THOMAS | 8 |

Tabelle 3: Die am häufigsten eingegebenen Passwörter

| Platz | Passwort |
|---|---|
| 1 | 123456 |
| 2 | Password |
| 3 | 12345678 |
| 4 | qwerty |
| 5 | 12345 |
| 6 | 123456789 |
| 7 | letmein |
| 8 | 1234567 |
| 9 | football |
| 10 | iloveyou |

Tabelle 4: Die am häufigsten genutzten Passwörter laut SplashData

mit solchen „typischen Passwörter“ füllen. Die vielen Vornamen, welche in das Passwortfeld eingetragen wurden, lassen sich jedoch nicht erklären. Wahrscheinlich hat ein Webcrawler eine sogenannte Wörterbuchattacke ausgeführt, sprich er hat Passwörter probiert, welche man in einem Wörterbuch auffinden kann. Hierzu zählen selbstverständlich auch Vornamen. Beeindruckend ist jedoch die Tatsache, dass Webcrawler auch meinen Vornamen als Passwort ausprobiert haben. Meinen Vornamen herauszufinden ist für einen Webcrawler ein leichtes Spiel: Der Meta-Tag *author* in der HTML-Seite enthält den Wert *Thomas Brixen*. Das beeindruckende und zugleich unerklärliche ist jedoch, dass kein einziges mal, weder als Benutzernamen, noch als Passwort, das

Wort *Brixen* probiert wurde, obwohl es, wie auch mein Vorname, der Wert des Metatags *author* ist. Selbstverständlich könnte ein Webcrawler den Inhalt des Metatags durch einen Filter laufen lassen und merken, dass *Brixen*, im Gegensatz zu *Thomas*, kein Name ist. Des Weiteren könnte ein Webcrawler auch anhand der IP-Adresse den ungefähren Standort des Webservers zurückverfolgen und würde dann merken, dass *Brixen* eine Stadt im unmittelbaren Umfeld ist. Beides rechtfertigt jedoch nicht, das Wort *Brixen* als einen potentiellen Benutzernamen oder Passwort auszuschließen und deshalb nicht zu probieren.

| **User-Agent (nur Browser)** | **Anzahl an Loginversuchen** |
|---|---|
| Firefox/40.1 | 3.755 |
| Firefox/3.0.15 | 3.152 |
| *Weitere User-Agents*[36] | 144 |

Tabelle 5: Die User-Agents (nur Browser) mit welchen Loginversuche getätigt wurden

Aus Tabelle 5 lässt sich des Weiteren eindeutig erkennen, dass für die Angriffe hauptsächlich zwei Versionen des Browsers Firefox, nämlich die Versionen 40.1 und 3.0.15, verantwortlich waren. Jedoch veröffentlichte Mozilla beispielsweise nie eine Version 40.1 von Firefox[36]. Nach kurzer Recherche zeigt sich, dass der User-Agent Firefox/40.1 dafür bekannt ist, weltweit Loginversuche auf Webseiten, welche Wordpress verwenden, zu tätigen[37]. Viele vermuten hinter Firefox/40.1 ein Botnetz[37], zu welchem Endgeräte auf der ganzen Welt zusammengeschlossen wurden[38]. Diese Vermutung bestätigte sich im Verlaufe dieses Versuches. In der Zeit vom 25. bis zum 28. Februar 2018 wurde VEVETA Opfer eines Angriffes eines Botnetzes. Endgeräte aus 152 verschiedenen Ländern versuchten sich über die Wordpressloginform anzumelden und tappten dabei in die Falle: Die Brute-Force-Tarpit fing sie und loggte ihre Loginversuche mit. Hierbei schienen fast ausschließlich User-Agent mit dem Browser Firefox/40.1 auf.

[37] Ein Botnetz ist eine Ansammlung von Endgeräten, Router, PCs, Smartphones, etc., welche von einem Angreifer aus der Ferne kontrolliert werden können. Durch eine Schadsoftware, welche beispielsweise zuvor über E-Mails verbreitet wurde, werden nach und nach immer mehr Endgeräte in dieses Botnetz aufgenommen. Ein großes Botnetz kann schnell mehrere Tausend Endgeräte umfassen. Meist werden Botnetze für einen DDoS-Angriff verwendet, bei welchem zu einem Zeitpunkt X die infizierten Endgeräte simultan eine Ressource auf einem Webserver anfragen, mit dem Ziel, dass dieser Webserserv die Anfragen nicht stemmen kann und zusammenbricht.

Die Angriffe des Botnetzes kamen hierbei, wie bereits erwähnt, aus 152 verschiedenen Ländern, was in Abbildung 6 nochmals verdeutlicht wird:

Abbildung 6: Länder aus denen der Angriff des Botnetzes stammte (schwarz markiert)

Hierbei kann man in Abbildung 6 deutlich erkennen, dass nahezu jedes Land Bestandteil dieses Botnetzes ist. Sowohl Internetnutzer in Schwellenländern als auch in Industriestaaten wurden Opfer einer schadhaften Software, welche dann später diesen Angriff koordiniert hat.

# 4 Schlussteil

## 4.1 Zusammenfassung der Ergebnisse

Das Internet als ein immer stärker wachsendes Medium beinhaltet bereits heutzutage eine endlos scheinende Menge an Informationen. Menschen sind mit dieser Fülle an Informationen schon seit langem überfordert. Es wurden Mittel geschaffen diese Menge an Informationen für uns Menschen aufzubereiten, nämlich Webcrawler. Sie sind, auch wenn wir sie kaum bemerken, zu einem der wichtigsten Hilfsmittel in unserer heutigen Onlinewelt geworden. Doch wie so vieles haben auch sie Schattenseiten. Einige Webcrawler versuchen Inhalte

von Webseiten zu kopieren, Passwörter zu knacken oder E-Mail-Adressen für einen späteren Versand von Spamnachrichten zu sammeln.
Durch einen aggressiven Aufruf von Webseiten versuchen bösartige Webcrawler so schnell als möglich an eine Fülle an Informationen zu gelangen und nehmen dabei auch einen Serverabsturz in Kauf. Regeln der robots.txt werden hierbei bewusst verletzt, ein Serveradministrator ist in solchen Fällen oft machtlos.
Das Versenden von Spam- oder Scramnachrichten scheint immer noch ein lukratives Business zu sein. Öffentlich einsehbare E-Mail-Adressen werden gestohlen und landen in den Listen von Harvestern. Es finden sich in den Postfächern der Opfer immer wieder fragwürdige aber echt erscheinende Mails. Diese Mails gehören ohne zu zögern in den Papierkorb, das bloße Öffnen dieser könnte bereits einen Virus starten. Das Imitieren einer E-Mail von einem echten Unternehmen bedeutet auch für dieses einen großen Schaden. Die sinkenden Reputation nach so einem Vorfall ist für ein weltweit operierenden Unternehmen meist der größte finanzielle Schaden, den sie sich vorstellen können.
Die in den Mails enthaltenen Viren können unter Umständen auch das betroffene Endgerät übernehmen und in ein Botnetz einhängen. Dieses kann dann mit einem Klick aktiviert werden und von praktisch jedem Land der Welt aus starten Endgeräte DDoS-Attacken oder Brute-Force-Versuche auf Loginformen von populären System, wie etwa Wordpress. Listen von beliebten Kombinationen aus Benutzernamen und Passwörtern werden hier meist über Tage hinweg ausprobiert. Das Opfer ist diesen Angriffen schutzlos ausgeliefert, da die vermeintlichen Angreifer normale Endgeräte sind, welche von unwissenden Nutzern wie Du und Ich verwendet werden und man sie deshalb nicht aussperren kann. Das Opfer kann hierbei nur hoffen, dass seine Kombination aus Benutzernamen und Passwort nicht erraten wird, was bei einem starken Passwort jedoch sehr unwahrscheinlich ist.

## 4.2 Ausblick

Das Durchführen dieses Versuches hat gezeigt, dass unzählige bösartige Webcrawler im Internet ihr Unwesen treiben. Die Zahl von ihnen wird täglich größer. Das Betreiben einer Tarpit als Schutzmaßnahme vor Webcrawlern und ihren Angriffen ist jedoch nicht rentabel. Webcrawler verändern sich über die Zeit, werden schlauer, erkennen Gefahren und reagieren darauf. Um mit den Webcrawlern Schritt halten zu können müssen sich auch die Tarpits dementsprechend weiterentwickeln aber sie hinken dabei den Webcrawlern immer einen Schritt hinterher: Auf ein Angriffsmuster kann erst reagiert werden, nach-

dem es schon einmal angewendet wurde, doch dann ist es meist schon zu spät. Das Betreiben und Weiterentwickeln von Tarpits ist ein ressourcen- und zeitaufwendiges Unterfangen, dessen Kosten ein Unternehmen erst einmal stemmen muss. Das investierte Geld wäre in qualitative Antivirensoftware, Spamfilter oder dergleichen besser investiert. Auch kann eine einzige Tarpit niemals gegen die schier endlose Anzahl an Webcrawlern aufkommen. Fängt man einen, können auf dem Webserver des Betreibers des Webcrawlers noch zehn weitere auf ihren Einsatz warten. Ein Verbund von Tarpits wäre erforderlich, dessen Koordination erneut Unsummen verschlingen würde. Des Weiteren ist zu bedenken, dass sich auch die Betreiber von Webcrawlern zusammenschließen könnten und ein gezielter Angriff von Webcrawlern, welche über ein oder gar mehrere Botnetze betrieben werden, nahezu jeden Webserver in die Knie zwingen könnte.
So interessant die Vorstellung einer Tarpit und dessen Konzept der „Rache“ an einem Webcrawler auch klingt, ist die Umsetzung zum jetzigen Zeitpunkt wirtschaftlich nicht rentabel und praktisch auch nahezu unmöglich. Hier heißt es einfach abwarten was die Zukunft bringt. Doch spätestens dann, wenn die rapiden Fortschritte in der künstlichen Intelligenz Früchte tragen oder die Theorien des Quantencomputers zur Realität werden, müssen wir uns alle Gedanken über unsere Sicherheit im Internet machen, denn unsere jetzigen Sicherheitskonzepte werden bei weitem nicht mehr ausreichend sein.

# Literatur


[1] I. Zeifman. (24. Jan. 2017). Bot traffic report 2016, Adresse: https://www.incapsula.com/blog/bot-traffic-report-2016.html (besucht am 12. 02. 2018).

[2] P. Christensson. (14. Feb. 2014). Bot definition, Adresse: https://techterms.com/definition/bot (besucht am 12. 02. 2018).

[3] ——, (2006). Spider definition, Adresse: https://techterms.com/definition/spider (besucht am 12. 02. 2018).

[4] S. Dhenakaran und K. T. Sambanthan, »Web crawler - an overview«, *International Journal of Computer Science and Communication*, März 2011. Adresse: http://csjournals.com/IJCSC/PDF2-1/Article_49.pdf.

[5] I. L. Stats. (). Total number of websites, Adresse: http://www.internetlivestats.com/ (besucht am 17. 02. 2018).

[6] J. Atwood. (25. März 2011). Eeeeek where is my slow performance??? [closed], Adresse: https://meta.stackexchange.com/questions/84695/eeeeek-where-is-my-slow-performance/84701#84701 (besucht am 24. 02. 2018).

[7] T. Everts. (9. Aug. 2017). The average web page is 3mb. how much should we care?, Adresse: https://speedcurve.com/blog/web-performance-page-bloat/.

[8] M. Koster. (). A standard for robot exclusion, Adresse: http://www.robotstxt.org/orig.html (besucht am 18. 03. 2018).

[9] ——, (2011). Historical web services, Adresse: http://www.greenhills.co.uk/historical.html.

[10] W. T. Surveys. (24. März 2018). Usage of web servers for websites, Adresse: https://w3techs.com/technologies/overview/web_server/all (besucht am 24. 03. 2018).

[11] C. Hoffman. (28. Sep. 2016). What is a browser's user agent?, Adresse: https://www.howtogeek.com/114937/htg-explains-whats-a-browser-user-agent/ (besucht am 04. 03. 2018).

[12] W. Recommendation. (). Robots and the meta element, Adresse: https://www.w3.org/TR/html4/appendix/notes.html#h-B.4.1.2 (besucht am 24. 03. 2018).

[13] M. Koster. (). About the robots <meta> tag, Adresse: http://www.robotstxt.org/meta.html (besucht am 24. 03. 2018).

[14] I. P. T. Council. (). Solutions to address the challenges of communicating digital rights and permissions, Adresse: http://www.the-acap.org/index.php (besucht am 24. 03. 2018).

[15] iTWire. (18. März 2008). Acap content protection protocol "doesn't workß-ays google ceo, Adresse: https://www.itwire.com/your-it-news/home-it/17206-acap-content-protection-protocol-qdoesnt-workq-says-google-ceo (besucht am 24. 03. 2018).

[16] lunapark. (11. Dez. 2017). Suchmaschinenmarktanteile weltweit 2017, Adresse: https://www.luna-park.de/blog/9907-suchmaschinen-marktanteile-weltweit-2014/ (besucht am 25. 03. 2018).

[17] Google. (). Googlebot, Adresse: https://support.google.com/webmasters/answer/182072?hl=de (besucht am 25. 03. 2018).

[18] R. Zimmermann. (18. Apr. 2017). Crawling und indexierung von webseiten: Theorie und praxis, Adresse: https://www.more-fire.com/blog/crawling-und-indexierung-von-webseiten-theorie-und-praxis/ (besucht am 04. 04. 2018).

[19] M. Gaffan. (21. Aug. 2012). Top 10 bots you should know about, Adresse: https://www.incapsula.com/blog/know-your-top-10-bots.html (besucht am 25. 03. 2018).

[20] Google. (). Google-crawler, Adresse: https://support.google.com/webmasters/answer/1061943?hl=de (besucht am 25. 03. 2018).

[21] I. Zeifman. (24. Juli 2014). Mr. hack: Googlebot's unruly alter ego, Adresse: https://www.incapsula.com/blog/googlebot-study-mr-hack.html.

[22] J. Gallenbacher, *Abenteuer Internet, IT zum Anfassen für alle von 9 bis 99 - vom Navi bis Social Media*, 4. Aufl. Springer, 2017, ISBN: 978-3-662-53964-4.

[23] "Wikipedia". (8. Feb. 2017). Teergrube (informationstechnik), Adresse: https://de.wikipedia.org/wiki/Teergrube_(Informationstechnik) (besucht am 29. 03. 2018).

[24] P. Christensson. (7. Sep. 2013). Honeypot definition, Adresse: https://techterms.com/definition/honeypot (besucht am 05. 05. 2018).

[25] T. Eggendorfer, »E-mail-adressenjägern auf webseiten eine falle stellen«, *Linux-Magazin*, Juni 2004. Adresse: http://www.linux-magazin.de/ausgaben/2004/06/ernte-nein-danke/.

[26] A. Mitra. (19. Sep. 2015). What is tarpit and how does it improve security ?, How does tarpit work?, Adresse: https://computersecuritypgp.blogspot.it/2015/09/what-is-tarit-and-how-it-enhances.html (besucht am 03. 29. 2018).

[27] T. Eggendorfer, »Mit teergruben aktiv gegen spammer vorgehen – welche wirken tatsächlich?«, *Linux Magazin*, Jan. 2007. Adresse: http://www.linux-magazin.de/ausgaben/2007/01/her-mit-dem-abfall/.

[28] T. Eggendorfer, Hrsg., *The Conference for Unix, Linux and Open Source Professionals, AUUG 2005*, **presentedat**Stopping Spammers' Harvesters using a HTTP-tar pit - Details on implementation and a real-world experiment, Okt. 2005. Adresse: https://books.google.it/books?id=iJw5zAu7LncC&pg=PA68&lpg=PA68&dq=http+tarpit&source=bl&ots=DNAI72u8Fa&sig=qBLDp--ttBS-RD3p0G1b89gSJew&hl=de&sa=X&ved=0ahUKEwjnqOe7oJHaAhWKiaYKHT2sAGEQ6AEIggEwCQ#v=onepage&q=http%20tarpit&f=false (besucht am 29. 03. 2018).

[29] M. web docs. (24. Juli 2017). Last-modified, Syntax, Mozilla, Adresse: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Last-Modified (besucht am 02. 04. 2018).

[30] Statista. (Jan. 2018). Ranking der 10 content-management-systeme (cms) weltweit nach marktanteil im januar 2018, Adresse: https://de.statista.com/statistik/daten/studie/320670/umfrage/marktanteile-der-content-management-systeme-cms-weltweit/ (besucht am 03. 04. 2018).

[31] crunchbase. (). Dotmic, Adresse: https://www.crunchbase.com/organization/dotmic#section-overview (besucht am 04. 04. 2018).

[32] distil networks. (). Dotbot/1.1 user agent string, Adresse: https://www.distilnetworks.com/bot-directory/bot/dotbot1-1/ (besucht am 04. 04. 2018).

[33] S. Brück. (9. Juni 2017). Offiziellegewinner brief von internacionale lotto commission (admin@dealer.com), Adresse: https://vorsicht-email.de/beitrag/2017/06/09/offiziellegewinner-brief-von-internacionale-lotto-commission-admindealer-com/ (besucht am 11. 04. 2018).

[34] W. I. Group. (22. März 2018). Notice: Malicious email scam, Adresse: http://wolseleyindustrialgroup.com/news/notice-malicious-email-scam/ (besucht am 11. 04. 2018).

[35] simplepedia. (28. Jan. 2016). Vorschussbetrug, Adresse: https://www.scambaiter-forum.info/wiki:vorschussbetrug (besucht am 11. 04. 2018).

[36] Mozilla. (). Firefox releases, Adresse: https://www.mozilla.org/en-US/firefox/releases/ (besucht am 14. 04. 2018).

[37] Haroldoland. (16. Feb. 2017). Firefox 40.1 and wordpress attacks, Adresse: https://haroldoland.blogspot.it/2017/02/firefox-401-and-wordpress-attacks.html (besucht am 14. 04. 2018).

[38] M. Maunder. (11. Apr. 2017). Thousands of hacked home routers are attacking wordpress sites, Adresse: https://www.wordfence.com/

blog/2017/04/home-routers-attacking-wordpress/ (besucht am 14. 04. 2018).

# Abbildungsverzeichnis



# Tabellenverzeichnis