



A Walk on the Web's Wild Side

STUDIENARBEIT

für die Prüfung zum

Bachelor of Science

des Studiengangs Informatik
Studienrichtung Angewandte Informatik

an der

Dualen Hochschule Baden-Württemberg Karlsruhe

von

**Samuel Philipp
Daniel Brown
Jan-Eric Gaidusch**

17. April 2017

Bearbeitungszeitraum

6 Monate

Matrikelnummern

9207236, 3788021, 8296876

Kurs

TINF14B2

Ausbildungsfirma

Fiducia & GAD IT AG

Gutachter der Studienakademie

Dr. Martin Johns

Abstract

Erklärung

(gemäß §5(3) der „Studien- und Prüfungsordnung DHBW Technik“ vom 29.9.2015)

Wir versichern hiermit, dass wir unsere Studienarbeit mit dem Thema:

„A walk on the web’s wild side“

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben. Wir versichern zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Karlsruhe, den 17. April 2017

Ort, Datum

Samuel Philipp

Karlsruhe, den 17. April 2017

Ort, Datum

Daniel Brown

Karlsruhe, den 17. April 2017

Ort, Datum

Jan-Eric Gaidusch

Inhaltsverzeichnis

Abkürzungsverzeichnis	V
Abbildungsverzeichnis	VI
Tabellenverzeichnis	VII
Listings	VIII
1 Einleitung	1
1.1 Einführung	1
1.2 Hintergrund	1
1.3 Team	1
1.4 Aufgabenstellung	2
1.5 webifier	2
2 Grundlagen	4
2.1 Frontend Technologien und Frameworks	4
2.2 Backend Technologien und Frameworks	4
2.3 Technologien und Frameworks der Tests	6
2.4 Angriffstypen	6
2.4.1 Malware	6
2.4.2 Request Header Investigation	6
2.4.3 JavaScript Port Scanning	6
2.4.4 JavaScript IP Scanning	6
2.4.5 Clickjacking	7
2.4.6 Phishing	7
3 Konzept	8
3.1 Gesamtkonzept	8
3.1.1 webifier Tests	8

3.1.2	webifier Tester	8
3.1.3	webifier Plattform	8
3.1.4	webifier Mail	8
3.1.5	webifier Data	8
3.1.6	webifier Statistics	8
3.2	Testarten	9
3.2.1	Virensan	9
3.2.2	Vergleich in verschiedenen Browsern	9
3.2.3	Test auf Port Scanning	9
3.2.4	Test auf IP Scanning	9
3.2.5	Link Checker	9
3.2.6	Google Safe Browsing	10
3.2.7	Überprüfung des Zertifikats	10
3.2.8	Erkennung von Phishing	10
3.2.9	Screenshot	10
4	Umsetzung	11
4.1	Gesamtanwendung	11
4.1.1	webifier Tests	11
4.1.2	webifier Tester	11
4.1.3	webifier Plattform	11
4.1.4	webifier Mail	11
4.1.5	webifier Data	11
4.1.6	webifier Statistics	11
4.2	Tests	12
4.2.1	Virensan	12
4.2.2	Vergleich in verschiedenen Browsern	12
4.2.3	Test auf Port Scanning	12
4.2.4	Test auf IP Scanning	12
4.2.5	Linkchecker	12
4.2.6	Google Safe Browsing	12
4.2.7	Überprüfung des Zertifikats	12
4.2.8	Erkennung von Phishing	12
4.2.9	Screenshot	12
5	Analyse	13

6 Ausblick	14
6.1 Weitere Tests	14
6.2 Weitere Module	14
7 Fazit	15
7.1 Zusammenfassung	15
7.2 Bewertung der Ergebnisse	15
Literaturverzeichnis	X

Abkürzungsverzeichnis

WWW World Wide Web

JVM Java Virtual Machine

API Application Programming Interface

Abbildungsverzeichnis

1	Secutitysquad - Logo	1
2	webifier - Logo	2

Tabellenverzeichnis

Listings

1 Einleitung

1.1 Einführung

TODO Samuel

1.2 Hintergrund

TODO Jani

1.3 Team



Abbildung 1: Secutitysquad - Logo

1.4 Aufgabenstellung

Anbieter von zwielichtigen Web-Angeboten greifen ihre User mit diversen Client-seitigen Methoden an. Beispiele für solche Angriffe sind Malware Downloads, Phishing, JavaScript Intranet Angriffe, oder Browser Exploits.

Ziel der Arbeit ist eine systematische Untersuchung der Aktivitäten von semi-legalen Webseiten im World Wide Web (WWW). Das erwartete Ergebnis ist ein Prüfportal, auf dem jene Webseiten automatisiert analysiert werden und Ergebnisse präsentiert werden sollen.

Nach dem ersten Schaffen einer Übersicht von interessanten Zielen, wie z.B. One-Click-Hoster oder File-sharing Sites sollen ausgewählte Webseiten manuell untersucht werden. Außerdem sollen verschiedene Angriffsszenarien zur weiteren Prüfung ausgewählt werden. Der Untersuchungsprozess der Webseiten soll im Verlauf dieser Arbeit stückweise automatisiert und in den Rahmen einer Prüfanwendung gebracht werden.

Abschließend sollen eine Vielzahl von Webseiten mit der Anwendung getestet und die Ergebnisse ausgewertet und dokumentiert werden.

1.5 webifier



Abbildung 2: webifier - Logo

webifier ist eine Anwendung, mit der Webseiten auf deren Seriosität und mögliche clientseitige Angriffe auf den Nutzer geprüft werden können. Sie besteht aus mehreren eigenständigen Teilanwendungen. Im Zentrum steht der Tester, welcher die einzelnen Tests verwaltet, ausführt und anschließend die Ergebnisse auswertet. Jeder einzelne Test ist eine weitere isolierte Teilanwendung des Testers. So kann jeder Test unabhängig von allen anderen betrieben werden.

Die Plattform ist eine Webanwendung welche den Endnutzern eine grafische Oberfläche zur Verfügung stellt, um Webseiten zu überprüfen. Im Hintergrund setzt die Plattform auf den Tester auf. webifier Mail ist ein Dienst mit dem Links aus E-Mails überprüft werden können. Anschließend erhält der Sender eine E-Mail mit den Resultaten zurück.

Eine weitere Teilanwendung von webifier ist das Data-Modul. Es stellt eine Schnittstelle für den Tester bereit, um alle Testergebnisse sammeln zu können. Das Statistik-Modul ist die letzte Teilanwendung von webifier. Es setzt auf das Data-Modul auf und stellt Funktionen zur Auswertung aller Testergebnisse bereit.

Um die Techniken und Algorithmen von webifier verstehen zu können sind einige Grundlagen erforderlich, welche nun im nächsten Kapitel genauer vorgstellt werden.

2 Grundlagen

In diesem Kapitel werden die Grundlagen, welche für das weitere Verständnis der Arbeit und der gesamten Anwendung notwendig sind, näher beschrieben. Zunächst werden die verschiedenen Technologien und Frameworks, sowohl des Frontends, als auch des Backends dargestellt. Anschließend werden einige gängige Angriffstypen im WWW erläutert, welche webifier überprüft.

2.1 Frontend Technologien und Frameworks

- HTML
- CSS
- JavaScript
- jQuery
- Bootstrap

2.2 Backend Technologien und Frameworks

In diesem Abschnitt werden nun alle Technologien und Frameworks vorgestellt welche in den Backends der einzelnen Teilanwendungen zum Einsatz kamen.

Wohl am häufigsten kam die Programmiersprache Java zum Einsatz. Java ist eine universal einsetzbare, nebenläufige, klassenbasierte und objektorientierte Programmiersprache. Sie wurde möglichst einfach gestaltet um von vielen Entwicklern genutzt zu werden. In ihrer Syntax ähnelt sie den Programmiersprachen C und C++. Außerdem ist sie stark und statisch typisiert. Vorallem aber zeichnet sich Java durch seine plattformunabhängigkeit aus. Diese wird dadurch umgesetzt, dass Java-Quellcode in plattformunabhängigen Byte-Code kompiliert wird, welcher von einer Java Virtual Machine (JVM) ausgeführt wird. Java ist eine Hochsprache, die mit Hilfe des so genannten „Garbage Collectors“ eine automatische Speicherverwaltung bereitstellt.¹

In einigen Teilprojekten wurde das auf Java basierende *Spring*-Framework verwendet. *Spring* stellt eine vereinfachte Möglichkeit auf den Zugriff auf viele Application Programming Interface (API) der Standard-Version zur Verfügung. Ein weiterer wesentlicher Bestandteil des *Spring*-Frameworks ist die *Dependency Injection*. Hierbei suchen sich Objekte ihre Referenzen nicht selbst, sondern bekommen diese Anhand einer Konfiguration injiziert. Dadurch sind sie eigenständig und können in verschiedenen Umgebungen eingesetzt werden. Des weiteren bringt *Spring* eine Unterstützung für aspektorientierte Programmierung mit, wodurch mit verschiedenen Abstraktionsschichten einzelne Module abgekapselt werden können.²

Aufbauend auf dem *Spring* Basis-Modul werden noch weitere Module, wie beispielsweise Spring Security, Spring Boot, Spring Integration, Spring Data, Spring Session oder Spring Web Services.³ ...

TODO Samuel

- MongoDB

TODO Samuel

- Gradle

TODO Jani

- REST

TODO Jani

- Docker

TODO Jani

¹ Vgl. Gosling u. a. (2014), S. 1

² Vgl. Wolff (2011), S. 2

³ Vgl. Cosmina (2016), S. 2

- R
TODO Jani

2.3 Technologien und Frameworks der Tests

- Phantom JS
- Bro
TODO Jani
- Python
- HTtrack
TODO Samuel
- Resemble JS
TODO Samuel

2.4 Angriffstypen

2.4.1 Malware

TODO Samuel

2.4.2 Request Header Investigation

2.4.3 JavaScript Port Scanning

TODO Jani

2.4.4 JavaScript IP Scanning

TODO Jani

2.4.5 Clickjacking

TODO Jani

2.4.6 Phishing

TODO Samuel

3 Konzept

3.1 Gesamtkonzept

3.1.1 webifier Tests

TODO Jani

3.1.2 webifier Tester

TODO Samuel

3.1.3 webifier Platform

3.1.4 webifier Mail

3.1.5 webifier Data

TODO Samuel

3.1.6 webifier Statistics

TODO Jani

3.2 Testarten

3.2.1 Virensan

TODO Samuel

- Httrack (Umsetzung)
- Download aller Dateien der Webseite
- Scannen der Heruntergeladenen Dateien
 - Clamav (Umsetzung)
 - AVG (Umsetzung)
 - CAV (Umsetzung)

3.2.2 Vergleich in verschiedenen Browsern

3.2.3 Test auf Port Scanning

TODO Jani

3.2.4 Test auf IP Scanning

TODO Jani

3.2.5 Link Checker

- herausfiltern aller Links und nachgeladenen Ressourcen

3.2.6 Google Safe Browsing

3.2.7 Überprüfung des Zertifikats

TODO Samuel

- Auslesen der relevanten Informationen des Zertifikates der Webseite
- Validierung des Zertifikates

3.2.8 Erkennung von Phishing

TODO Samuel

- Herausfiltern der Schlagwörter
- Finden möglicher Duplikate der Webseite
 - Erstes Schlagwort zu Top Level Domains
 - * com
 - * ru
 - * net
 - * org
 - * de
 - Websuche nach den Schlagwörtern mittels Suchmaschinen
 - * DuckDuckGo
 - * Ixquick
 - * Bing

3.2.9 Screenshot

TODO Jani

4 Umsetzung

4.1 Gesamtanwendung

4.1.1 webifier Tests

TODO Jani

4.1.2 webifier Tester

TODO Samuel

4.1.3 webifier Platform

4.1.4 webifier Mail

4.1.5 webifier Data

TODO Samuel

4.1.6 webifier Statistics

TODO Jani

4.2 Tests

4.2.1 Virenscan

TODO Samuel

4.2.2 Vergleich in verschiedenen Browsern

4.2.3 Test auf Port Scanning

TODO Jani

4.2.4 Test auf IP Scanning

TODO Jani

4.2.5 Linkchecker

4.2.6 Google Safe Browsing

4.2.7 Überprüfung des Zertifikats

TODO Samuel

4.2.8 Erkennung von Phishing

TODO Samuel

4.2.9 Screenshot

TODO Jani

5 Analyse

6 Ausblick

6.1 Weitere Tests

6.2 Weitere Module

7 Fazit

7.1 Zusammenfassung

7.2 Bewertung der Ergebnisse

Literaturverzeichnis

Cosmina, Iuliana (2016):

Pivotal Certified Professional Spring Developer Exam: A Study Guide, 3. Auflage,
Apress

Gosling, James u. a. (2014):

The Java Language Specification - Java SE 8 Edition, 5. Auflage, Addison-Wesley

Wolff, Eberhard (2011):

Spring 3 – Framework für die Java Entwicklung, 3. Auflage, dpunkt.verlag