Völlig überarbeitete & aktualisierte **2. Auflage**

Tim Philipp Schäfers

Hacking im Web

Denken Sie wie ein Hacker und schließen Sie die Lücken in Ihren Webapplikationen

- Angriffe verstehen: Session Hijacking, Cross-Site-Request-Forgery, Cross-Site-Scripting, File Inclusion, SQL-Injection, UI-Redressing und mehr
- Eigene Testumgebung aufbauen und Kenntnisse direkt umsetzen
- 35 Sicherheits-Tools kennenlernen und zum Aufspüren von Lücken nutzen

FRANZIS

Tim Philipp Schäfers, Jahrgang 1995, ist seit seinem Abitur als Security-Consultant tätig und berät Firmen im Bereich Websicherheit/Webhacking. Schäfers deckte gravierende Sicherheitslücken bei Unternehmen wie PayPal, Facebook, Google, der deutschen Telekom und vielen weiteren Unternehmen verantwortlich auf. Er ist Mitgründer und Mitbetreiber des Webprojekts „Internetwache.org“, das sich ebenfalls mit der Sicherheit von Webapplikationen und Webhacking beschäftigt. Seit Ende 2015 schreibt Schäfers zudem auf dem Web-Portal golem.de über die Themen IT-Sicherheit und Privatsphäre.

Tim Philipp Schäfers

Hacking im Web

Denken Sie wie ein Hacker und schließen Sie die Lücken in Ihren Webapplikationen

- Angriffe verstehen: Session Hijacking, Cross-Site-Request-Forgery, Cross-Site-Scripting, File Inclusion, SQL-Injection, UI-Redressing und mehr
- Eigene Testumgebung aufbauen und Kenntnisse direkt umsetzen
- 35 Sicherheits-Tools kennenlernen und zum Aufspüren von Lücken nutzen

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte Daten sind im Internet über *http://dnb.ddb.de* abrufbar.

Autor: Tim Philipp Schäfers
Programmleitung: Dr. Markus Stäuble, Benjamin Hartlmaier
Lektorat: Hans Hartings
Satz: DTP-Satz A. Kugge, München
art & design: www.ideehoch2.de

ISBN 978-3-645-20636-5

Vorwort

Kaum ein Monat vergeht, in dem nicht bekannt wird, dass ein Unternehmen Opfer eines Hackerangriffs geworden ist. Oftmals wird die Webseite verunstaltet, Social-Media-Accounts geraten unter die Kontrolle von unbekannten Angreifern, oder es wird gar die gesamte Onlinedatenbank eines Unternehmens kopiert. Als Einstiegspunkt für Angreifer dient oft das Web, genauer die Webapplikationen, die auf Servern im Internet betrieben werden. Im Handumdrehen kann die Webseite eines Unternehmens zum Einfallstor werden oder der Webshop zur Goldgrube für Onlinekriminelle. Solche Vorgänge schaden dem Geschäft meist erheblich.

Die (Un-)Sicherheit Ihrer Webapplikationen entscheidet letztlich auch über den Erfolg oder Misserfolg Ihres Unternehmens, denn oftmals findet über diese Anwendungen die Abwicklung ganzer Geschäfts(teil)prozesse statt. Genau um diese Thematik soll es in diesem Buch gehen: die Sicherheit von Webanwendungen mithilfe eines sicheren Programmcodes und sicherer Konfiguration von Webservern zu gewährleisten.

Längst wird für eine sichere Webapplikation mehr benötigt als Standardlösungen oder Firewalls, vielmehr ist ein hohes Maß an Wissen nötig, um überhaupt nachvollziehen zu können, was mögliche Angreifer auf Applikationsebene alles versuchen, um Systeme zu hacken. Da verschiedene Arten von Angriffen zur Verfügung stehen, geht dieses Fachbuch auf die gängigsten Angriffsvektoren beim Webhacking ein und versucht zu erklären, wie ihnen begegnet werden kann.

Die Komplexität und das schnelle Wachstum des Webs und der damit verbundenen Technologien haben seit seiner Entstehung für einiges Durcheinander gesorgt. Dieser Wirrwarr wirkt sich auf die Sicherheit von Systemen und Webapplikationen negativ aus. Es gibt im Web Standards, die nicht eingehalten werden, gelebte Regeln (die eigentlich nur Vorschläge waren) und sicherlich auch eine Menge verzweifelter Entwickler, die sich fragen, wie sie da den Durchblick behalten können.

»Hacking im Web« soll hier weiterhelfen: Es werden Angriffsmöglichkeiten aufgezeigt, zahlreiche Tipps gegeben und Vorschläge zur Behebung von Lücken gemacht, damit Sie wissen, wie auf mögliche Gefahren reagiert werden kann. Dadurch können Angriffsflächen bereits während der Architektur- und Anwendungsentwicklung minimiert werden.

Darüber hinaus soll Ihnen dieses Buch das notwendige Werkzeug an die Hand geben, um selbst Angriffe auf Ihre Webapplikationen zu Testzwecken durchführen zu können.

In der zweiten, überarbeiteten Auflage von »Hacking im Web« wurden dazu ein Kapitel zur Einrichtung einer Testumgebung eingefügt und zudem die Nutzung von Tools stärker in den einzelnen Kapiteln berücksichtigt. Sofern Sie also selbst die vorgestellten Schwachstellen ausnutzen möchten, empfiehlt es sich, die Schritte aus Kapitel 3 zu befolgen und die in den Kapiteln vorgestellten Schwachstellen direkt selbst auszuprobieren. Neben inhaltlichen Anpassungen in den übrigen Kapiteln wurden mit Blick auf neue Technologien (Cloud-Speicher) damit verbundene Schwachstellen und sie auszunutzende Angriffsvektoren hinzugefügt.

Das frühere Kapitel zum Basiswissen (Kapitel 3 der ersten Auflage) wurde aus dem Buch entfernt. Es lässt sich mittlerweile auf der Webseite zum Buch kostenfrei abrufen: *https://hacking-im-web.de/basiswissen.pdf*

Begleitend zum Buch wird es Kurz-Videos zum Thema Web- und IT-Sicherheit geben, die nach Veröffentlichung des Buches erscheinen. Um auf dem Laufenden zu bleiben, empfiehlt es sich, die Webseite zum Buch (*https://hacking-im-web.de*) periodisch abzurufen oder dem Autor auf Twitter zu folgen (*https://twitter.com/TimPhSchaefers*).

Nun wünsche ich Ihnen erst einmal viel Freude bei der Lektüre und Erfolg bei der praktischen Umsetzung!

Tim Philipp Schäfers, Detmold im Sommer 2018

Danksagung

Dieses Buch konnte nur in dieser Form erscheinen, weil zahlreiche Personen mich in meinem Vorhaben, einiges über die Sicherheit von Webapplikationen zu Papier zu bringen, unterstützt haben.

Ein großes Dankeschön für geniale Ideen, geistreiche Einfälle und alle sonstigen Formen der Unterstützung geht an meine Freunde und meine gesamte Familie.

Des Weiteren gilt mein ganz besonderer Dank Rico Walde und Sebastian Neef. Beide haben mit ihrem fundierten Fachwissen und durch hilfreiche Rückmeldungen zu einzelnen Kapiteln maßgeblich dazu beigetragen, dass dieses Fachbuch hoffentlich ausgeglichene Anteile an Theorie und Praxis aufweist und noch dazu angenehm zu lesen ist.

Zudem möchte ich Dr. Markus Stäuble, dem ehemaligen Lektor und Programmleiter »Professional Series« beim Franzis Verlag, danken. Seine Ausdauer, seine Anregungen und Tipps und nicht zuletzt auch die Begeisterung für das Thema haben mich als recht jungen Erstautor 2016 motiviert, das Buchprojekt durchzuführen und mit dem Buch Teil der Serie zum Thema »Hacking« im Franzis Verlag zu werden.

2018 erscheint dieses Buch in zweiter Auflage. Daran haben weitere Personen mitgearbeitet und mich unterstützt. Mein Dank gilt vor allem Benjamin Hartlmaier und dem Team vom Franzis Verlag, die immer ein offenes Ohr hatten und mich bei der Überarbeitung des Buches tatkräftig unterstützt haben.

Außerdem möchte ich Hans Hartings ganz herzlich danken, der durch seine zahlreichen Hinweise im Rahmen des Lektorats maßgeblich zur Steigerung der sprachlichen Qualität dieses Buches beigetragen hat und insbesondere in der Endphase des Projektes mir mit Rat und Tat zur Seite stand.

Abschließend möchte ich Ihnen, liebe Leser, stellvertretend für eine ganze Reihe von Webnutzern danken! Das moderne Web existiert nicht ohne neugierige Entwickler, mutige Gestalter, gewiefte Blogger, clevere Hacker oder skeptische Mitleser. In der menschlichen Evolution gab es wohl kaum eine andere Kommunikations- und Verbindungsart, die so vielfältig, umfassend und allgegenwärtig war oder ist wie das Web — deshalb: Danke, dass Sie alle es täglich nutzen und es so zu dem machen, was es ist!

Inhalt

Einleitung

Webapplikationen sind im Geschäftsalltag in nahezu jedem Unternehmen allgegenwärtig: Das fängt bei einem einfachen Ticketsystem für Supportleistungen an, führt über die Firmenwebsite zu einer komplexen Webanwendung für die Bestellung der Waren durch Kunden (E-Commerce) und reicht bis hin zu sensiblen Gehaltsabrechnungssystemen. Viele Anwendungen haben heutzutage Programmierschnittstellen (APIs) mit dem Web, und auch der Austausch von Unternehmensdaten mit Dritten wird immer mehr die Regel.

Was wäre, wenn plötzlich der Webshop oder die Internetseite Ihres Unternehmens für mehrere Tage nicht erreichbar wäre, oder das Banner einer Hackergruppe darauf thronte? Wie gehen Sie damit um, wenn plötzlich Ihre Kundendatenbank in ominösen Onlineforen auftaucht oder vertrauliche E-Mails in sozialen Netzwerken für einen Sturm der Entrüstung (Shitstorm) sorgen?

Eines der prominentesten Beispiele für die gravierenden Folgen eines Hackerangriffs ist der Hack von Yahoo im Jahr 2013. Bei diesem Hack gelang es Unbekannten die Daten von drei Milliarden Nutzern zu kopieren [1]. Der Angriff ist ausgerechnet kurz vor dem Verkauf des Unternehmens Yahoo an Verizon bekannt geworden und hat abschließend sogar die US-Börsenaufsicht auf den Plan gerufen [2]. Das Unternehmen sollte ursprünglich für über 4,8 Milliarden US-Dollar verkauft werden – durch

das Bekanntwerden des Hacks wurde der Kaufpreis von Yahoo nachträglich auf 4,48 Milliarden US-Dollar (um circa 300 Millionen US Dollar) gesenkt, und in der Folge des Hacks ist die ehemalige Chefin von Yahoo (Marissa Mayer) zurückgetreten [3]. Wie sich an dem Beispiel zeigt, kann ein Hack auf ein Unternehmen immense Auswirkungen haben, nicht zuletzt haben neben dem Betreiber vor allem auch die Nutzer dieser Applikationen unter solchen Raubzügen im digitalen Raum zu leiden.

Der australische Sicherheitsforscher Troy Hunt hat eine Webseite zu Datenleaks erstellt, auf der Sie durch Eingabe Ihrer E-Mail-Adresse prüfen können, ob Sie selbst schon einmal von einem »Data Breach« betroffen waren:

```
https://haveibeenpwned.com/
```

Trotz all dieser Hacks hat sich recht wenig getan, schließlich lesen wir Nachrichten zu solchen Hacks immer wieder. Umso wichtiger ist es, dass Sie sich mit Ihren Systemen beschäftigen und solchen Angriffen präventiv begegnen, indem Sie sie stärker sichern.

1.1 Der Ansatz

In diesem Buch soll ganz pragmatisch gezeigt werden, wie die Programmierung einer sicheren Webapplikation möglich ist. Mittels realer Sicherheitslücken und durch Codebeispiele wird deutlich gemacht, wie schnell Hacker einer sicher geglaubten Webanwendung habhaft werden können. Um dies zu erreichen, wird immer wieder die Perspektive eines potenziellen Angreifers eingenommen und beispielsweise an Hand von Angriffen auf die Testumgebung gezeigt, welche Folgen Angriffe haben können. Dadurch soll die Bedrohung realistisch nachgeahmt und verdeutlicht werden, dass sie nicht nur durch IT-Profis realisiert werden kann, sondern durch jeden, der genug Zeit aufwendet und Interesse aufbringt. Letzteres ist auch der Grund, weshalb es solche Angriffe täglich tausendfach gibt.

1.2 Das Ziel

Für Entwickler oder IT-Sicherheitsverantwortliche ist es immer wieder sinnvoll, sich in die Lage eines Angreifers zu versetzen, um die eigene Anwendung besser gegen Angriffe zu schützen. Ziel des Buchs ist also die Schaffung eines Bewusstseins dafür, dass jeder geschriebene Code zum Einfallstor werden kann, wenn man ihn nicht so genau und korrekt implementiert wie notwendig. Das Buch zielt darauf ab, dass Sie um die Ecke zu denken lernen und noch nicht zufrieden sind, wenn eine Funktionali-

tät wie gewünscht implementiert wurde, sondern erst, wenn sie ausschließlich wie gewünscht verwendet werden kann und somit »sicher« ist.

1.3 Der Aufbau

Zu Anfang werden die gemeinsamen Grundlagen für das weitere Verständnis des Buchs gelegt. Es wird auf den historischen Entwicklungsprozess der Webanwendungen und des Webs eingegangen (siehe Kapitel 2, »Evolution des Webs«) und anschließend darauf, wie sich der Aspekt der Websicherheit in den letzten Jahren verändert hat.

Danach wechseln wir durch das Einrichten der Testumgebung direkt in die Praxis (siehe Kapitel 3, »Einrichtung der Testumgebung«). Die Testumgebung dient dazu, Angriffe zu beschriebenen Angriffsvektoren selbst durchzuführen und zu Testzwecken mit den über die in Kapitel 14 beschriebenen Angriffstools zu verfügen.

Im anschließenden Kapitel beginnt die Auseinandersetzung mit verschiedenen Angriffsvektoren. Wir werden schauen, wie man sogenannten Sessions habhaft werden kann und welche Möglichkeiten zum Schutz bestehen (siehe Kapitel 4, »Session-Angriffe«).

Das darauffolgende Kapitel widmet sich der Problematik des sogenannten Cross-Site-Scriptings. Wir werden verschiedene Techniken betrachten, um anschließend mögliche Schutzmaßnahmen zu evaluieren (siehe Kapitel 5, »Cross-Site-Scripting (XSS)«).

In Kapitel 6 wird es um nachgelagerte Datensysteme gehen. Wir werden feststellen, dass Angreifer bei unsicheren Applikationen mittels SQL-Injection Zugriff auf Datenbanken erlangen können und dass es weitere Speichermöglichkeiten gibt, bei denen Angreifer an Daten gelangen können. Natürlich betrachten wir auch, wie sich dieser unautorisierte Zugriff vermeiden lässt (siehe Kapitel 6, »Angriffe auf nachgelagerte Datenbanksysteme«).

Authentifizierungssysteme gelten neben dem Session-Management als Kernstück vieler Webapplikationen, da darüber Zugriffe auf Funktionen und Accounts verwaltet werden. Wir werden schauen, wie man ihrer habhaft werden kann und wie es möglich ist, diese Prozesse sicher zu gestalten. Dazu werden wir unter anderem auf die Möglichkeiten des Passworthashings eingehen (siehe Kapitel 7, »Sicherheit von Authentifizierungsmechanismen«).

In Kapitel 8 werden wir schauen, inwieweit die Einbindung von Dateien durch den Nutzer problematisch werden kann. Viele Webapplikationen sind heutzutage interaktiv und bieten neben Dateiuploads sogar die Möglichkeit, Dateien aus dem Inter-

net einzubinden — dabei können viele Schwachstellen entstehen, die wir gemeinsam ausräumen wollen (siehe Kapitel 8, »File Inclusion«).

Wir werden sehen, dass einige Fehler in Applikationen auf ganz simplen Missverständnissen beim Geschäftsprozess beruhen. Diese logischen Fehler können mittlerweile so umfassend sein, dass wir ihre Gefahren sowie Möglichkeiten zur Abhilfe in einem eigenen Kapitel behandeln werden (siehe Kapitel 9, »Logische Fehler«).

Informationen können Angreifern sehr weiterhelfen, sowohl bei bereits aus den vorherigen Kapiteln bekannten Angriffsvektoren als auch bei Angriffen mittels Phishing oder Social Hacking. Was genau das ist, und ab welchem Grad Informationen als sicherheitskritisch zu betrachten sind, sehen wir uns in Kapitel 10, »Informationspreisgabe«, an.

Vor einigen Jahren wurde Cross-Site-Scripting in Fachkreisen belächelt — mittlerweile hat sich dort allerdings herumgesprochen, welche erheblichen Gefahren davon ausgehen. Heutzutage wird UI-Redressing, vielen auch unter dem Begriff »Clickjacking« bekannt, schlichtweg unterschätzt. Viele Entwickler denken, dass auf dem Zielsystem keine Veränderungen vorgenommen werden, insoweit bestünde auch keine große Gefahr. Das ist allerdings ein folgenschwerer Irrtum, denn Clickjacking und Co. bedrohen die Nutzer direkt, teilweise sogar so perfekt, dass man als Seitenbetreiber kaum etwas davon mitbekommt. Es gibt jedoch Abhilfe, die wir uns gemeinsam anschauen möchten (siehe Kapitel 11, »UI-Redressing«).

Am Ende des Hauptteils möchten wir zwei Dinge genauer betrachten, zum einen die Kombination von bereits bekannten Angriffen und zum anderen sonstige Angriffsvektoren, die nicht direkt Webapplikationen betreffen, letztlich aber auch die Sicherheit von Webangeboten gefährden können (siehe Kapitel 12, »Weitere Angriffsarten«).

Anschließend möchten wir einen zusammenfassenden Blick zurückwerfen und einige »goldene Regeln« aufstellen. Wenn sich jeder an diese halten würde, wäre schon viel in puncto Websicherheit gewonnen (siehe Kapitel 13, »Die 10 wichtigsten Regeln für Entwickler und Sicherheitsverantwortliche«).

Es folgt eine langes Kapitel zu den Tools. Es gibt eine ganze Reihe von nützlichen Tools, die helfen können, Angriffe durchzuführen oder Lücken aufzuspüren. Wir wollen uns die bekanntesten Werkzeuge anschauen, um das zuvor erworbene Wissen anzuwenden und in unseren eigenen Systemen nach Lücken suchen zu können (siehe Kapitel 14, »Tools«).

Bug-Bounty-Programme können eine organisatorische Maßnahme sein, um Sicherheitslücken in Webapplikationen zu schließen. Wie genau Bug-Bounty-Programme

funktionieren und was dabei zu beachten ist, möchten wir in Kapitel 15 näher beleuchten (siehe Kapitel 15, »Bug-Bounty-Programme«).

An einigen Stellen in diesem Buch werden Sie Methoden kennenlernen, die sich in der realen Welt kaum für das Gute einsetzen lassen. Welche Möglichkeiten es gibt, diese Fertigkeiten dennoch sinnvoll einzusetzen, werden wir in Kapitel 16 behandeln (siehe Kapitel 16, »Legal Webhacking durchführen«).

Das Internet der Dinge breitet sich rasant aus, und Kontrollsysteme von Industrieanlangen sind längst vernetzt und lassen sich auch über das Web steuern. Wir möchten in Kapitel 17 über sogenanntes SCADA-Hacking sprechen und sehen, welche Angriffsmöglichkeiten sich daraus ergeben (siehe Kapitel 17, »SCADA-Hacking«).

Sie werden feststellen, dass die meisten Kapitel nach dem gleichen Schema aufgebaut sind. Zunächst soll die Problematik der Lücke deutlich werden, häufig anhand von Codebeispielen, konkreten Fällen und einfachen Erklärungen, damit im Anschluss eine mögliche Abhilfe betrachtet werden kann.

1.4 Die Grenzen

Es gibt keine absolute Sicherheit — aber immer Maßnahmen zur ihrer Erhöhung.

Dieses Buch beschäftigt sich nicht oder kaum mit Lösungen im Rahmen der IT-Infrastruktur (etwa Firewalls) oder des Sicherheitsdesigns (Risikomanagement), sondern handelt primär vom Schreiben eines sicheren Codes, also dem Programmieren und der sicheren Konfiguration von Webservern. Es soll dabei helfen, ein Verständnis der Anatomien von Sicherheitslücken in Webanwendungen zu erlangen. Sie sollen dafür sensibilisiert werden, Ihren Code aufmerksamer zu implementieren und Konfigurationen bewusster vorzunehmen, um die Wahrscheinlichkeit einer angreifbaren Sicherheitslücke zu verringern.

Einige Problematiken spezieller Programmiersprachen, wie C/C++, werden in diesem Buch nicht behandelt, da sie extrem komplex sind.

Es ist unmöglich, sämtliche Angriffsvektoren und alle denkbaren Codevariationen in einem Buch abzubilden. Die Logik vieler Lücken ist jedoch ähnlich, und die Bandbreite der in diesem Buch beschriebenen Lücken so groß, dass Sie nach der Lektüre in der Lage sein sollten, durch kreatives Testen und den Transfer des Wissens weitere Lücken selbst zu identifizieren.

Der mittlerweile recht umfangreiche Komplex der Websicherheit entwickelt sich dynamisch weiter. Täglich gibt es neue Tricks aus der dunklen Ecke des Internets und Whitepapers mit neuen Erkenntnissen aus diesem Bereich. Insofern empfiehlt

es sich, dass Sie sich auch über das Buch hinaus informieren und sich auf dem aktuellen Stand halten.[1]

Ich werde an geeigneter Stelle Hinweise auf relevante Webprojekte oder weiterführende Literatur geben (siehe Kapitel »Literaturempfehlungen«). Weitere Quellen können Sie dem angehängten Quellenverzeichnis entnehmen.

Eine letzte, in der Fachliteratur manchmal vernachlässigte Grenze möchte ich abschließend noch benennen: die der Akzeptanz seitens des Nutzers.

Als Sicherheitsverantwortlicher oder Person mit Interesse an IT-Sicherheit neigt man zu Maßnahmen, die zwar unter dem Gesichtspunkt der Sicherheit hervorragend, für den Nutzer aber so lästig sind, dass dieser vom Einhalten dieser Regeln absieht oder die entsprechende Anwendung sogar nicht mehr nutzt. Letzteres wäre gerade im Bereich des Webs ein großes Übel, denn Webanwendungen sind dazu da, benutzt zu werden.

Die Empfehlungen in diesem Buch sind fast immer auf größtmögliche Sicherheit angelegt. Deren Befolgung führt nicht zwangsläufig zu einer Optimierung im Sinne der Benutzbarkeit. Bei Veränderungen sollte somit (im Sinne eines ausgewogenen Change-Managements) dafür gesorgt werden, die Akzeptanz der Nutzer zu gewährleisten.

Die sicherste Webapplikation der Welt bringt nichts, wenn sie nahezu »unbenutzbar« wird. Es besteht beispielsweise oft ein Zielkonflikt zwischen Sicherheit und Benutzerfreundlichkeit/Bequemlichkeit (Usability). Es gibt eine Vielzahl weiterer Zielkonflikte, darunter an prominenter Stelle jener zwischen Sicherheit und Rentabilität.

Generell gilt es, diese Zielkonflikte durch Kompromisse aufzulösen und abzuwägen, ob fragliche Ziele in einem noch vertretbaren Maße erreicht und sonstige vitale Interessen nicht erheblich missachtet werden.

1.5 Das (Kern-)Problem

Gleich zu Anfang möchte ich auf ein mögliches Kernproblem zu sprechen kommen, wenn es um die Sicherheit von Webapplikationen geht:

Viele Menschen (auch erfahrene Entwickler) denken, dass Applikationen etwas »falsch machen« oder die Sicherheit aufgrund der Applikation selbst nicht gegeben

[1] Grundsätzlich möchte ich bereits an dieser Stelle auf das »Open Web Application Security Project« (OWASP) aufmerksam machen, bei dem ich selbst Mitglied bin. Es ist eine Non-Profit-Organisation, die sich gezielt mit dem Thema Sicherheit von Anwendungen im WWW beschäftigt. Mehr dazu unter *https://owasp.org/*.

ist. Das ist jedoch weit gefehlt, denn Applikationen (und Computer generell) tun immer nur das, wozu sie angewiesen wurden.

Sie folgen streng ihrem Ablauf und funktionieren, rein logisch betrachtet, wie sie es sollen. Nur wenn ein Fehler in der Verarbeitung einer Anweisung vorliegt, kommt es auch zu einem Fehler in der Applikation. Und Anweisungen kommen fast ausschließlich von dem Benutzer einer Anwendung.

Wir werden in diesem Buch lernen, dass die meisten Fehler erst dann zu Sicherheitsrisiken werden, wenn Applikationen von den Entwicklern (während des Programmiervorgangs) nicht klar angewiesen wurden, wie sie zu funktionieren haben.

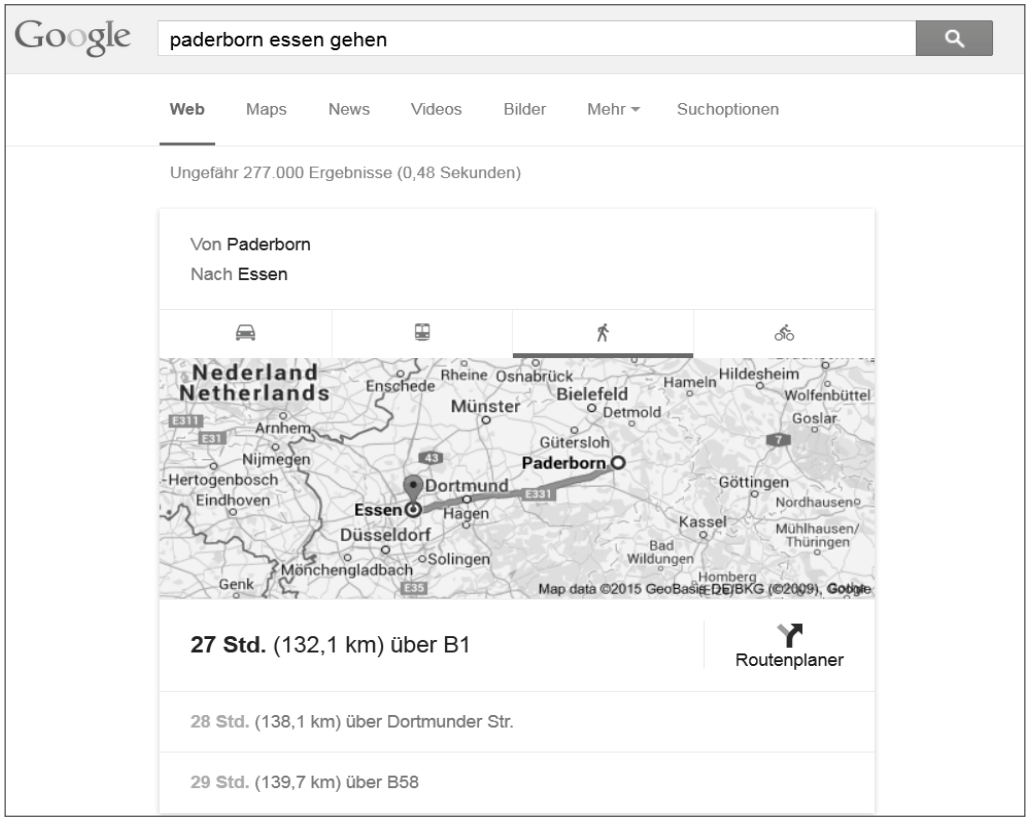


Bild 1.1: Ein typischer »Fehler«, entstanden durch die doppelte Bedeutung des Worts »Essen«.

Die hier abgebildete Google-Suche (von 2015) zeigt mustergültig, wie schnell es zwischen dem Nutzer und einer Anwendung zu Missverständnissen kommen kann. Es

bleibt in der Suchformulierung unklar, ob wir »in Paderborn essen gehen« möchten oder von »Paderborn bis nach Essen gehen« möchten. Was in diesem Beispiel harmlos und ein Fehler ohne große Auswirkung ist, kann bei sensiblen Eingaben in komplexen Applikationen schnell zu einem Sicherheitsrisiko werden.

Angreifer machen sich oftmals genau eine solche Systematik zunutze. Die Applikation kann dafür nichts, vielmehr sind es Entwickler und Programmierer, die nicht alle möglichen Fälle bedacht oder aufgefangen haben. Welche Folgen das genau haben kann und wie es uns gelingt, bestmöglichsten Schutz vor solchen »Missverständnissen« zu erhalten, wollen wir gemeinsam in diesem Buch genauer untersuchen.

Mittlerweile (2018) hat Google seiner Suchmaschine die Doppelbedeutung des Wortes »Essen« beigebracht. Einen Hintergrundartikel zu dem Thema finden Sie auf dem Blog zum Buch: *https://hacking-im-web.de/blog-semantik-websicherheit.html*

2

Evolution des Webs

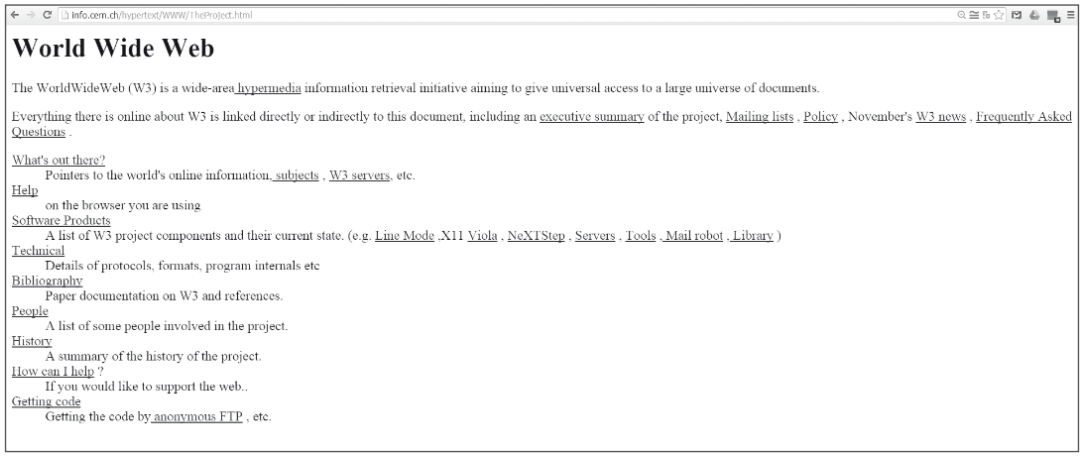
Bevor wir uns auf die Technik von Webapplikationen und deren Lücken konzentrieren, soll diese Einführung über den historischen Hintergrund erklären, wieso bis heute viele Webapplikationen unsicher sind und weshalb die (Un-)Sicherheit dieser Systeme von besonderer Bedeutung ist. Um das zu erreichen, werden die wichtigsten Entwicklungsstufen des Webs kurz umrissen.

2.1 Die Anfänge des Webs

1989 beschäftigte sich Tim Berners-Lee intensiv damit, Informationen und Ergebnisse von Forschungsprojekten zu »vernetzen«. Er entwickelte die Grundzüge des Webs, wie wir es heute kennen. In seinem Konzept baute er auf das bereits seit mehreren Jahren existierende Internet auf und entwickelte die Idee von abrufbaren Webseiten [4].

Wer sich mit Webseiten beschäftigt, weiß, dass es heute grundsätzlich zwei Typen von Webseiten gibt: zum einen die statischen und zum anderen die dynamischen Webseiten (meist Webapplikationen).

Zu Beginn des Webs gab es ausschließlich statische Internetseiten, das sind jene, bei denen nur eine geringe Interaktion möglich ist. Einige von ihnen waren sogar reine Textdateien (`.txt`). Nach und nach wurden die meisten Webseiten allerdings mit einer abstrakten Auszeichnungssprache (HTML) strukturiert und enthalten oft verschiedene Elemente, die auf Abruf angezeigt werden können. Das Suchen, Eingeben und Verarbeiten von Daten ist auf diesen Webseiten allerdings nicht möglich, sie sind statisch.


info.cern.ch/hypertext/WWW/TheProject.html

World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
Pointers to the world's online information, subjects , W3 servers, etc.
Help
on the browser you are using
Software Products
A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)
Technical
Details of protocols, formats, program internals etc
Bibliography
Paper documentation on W3 and references.
People
A list of some people involved in the project.
History
A summary of the history of the project.
How can I help ?
If you would like to support the web..
Getting code
Getting the code by anonymous FTP , etc.

Bild 2.1: Die Webseite des CERN-Instituts [5], eine der ersten Webseiten überhaupt — sie war statisch und verwies auf weitere Informationen über das WWW-Projekt.

Der Aufruf solcher Webseiten[2] funktioniert bis heute nach folgendem Prinzip: Ein Benutzer navigiert zu einer bestimmten Stelle im Web, lädt ein Dokument herunter, der Inhalt wird interpretiert und dann dem Benutzer angezeigt. Die Übertragung von Inhalten erfolgte also zunächst überwiegend von Server zu Client. Schauen Sie einmal selbst auf der ersten Webseite überhaupt vorbei:

http://info.cern.ch/hypertext/WWW/TheProject.html

Mit der Zeit versuchte man, Möglichkeiten zu entwickeln, die Benutzerinteraktion zu steigern: Der Benutzer sollte dazu in der Lage sein, durch Eingabe von Daten die Website zu verändern oder Aktionen durchzuführen. Erstrebenswert erschien unter anderem das Verwenden eines Formulars oder die Suche nach gewissen Inhalten

[2] Das Internet selbst ist eine Plattform für Dienste, etwa das Web. Im Web gibt es Seiten, deshalb der Begriff »Webseiten«. »Internetseiten« ist streng genommen ein Wort der Umgangssprache und wird deshalb in diesem Fachbuch vermieden.

innerhalb einer Website. Was sich heute banal anhört, da wir es nahezu täglich nutzen, erschien damals bahnbrechend.

Bereits 1991 wurde ein solches Vorgehen von einer Datenbankanwendung an der Stanford-Universität genutzt. Der erste Gebrauch dieser Applikation wurde auf deren Website umfassend dokumentiert [6]. Sie gilt bis heute als wohl erste Webapplikation, da sie in der Lage war, eine Eingabe des Nutzers serverseitig zu verarbeiten, Dokumente zu bearbeiten und diese wieder an den Nutzer auszugeben.

Im Laufe der Jahre (etwa um 1993) war es erstmals möglich, auch völlig programmiersprachenunabhängig eine Eingabe des Endnutzers serverseitig zu verarbeiten. Dies wurde zur Erstellung von dynamischen Inhalten genutzt.

Wenig später folgten weitere Entwicklungen. So wurde PHP (Hypertext PreProcessor) 1995 als Programmiersprache äußerst beliebt und zusammen mit dem neu geschaffenen Betriebssystem »Apache HTTP Server« zu einer Art Standard im Web. Bis heute ist der »Apache Server« laut Netcraft Webserver-Survery der meistgenutzte Webserver im Internet [7] und PHP eine der beliebtesten Programmiersprachen im Web[3].

Mit diesen Technologien und Möglichkeiten nahm die Evolution des Webs Fahrt auf, und der Grundstein für moderne Webanwendungen war gelegt. Es schien nur noch eine Frage der Zeit, bis neben Forschungseinrichtungen auch Unternehmen die Technologien nutzen wollten, um Vorteile für ihre Geschäftsprozesse zu erzielen.

2.2 Die Boomphase des Webs

Und genau so kam es: 1994 entwickelten zwei Studenten der Stanford-Universität die Idee, eine Art Übersicht über das Web zu schaffen, schließlich sah man sich einem gewaltigen Zuwachs an Webseiten und Informationen ausgesetzt. Wie sollte man den Überblick behalten und die Sachen finden, die man suchte?

David Filo und Jerry Yang schufen mit Yahoo! eine der ersten bekannteren Webanwendungen. Sie wurde zur zentralen Anlaufstelle für viele neue und neugierige Nutzer, die das Web entdeckten. Ebenfalls 1994 gründeten die beiden Studenten ihr eigenes Unternehmen mit dem Namen »Jerry and David's guide to the World Wide Web«, später wurde es in »Yahoo!« umbenannt.

Das war offenbar der Moment in der Geschichte des Webs, an dem erstmals klar war, dass es nicht länger eine Randerscheinung von einigen Akademikern oder

[3] Das ist der Hauptgrund, weshalb unser Setup für die Testumgebung ebenfalls auf diesen Technologien basiert (dazu später mehr).

Wissenschaftlern bleiben würde, sondern dass immer mehr Innovationen in dem Bereich stattfinden würden. Viele große Unternehmen unterschätzten das Web jedoch zunächst und begannen erst recht spät, sich dafür zu interessieren. Dann wurden massenhaft Start-ups im Bereich der Informationsverarbeitung und Informationstechnologie in der Hoffnung gegründet, eine entscheidende Innovation im Bereich Web zu erzielen. Damit begann die Zeit der sogenannten »New Economy«, der Boomphase des Webs.

Diese Ära endet mit einigen Erfolgsgeschichten, deren Ergebnis man heute noch sehen kann. Zu dieser Zeit entstanden Unternehmen wie Google, eBay, Amazon und Web.de, die heute erfolgreich am Markt etabliert sind (weitere siehe Tabelle unten). Neben den rein kommerziellen Erfolgen wurde auch technisch einiges Neues geschaffen. Viele der damaligen Technologien werden noch heute, allerdings in leicht abgewandelter Form, genutzt.

Die wichtigsten Meilensteine sind in der nachfolgenden Tabelle dargestellt.

Jahr	Spezifikation	Server-technologie	Client-technologie	Dienst
1994	HTTP-Cookie		Netscape Navigator 0.9	Yahoo IMDb
1995	HTML 2.0 JavaScript Java 1.0	Apache HTTP Cold Fusion PHP	Internet Explorer 1 JavaScript	Amazon eBay Web.de
1996	HTTP/1.0 Frames CSS	Active Server Pages (ASP)	Macromedia Flash	HotMail Archive.org
1997	HTML 4.0 JavaScript Standardisierung		Netscape Composer Internet Explorer 4	GMX NETFLIX
1998	DOM CSS 2	Zope		Google PayPal

Diese Phase war für das Web von besonderer Bedeutung, denn durch die enormen Investitionen und die Weiterentwicklung der Lösungen wurde das Web alltäglicher, und viele Standards wurden schnell weiterentwickelt. Immer mehr Menschen in den Industrienationen hatten einen Internetanschluss, zeitgleich wuchs die Anzahl der

erreichbaren Applikationen im Web. Mitte der 1990er-Jahre bis zur Jahrtausendwende expandierte das Web sehr stark.

2.3 Webapplikationen heute

Die Expansion setzte sich auch nach der Jahrtausendwende fort, und das Angebot im Web ist inzwischen unfassbar groß. Durch die neuen Technologien und einige findige Unternehmen erhielt auch der weniger technikversierte Nutzer die Möglichkeit, Inhalte im Web zu erstellen.

Webapplikationen sind heute nicht mehr wegzudenken: Ob wir nur kurz online einen Blick auf das Wetter werfen, uns mit unseren Freunden über ein soziales Netzwerk schreiben, etwas zum Geburtstag unserer Freunde im Onlineshop kaufen, einem Onlinemeeting beitreten oder etwas in der Vertriebsdatenbank ändern — immer nutzen wir auch Webapplikationen.

Durch das mobile Internet und Flatrates wurde die Nutzung des Webs im Consumerbereich weiter intensiviert. Webapplikationen sind weit verbreitete Anwendungen im Businessbereich (sowohl Business-to-Business als auch Business-to-Consumer) und werden zunehmend auch von Behörden (E-Government) genutzt.

Nachfolgend sind typische Anwendungen und entsprechende Dienste des heutigen Webs aufgelistet:

- Suche nach Informationen (Nachrichten, Lexika, Wetter etc.)
- Websuche (Suchmaschinen: google.com / yahoo.com / bing.com)
- Soziales Vernetzen (facebook.com / twitter.com / myspace.com)
- E-Mailing (yahoo.com / web.de / gmail.com)
- Einkaufen (E-Commerce: amazon.com / zalando.de)
- Auktionen und Marktplätze (ebay.com)
- Onlinebanking (paypal.com)
- Handel mit Wertpapieren/Kryptowährungen (Bitcoin)
- Onlinewetten/Gewinnspiele/Onlinespiele (Browserspiele)
- Blogs (wordpress.com, blogger.com)
- Medien (spiegel.de / zeit.de / youtube.com)

Durch das Aufkommen von Smartphones wurde ein zusätzlicher Markt geschaffen. Viele Apps auf Smartphones sind mit Schnittstellen zum Web ausgestattet.[4]

Es gibt für sehr viele Systeme mittlerweile Schnittstellen im Web, so sind etwa ERP-Systeme über Webapplikationen erreichbar. Auch die Konfiguration von Hardware wird über Webanwendungen durchgeführt, und selbst im Bereich der Industriemaschinensteuerung (ICS) oder der Auswertung und Kontrolle (SCADA) gibt es die Tendenz, Steuerungssysteme über Webapplikationen bedienbar zu machen. Der Vorteil von Webapplikationen ist schlichtweg, dass diese auf nahezu jedem Endgerät lauffähig sind und kaum Ressourcen verbrauchen.

Alles in allem ist in der ersten Dekade des 21. Jahrhunderts der Siegeszug des Webs und der von Webapplikationen unübersehbar, es ist der wohl wichtigste, zumindest aber der meistgenutzte, Dienst im Internet geworden.

Das Web ist eine Technologie, die das Leben mehrerer Milliarden von Menschen täglich stark beeinflusst und von ihnen vielfältig genutzt wird. Zudem bauen Unternehmen ihr komplettes Geschäftsmodell darauf auf, und nahezu jedes Unternehmen nutzt Webapplikationen zur Unterstützung des laufenden Betriebs oder der Repräsentation im Web — aus diesem Grund ist die Sicherheit solcher Systeme sehr wichtig.

2.4 Webapplikationen in der Zukunft

Da es weiterhin einen Zuwachs an Vernetzung zwischen verschiedenen Systemen geben wird, ist zu vermuten, dass das Web ebenfalls in diese Bereiche integriert wird. Mittlerweile wurde mit dem »Internet der Dinge« eine ähnliche Goldgräberstimmung heraufbeschworen, wie sie einst zu den Boomzeiten des Webs herrschte — viele Hersteller wollen ihre Geräte »smart« machen, dazu nutzen sie meist auch Webapplikationen oder Apps.

Es ist sehr wahrscheinlich, dass die Auswertung und vor allem die Darstellung von großen Datenmengen (»Big Data«) durch Webapplikationen erfolgen wird, da diese einfach zu bedienen und nahezu von jedem Endgerät aus zugänglich sind.

Im Bereich des E-Government ist damit zu rechnen, dass langfristig eine vermehrte Authentifizierung über das Internet stattfinden wird, Webapplikationen würden eine benutzerfreundliche und plattformunabhängige Möglichkeit darstellen, diese Idee umzusetzen.

---

[4] Die Sicherheit von mobilen Apps ist ein sehr wichtiges, jedoch eigenes Thema, das in diesem Buch nicht vertiefend behandelt wird. Viele der später aufgezeigten Lücken sind aber in ähnlicher Form auch im Bereich der App-Sicherheit möglich.

Die Relevanz des Webs wird also angesichts dieser kurz genannten Beispiele in keinem Fall abnehmen, es ist eher stark davon auszugehen, dass es mit anderen Technologien zusammen seine Relevanz weiter ausbauen wird.

2.5 Entwicklung der Websicherheit

Websicherheit war fast immer ein eher beiläufiges, ja offenbar für viele unnötiges Thema. Zu Beginn des Webs spielte es gar keine Rolle, denn bei statischen Webseiten gibt es kaum Möglichkeiten, auf Anwendungsebene Angriffe durchzuführen. Es war schlicht nicht möglich, als Client komplexe Anweisungen an den Webserver oder an Subsysteme zu schicken, wenn es sich um eine statische Seite (etwa HTML-Dokumente oder TXT-Dateien) handelte.

Ein weiterer Faktor — und wohl ein bedeutender Grund dafür, dass man sich nicht besonders stark um die Sicherheit des Webs kümmerte — war, dass man sich vertraute; die Nutzer der ersten Stunde waren allesamt Forscher und kannten sich sehr gut. Mehr als die »Vernetzung« von Wissen stellte das Web zunächst nicht dar; es war eine Art wissenschaftlicher Versuch, bei dem damals nicht abzusehen war, was dabei herauskommen würde.

Nach dem Aufkommen der ersten größeren Webapplikationen und der Kommerzialisierung Mitte der 1990er-Jahre wurde dann allerdings nach und nach deutlich, dass Sicherheit doch ein extrem wichtiges Thema sein muss — lange Zeit hatte man sich eher darauf konzentriert, auf Netzwerk- bzw. Infrastrukturebene Sicherheit zu schaffen. Viele Unternehmen setzen bis heute teure Enterprise-Firewalls oder komplexe Netzwerk-Firewalls ein, führen die neuesten Softwarepatches durch, verschlüsseln ihre Kommunikation und führen immer wieder Penetration-Tests durch. Sie denken, dass sie sicher vor Angriffen seien, schließlich haben sie eine moderne Firewall, viele Schutzmechanismen und im besten Fall weitreichende Sicherheitsstandards. Grundsätzlich bietet diese Art von Maßnahmen auch einen gewissen Schutz, etwa vor sogenannten »Distributed Denial of Service«-Attacken (DDOS, massenhaften Anfragen, die eine Dienstüberforderung verursachen), unberechtigten Zugriffen und Man-in-the-Middle-Angriffen (MITM-Angriffen), trotzdem können diese Schutzwälle und Maßnahmen allein niemals die Sicherheit eines Systems mit Webapplikationen garantieren, da sie auf der Transportebene, nicht aber auf der Applikationsebene ansetzen.

Meist fungieren zum Beispiel die Firewalls nur als eine Art »Türsteher«, durch den gewisse Zugriffe auf Netzwerkdienste zugelassen oder eben unterbunden werden. Diese konventionellen Schutzmaßnahmen stoßen aber bei der Sicherheit von Webanwendungen an ihre Grenzen, da sie oft ausschließlich auf Netzwerkprotokollebene

analysieren, aber nicht beobachten, was nach dem Durchkommen tatsächlich passiert und welche möglicherweise schädlichen Zeichen ein Parameter beinhaltet.

Da es immer wieder zu Zwischenfällen und Hacks größeren Ausmaßes kam, fand Anfang der 2000er ein Umdenken statt, das bis heute anhält. Man erkannte, dass gerade im Bereich des Webs eine erhebliche Gefahr bestand, denn:

Die Hürde für Angreifer ist oft gering[5] — der Schaden kann jedoch enorm sein.[6]

Es stellt sich also folgende Frage: Wie schafft man es, Webanwendungen sicher zu machen?

Die Antwort fällt uns nicht leicht, denn das Web war in der Entwicklung schlicht schneller als seine Macher. Dass sich aus dieser Idee einmal so viel entwickeln würde, war nicht absehbar, deshalb gab es kaum verbindliche Sicherheitsstandards, sondern lediglich RFC (Requests For Comment) oder BCP (Best Current Practice). Nach und nach etablierten sich jedoch auch verbindlichere Internetstandards (STD).

2001 wurde vor allem durch die Gründung des »Open Web Application Security Project« (OWASP) erstmals vermehrt über das Thema Sicherheit von Webapplikationen diskutiert. Bis heute stellt diese Organisation eine entscheidende Säule im Bereich der Websicherheit dar. So werden beispielsweise die sogenannten »OWASP Top 10« veröffentlicht, bei denen es sich um die Darstellung der zehn größten Bedrohungen bzw. Sicherheitslücken von Webapplikationen handelt. [8]

Erstaunlich ist, dass es bis heute zu vielen Hacks von Webapplikationen über bereits bekannte Angriffsvektoren kommt, obwohl das Thema und entsprechende Problematiken mittlerweile seit weit über zehn Jahren diskutiert werden. Daran wird jedoch noch einmal deutlich, dass es zum einen kein Allheilmittel gibt und zum anderen offenbar noch keine hinreichende Sensibilisierung aller Entwickler in dem Bereich stattgefunden hat. Oftmals scheint der Fokus beim Lehren einer Programmiersprache nicht in erster Linie auf Sicherheit, sondern auf Funktionsfähigkeit zu liegen.

Eine weitere Entwicklung ist, dass Applikationen, die man noch vor fünf Jahren für sicher hielt, heute möglicherweise unsicher geworden sind, weil neue Angriffsvektoren entdeckt wurden. Es bedarf also wachsamer und informierter Programmierer, um eine fortdauernde Sicherheit zu gewährleisten, da das technische Umfeld sehr dynamisch ist.

---

[5] Oftmals ist nicht einmal das Umgehen einer komplexen Firewall oder die Kenntnis der im Hintergrund arbeitenden IT-Infrastruktur vonnöten, ein simpler Webbrowser mit einigen Erweiterungen genügt, um Schaden anzurichten, wie wir später sehen werden.

[6] Überlegen Sie bitte einmal selbst, was der Diebstahl Ihrer Kundendaten oder die Verunstaltung der Webseite für Ihr Unternehmen bedeuten würde.

Zwischenzeitlich glaubte man, das Problem von Lücken in Webapplikationen weitgehend in den Griff zu bekommen, indem man sogenannte Web-Application-Firewalls (WAFs) entwickelte. Diese Systeme sollten ähnlich wie Firewalls arbeiten und zum Beispiel auffällige Anfragen blockieren, indem die Inhalte genauer betrachtet und nicht bloß Netzwerkpakete dahingehend kontrolliert wurden, ob der Dienst oder das Protokoll zulässig ist. Tatsächlich führte das aber nicht wirklich zum Erfolg, denn zum einen ist die Implementierung eines solchen Systems mit sehr viel Aufwand verbunden, weil es praktisch vor den Webserver geschaltet werden muss und dadurch die Komplexität des Systems erhöht. Und zum anderen gab es immer wieder Zwischenfälle, bei denen die Filterregeln dieser Firewalls einfach umgangen werden konnten. Diese WAFs dürften also nur als »Auffangnetz« infrage kommen und halten möglicherweise unerfahrene Angreifer auf. Es ist allerdings mehr Verlass auf wirklich sicheren Code.

Die neuesten Tendenzen im Bereich der Websicherheit bilden sogenannte Bug-Bounty-Programme, bei denen Sicherheitsexperten entlohnt werden, wenn sie Unternehmen eine Lücke in ihrer Applikation verantwortlich melden. Damit wollen Unternehmen dem florierenden Schwarzmarkt im Bereich der Lücken Einhalt gebieten und ihre Dienste sicherer machen, um einen Hack und einen damit verbundenen Imageschaden zu vermeiden.

Unternehmen, die ein solches System im Einsatz haben, etwa Google [9] und Facebook [10], blieben bis jetzt sogar weitgehend unversehrt. Vielleicht ist diese Art der Websicherheit also auch ein Modell für die Zukunft.

Insgesamt ist das Web besonders aufgrund der Durchdringung vieler Lebensbereiche ein bedeutender Teil des Alltags von Privatpersonen und Unternehmen geworden. Diesen vor potenziellen Angreifern zu schützen, erweist sich als zwingend notwendig. Der Sicherheit von Webapplikationen sollte also eine sehr hohe Relevanz zukommen, insbesondere in Unternehmen, in denen die eigentliche Wertschöpfung durch Produkte im Web erreicht oder entscheidend unterstützt wird.

Epilog

Hier endet das Buch »Hacking im Web« — was wohl nie enden wird, ist das Hacking im Web selbst. Solange es das Web gibt, gibt es (vermutlich) immer wieder Angreifer und Angriffe — aber auch Verteidigungsmöglichkeiten.

Bereits heute kann man sehen, welche Folgen es haben kann (Masshacking), wenn Schwachstellen oder Zero-Day-Exploits populärer Produkte oder Webapplikationen im Web auftauchen. Leider ist mit einer weiteren Professionalisierung im Bereich Cybercrime zu rechnen, also sollten Sie gewappnet sein, sich die Hinweise in diesem Buch zu Herzen nehmen und Ihre Webapplikationen entsprechend sichern oder anderen dabei helfen, das zu tun.

Bislang geht es bei diesem Kampf meist »nur« um jede Menge Daten, Geld und Pixel — das ist bereits ärgerlich genug. Allerdings scheint in naher Zukunft ein noch viel bedrohlicheres Szenario möglich: Niemand kann genau sagen, was nach dem Einzug von Smart Home, Industrie 4.0 und dem Internet der Dinge für Gefahren entstehen könn(t)en.

Die Folgen wären in jedem Fall verheerender, denn dann geht es nicht mehr um nur Einsen und Nullen in Computersystemen oder Pixel auf Bildschirmen, sondern um Auswirkungen in der realen Welt: die Sicherheit von Maschinen, Anlagen und allgemein kritischer Infrastruktur — also auch um unsere persönliche Sicherheit.

Bleibt zu hoffen, dass es immer kluge Menschen geben wird, die im richtigen Moment auf der richtigen Seite stehen.

Gleiches erbitte ich mir auch von Ihnen, den Lesern dieses Buchs: Seien Sie wachsam, bleiben Sie kritisch und denken Sie mit. Sicherheit generell kann nie als statischer Faktor betrachtet werden, sondern ist ein Weg!

Wir haben einen kleinen Teil des Wegs (im Bereich Web) gemeinsam beschritten. Ich hoffe sehr, dass Ihnen unsere gemeinsame Reise und der Besuch so mancher Sehenswürdigkeit (Schwachstelle) gefallen hat und Sie einiges Neues erfahren haben.

Ihre Kommentare

Das Buch hat immer noch den Ruf eines Einweg-Kommunikationsmediums, schließlich gibt es hier — anders als im Web — nicht die Möglichkeit, einen Kommentar zu schreiben oder Nachfragen zu stellen.

E-Mail-Schlüssel für die Kommunikation

Zur Verschlüsselung von E-Mails ist ein öffentlicher Schlüssel unter folgender URL zu finden (oder auf bekannten Schlüsselservern):
http://hacking-im-web.de/0x3F8F7EB1.asc

Ich habe eine Webpräsenz erstellt, auf der ich immer wieder Artikel zu aktuellen Themen der Websicherheit veröffentlichen werde und auf der sich weitergehende Informationen zum Thema Webhacking, Websicherheit und Onlinekriminalität (auch nach Buchveröffentlichung) finden werden.

Besuchen Sie mich doch einfach auf *hacking-im-web.de*.

Ich würde mich sehr freuen, wenn Sie mir mitteilten, wie Sie dieses Buch fanden. Über Anfragen, Rückmeldungen, fachliche Diskurse und Kritik jeder Art freue ich mich und nehme diese gern über folgende E-Mail-Adresse entgegen: *buch@tim-philipp-schaefers.de*

Sie werden von mir hören — mit Sicherheit.

Tim Philipp Schäfers, Detmold im Juli 2018

Abkürzungsverzeichnis

AJAX = Asynchronous JavaScript and XML

ANSI = American National Standards Institute

API = Application Programming Interface

ASCII = American Standard Code for Information Interchange

ASP.NET = Active Server Pages .NET

BCP = Best Current Practice

CAPTCHA = Completely Automated Public Turing test to tell Computers and Humans Apart

CGI = Common Gateway Interface

CMS = Content-Management-System

CORS = Cross-Origin Sharing Standard

CSRF = Cross Site Request Forgery

CSS = Cascading Style Sheets

CSP = Content-Security-Policy

CSV = Comma-separated Values

CVE = Common Vulnerabilities and Exposures

DB = Datenbank

DBMS = Datenbank-Management-System

DDOS = Distributed Denial of Service

DIT = Directory Information Tree

DOM = Document Object Model

DOS = Denial of Service

DTD = Document Type Definition

DVWA = Damn Vulnerable Web Application

ECMA-Script = European Computer Manufactures Association Script

ERP = Enterprise Resource Planning

GUI = Graphical User Interface

HSTS = HTTP Strict Transport Security

HTML = Hypertext Markup Language

HTTP(S) = Hypertext Transfer Protocol (Secure)

HMI = Human Machine Interface

IDOR = Insecure Direct Object References

IDS = Intrusion Detection System

IETF = Internet Engineering Task Force

IP = Internet Protocol

JSF = JavaServer Faces

JSON = JavaScript Object Notation

JSTL = JavaServer Pages Standard Tag Library

LDAP = Lightweight Directory Access Protocol

LFI = Local File Inclusion

LSO = Local Shared Object

MD5 = Message-Digest Algorithm 5

MITM = Man-in-the-Middle

OCR = Optical Character Recognition

OWASP = Open Web Application Security Project

PHP = Hypertext Preprocessor

PoC = Proof of Concept

PLC = Programmable Logic Controller

RCE = Remote Code Execution

RFC = Request for Comment

RFI = Remote File Inclusion

RTU = Remote Terminal Units

SGML = Standard Generalized Markup Language

SHA1 = Secure Hash Algorithm

SID = Session Identifier

SIEM = Security Information and Event Management

SMTP = Simple Mail Transfer Protocol

SOP = Same-Origin-Policy

SQL = Structured Query Language

SSL/TLS = Secure Socket Layer

SSO = Single Sign-On

STD = Standard

STS = Strict Transport Security

SVG = Scalable Vector Graphics

TCP = Transmission Control Protocol

TLD = Top Level Domain

TLS = Transport Layer Security

UID = User Identifier

URI = Uniform Resource Identifier

URL = Uniform Resource Locator

UTF-x = Unicode Transformation Format

VBScript = Visual Basic Script

VBA = Visual Basic for Applications

VPN = Virtual Private Network

W3C = World Wide Web Consortium

WAF = Web Application Firewall

WWW = World Wide Web

XFO = X-Frame-Options

XHR = XMLHttpRequest-Technologie

XHTML = Extensible Hypertext Markup Language

XML = Extensible Markup Language

XPath = XML Path Language

XSS = Cross-Site-Scripting

XXE = XML External Entity

Index

S

Y

Z

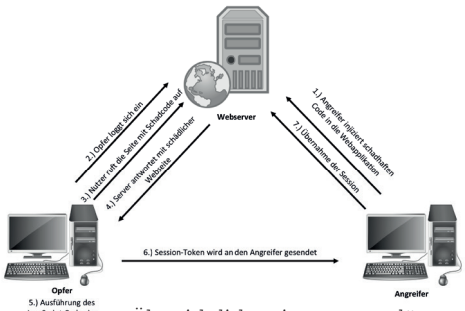
Tim Philipp Schäfers

Hacking im Web

Der Erfolg des E-Commerce hat auch seine Schattenseiten: Hackerangriffe im Web gehören inzwischen zum Alltag. Es geht dabei nicht nur um unsichere Firewalls oder Fehler in Betriebssystemen, häufig stellt die selbst programmierte Webapplikation das größte Einfallstor dar. Um sich vor Hackern zu schützen, ist es wichtig, wie ein Hacker zu denken. In diesem Buch lernen Sie die häufigsten Angriffsmethoden kennen und erhalten Tipps, wie Sie sich dagegen schützen können. Analysieren Sie Ihren Programmcode auf Schwachstellen und schließen Sie die Lücken gleich in der Implementierungsphase.

AUS DEM INHALT

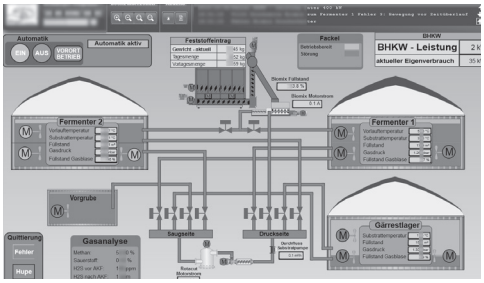
- Aufbau einer Testumgebung
- Session-Angriffe
- Cross-Site-Scripting
- **Injections:** Code, Cookie, http-Header, LDAP, SMTP-Header, SQL und XPath
- Grundlagen von LDAP & XPath
- Sicherheit von Authentifizierungsmechanismen
- Kryptografische Hashfunktionen
- Passwortcracking
- File Inclusion
- Zugriffsrechte
- Cookie-Manipulation
- Informationspreisgabe
- UI-Redressing
- Google-Bombing und -Dorking
- Exploits
- Hackertools



Übersichtliche Diagramme erläutern das Vorgehen bei einem Angriff.

Die wichtigsten Angriffsvektoren

Durch die Kombination verschiedenster Technologien wie Browser, HTML, JavaScript, PHP, Java und SQL in Webanwendungen sind die potenziellen Schwachstellen quasi unzählbar. Ob SQL-Injection, Cross-Site-Scripting oder Session-Hijacking: Lernen Sie die Funktionsweise dieser Angriffe kennen, stellen Sie Ihr Können beim Angreifen der Testumgebung unter Beweis und schützen Sie sich mit den aufgeführten Tipps erfolgreich vor Angriffen.



Anfällige Webanwendungen gefährden auch Industrie-4.0-Betriebe wie Biogasanlagen.

Werkzeuge kennen und nutzen

Entwickler sind keine Sicherheitsexperten und können nicht jede Schwachstelle der eingesetzten Programmiersprache und Bibliotheken kennen. Umso wichtiger ist es, die entstandene Webanwendung auf ihre Schwachpunkte zu testen. Schäfers stellt in einem ausführlichen Anhang zahlreiche Werkzeuge vor, mit denen Sie effektiv nach Schwachstellen suchen können.

ISBN 978-3-645-20636-5