Kapitel 1
Einleitung

Es vergeht kein Tag, an dem nicht über Sicherheitslücken, Hacking-Angriffe und Cybercrime berichtet wird. Das Thema ist allgegenwärtig und greift immer stärker in unser Leben ein. Wir sind abhängig von den uns umgebenden IT-Systemen. Die Wirtschaft könnte ohne funktionierende Kommunikations- und Datenverarbeitungssysteme und die zahlreichen Dienste im Internet nicht mehr arbeiten. Die Schattenseite des rasanten Wachstums von Internet und Co sind die zahlreichen Probleme und Schwachstellen, die Kriminelle ausnutzen können, um Schaden anzurichten und mit den gestohlenen Daten Profit zu machen.

Es ist mittlerweile ein ganzer Industriezweig entstanden, der sich mit der Suche und Ausnutzung von sicherheitskritischen Schwachstellen in diversen Systemen beschäftigt und damit auch gutes Geld verdient. Täglich entstehen mehr als einhunderttausend neue Varianten von Computerviren, mit denen Systembetreiber, Endbenutzer und Hersteller von Antivirensoftware zu kämpfen haben.

Aber warum gibt es so viele Schwachstellen? Können Systeme nicht einfach sicherer gebaut werden? Die einfache Antwort ist, wir haben ein großes Bewusstseins- und Qualitätsproblem in der Softwareentwicklung. Die Komplexität von großen Softwareprodukten mit mehreren Millionen Zeilen Quellcode verhindert umfangreiche Tests, sodass Anwendungen und Betriebssysteme oft in schlechter Qualität ausgeliefert werden. Das Bewusstsein für sicheres Softwaredesign und Begriffe wie *Secure Coding* sind in vielen Entwicklungsabteilungen leider noch nicht angekommen.

1.1 Über dieses Buch

In diesem Buch möchten wir das Thema sichere Softwareentwicklung von zwei Seiten beleuchten. Einerseits gibt es in der Entwicklung von Programmen Möglichkeiten und Wege, sichere Systeme zu entwerfen. Andererseits gibt es die Seite der Angreifer, die ständig versuchen, jede noch so kleine Schwachstelle in einem Programm sofort auszunutzen und diese mit Exploits anzugreifen, die letztlich wiederum eine Art von Software sind.

Gute Softwareentwickler sollten beide Seiten kennen, um sichere Systeme zu bauen. Allein das Wissen um die Kreativität, die Tools und Möglichkeiten von Angreifern kann bereits ein erster Schritt in die richtige Richtung sein. Ebenso sind die in moder-

nen Betriebssystemen eingebauten Schutzmaßnahmen gegen Exploits oft nur bedingt wirksam. Wirksamer Schutz ist nur mit hoher Qualität in der Softwareentwicklung möglich.

Der erste Teil des Buchs setzt sich mit der Entwicklung sicherer Software auseinander, der zweite Teil behandelt die verschiedenen Möglichkeiten, um Softwarefehler aktiv durch die Entwicklung von Exploits auszunutzen.

In **Kapitel 2** zeigen wir Ihnen einen vollständigen Angriff auf eine IT-Infrastruktur. Dabei werden Sie die einzelnen Schritte des Angriffs und die Probleme und Schwachstellen, die einen Angriff erst ermöglichen, kennen lernen.

In **Kapitel 3** stellen wir Ihnen das Konzept der sicheren Softwareentwicklung vor. Anhand eines Prozessmodells zeigen wir jene Praktiken, die Ihnen helfen sollen, Softwaresysteme möglichst ohne Sicherheitslücken zu erstellen. Ein Überblick über die häufigsten Angriffe gegen Webapplikationen und mögliche Verteidigungsmechanismen rundet das Ganze ab.

In **Kapitel 4** beschreiben wir den praktischen Ablauf und die notwendigen Werkzeuge für die sichere Softwareentwicklung. Dazu gehören Methoden der *agilen Softwareentwicklung* genauso wie ein Überblick über die in diesem Buch verwendeten Programmiersprachen *C* und *Java*. Zusätzlich werden Techniken wie *Quellcode-Versionierung*, *Debugging*, automatisiertes Testen und *Continuous Integration* angesprochen.

In **Kapitel 5** beschäftigen wir uns mit *Reverse Engineering*. Wir werden also eine bestehende Applikation analysieren, um potenzielle Schwachstellen zu finden oder Sicherheitsmechanismen zu umgehen. Je nach Programmiersprache setzen wir hierfür unterschiedliche Techniken und Tools ein. Dabei werden wir aus der Sicht des Softwareentwicklers erklären, wie man solche Analysen verhindern oder zumindest erschweren kann.

In **Kapitel 6** geben wir Ihnen konkrete Anleitungen zur sicheren Codierung und gehen näher auf die *Kapselung* von Funktionalitäten, den sicheren Umgang mit Daten (*Input Validation*, *Representation* und *Output Encoding*) und das richtige Verhalten in Fehlersituationen ein. Einen Schwerpunkt bilden die *kryptografischen APIs* und deren richtiger Einsatz. Schließlich stellen wir Ihnen noch Techniken und Tools zur statischen Codeanalyse vor.

In **Kapitel 7** erklären wir Ihnen das Konzept der architekturbasierten Risikoanalyse und erörtern die grundlegendsten Designprinzipien für die sichere Softwareentwicklung. Anschließend stellen wir Ihnen die wichtigsten Designkonzepte für sichere Webapplikationen vor. Ausgehend vom *HTTP-Protokoll* werden wir Themen wie clientseitige Controls von Webapplikationen, Zugriffskontrolle auf Ressourcen (*Authentifizierung*, *Session Management* und *Autorisierung*), sichere Datenspeicherung und die Verhinderung von Browserangriffen (*Cross Site Scripting*, *Cross Site Request Forgery*) besprechen.

In **Kapitel 8** erläutern wir wichtige Konzepte der *Kryptografie*. Wir erklären *Verschlüsselungsfunktionen, Hash-Funktionen, Message Authentication Codes* und *digitale Signaturen*. Abschließend stellen wir kurz vor, welche Algorithmen und Schlüssellängen derzeit vom *National Institute of Standards and Technology (NIST)* empfohlen werden und in welchen Dokumenten Sie sich rasch einen Überblick über Standards und Empfehlungen verschaffen können.

In **Kapitel** 9 suchen Sie selbst nach Sicherheitslücken in einer Windows-Anwendung. Dabei lernen Sie Tools und Methoden zum Auffinden potenzieller Schwachstellen kennen. Sie schreiben Ihren eigenen *Fuzzer* und arbeiten mit Tools wie *Debugger* und *Disassembler*.

In **Kapitel 10** des Buchs nutzen Sie die gefundenen Schwachstellen aus und entwickeln Ihren ersten *Windows-Exploit-Code*. Der Exploit ist grundsätzlich lauffähig, wird aber durch die Standard-Schutzmechanismen eines modernen Betriebssystems, wie zum Beispiel die *Address Space Layout Randomization (ASLR)* verhindert.

In **Kapitel 11** lernen Sie dann die derzeit vorhandenen Schutzmechanismen auf Betriebssystemebene kennen. Sie verstehen die Funktionsweise von Mechanismen wie *ASLR, SafeSEH, SEHOP, DEP* und können diese aktivieren oder auch ausschalten.

In **Kapitel 12** wechseln Sie wieder die Seiten und versuchen, die in Kapitel 11 implementierten Schutzmaßnahmen wieder zu umgehen. Dort werden Sie auch den in Kapitel 10 entwickelten *Exploit* um Elemente ergänzen, die die Robustheit erhöhen, sodass dieser auf verschiedenen Betriebssystemversionen lauffähig wird.

In **Kapitel 13** behandeln wir einen konkreten Exploit auf einem Linux-System. Für die vorgestellte Anwendung lässt sich ein *Format String Exploit* entwickeln. Der Exploit umgeht alle aktivierten Schutzmaßnahmen wie ASLR und DEP unter Linux und ist damit universell einsetzbar.

Schlussendlich stellen wir Ihnen in **Kapitel 14** einige *Real World Exploits* der letzten Jahre vor. Darunter befinden sich klingende Namen wie *Heartbleed, Spectre, Meltdown, OpenFuck, Stagefright, Eternal Blue* und *WannaCry*.

1.2 Zielgruppe

Dieses Buch richtet sich an Softwareentwickler, Qualitäts- und Sicherheitsverantwortliche, die bereits über ein Grundwissen in der Entwicklung von Software verfügen und die ihr Wissen erweitern wollen. Idealerweise bringen Sie Kenntnisse in den Programmiersprachen C bzw. C++ und Java mit.

Nicht im Fokus dieses Buchs sind reine Anwender von IT-Systemen. Natürlich spielt auch der User eines Systems eine wichtige Rolle in puncto Sicherheit. Sicherheitsgeschultes Personal kann viele der täglichen Angriffe, man denke nur an die unzähligen Phishing-Mails, durch geeignetes Verhalten verhindern.

1.3 Wie Sie mit dem Buch arbeiten

Wir empfehlen Ihnen, abhängig von Ihren fachlichen Vorkenntnissen, verschiedene Pfade durch das Buch. Wenn Sie Softwareentwickler sind, können Sie die Kapitel in der vorgegebenen Reihenfolge einfach durcharbeiten. Sollten Sie bereits erste Erfahrungen mit Software-Exploits gemacht haben, so empfehlen wir, nach Kapitel 2 direkt in den zweiten Teil des Buchs zu springen: Arbeiten Sie die Kapitel 9 bis 14 durch, und kehren Sie dann wieder zu Kapitel 3 zurück, um die entwicklungsrelevanten Kapitel zu erkunden.

Der Fokus des Buchs liegt auf den nachvollziehbaren Praxisbeispielen, für die Sie sich genügend Zeit nehmen sollten, da einige Beispiele und Teilschritte technisch recht anspruchsvoll sind.

Wir hoffen, dass wir mit diesem Buch einen Beitrag dazu leisten können, die zukünftige Zahl der sicheren Softwaresysteme deutlich zu erhöhen.

1.4 Die Autoren

Klaus Gebeshuber (*https://fh-joanneum.at/hochschule/person/klaus-gebeshuber*) ist Professor für *IT-Security* an der FH JOANNEUM in Kapfenberg, Österreich. Seine Schwerpunkte liegen im Bereich *Netzwerksicherheit, Industrial Security, Security-Analysen* und *Ethical Hacking*. Er hält zahlreiche Industriezertifizierungen im Umfeld von IT-Security, Netzwerksicherheit und Penetration-Testing. In diesem Buch deckt er die Themen Exploit-Entwicklung, Einsatz von Schutzmaßnahmen und deren Umgehung in den Kapiteln 2 und 9 bis 14 ab.

Egon Teiniker (*https://www.fh-joanneum.at/hochschule/person/egon-teiniker*) ist Professor für *Software Engineering* an der FH JOANNEUM in Kapfenberg, Österreich. Seine Schwerpunkte liegen im Bereich *Software Design* und *Software Security*. Weiter ist er seit vielen Jahren als Gastdozent an der Hochschule Bremen tätig, wo er Methoden zur Entwicklung komplexer Softwaresysteme und sichere Softwareentwicklung lehrt. Sein Beitrag in diesem Buch erstreckt sich von Kapitel 3 bis Kapitel 7. Dabei werden die Themenbereiche sichere Softwareentwicklung (Implementierung und Design) sowie *Reverse Engineering* abgedeckt.

Wilhelm Zugaj (*https://www.fh-joanneum.at/hochschule/person/wilhelm-zugaj/*) ist Professor für *Kryptografie* und *Datenbanktechnologien* an der FH JOANNEUM in Kapfenberg, Österreich. Seine Schwerpunkte liegen im Bereich *Verschlüsselungsverfahren, Big Data Security* und *Big Data Analytics*. Hier ist er in Forschungsförderprojekten leitend und forschend tätig. In diesem Buch deckt er das Thema Kryptografie in Kapitel 8 ab.