



Zertifikatsprogramm - Z213

Netzwerkhacking

- Bedrohungen der IT-Sicherheit
- Grundlagen von Hacking-Techniken
- IT-Verteidigungsmaßnahmen
- Penetration Testing
- Fallbeispiele zur Behandlung von netzwerkzentrierten Angriffen

Prof. Dr. Martin Rieger

Patrick Eisoldt, M.Eng.

David Schlichtenberger, M.Sc.

Benjamin Welte, M.Eng.

Michael Kotzjan, B.Sc.

Patrick Geiger

Prof. Dr. Martin Rieger
rieger@hs-albsig.de
+49 (0) 07571 / 732 - 9124

Benjamin Welte, M.Eng.
welte@hs-albsig.de
+49 (0) 07571 / 732 - 9554

Netzwerk – Missbrauch – Hacking

Studienbrief 1: Bedrohungen der IT-Sicherheit

Studienbrief 2: Grundlagen von Hacking-Techniken

Studienbrief 3: IT-Verteidigungsmaßnahmen

Studienbrief 4: Penetration Testing

Studienbrief 5: Fallbeispiele zur Behandlung von netzwerk-
zentrierten Angriffen

Autoren:

Prof. Dr. Martin Rieger

Patrick Eisoldt, M.Eng.

David Schlichtenberger, M.Sc.

Benjamin Welte, M.Eng.

Christian Schneider, M.Eng.

Michael Kotzjan, B.Sc.

Patrick Geiger

2. Auflage

Hochschule Albstadt-Sigmaringen

© 2021 Hochschule Albstadt-Sigmaringen
Institut für wissenschaftliche Weiterbildung
Open C³S | Zertifikatsprogramm
Gartenstraße 15
72458 Albstadt

2. Auflage (2021-09-03)

Das Werk einschließlich seiner Teile ist urheberrechtlich geschützt. Jede Verwendung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung der Verfasser unzulässig und strafbar. Das gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Um die Lesbarkeit zu vereinfachen, wird auf die zusätzliche Formulierung der weiblichen Form bei Personenbezeichnungen verzichtet. Wir weisen deshalb darauf hin, dass die Verwendung der männlichen Form explizit als geschlechtsunabhängig verstanden werden soll.

Inhaltsverzeichnis

Einleitung zu den Studienbriefen	6
I. Abkürzungen der Randsymbole und Farbkodierungen	6
II. Zu den Autoren	7
III. Modulehrziele	9
Studienbrief 1 Bedrohungen der IT-Sicherheit	11
1.1 Lernziele	11
1.2 Schwachstellen, Bedrohungen und Angriffe	11
1.3 Grundlegende Begriffe zur IT-Sicherheit	16
1.4 Ablauf von Angriffen	21
1.4.1 Kill Chain	21
1.4.2 Zusammenfassung der Modelle Post-Exploitation-Phase	23
1.4.3 Portscan auf das Ziel	25
1.4.4 Zugriff erlangen	40
1.4.5 Erstellung eines Trojaners/Hintertür	66
1.4.6 Schadsoftware in Android Anwendungen	72
1.4.7 Command-and-Control	80
1.4.8 Schadcode nachladen	82
1.4.9 Antiforensische Maßnahmen	83
1.5 Beispiele für Phasen eines Sicherheitsvorfalls	87
1.6 Beispiel Spear-Phishing-Angriffs	100
1.6.1 Zielsetzung	100
1.6.2 Empfängerkreis definieren: Reconnaissance	100
1.6.3 Informationen über Empfänger sammeln: Reconnaissance / Weaponization	101
1.6.4 Technische Anforderungen des Angriffs definieren und vorbereiten: Weaponization	102
1.7 Beispiel eines Angriffs durch Emotet	104
1.7.1 Reconnaissance und Weaponization	105
1.7.2 Delivery: Malspam	105
1.7.3 Lateral Movement	106
1.7.4 Exploitation	106
1.7.5 Installation	107
1.7.6 Command-and-Control	109
1.7.7 Weitere Wirkungen	110
1.8 Zusammenfassung	111
Studienbrief 2 Grundlagen von Hacking-Techniken	113
2.1 Schadsoftware	113
2.1.1 Computerviren	114
2.1.2 Würmer	117
2.1.3 Trojanisches Pferd	120
2.1.4 Ransomware	121
2.2 Bedrohungen im Rechnernetz	125
2.2.1 Angriffe auf Schicht 2	127
2.2.2 Angriff auf Schicht 3: IP Spoofing	130
2.2.3 Angriff auf Schicht 4: SYN Flooding	132
2.3 Bedrohungen aus dem Internet	134
2.3.1 Angriff auf einen Browser	135
2.3.2 E-Mail-Client Angriffe	138
2.3.3 Spam-Mails und Phishing-Mails	139
2.3.4 Botnetze	142
2.3.5 DoS und DDoS	146
2.4 Social Engineering	150
2.4.1 Grundlagen Social Engineering	150

2.4.2	Angriffsmethoden Social Engineering	154
2.4.3	Beispiele für Social Engineering Angriffe	157
2.5	Hardware-Tools	158
2.5.1	Raspberry Pi	158
2.5.2	WiFi Pineapple	159
2.5.3	Rubber Ducky	159
2.5.4	Teensy	160
2.5.5	LAN Turtle	169
2.5.6	Proxmark	170
2.5.7	SDR (Software Defined Radio)	170
2.5.8	Keylogger	171
2.6	Speicherung einer Router-Konfiguration ohne Zugangsdaten	171
2.6.1	Ansatz	172
2.6.2	Test am eigenen Router	173
2.6.3	Anpassen der gespeicherten Datei	174
2.6.4	Anwendung von RouterPassView	175
2.6.5	Was macht das Programm RouterPassView?	177
2.6.6	Nebenbemerkung	177
Studienbrief 3 IT-Verteidigungsmaßnahmen		179
3.1	Prinzipien und grundlegende Mechanismen	179
3.2	Der BSI-Sicherheitsprozess	187
3.2.1	Analyseverfahren	187
3.2.2	Grundschutz	194
3.3	Bausteine zur IT-Sicherheit	195
3.3.1	Sichere Software-Entwicklung und deren Betrieb	195
3.3.2	Schutzmaßnahmen eines PC-Clients	201
3.3.3	Nutzung einer Sandbox	209
3.3.4	Kryptografische Verfahren	218
3.3.5	Zertifikate	233
3.3.6	Konzepte zur Netzwerksicherheit	244
3.4	Reaktion auf Sicherheitsvorfälle (Incident Response)	277
3.4.1	Strukturelle Organisation	278
3.4.2	Incident Response Prozess	280
3.4.3	Vorbereitung	280
3.4.4	Erkennung	284
3.4.5	Analyse	290
3.4.6	Eindämmung, Beseitigung & Wiederherstellung	296
3.4.7	Nachbereitende Phase	298
3.4.8	Checkliste für die Vorfallsbehandlung	301
3.4.9	Empfehlungen für die Vorfallsbehandlung	302
Studienbrief 4 Penetration Testing		305
4.1	Methodik für die Durchführung von Penetrationstests	305
4.2	Beispiele	312
4.2.1	Informationsbeschaffung mit Nmap	313
4.2.2	Passwort-Cracking	314
4.2.3	ARP-Spoofing	317
4.2.4	ARP-Spoofing mit DNS-Spoofing und Phishing	321
4.3	Vulnerability Assessment	326
4.3.1	Abgrenzung zum Pentesting	326
4.3.2	OpenVAS	327
Studienbrief 5 Fallbeispiele zur Behandlung von netzwerkzentrierten Angriffen		333
5.1	Herangehensweise in Beispielen	333
5.1.1	Vorfallsbehandlung eines Ransomware-Angriffs	333
5.1.2	Vorfallsbehandlung eines SSH-Wurms	340

5.2	Implementierung einer Ransomware	352
5.2.1	Installationshinweise	352
5.2.2	Begriffsdefinition	352
5.2.3	Implementierung	352
5.2.4	Ergebnisse	362
5.2.5	Ausführung	363
5.2.6	Erweiterung	365
5.3	Beispiel Keylogger	366
5.3.1	Ausführung	369
5.3.2	Unsichtbare Ausführung	370
5.3.3	Eigenständige Ausführung	370
5.4	Weitere Informationen zu Prozessen ermitteln	371
5.4.1	Welcher Prozess nutzt eine Datei?	371
5.4.2	Mit welchen Parametern wurde ein Prozess geöffnet?	371
5.4.3	Welche Ressourcen nutzt ein Prozess?	372
5.4.4	Analyse mit Cuckoo	374
Liste der Lösungen zu den Kontrollaufgaben		387
Verzeichnisse		403
I.	Abbildungen	403
II.	Beispiele	407
III.	Definitionen	408
IV.	Exkurse	409
V.	Kontrollaufgaben	409
VI.	Tabellen	409
VII.	Literatur	410
Stichwörter		415

Einleitung zu den Studienbriefen**I. Abkürzungen der Randsymbole und Farbkodierungen**

Beispiel	B
Definition	D
Exkurs	E
Kontrollaufgabe	K
Merksatz	M
Quelltext	Q

II. Zu den Autoren



Prof. Dr. Martin Rieger studierte Elektro- und Informationstechnik an der Technischen Universität München und schloss an derselben Hochschule die Promotion mit Auszeichnung ab. Ein Schwerpunkt seiner Forschungsarbeit lag in der Erstellung von Methoden und Werkzeugen zur Modellierung sowie Analyse und Optimierung elektrischer Schaltungen. Er war fünf Jahre Leiter des Labors für schnelle Analog-ICs in der IC-Entwicklungsabteilung des Forschungs- und Entwicklungszentrums der Firma Thomson Multimedia in Villingen. In der Zeit bei Thomson Multimedia war er Erfinder bzw. Miterfinder an 15 deutschen, 12 europäischen und 8 weltweiten Patenten.

Seit 1993 ist er als Professor an der Fakultät Engineering der Hochschule Albstadt-Sigmaringen auf den Gebieten Informatik und Informationstechnik tätig. Im Labor für Eingebettete Systeme und IT-Sicherheit betreibt er anwendungsnahe Forschung auf den Gebieten Embedded Systems und IT-Sicherheit. Er hatte über viele Jahre an der Hochschule Albstadt-Sigmaringen Positionen als Studiendekan, Prodekan, Prorektor und Rechenzentrumsleiter inne.

Prof. Dr. Rieger ist Initiator und Gründungs-Studiendekan des Master-Studiengangs Digitale Forensik, der in Kooperation mit der Friedrich-Alexander Universität Erlangen-Nürnberg und der Ludwig-Maximilians-Universität München betrieben wird.

Er leitet das vom BMBF geförderte Zertifikatsprogramm Open-C³S, das 35 Hochschul-Zertifikatsmodule auf dem Gebiet Cybersicherheit anbietet und das kooperativ von der Hochschule Albstadt-Sigmaringen, der Friedrich-Alexander-Universität Erlangen-Nürnberg, der Freien Universität Berlin, und der Ludwig-Maximilians-Universität München, getragen wird.



Patrick Eisoldt, M.Eng. hat an der Hochschule Albstadt-Sigmaringen und der Glyndŵr University in Wales studiert. 2012 schloss er erfolgreich sein Masterstudium Systems Engineering ab. Im Rahmen seiner Master-Thesis konzipierte und realisierte er einen prototypischen Editor zur Projektierung von Prozessleitsystemen der Firma Siemens nach dem Ursache-Wirkung-Prinzip. Von November 2010 bis August 2011 unterstützte er das Institut für Wissenschaftliche Weiterbildung bei der Erstellung von Studienbriefen für den Studiengang Digitale Forensik.

Seit 2012 ist er für das Open Competence Center for Cyber Security als Modulentwickler tätig.



David Schlichtenberger studierte Medien- und Kommunikationsinformatik an der Hochschule Reutlingen. Nach seinem Masterstudium arbeitete er einige Jahre als Webentwickler und Kundenberater für Internetservices. Seit November 2014 ist er als akademischer Mitarbeiter an der Hochschule Albstadt-Sigmaringen am Institut für Wissenschaftliche Weiterbildung beschäftigt.



Benjamin Welte absolvierte den Bachelor-Studiengang Kommunikations- und Softwaretechnik an der Hochschule Albstadt-Sigmaringen. Während des Bachelors arbeitete er als Werkstudent im Bereich der Softwareentwicklung.

Seit Februar 2016 arbeitet er als Modulentwickler und Tutor im Zertifikatsprogramm des Open Competence Center for Cyber Security. Innerhalb dieser Zeit absolvierte er ebenfalls den Master-Studiengang Systems Engineering an der Hochschule Albstadt-Sigmaringen.



Christian Schneider hat an der Hochschule Albstadt-Sigmaringen Systems Engineering studiert. Während seines Bachelorstudiums arbeitete er für einen Antivirenhersteller und als Webentwickler. Weiter betreute er, im Rahmen von Vorlesungen, Studenten als Tutor.

Von 2016 bis 2017 unterstützte er als wissenschaftlicher Mitarbeiter das Institut für wissenschaftliche Weiterbildung bei der Erstellung von Studienbriefen.



Michael Kotzjan studierte Allgemeine Informatik an der Hochschule Furtwangen. Während seines Bachelorstudiums arbeitete er als Softwaretester im Bereich der Medizintechnik und betreute als Tutor Studenten im Rahmen von Vorlesungen.

Seit 2019 absolviert er den Master-Studiengang Systems Engineering und unterstützt als wissenschaftlicher Mitarbeiter das Institut für wissenschaftliche Weiterbildung bei der Erstellung von Studienbriefen.



Patrick Geiger studiert IT-Security an der Hochschule Albstadt-Sigmaringen. Während des Studiums unterstützte er als studentische Hilfskraft das Zertifikatsprogramm in den Modulen der digitalen Forensik. Seit März 2019 arbeitet er am Institut für wissenschaftliche Weiterbildung an Studienbriefen und Modulentwicklungen mit.

III. Modullehrziele

Dieses Modul soll zeigen, wie einfach es in vielen Fällen ist, einen Angriff durchzuführen. Wir wollen vermitteln, worauf ein erfolgreicher Angriff basiert:

- Ansatz des Angriffs
- Mechanismen von Schadsoftware
- Stufenweises Vorgehen
- Schadsoftware tarnt sich
- Schadsoftware verankert sich im System
- Schadsoftware ist „fernsteuerbar“

Es soll auch gezeigt werden, wie vorgegangen werden kann, wenn ein Angriff bereits eingetreten ist oder eventuell sogar noch aktiv ist.

- Incident Response, „Feuerwehr“
- Forensische Ermittlungen

Die bekannten Prinzipien zum Aufbau sicherer digitaler Infrastruktur werden erlernt:

- Bausteine sicherer digitaler Netze: Zutrittskontrolle, Zugriffskontrolle, Verschlüsselung, Firewalls
- Analyse und Design der erforderlichen Sicherheitsmaßnahmen
- Schutzmaßnahmen für Clients
- Konzepte zur Netzwerksicherheit

Wir lernen den Nutzen einer ständigen Überwachung der Netzwerksicherheit kennen:

- Network Security Monitoring
- Vorgänge erfassen
- Vorgänge detektieren
- Vorgänge analysieren

Es ist sinnvoll, Systeme selbst zu testen (Penetrationstests) und somit bereits möglichst früh einen Großteil der Schadsoftware auszusperren.

- Schließen der Sicherheitslücken
- Schulung der Mitarbeiter

Studienbrief 1 Bedrohungen der IT-Sicherheit

1.1 Lernziele

Zunächst sollen wichtige Begriffe im Umfeld der IT-Sicherheit und der IT-Angriffe vorgestellt werden, damit sich der Leser in diesen Begrifflichkeiten ausdrücken kann. Dann wird in Beispielen gezeigt, wie einfach es in vielen Fällen ist, einen Angriff durchzuführen. Wir wollen vermitteln, worauf ein systematisch geplanter und durchgeführter Angriff basiert.

- Ansatz des Angriffs
- Mechanismen von Schadsoftware
- Stufenweises Vorgehen
 - Schadsoftware tarnt sich
 - Schadsoftware verankert sich im System
 - Schadsoftware ist „fernsteuerbar“

1.2 Schwachstellen, Bedrohungen und Angriffe

Definition 1.1: Schwachstelle; Verwundbarkeit

Unter einer Schwachstelle (engl. *weakness*) verstehen wir eine Schwäche eines Systems oder einen Punkt, an dem das System verwundbar werden kann. Eine Verwundbarkeit (engl. *vulnerability*) ist eine Schwachstelle, über die die Sicherheitsdienste des Systems umgangen, getäuscht oder unautorisiert modifiziert werden können.

D

Beispielsweise zählt die Diebstahlgefahr zu den physischen Schwachstellen mobiler Geräte. Weitere Schwachstellen können durch die unsachgemäße Nutzung des Systems, durch Softwarefehler oder auch durch unsichere Kommunikationsverbindungen entstehen. Typische softwarebedingte Schwachstellen, die häufig zu einer Verwundbarkeit werden, sind Programme, in denen aufgrund einer fehlenden Eingabeüberprüfung Pufferbereichsüberläufe nicht korrekt abgefangen werden.

Definition 1.2: Bedrohung

Eine Bedrohung (engl. *threat*) des Systems zielt darauf ab, eine oder mehrere Schwachstellen oder Verwundbarkeiten auszunutzen, um einen Verlust der Datenintegrität, der Informationsvertraulichkeit oder der Verfügbarkeit zu erreichen oder um die Authentizität von Subjekten zu gefährden.

D

Unter dem Risiko (engl. *risk*) einer Bedrohung verstehen wir die Wahrscheinlichkeit (oder relative Häufigkeit) des Eintritts eines Schadensereignisses, verknüpft mit der dadurch hervorgerufenen Höhe des potentiellen Schadens. Hier verstehen wir Risiko als Produkt aus Schaden und Wahrscheinlichkeit. Also:

Risiko

Definition 1.3: Risiko

$$\text{Risiko} = \text{Schaden} * \text{Wahrscheinlichkeit}$$

D

Studienbrief 2 Grundlagen von Hacking-Techniken

2.1 Schadsoftware

Spezielle Klassen von Bedrohungen eines IT-Systems gehen von Viren, Würmern und Trojanischen Pferden aus, die wir als Schadsoftware (engl. *malware* oder *malicious code*) bezeichnen. Es handelt sich hierbei um Programme, die festgelegte, dem Benutzer jedoch verborgene, Funktionen ausführen. Die Festlegung der zusätzlichen und beabsichtigten (i. d. R. böswilligen) Funktionalität unterscheidet diese Klasse von Programmen von den so genannten Wanzen (engl. *bugs*), deren Fehlverhalten meist von nicht beabsichtigten Programmierfehlern verursacht wird.

Im Folgenden finden Sie eine Auflistung möglicher Architektur-Schwächen von Software:

- Die gemeinschaftliche Nutzung von Programmen und Daten (engl. *sharing*). Eine solche gemeinsame Nutzung wird u. a. über eine zum Teil sehr freizügige Netzwerkfreigabe oder über öffentliche Verzeichnisse ermöglicht. Architektur-Schwächen
- Eine weitere wesentliche Eigenschaft ist die *universelle Interpretierbarkeit* von Daten. Sie besagt, dass kein Unterschied zwischen Daten und Programmen besteht. Programme können demnach wie Daten leicht modifiziert werden, indem man sie beispielsweise um einen Schadenstil erweitert.
- Als dritte wesentliche Eigenschaft ist die Transitivität des Informationsflusses zu nennen. Falls eine Information vom Rechner A zum Rechner B fließt und es Möglichkeiten gibt, Informationen von B zu einem Rechner C fließen zu lassen, so kann die Information auch von A zu C weitergeleitet werden. Die Transitivität wird insbesondere von Würmern ausgenutzt, um sich auf diese Weise zu verbreiten.

Zur Abwehr der Bedrohungen kann man drei Ansätze verfolgen, die in der Praxis in verschiedenen Kombinationen eingesetzt werden:

Techniken zur Bedrohungsabwehr

- Techniken zur Verhinderung (engl. *prevention*) von Angriffen wie beispielsweise die Verwendung stark typisierter Programmiersprachen, Code-Verifikation oder die Beschränkung von Zugriffen aufgrund von systembestimmten Regeln.
- Die zweite Klasse von Ansätzen beschäftigt sich mit Techniken zur Erkennung und frühzeitigen Abwehr möglicher Bedrohungen. Als Beispiele seien statische Analysen des Programmcodes genannt, um z. B. Code-Stücke, die eine Buffer-Overflow-Schwachstelle aufweisen (u. a. Kopieren von Zeichenfolgen ohne die Längenangaben zu prüfen), zu identifizieren und zu korrigieren. Aber auch die bekannten Viren-Scanner und Intrusion Detection Systeme gehören in diese Klasse von Techniken, die anhand charakteristischer Merkmale versuchen, potentielle Schadsoftware zu erkennen.
- Zu der dritten Klasse von Ansätzen zählen Techniken, die darauf abzielen, die Auswirkungen von eingetretenen Bedrohungen zu begrenzen (engl. *mitigation*). Dazu zählen Isolierungsansätze wie das Sandboxing, die darauf abzielen, Anwendungsbereiche gegeneinander abzuschotten, so dass ein Schaden, der bei der Ausführung eines Anwendungsprogramms auftritt, sich nicht auf andere Anwendungsbereiche ausbreiten und womöglich das ganze System beeinträchtigen kann.

Studienbrief 3 IT-Verteidigungsmaßnahmen

3.1 Prinzipien und grundlegende Mechanismen

Definition 3.1: Security Engineering

„Beim Security Engineering geht es um die Entwicklung sicherer Systeme, die auch im Fehlerfall, durch Missgeschicke oder Böswilligkeit, zuverlässig bleiben. Security Engineering adressiert Werkzeuge, Prozesse und Methoden für den Entwurf, die Implementationen, den Test von IT-Systemen und zur Adaption des Systems, falls in der Systemumgebung Änderungen auftreten.“ (frei übersetzt nach [Anderson, 2008] S.3)

D

Security Engineering ist ein Teilbereich der Ingenieurwissenschaften, welcher darauf abzielt, bereits ab der Entwurfsphase, anhand von bewährten Prinzipien und Mechanismen, sichere und robuste IT-Systeme zu entwickeln. Bei der Entwicklung von komplexer Software sind zahlreiche Akteure involviert, die jeweils eine unterschiedliche Sichtweise auf das IT-System haben. Über Security Engineering wird versucht, diese Akteure in einen sicheren Entwicklungsprozess mit einzubeziehen, um in IT-sicherheitsrelevanten Aspekten die richtigen Entscheidungen zu treffen. Insbesondere durch die stetig wachsende Geschwindigkeit in der neue IT-Systeme entwickelt werden (vgl. auch Agile Softwareentwicklung [Baca and Carlsson, 2011]) und durch Erstellung von Gesamtsystemen aus modularisierten Teilen, ist das Risiko, sicherheitsrelevante Aspekte zu vernachlässigen, vorhanden.

Die wichtigsten Prinzipien des Security Engineering sind das

- Erlaubnisprinzip
- Vollständigkeitsprinzip
- Need-to-know-Prinzip
- Prinzip der Benutzerakzeptanz

Das Erlaubnisprinzip (engl. *fail-safe defaults*) fordert, dass grundsätzlich jeder Zugriff verboten ist (default deny) und nur durch eine explizite Erlaubnis ein Zugriffsrecht gewährt werden kann.

Das Vollständigkeitsprinzip (engl. *complete mediation*) besagt, dass jeder Zugriff auf seine Zulässigkeit zu prüfen ist. Ein System, in dem Zugriffsrechte zur Nutzung von Dateien vergeben werden und eine Rechteüberprüfung nur beim Öffnen einer Datei durchgeführt wird, erfüllt beispielsweise nicht das Vollständigkeitsprinzip. In solchen Systemen ist es möglich, dass ein Benutzer eine Datei über lange Zeiträume geöffnet hält und damit das Zugriffsrecht darauf nutzt, obwohl der Besitzer der Datei zwischenzeitlich dem Benutzer dieses Recht bereits wieder entzogen haben kann.

Das Prinzip der minimalen Rechte (engl. *need-to-know*) fordert, dass jedes Subjekt nur genau die Zugriffsrechte erhalten darf, die es zur Erfüllung seiner Aufgaben benötigt. Systeme, in denen ein Superuser unbeschränkte Rechte besitzt, sind ein offensichtliches Beispiel für die Nicht-Einhaltung des Need-to-know Prinzips.

Das Prinzip der Benutzerakzeptanz (engl. *economy of mechanism*) fordert, dass die eingesetzten Sicherheitsmechanismen einfach zu nutzen sein müssen und dass sie automatisch und routinemäßig angewendet werden.

Prinzip der Benutzerakzeptanz

Studienbrief 4 Penetration Testing

4.1 Methodik für die Durchführung von Penetrationstests

In diesem Kapitel wird eine in fünf Phasen gegliederte Methodik zur Durchführung von Penetrationstests vorgestellt. Diese Methodik berücksichtigt die bisher diskutierten Aspekte und wurde im Hinblick auf eine möglichst große Allgemeingültigkeit entwickelt. Zentraler Bestandteil der Methodik ist eine strukturierte Vorgehensweise zur Durchführung von Penetrationstests, auf dessen Basis individuelle Ablaufpläne für konkrete Penetrationstests gebildet werden können.

5 Phasen Methodik

Anforderungen an eine Methodik für die Durchführung von Penetrationstests

Die Methodik beschreibt und strukturiert die Durchführung eines Penetrationstests im Auftrag. Daher muss immer wieder eine Orientierung auf die Ziele des Auftraggebers erfolgen bzw. Sorge getragen werden, dass diese Orientierung nicht außer Acht gelassen wird.

Dies bedeutet beispielsweise, dass die zur Erreichung der Ziele notwendigen Prüfungsschritte beschrieben werden, bzw. dass erläutert wird, ob die Ziele überhaupt sinnvoll mit einem Penetrationstests erreicht werden können. Des Weiteren müssen Maßnahmen zur Einhaltung der rechtlichen Bestimmungen (zur Beachtung von organisatorischen bzw. personellen Voraussetzungen) für die Durchführung von Penetrationstests enthalten sein.

Darüber hinaus sollte eine Methodik zur Durchführung von Penetrationstests die nur begrenzt zur Verfügung stehende Zeit berücksichtigen und muss daher eine Bewertung des potenziellen Risikos bzw. einen Kosten/Nutzen Vergleich enthalten.

Für die Gruppierung der einzelnen Prüfungsschritte empfiehlt sich ein modulatorientierter Ansatz, wie z. B. der des OSSTMMs¹. Dieser erlaubt eine thematische Gruppierung der durchzuführenden Schritte innerhalb eines Penetrationstests. Dies dient zum einen der Übersichtlichkeit, zum anderen kann so durch die Auswahl bzw. das Auslassen von bestimmten Modulen ein angemessener Penetrationstest zusammengestellt werden.

Prüfmodelle

Im Rahmen eines konkreten Penetrationstests können aus wirtschaftlichen Gründen oftmals nicht alle möglichen Prüfungsmodule bearbeitet werden. Dies würde zwar eine vollständige Prüfung sicherstellen, aber auch zu einem hohen zeitlichen Aufwand führen, der u. U. nicht mit den Zielen des Auftraggebers vereinbar bzw. nicht den konkreten Sicherheitsanforderungen angemessen ist. Bei sehr hohen Sicherheitsanforderungen sollte ein möglichst vollständiger Test durchgeführt werden. D. h., dass alle bzw. die meisten Module angewendet werden müssen und dass alle Systeme des Auftraggebers mit in den Test eingeschlossen werden. Bei niedrigen Sicherheitsanforderungen können hingegen bestimmte Module ausgelassen werden und/oder nur exponierte bzw. von „außen“ sichtbare Systeme geprüft werden. Der Umfang des Penetrationstests sollte dabei durch wirtschaftliche Überlegungen bestimmt werden. Es gilt die Kosten bzw. Risiken der Prüfungstätigkeiten mit den möglichen Kosten bzw. Risiken, die durch einen Angriff entstehen könnten, abzuwägen.

¹ OSSTMM = Open Source Security Testing Methodology Manual, z. B. https://scadahacker.com/library/Documents/Assessment_Guidance/OSSTMM-3.0.pdf [Stand: 15.08.2021]

Studienbrief 5 Fallbeispiele zur Behandlung von netzwerkzentrierten Angriffen

5.1 Herangehensweise in Beispielen

5.1.1 Vorfallsbehandlung eines Ransomware-Angriffs

Eine Ransomware ist ein Computer-Virus, der Dateien im Dateisystem oder den gesamten Client verschlüsselt und erst gegen Zahlung eines Lösegelds die Dateien wieder entschlüsselt. GandCrab ist eine solche Ransomware die seit dem Jahr 2018 in verschiedenen Varianten bis heute auftritt. Sie galt als eine der höchsten IT-Sicherheitsbedrohungen im Jahr 2018^{1 2}.

Der Security Incident Prozess nach NIST besteht aus mehreren Phasen, wie in Studienbrief 3 ausführlich beschrieben. Wie in Abb. 5.1 zu sehen ist, sind dies die Phasen Preparation, Detection & Analysis, Containment Eradication & Recovery und Post-Incident Activity. Diese Vorgehensmodell wir nun für die Vorfallsbehandlung eines GrandCrab-Angriffs angelegt.

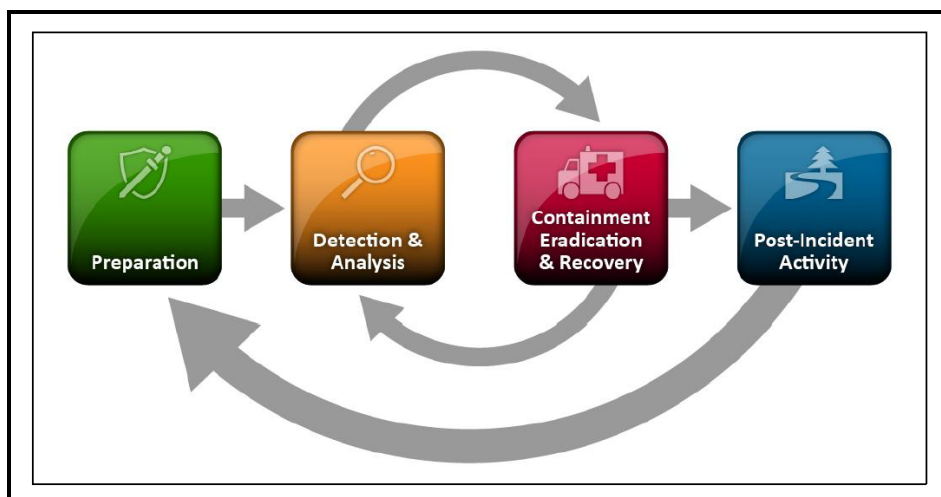


Abb. 5.1: Incident Response Life Cycle

Funktionsweise von GrandCrab

Die Ransomware GandCrab trat erstmals im Januar 2018 auf. Über das Jahr hinweg bis Anfang 2019 wurde GandCrab von den Angreifern immer wieder weiterentwickelt, sodass im Verlauf verschiedene Versionen der Ransomware zum Vorschein traten. Das Konzept von GandCrab hat sich während seiner Entwicklung nicht grundlegend verändert. Auf Grund seiner Funktionalität zielt GandCrab ausschließlich auf Benutzer mit einem Windows Betriebssystem ab.

Zur Ausbreitung verwendet GandCrab verschiedene Wege. Bei den Verbreitungsarten gehört der Versand von Phishing-Mails zu den häufigsten Methoden. Hier wird GandCrab in einer Bewerbung versteckt und als ausführbare Datei übertragen. Es kommt auch vor, dass GandCrab als Makro-Virus in Word-Dokumenten im Anhang von E-Mails versandt wird. Dies kann ebenfalls als Bewerbung aber auch als Rechnung getarnt sein. Das CERT-Bund hatte eine Warnmeldung veröffentlicht, in welcher auf die Gefahren von GandCrab und den Angriffsvektor via E-Mail hingewiesen wurde. Die Angreifer nutzen für den Versand der E-Mails Bots um eine

¹ GandCrab Ransomware (2018): Gandcrab Ransomware Attack <https://www.securonix.com/web/wp-content/uploads/2018/07/Securonix-Threat-Research-GandCrab-Ransomware-Attack.pdf> [Stand: 15.08.2021]

² Hausarbeit Simon Eicker (2019), Digitale Forensik

Verzeichnisse

I. Abbildungen

Abb. 1.1:	Schutzziele CIA	20
Abb. 1.2:	Cyber-Kill-Chain nach Lockheed Martin. Quelle: http://www.lockheedmartin.com/us/what-we-do/aerospace-defense/cyber/cyber-kill-chain.html	21
Abb. 1.3:	Malware wirkt überwiegend in den Phasen Installation, Command and Control sowie Actions on Objectives.	22
Abb. 1.4:	msf5-Suche nach dem Wort <i>wordpress</i>	58
Abb. 1.5:	msf5-Suche nach dem Modul <i>vsftpd</i>	58
Abb. 1.6:	Installation des SSH-Client-Tools PuTTY	66
Abb. 1.7:	Metasploit-Payload-Generator msfvenom	66
Abb. 1.8:	Ansicht von <i>msfconsole</i>	67
Abb. 1.9:	HTTP-Server in Python	68
Abb. 1.10:	Schließen der Verbindung in Currports ¹	68
Abb. 1.11:	Nutzung der Reverse Shell	69
Abb. 1.12:	Anwendung von <i>pyinstaller</i>	70
Abb. 1.13:	Erkennen des Backdoors in CurrPorts	71
Abb. 1.14:	Test des eigenen Backdoors auf Virustotal	71
Abb. 1.15:	APK mit <i>msfvenom</i> manipulieren	73
Abb. 1.16:	APK aus unbekannter Quelle installieren	73
Abb. 1.17:	Meterpreter Handler starten	74
Abb. 1.18:	APK mit <i>apktool</i> entpacken	74
Abb. 1.19:	<i>targetsdkVersion</i> in entpackter APK ändern	74
Abb. 1.20:	entpackte APK wieder packen	75
Abb. 1.21:	Keystore für die APK erstellen	75
Abb. 1.22:	neu erstellte APK signieren	75
Abb. 1.23:	APK mit <i>zipalign</i> abschließen	76
Abb. 1.24:	neu erstellte APK mit <i>msfvenom</i> manipulieren	76
Abb. 1.25:	Rechteanfrage der neuen APK	77
Abb. 1.26:	Meterpreter Handler starten	77
Abb. 1.27:	Ausgabe des Befehls <i>dump_calllog</i>	78
Abb. 1.28:	Stream des Befehls <i>webcam_share</i>	79
Abb. 1.29:	Ausgabe des Befehls <i>geolocate</i>	79
Abb. 1.30:	Darstellung eines Botnetzes	81
Abb. 1.31:	Schadcode unter Windows nachladen	82
Abb. 1.32:	Schadcode unter Linux nachladen	82
Abb. 1.33:	Windows MAC ändern Teil 1	83
Abb. 1.34:	Windows MAC ändern Teil 2	84
Abb. 1.35:	Windows MAC ändern Teil 3	85
Abb. 1.36:	Verwendung von <i>slacker.exe</i>	86
Abb. 1.37:	Verwendung von <i>timestomp</i>	87
Abb. 1.38:	Säuberung der Event Logs	87
Abb. 1.39:	Die einzelnen Phasen eines Sicherheitsvorfalls	87
Abb. 1.40:	IOProtect. Quelle: http://docplayer.org/11366940-Fortgeschrittene-angriffe-und-gegenmassnahmen-in-der-realitaet-sgrp-gv-11-september-2014-adrian-leuenberger.html (besucht 09-2018)	94
Abb. 1.41:	Ablauf eines APT	94
Abb. 1.42:	Inhalt des PDF	95
Abb. 1.43:	Zweck des Sandbox-Ausbruchs	95
Abb. 1.44:	Gefahr eines Pass-The-Hash Angriffs	97
Abb. 1.45:	Mimikatz 2.0	98
Abb. 1.46:	<i>psexec</i>	99
Abb. 1.47:	Verknüpfung von Daten aus Sozialen Netzwerken	102

Abb. 1.48: Generierung der URLs und Download der Emotet-Malware	107
Abb. 1.49: Emotet läuft als Dienst	108
Abb. 1.50: Command-and-Control Kommunikation über HTTP-Cookie	110
Abb. 1.51: Emotet lädt Trickbot nach; Trickbot lädt anschließend weitere Plugins	110
Abb. 1.52: Ryuk Auslieferung per Emotet – TrickBot-Kette	111
Abb. 2.1: Allgemeiner Aufbau eines Virus	114
Abb. 2.2: Vorgang Crypto-Ransomware	123
Abb. 2.3: Angriff auf Sicherheitsdienste	127
Abb. 2.4: Beispiel Netzwerk	129
Abb. 2.5: Beispiel für IP Spoofing	130
Abb. 2.6: IP Spoofing und hosts.equiv	131
Abb. 2.7: TCP 3-Wege-Handshake	132
Abb. 2.8: SYN Flooding	133
Abb. 2.9: Anzahl der Schwachstellen in Netzwerk, Betriebssystem und Anwendungen ([SANS, 2009])	134
Abb. 2.10: Drive-by-Downloads	138
Abb. 2.11: Phishing-E-Mail ([Heise security, 2012])	141
Abb. 2.12: Botnetz	143
Abb. 2.13: P2P-Botnetz	143
Abb. 2.14: Digital Attack Map	147
Abb. 2.15: DNS-Amplification Angriff	148
Abb. 2.16: Social Engineering Lifecycle	153
Abb. 2.17: Vier-Phasen-Konzeption einer Security-Awareness Kampagne [Fox, Dirk; Kaun, Sven, 2005]	153
Abb. 2.18: Angriff auf ein Amazon-Konto: Phishing Mail	155
Abb. 2.19: Angriff auf ein Amazon-Konto: Anmeldemaske	155
Abb. 2.20: XML Ausschnitt von einer RFID Karte ² (Fröhlich, 2013)	157
Abb. 2.21: Phishing Mail von den Hackern ³ (Kelm, 2016)	158
Abb. 2.22: Raspberry Pi	159
Abb. 2.23: WiFi Pineapple	159
Abb. 2.24: Rubber Ducky	160
Abb. 2.25: Teensy	161
Abb. 2.26: Ausführung eines PowerShell Skripts	162
Abb. 2.27: Gegenstück zur Reverse Shell	162
Abb. 2.28: Auswahl der Plattform in der Arduino Software	163
Abb. 2.29: Auswahl des USB Typs in der Arduino Software	164
Abb. 2.30: Ablauf des programms auf dem Teensy	168
Abb. 2.31: LAN Turtle	170
Abb. 2.32: Proxmark	170
Abb. 2.33: Software-Defined Radio	171
Abb. 2.34: Keylogger	171
Abb. 2.35: TP-LINK WR802N	172
Abb. 2.36: Ergebnis von curl	173
Abb. 2.37: Ausgabe mit HTTP-Header 'Referer'	173
Abb. 2.38: Verzeichnisinhalt mit backup.bin	173
Abb. 2.39: Hexdump von backup.bin	174
Abb. 2.40: Hexdump der fehlerhaften Antwort	174
Abb. 2.41: Hexdump der erfolgreichen Antwort	175
Abb. 2.42: Ansicht in RouterPassView	175
Abb. 2.43: Ausgabe von RouterPassView	176
Abb. 2.44: Burb Suite	177
Abb. 3.1: Sicherheitsinfrastruktur	181
Abb. 3.2: Sicherheitskette	181
Abb. 3.3: Perimeter Absicherung	184
Abb. 3.4: BSI-Sicherheitsprozess (s. [Eckert, 2008])	187
Abb. 3.5: Netztopologie Beispiel (s. [Eckert, 2008] S. 165 nach BSI-Grundschutzhandbuch)	188
Abb. 3.6: Formel Schadensausmaß	191
Abb. 3.7: Angreifer Modell „Passwort ausspähen“	193
Abb. 3.8: HTML-basierte Visualisierung von Snort Protokolldaten	199

Abb. 3.9: Architektur eines Host-based Intrusion Detection Systems	200
Abb. 3.10: GFI LANguard Management Console [GFI Software, 2014]	200
Abb. 3.11: Windows Firewall	202
Abb. 3.12: EICAR Browser Test	203
Abb. 3.13: Windows Defender 8	204
Abb. 3.14: Windows Defender 10	204
Abb. 3.15: Einstellungen für regelmäßige Updates unter Windows 10 (Windows-Update)	205
Abb. 3.16: Übermittlung von Updates)	206
Abb. 3.17: Einstellungen der Zugriffsrechten von USBSTOR.SYS	207
Abb. 3.18: Einstellungen von Gruppenrichtlinien (Dateidownloads gewähren oder verbieten)	207
Abb. 3.19: Windows Sandbox aktivieren	210
Abb. 3.20: Windows Sandbox	210
Abb. 3.21: Windows Sandbox Konfiguration	212
Abb. 3.22: Windows Sandbox Startskript	213
Abb. 3.23: VirtualBox - Einrichtung	214
Abb. 3.24: VirtualBox - Link auf Laufwerk erzeugen	215
Abb. 3.25: Inhalt der Datei drive0.vmdk	215
Abb. 3.26: Windows Datenträgerverwaltung	216
Abb. 3.27: VirtualBox - Datenträger hinzufügen	216
Abb. 3.28: VirtualBox - Snapshot erzeugen	217
Abb. 3.29: VirtualBox - Nutzung von EFI aktivieren	217
Abb. 3.30: VirtualBox - VT-x aktivieren	217
Abb. 3.31: Kryptographisches System ⁴	219
Abb. 3.32: DES-Verfahren (Data Encrytion Standard)	221
Abb. 3.33: DES-Verfahren (DES Permutationstabelle)	222
Abb. 3.34: DES-Verfahren (DES Runden)	222
Abb. 3.35: DES-Verfahren (DES Verschlüsselungsfunktion)	223
Abb. 3.36: DES-Verfahren (DES Substitutionstabelle)	224
Abb. 3.37: Unterschriften mit dem RSA-Verfahren ⁵	229
Abb. 3.38: Verschlüsselung mit dem RSA-Verfahren und öffentlichem Schlüsseltausch ⁶	231
Abb. 3.39: Struktur eines X.509 Zertifikats	234
Abb. 3.40: Aufbau eines X.509 Zertifikats	236
Abb. 3.41: Beispiel für ein Zertifikat	238
Abb. 3.42: Zertifikat 1	243
Abb. 3.43: Zertifikat 2	243
Abb. 3.44: Zertifikat 3	244
Abb. 3.45: Perimeter-Absicherung mit Firewall	247
Abb. 3.46: Zustandsbasierte Untersuchung ⁷	251
Abb. 3.47: Wirkung der Personal Firewall	252
Abb. 3.48: Windows Personal Firewall Regeln per GUI ansehen und verändern	253
Abb. 3.49: Windows Personal Firewall Regeln mittels PowerShell ansehen	253
Abb. 3.50: Optionen für Filterregeln	256
Abb. 3.51: End-to-Site VPN	260
Abb. 3.52: Site-to-Site VPN	261
Abb. 3.53: End-to-End VPN	261
Abb. 3.54: Häufigkeit der versuchten Passwörter	271
Abb. 3.55: Versuchte Benutzernamen	271
Abb. 3.56: Anzahl der Verbindungen pro IP-Adresse	271
Abb. 3.57: Weltkarte mit Herkunft der Angriffe	272
Abb. 3.58: Erfolge- / Fehlversuche-Rate	272
Abb. 3.59: Nachricht, welche von OpenSSH als Antwort auf einen Banner gesendet wird	273
Abb. 3.60: Fehlermeldung , welche von Kippo als Antwort auf einen Banner gesendet wird	273
Abb. 3.61: Starten der Docker Container	274
Abb. 3.62: Starten des eigentlichen Honeypot's	275
Abb. 3.63: Dashboard des OWASP Honeypots	275
Abb. 3.64: Credential Events	276
Abb. 3.65: File Change Events	276

Abb. 3.66: Incident Response Lifecycle n. [Government, 2011]	280
Abb. 4.1: Phasen des Penetrationstests	308
Abb. 4.2: Ausschluss der Module durch die Klassifikation	311
Abb. 4.3: Nmap-Scanner	313
Abb. 4.4: Zenmap GUI	314
Abb. 4.5: John the ripper - md5	315
Abb. 4.6: Ausgangssituation vor dem ARP-Spoofing	318
Abb. 4.7: Erfolgreiches Vergiften der ARP-Tabelle	318
Abb. 4.8: Schritte zum ARP-Spoofing	320
Abb. 4.9: Driftnet	321
Abb. 4.10: Szenario DNS-Spoofing mit Phishing-Website	323
Abb. 4.11: DNS-Spoof mit Phishing-Website	324
Abb. 4.12: Weiterleitung nach Eingabe der Anmeldedaten	324
Abb. 4.13: Anmeldedaten des Benutzers in der Konsole des Angreifers	325
Abb. 4.14: OpenVAS-Architektur [OpenVAS, 2015]	328
Abb. 4.15: Greenbone Security Assistant Webinterface	330
Abb. 4.16: Advanced Task Wizard	330
Abb. 4.17: Greenbone Security Report	331
Abb. 4.18: Greenbone Security Report Override	332
Abb. 5.1: Incident Response Life Cycle	333
Abb. 5.2: Incident Response Lebenszyklus nach SANS	341
Abb. 5.3: Generisches Einbruchmodell nach Muth ⁸	344
Abb. 5.4: Erkennung Wurmbefall in der Search Information Phase	346
Abb. 5.5: Erkennung Wurmbefall in der Search/Exploit Vuln. Phase	347
Abb. 5.6: Erkennung Wurmbefall in der Examine System Phase	348
Abb. 5.7: Erkennung Wurmbefall in der Download Item Phase	349
Abb. 5.8: Erkennung Wurmbefall in der Escalate Privileges Phase	349
Abb. 5.9: Erkennung Wurmbefall in der Execute Attack Phase	350
Abb. 5.10: Dateiverschlüsselungsvorgang	361
Abb. 5.11: Ausgangsdatei unverschlüsselt	362
Abb. 5.12: Ergebnisdatei verschlüsselt	363
Abb. 5.13: Ausschnitt des Servers beim Verschlüsseln	363
Abb. 5.14: Ausschnitt des Clients beim Verschlüsseln	364
Abb. 5.15: Darstellung der Dictionaries und den Status Bezahlte einen Client zuweisen	364
Abb. 5.16: Ausschnitt des Servers beim Entschlüsseln	364
Abb. 5.17: Ausschnitt des Clients beim Entschlüsseln	365
Abb. 5.18: Ablauf des Keyloggers	367
Abb. 5.19: Ausgabe der Servers	370
Abb. 5.20: Ausgabe der Clients	370
Abb. 5.21: Procmon - Dateiaktionen	371
Abb. 5.22: Anzeigen von Prozess Parametern	372
Abb. 5.23: Procmon - Typen	372
Abb. 5.24: Procmon - Verbindungen	373
Abb. 5.25: CurrPorts - Python Prozess	373
Abb. 5.26: Startseite von Cuckoo	374
Abb. 5.27: Reports zuletzt hochgeladener Dateien	375
Abb. 5.28: Beispielhafter Report	376
Abb. 5.29: Darstellung des Scores	376
Abb. 5.30: Blaue Kategorie	377
Abb. 5.31: Gelbe Kategorie	377
Abb. 5.32: Rote Kategorie	377
Abb. 5.33: Darstellung von Screenshots	378
Abb. 5.34: Hinweise auf Datenabgriff von E-Mail-Clients	378
Abb. 5.35: Hinweise auf Datenabgriff von FTP-Clients	378
Abb. 5.36: Ausgaben von Snort	379
Abb. 5.37: Ausgaben bezüglich Code Injection	379
Abb. 5.38: Weitere Inhalte des Reports	380

Abb. 5.39: Angelegte Dateien	380
Abb. 5.40: Verhaltensanalyse	381
Abb. 5.41: Upload einer Datei	382
Abb. 5.42: Auswahl der VM	383
Abb. 5.43: Status des Tasks	383
Abb. 5.44: Verwenden der Suchfunktion	384
Abb. 5.45: Einschätzung der Gefährdung	384
Abb. 5.46: Übersicht der Screenshots	384
Abb. 5.47: Nachweis der Ausführung in der Eingabeaufforderung	385
Abb. 5.48: Log der Ausgaben in der Eingabeaufforderung	385
Abb. 5.49: Logeintrag mit dem Befehl zum Erstellen der Datei	385
Abb. 5.50: Logeintrag der das Erstellen der Datei festhält	385
Abb. 5.51: Angelegte Datei test in Dropped Files	386
Abb. 5.52: Rückfrage bzgl. des Secrets	386
Abb. 53: Lösung: Vergleich der Modelle nach NIST und SANS	401

II. Beispiele

Beispiel 1.1: SYN-Scan	26
Beispiel 1.2: SYN-Scan mit Portbereichs-Angabe	27
Beispiel 1.3: SYN-Scan mit Timing-Parameter	28
Beispiel 1.4: SYN-Scan mit Parameter -V	29
Beispiel 1.5: SYN-Scan mit Port-Aufzählung	30
Beispiel 1.6: SYN-Scan mit Parameter -A	31
Beispiel 1.7: UDP-Scan mit Parameter -v	34
Beispiel 1.8: UDP-Scan mit Parameter -sUV	35
Beispiel 1.9: Zeitoptimierter UDP-Scan	37
Beispiel 1.10: Variante zu zeitoptimiertem UDP-Scan	38
Beispiel 1.11: UDP-Scan mit Parameter -A	39
Beispiel 1.12: Ausgabe von rpcinfo	40
Beispiel 1.13: Beispielhafte NFS-Konfiguration	41
Beispiel 1.14: NFS-Freigabe auf Metasploitable-VM	42
Beispiel 1.15: SMB-Freigaben prüfen	45
Beispiel 1.16: SMB-Freigabe verbinden	46
Beispiel 1.17: SSH-Zugriff auf SMB-Freigabe einrichten	49
Beispiel 1.18: smbmap-Aufruf	52
Beispiel 1.19: Metasploit aktualisieren	56
Beispiel 1.20: Ausführen von Quelltext 1.1	63
Beispiel 2.1: ILOVEYOU	118
Beispiel 2.2: Lovsan- bzw. Blaster-Wurm	119
Beispiel 2.3: „Staatstrojaner“ ([Wikipedia, 2015a] / [CCC, 2011])	120
Beispiel 2.4: Mit MAC-Adressen unter Linux arbeiten	127
Beispiel 2.5: Eine echte Spam-Mail	139
Beispiel 2.6: Spam Mail	140
Beispiel 2.7: Mirai-Botnet	144
Beispiel 2.8: Computer Based Social Engineering	150
Beispiel 2.9: Human Based Social Engineering	151
Beispiel 2.10: Reverse Social Engineering	151
Beispiel 2.11: CEO Fraud	152
Beispiel 3.1: Schadensszenario: Beeinträchtigung des informationellen Selbstbestimmungsrecht	189
Beispiel 3.2: Schadensszenario: Beeinträchtigung der persönlichen Unversehrtheit	189
Beispiel 3.3: Schadensszenario: Negative Auswirkungen auf das Ansehen	189
Beispiel 3.4: Schadensszenario: Finanzielle Auswirkungen	189
Beispiel 3.5: Beispiel zur Risikoberechnung: Passwort ausspähen	193
Beispiel 3.6: Maßnahme M2.363 Schutz gegen SQL-Injection	195

Beispiel 3.7: Substitutionstabelle	224
Beispiel 3.8: Primzahl-Faktorisierung	226
Beispiel 3.9: Die Eulersche Funktion	226
Beispiel 3.10: RSA-Verschlüsselung	227
Beispiel 3.11: Filterregeln aufstellen für eine Workstation die über eine Internet-Verbindung verfügt. . .	257
Beispiel 3.12: Statische Paketfilterung Logging aller ICMP-Pakete	258
Beispiel 3.13: Dynamische Paketfilterung mit Netfilter	259
Beispiel 3.14: Security Incident	277

III. Definitionen

Definition 1.1: Schwachstelle; Verwundbarkeit	11
Definition 1.2: Bedrohung	11
Definition 1.3: Risiko	11
Definition 1.4: Angriff	12
Definition 1.5: IT-System	16
Definition 1.6: Geschlossene und offene Systeme	17
Definition 1.7: Safety	17
Definition 1.8: Security	17
Definition 1.9: Datenschutz (Privacy)	18
Definition 1.10: Datensicherheit (Protection)	18
Definition 1.11: Zugriff	19
Definition 1.12: Zugriffsrecht	19
Definition 1.13: Command-and-Control-Server	80
Definition 2.1: Computervirus	114
Definition 2.2: Wurm	118
Definition 2.3: Trojanisches Pferd	120
Definition 2.4: Sniffing ([Russel and Cunningham, 2002])	125
Definition 2.5: Spoofing ([Russel and Cunningham, 2002])	126
Definition 2.6: Replay ([Russel and Cunningham, 2002])	126
Definition 2.7: Denial-of-Service (DoS) ([Handley and Rescorla, 2006])	132
Definition 2.8: Malsite	135
Definition 2.9: Drive-by-Download	137
Definition 2.10: Bot	142
Definition 2.11: Botnetz	142
Definition 2.12: Social-Engineering	150
Definition 3.1: Security Engineering	179
Definition 3.2: Perimeter	184
Definition 3.3: Verschlüsselung	186
Definition 3.4: Sicherheit eines Kryptosystem [Hromkovic et al., 2010]	186
Definition 3.5: Kryptographie, Kryptoanalyse, Kryptologie	218
Definition 3.6: Paketfilter	244
Definition 3.7: Applikationsfilter	246
Definition 3.8: Personal Firewall ⁹	249
Definition 3.9: iptables ¹⁰	254
Definition 3.10: VPN ¹¹	260
Definition 3.11: Honeypot ¹²	264
Definition 3.12: Security Incident	277
Definition 3.13: Incident Response	277
Definition 3.14: Computer Security Incident Response Team	278
Definition 4.1: Wiederholung ARP	317
Definition 4.2: Wiederholung Gratuitous ARP	317
Definition 4.3: Vulnerability Assessment	326
Definition 4.4: Vulnerability Scan	326
Definition 5.1: Ransomware	352

IV. Exkurse

Exkurs 2.1: Erster Verschlüsselungstrojaner	123
Exkurs 2.2: (D)DoS-Angriffe	134
Exkurs 3.1: ISO/IEC 27001 auf Basis des BSI-Grundschutzes	194
Exkurs 4.1: Social Engineering Toolkit	325
Exkurs 5.1: Verschlüsselungsverfahren	353
Exkurs 5.2: Initialvektor	361

V. Kontrollaufgaben

Kontrollaufgabe 1.1: Sicherheitsrisiken veralteter Browser	16
Kontrollaufgabe 1.2: Konkretisierung von Schutzzielen	21
Kontrollaufgabe 1.3: Schwächen der NFS-Konfiguration in der Metasploitable-VM	45
Kontrollaufgabe 1.4: Schwächen der SMB-Konfiguration in der Metasploitable-VM	51
Kontrollaufgabe 1.5: Zugriff mittels smbmap	52
Kontrollaufgabe 1.6: Suchmaschine mit erweiterten Suchfunktionen nutzen	90
Kontrollaufgabe 1.7: Erweiterter Scan auf Kali-VM	92
Kontrollaufgabe 1.8: Bannergrabbing mit nmap	92
Kontrollaufgabe 2.1: Unterschied Schadsoftwaretypen	121
Kontrollaufgabe 2.2: Recherche zu Botnetzen	146
Kontrollaufgabe 3.1: Prinzipien des Security Engineerings	180
Kontrollaufgabe 3.2: Unterscheidung Zugangs- und Zugriffskontrolle	181
Kontrollaufgabe 3.3: Sicherheitsfunktionen	184
Kontrollaufgabe 3.4: Teilschritte BSI-Sicherheitskonzept	194
Kontrollaufgabe 3.5: HIDS vs. NIDS	201
Kontrollaufgabe 3.6: Symmetrische Verschlüsselung	225
Kontrollaufgabe 3.7: RSA-Verschlüsselung	228
Kontrollaufgabe 3.8: RSA-Anwendung für elektronische Unterschrift	230
Kontrollaufgabe 3.9: Dokument von Alice an Bob	231
Kontrollaufgabe 3.10: Zertifikatsinhalt	242
Kontrollaufgabe 3.11: Vor- und Nachteile verschiedener Realisierungsformen von Paketfiltern	245
Kontrollaufgabe 3.12: HTTPS-Server	246
Kontrollaufgabe 3.13: Firewalls	249
Kontrollaufgabe 3.14: Windows Personal Firewall	254
Kontrollaufgabe 3.15: Regeln für HTTP- und HTTPS-Freischaltung	259
Kontrollaufgabe 3.16: VPN-Anwendungen	261
Kontrollaufgabe 3.17: Sicherheitsvorfall Universitätsklinik	280
Kontrollaufgabe 3.18: Vorfallsprävention	281
Kontrollaufgabe 3.19: Aufgaben eines Incident Response Teams	283
Kontrollaufgabe 3.20: Blacklisten überprüfen	288
Kontrollaufgabe 3.21: Priorisierung von Vorfällen	296
Kontrollaufgabe 4.1: Szenario „Haupt-Filiale“: I-Module	309
Kontrollaufgabe 4.2: Szenario „Haupt-Filiale“: E-Module	309
Kontrollaufgabe 4.3: Passwort-Cracking	316
Kontrollaufgabe 4.4: ARP-Tabelle des Angriffsziels	320
Kontrollaufgabe 4.5: Vulnerability Assessment oder Penetrationstest	327
Kontrollaufgabe 5.1: Vergleich der Modelle nach NIST und SANS	343

VI. Tabellen

Tabelle 1.1: Ziele und Aktivitäten eines Angreifers	24
Tabelle 1.2: Übersicht einiger Google Befehle	89
Tabelle 2.1: Bedrohte Sicherheitsdienste	126

Tabelle 2.2: ARP-Tabelle	128
Tabelle 3.1: BSI-Schutzkategorien	188
Tabelle 3.2: Schutzbedarf „niedrig bis mittel“	190
Tabelle 3.3: Schutzbedarf „sehr hoch“	190
Tabelle 3.4: Checkliste nach [Government, 2011]	302
Tabelle 4.1: Übersicht der Module zur Informationsbeschaffung	309
Tabelle 4.2: Übersicht der Module für aktive Eindringversuche	310
Tabelle 4.3: Beispiel Penetrationstest	311
Tabelle 4.4: Vergleich Sicherheitsprüfungen	327
Tabelle 1: Lösung Kontrollaufgabe	389

VII. Literatur

Allen, Malcom. Social Engineering: A Means To Violate A Computer System, 2006. <https://www.sans.org/reading-room/whitepapers/engineering/social-engineering-means-violate-computer-system-529>.

Ross Anderson. *Security engineering*. John Wiley & Sons, 2008.

Dejan Baca and Bengt Carlsson. Agile development with security engineering activities. In *Proceedings of the 2011 International Conference on Software and Systems Process, ICSSP '11*, pages 149–158, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0730-7. doi: 10.1145/1987875.1987900. URL <http://doi.acm.org/10.1145/1987875.1987900>.

Bernhard C. Witt. Grundlagen des Datenschutzes und der IT-Sicherheit, 2017. https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui/datenschutz/Loesung2017-07-17.pdf.

D. J. Bernstein. Syn cookies, 1996. <http://cr.yp.to/syncookies.html>.

BSI - Bundesamt für Sicherheit in der Informationstechnik. Konzeption von Sicherheitsgateways - der richtige Aufbau und die passenden Module für ein sicheres Netz, 2005.

BSI - Bundesamt für Sicherheit in der Informationstechnik. Zertifizierung nach ISO 27001 auf der Basis von IT-Grundschutz, 2012a. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Zertifikat/ISO27001/Auditierungsschema.pdf?__blob=publicationFile.

BSI - Bundesamt für Sicherheit in der Informationstechnik. Zertifizierte IT-Sicherheit, 2012b. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Broschueren/Zertifizierte-IT-Sicherheit.pdf?__blob=publicationFile.

BSI - Bundesamt für Sicherheit in der Informationstechnik. *Sichere Nutzung von PCs unter Microsoft Windows 7*. 2013. https://www.bsi-fuer-buerger.de/SharedDocs/Downloads/DE/BSIFB/Publikationen/BSI-E-CS_001.pdf?__blob=publicationFile.

BSI - Bundesamt für Sicherheit in der Informationstechnik. Zertifizierung nach ISO 27001 auf der Basis von IT-Grundschutz, 2014. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Zertifikat/ISO27001/Zertifizierungsschema.pdf?__blob=publicationFile.

BSI (Bundesamt für Sicherheit in der Informationstechnologie). Die Lage der IT-Sicherheit in Deutschland 2015, 2015. <https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Lageberichte/Lagebericht2015.pdf>.

BSI (Bundesamt für Sicherheit in der Informationstechnologie). Ransomware, 2016. https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Cyber-Sicherheit/Themen/Ransomware.pdf?__blob=publicationFile&v=2.

Fort- und Weiterbildung

Neue Bedrohungsszenarien stellen Sicherheitsexperten und IT-Verantwortliche in Unternehmen und einschlägigen Behörden vor immer größere Herausforderungen. Neue Technologien und Anwendungen erfordern zusätzliches Know-how und personelle Ressourcen.

Zur Erhöhung des Fachkräftepools und um neues Forschungswissen schnell in die Praxis zu integrieren, haben sich die im Bereich lehrenden und forschenden Verbundpartner zum Ziel gesetzt, ein hochschuloffenes transdisziplinäres Weiterbildungsprogramm im Sektor Cyber Security zu entwickeln. Auf der Grundlage kooperativer Strukturen werden wissenschaftliche Weiterbildungsmodulen im Verbund zu hochschulübergreifenden Modulpaketen und abschlussorientierten Ausbildungslinien konzipiert und im laufenden Studienbetrieb empirisch getestet.

Die Initiative soll High Potentials mit und ohne formale Hochschulzugangsberechtigung über innovative Weiterbildungsangebote (vom Zertifikat bis zum Masterprogramm) zu Sicherheitsexperten aus- und fortbilden. Hierzu werden innovative sektorale Lösungen zur Optimierung der Durchlässigkeit von beruflicher und hochschulischer Bildung entwickelt und für eine erfolgreiche Implementierung vorbereitet. Unter prominenter Beteiligung einschlägiger Verbände, der Industrie sowie Sicherheits- und Ermittlungsbehörden verfolgt die Initiative das Ziel, im deutschsprachigen Raum eine Generation von Fachkräften wissenschaftlich aus- und weiterzubilden, die unser Internet schützen kann.

Open Competence Center for Cyber Security

Open C³S ist aus dem Verbundvorhabens Open Competence Center for Cyber Security entstanden. Das Gesamtziel des Programms war die Entwicklung eines hochschuloffenen transdisziplinären Programms wissenschaftlicher Weiterbildung im Sektor Cyber Security. Das Bundesministerium für Bildung und Forschung (BMBF) fördert das Großprojekt im Rahmen des Wettbewerbs „Aufstieg durch Bildung: offene Hochschulen“, der aus BMBF-Mitteln und dem Europäischen Sozialfonds finanziert wird.

Neun in Forschung und Lehre renommierte Hochschulen und Universitäten aus dem gesamten Bundesgebiet haben sich zum Ziel gesetzt, Online-Studiengänge auf dem Gebiet der Cybersicherheit zu entwickeln. Dieses Konzept soll den Studierenden ermöglichen, sich berufsbegeleitend auf hohem Niveau wissenschaftliche Qualifikationen anzueignen und akademische Abschlüsse zu erlangen. Beruflich erworbene Kompetenzen können eingebracht werden. Die Bezeichnung „Open“ steht auch für die Öffnung des Zugangs zu akademischer Bildung ohne klassischen Hochschulzugang.

Mission der Initiative ist es, dringend benötigte Sicherheitsexperten aus- und fortzubilden, um mit einer sicheren IT-Infrastruktur die Informationsgesellschaft in Deutschland und darüber hinaus zu stärken.

Umsetzungsnahes Wissen ist ein wesentlicher Schlüssel um der wachsenden digitalen Bedrohung zu begegnen. Solange wir nicht in der Lage sind, Systeme hinreichend zu härten, Netzwerke sicher zu designen und Software sicher zu entwickeln, bleiben wir anfällig für kriminelle Aktivitäten. Unser Ziel ist es, die Mitarbeiter von heute zu Sicherheitsexperten und Führungskräften von morgen auszubilden und dafür zu sorgen, dass sich die Zahl und die Fertigkeiten dieser Experten nachhaltig erhöht.

Z213 Netzwerkhacking

Dieses Modul soll zeigen, wie einfach es in vielen Fällen ist, einen Angriff durchzuführen. Den Teilnehmenden wird vermittelt, worauf ein erfolgreicher Angriff basiert. Dabei wird auf den Ansatz des Angriffs und die Mechanismen von Schadsoftware eingegangen. Den Teilnehmenden wird das stufenweise Vorgehen der Schadsoftware vermittelt.

Es wird zudem aufgezeigt, wie vorgegangen werden kann, wenn ein Angriff bereits eingetreten oder eventuell sogar noch aktiv ist. Hierzu werden die Themen „Incident Response“ als „Feuerwehr“ und die forensischen Ermittlungen näher eingegangen.

Mithilfe des Studienbriefs lernen die Teilnehmenden die bekannten Prinzipien zum Aufbau sicherer digitaler Infrastruktur kennen. Hier werden die einzelnen Bausteine der Netze, die Analyse und das Design der erforderlichen Sicherheitsmaßnahmen, aber auch die Schutzmaßnahmen für Clients sowie die Konzepte zur Netzwerksicherheit, eingehend betrachtet.

Die Teilnehmenden lernen den Nutzen einer ständigen Überwachung der Netzwerksicherheit kennen, zu dem das Network Security Monitoring, die Erfassung und Detektierung sowie die Analyse der Vorgänge gehören.

Es ist sinnvoll, Systeme selbst zu testen (Penetrationstests) und somit bereits möglichst früh einen Großteil der Schadsoftware auszusperren. Deshalb wird im Studienbrief auf die Schließung der Sicherheitslücken sowie die Schulung der Mitarbeiter eingegangen.

Zertifikatsprogramm

Die Zertifikatsmodule auf wissenschaftlichem Niveau und mit hohem Praxisbezug bilden ein passgenaues Angebot an Qualifikation und Spezialisierung in der nebenberuflichen Weiterbildung. Damit können einzelne Module nebenberuflich studiert werden. Durch die Vergabe von ECTS-Punkten können sie auf ein Studium angerechnet werden.

<https://zertifikatsprogramm.de>