



Sebastian
Kübeck

Web-Sicherheit

Wie Sie Ihre Webanwendungen
sicher vor Angriffen schützen

Inhaltsverzeichnis

Einleitung	11
Teil I Grundlagen der Informationssicherheit	19
I Wie konnte es nur so weit kommen?	
– Eine kurze Geschichte der Computerkriminalität	21
I.1 Frühe Computer	21
I.2 Telefonnetze und Phreaker	21
I.3 Time-Sharing	22
I.4 Die Bedeutungen des Wortes »Hacker« im Laufe der Zeit	23
I.5 Computernetze	24
I.6 Die unselige Verbindung von Computern und Telefonnetz	25
I.7 Der BTX-Hack	27
I.8 Das Internet	28
I.9 Vom Wurm zum Bot-Netz	30
I.10 Social Engineering	33
I.10.1 Spam- oder Junk-Mails	34
I.10.2 Verbreitung von Phishing-Mails	36
I.10.3 Scareware	37
I.11 Die Schattenwirtschaft im Internet	38
I.12 Aktivitäten von Geheimdiensten im Internet	39
2 Auswirkungen von Sicherheitsvorfällen auf Unternehmen und Organisationen	41
2.1 Was kann eigentlich passieren?	41
2.1.1 Verbreitung von Schadsoftware	41
2.1.2 Verbreitung von Spam	42
2.1.3 Verbreitung von Phishing-Mails	42
2.1.4 Verbreitung von illegalen Daten	42
2.1.5 Diebstahl von personenbezogenen Daten	43
2.1.6 DDoS-Attacken	43
2.1.7 Angriff von innen	44
2.2 Was passiert, wenn etwas passiert?	44
2.2.1 Auswirkung auf den Betrieb	45
2.2.2 Personenbezogene Daten	45
2.3 Vorbereitungen für den Ernstfall	46

3	Warum sich keiner betroffen fühlt und niemand etwas dagegen tut	47
3.1	Die Prospect-Theorie	47
3.2	Ökonomische Erklärungsversuche	49
3.2.1	Sicherheit in kommerzieller Software und bei IT-Dienstleistern	50
3.2.2	Sicherheit als Verkaufsargument	50
3.2.3	Sicherheit in Open-Source-Software	51
3.2.4	Sicherheit durch Webseitenbetreiber	51
3.3	Schlussfolgerungen	52
4	Grundprinzipien der Informationssicherheit	55
4.1	Vertrauliche Daten	55
4.2	Öffentlich zugängliche Daten	56
4.3	Öffentlich veränderbare Daten	57
4.4	Verfügbarkeit	58
4.5	Sicherheit als Wettbewerbssituation	58
4.6	Es gibt keine perfekte Sicherheit	59
4.7	Das Prinzip der tiefgreifenden Verteidigung (Defense In-Depth)	60
4.8	Das Prinzip des schwächsten Gliedes (Weakest Link)	61
4.9	Das Prinzip der minimalen Angriffsfläche (Minimum Attack Surface Area)	62
4.10	Das Aufteilungsprinzip (Compartmentalization)	63
4.11	Wenn Sicherheitssysteme fehlschlagen	63
4.12	Beurteilung von Sicherheitsmaßnahmen	65
4.13	Sicherheitstheater	66
5	Authentifizierung und Autorisierung	69
5.1	Begriffserklärungen	69
5.2	Biometrische Verfahren	70
5.3	Authentifizierung mithilfe eines Gegenstandes (Tokens)	72
5.4	Passwortauthentifizierung	75
5.4.1	Passwortkomplexität	75
5.4.2	Social Engineering	76
5.4.3	Dumpster Diving	76
5.4.4	Brute-Force-Attacken	76
5.4.5	Wörterbuchattacken	77
5.4.6	Mehrfach verwendete Passwörter	78
5.4.7	Password Sniffing	79
5.4.8	Zurücksetzen von Passwörtern	79
5.5	Einsatz des richtigen Authentifizierungsverfahrens	79

6	Sichere Nachrichtenübermittlung und -speicherung	81
6.1	Das Problem der sicheren Nachrichtenübermittlung	81
6.2	Das Problem der sicheren Datenspeicherung.	84
6.3	Grundregeln beim Einsatz von Verschlüsselungsverfahren	84
6.3.1	Setzen Sie nie ein selbst erfundenes kryptografisches Verfahren ein	84
6.3.2	Bevorzugen Sie öffentlich bekannte und gut getestete kryptografische Verfahren.	85
6.3.3	Setzen Sie nie veraltete oder bewusst geschwächte Verschlüsselungsverfahren ein!	86
6.3.4	Sorgen Sie für eine ausreichende Schlüssellänge	86
6.4	Elementare kryptografische Verfahren	87
6.4.1	Hashfunktionen	87
6.4.2	Zufallszahlengeneratoren	88
6.4.3	Symmetrische Verschlüsselungsverfahren	90
6.4.4	Message Authentication Codes (MACs)	92
6.4.5	Asymmetrische Verschlüsselungsverfahren	92
6.4.6	Digitale Signatur	93
6.4.7	Public Key Infrastructure (PKI).	94
6.4.8	Das SSL- bzw. TSL-Protokoll.	94
6.4.9	Verschlüsselung gespeicherter Daten und E-Mail-Verschlüsselung.	97

Teil II Die häufigsten Schwachstellen und deren Vermeidung 101

7	Eine Auswahl der häufigsten Schwachstellen	103
7.1	Werkzeuge zum Schutz von Webanwendungen	103
7.1.1	Filtermechanismen zum Schutz vor Angriffen	103
7.1.2	Bibliotheken und Frameworks zur sicheren Entwicklung von Webapplikationen	106
8	Injection-Schwachstellen	107
8.1	SQL-Injection-Schwachstellen	107
8.1.1	Einschleusen in numerische Werte	107
8.1.2	Einschleusen in Strings	108
8.1.3	Beeinflussen der Anwendungslogik.	109
8.1.4	Auslesen von beliebigen Datenbankinhalten.	110
8.1.5	Blind-SQL-Injection	111
8.1.6	Überprüfen und Umwandeln von Daten aus unsicheren Quellen	113

8.1.7	Prepared-Statements	114
8.1.8	Abdeckfunktionen	116
8.2	Command-Injection-Schwachstellen	117
8.2.1	Weitere Injection-Schwachstellen	119
8.2.2	Maßnahmen gegen Injection-Angriffe	120
8.2.3	Verhindern von Injection-Schwachstellen	120
9	Cross-Site-Scripting-Schwachstellen (XSS-Schwachstellen)	123
9.1	Die Ursache von Cross-Site-Scripting-Schwachstellen	123
9.2	Fallbeispiel: Ausspähen von Zugangsdaten beim Onlinebanking ..	125
9.3	Cross-Site-Scripting-Schwachstellen in JavaScript	128
9.4	Persistente Cross-Site-Scripting-Verwundbarkeiten	131
9.5	Maßnahmen zum Schutz vor Cross-Site-Scripting-Angriffen	132
9.6	Verhindern von Cross-Site-Scripting-Angriffen	136
10	Cross Site Request Forgery (CSRF)	137
10.1	Die Ursache von Cross-Site-Request-Forgery-Schwachstellen	137
10.2	Vertrauliche Daten ausspähen mithilfe von Cross Site Request Forgery	138
10.3	Cross-Site-Request-Forgery-Angriffe auf interne Anwendungen ...	139
10.4	Verhindern von Cross-Site-Request-Forgery-Verwundbarkeiten ...	140
10.5	Clickjacking	142
10.5.1	Verhindern von Clickjacking-Angriffen	145
11	Fehlerhafte Authentifizierung und Session-Management	149
11.1	Speichern von Passwörtern	149
11.2	Authentifizierung durch den Browser	150
11.3	Formularbasierte Authentifizierung	151
11.4	Session-Management	152
11.5	Zwei-Faktor-Authentifizierung in Webanwendungen	154
11.6	Weiterführende Literatur	155
12	Fehlerhafte Autorisierung	157
12.1	Öffentlich zugängliche vertrauliche Informationen	157
12.2	Verhindern, dass vertrauliche Daten unbeabsichtigt ins Netz gestellt werden	158
12.3	Unsichere direkte Objektreferenzen	159
12.4	Path-Traversal-Verwundbarkeiten	162
12.4.1	Path-Traversal-Verwundbarkeiten bei Upload-Mechanismen	164
12.5	Unsichere Weiterleitungen	165
12.5.1	Absichern von unsicheren Weiterleitungen	166

13	Puffer- und Integer-Überläufe (Buffer and Integer Overflows)	167
13.1	Pufferüberläufe (Buffer Overflows)	167
13.1.1	Ausnutzen von Pufferüberläufen	168
13.1.2	Gegenmaßnahmen auf Betriebssystemebene	170
13.1.3	Verhindern von Pufferüberläufen.	170
13.2	Integer-Überläufe (Integer Overflows) und andere Rechenfehler.	171
13.2.1	Verhindern von Integer-Überläufen.	173
14	Denial-of-Service-Schwachstellen (DoS-Schwachstellen) in Webanwendungen	175
14.1	DDoS-Angriffe	175
14.2	Endlosschleifen und Rekursionen mit fehlerhafter Abbruchbedingung	176
14.3	Speicherlecks	178
14.3.1	Nicht geschlossene Ressourcen	178
14.4	Langsame Datenbankzugriffe	180
14.5	Regular-Expression-DoS-Schwachstellen (ReDoS- Schwachstellen)	180
14.5.1	Mehrdeutige reguläre Ausdrücke	182
14.5.2	Gruppieren von mehreren Alternativen mit Wiederholungen.	182
14.5.3	Verwundbare reguläre Ausdrücke in Bibliotheken und Webanwendungen	183
14.5.4	Verhindern von ReDoS-Schwachstellen.	184
15	Unsichere Konfiguration	187
15.1	Gefährliche Standardeinstellungen	188
15.2	Unsichere Protokolle und nicht verwendete Funktionalitäten	188
15.3	Übertriebene Auskunftsfreude.	189
15.4	Interne Fehlermeldungen.	190
15.5	Fehlende Softwareaktualisierungen.	191
15.6	Erste Schritte zu einer sicheren Konfiguration.	192
15.7	Weiterführende Informationen zur sicheren Konfiguration.	194
Teil III Testen und Absichern von Webanwendungen		195
16	Testen der Sicherheit von Webanwendungen	197
16.1	Sicherheitstests	197
16.2	Überprüfen öffentlicher Inhalte.	199
16.3	Überprüfen von Netzwerken	200
16.4	Überprüfen auf bekannte Schwachstellen.	201

17	Penetrationstests (Penetration Tests oder Pen Tests)	205
17.1	Verwendung von Proxies für Penetrationstests	205
18	Code Reviews	217
18.1	Datenflussanalyse	217
18.2	Suche nach verdächtigen Mustern im Quellcode	222
18.3	Die üblichen Verdächtigen	232
18.4	Werkzeuge zur Unterstützung von Code Reviews	235
19	Webapplikationen sicher entwickeln	239
19.1	So früh wie möglich dekodieren, so spät wie möglich kodieren	239
19.2	Validieren und Umwandeln in passende Datentypen	241
19.2.1	Steuerzeichen	249
19.3	So spät wie möglich kodieren	249
20	Beheben von Schwachstellen in bestehenden Webapplikationen . . .	253
20.1	Vorarbeiten zum sicheren Beheben von Schwachstellen	253
20.1.1	Automatisierte Tests	253
20.1.2	Produktivsetzung	254
20.2	Beispiel: Beheben von SQL-Injection-Schwachstellen in einem Servlet	255
20.2.1	Bereinigen der eingehenden Parameter	265
20.3	Beispiel: Beheben von Schwachstellen in einem JSP-Skript	274
20.3.1	Absichern des Skripts	282
20.4	Abschließende Bemerkungen	285
A	Quellcodes	287
A.1	Hilfsklassen für Datenbankzugriffe	287
A.2	Testimplementierungen von Webserver-Komponenten	293
B	Referenzen	309
B.1	Einleitung	309
B.2	Teil I – Grundlagen der Informationssicherheit	309
B.3	Die häufigsten Schwachstellen und deren Vermeidung	314
	Stichwortverzeichnis	319