

Daniel Wittig

Die produzentenrechtlichen Verkehrssicherungspflichten von Softwareproduzenten



Nomos

Schriften zum Medien- und Informationsrecht

herausgegeben von
Prof. Dr. Boris P. Paal, M.Jur.

Band 50

Daniel Wittig

Die produzentenrechtlichen Verkehrssicherungspflichten von Softwareproduzenten



Nomos



Onlineversion
Nomos eLibrary

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Zugl.: Münster, Univ., Diss., 2020

ISBN 978-3-8487-7042-7 (Print)

ISBN 978-3-7489-1091-6 (ePDF)

D 6

1. Auflage 2021

© Nomos Verlagsgesellschaft, Baden-Baden 2021. Gesamtverantwortung für Druck und Herstellung bei der Nomos Verlagsgesellschaft mbH & Co. KG. Alle Rechte, auch die des Nachdrucks von Auszügen, der fotomechanischen Wiedergabe und der Übersetzung, vorbehalten. Gedruckt auf alterungsbeständigem Papier.

Vorwort

Die vorliegende Arbeit wurde im August 2020 von der rechtswissenschaftlichen Fakultät der Westfälischen Wilhelms-Universität Münster (WWU) als Dissertation angenommen. Sie entstand neben meiner beruflichen Tätigkeit als Rechtsanwalt in einer mittelständischen Wirtschaftskanzlei in Paderborn und unter Betreuung durch meinen Doktorvater Herrn Prof. Dr. Hoeren, dem Direktor des Instituts für Informations-, Telekommunikations- und Medienrecht (ITM) an der WWU. Ihm Danke ich u. a. für seine teils harsche Kritik, die zur Fokussierung der Arbeit auf das Wesentliche führte und so maßgeblich zum Gelingen dieser beitrug. Ebenfalls danke ich Herrn Prof. Dr. Fabian Gieseke vom Lehrstuhl Maschinelles Lernen und Data Engineering für die rasche Erstellung des Zweitgutachtens. Mein Dank gilt auch meinen Arbeitskollegen, die Rücksicht auf meine nebenberufliche Promotion nahmen und mich nach Möglichkeit unterstützen.

Während meiner Zeit im Referendariat wurde ich von einem meiner Mentoren auf die Problematik der Haftung von Softwareproduzenten aufmerksam gemacht. Ich musste feststellen, dass wir die für die Haftung wesentlichen Verkehrssicherungspflichten kaum bestimmen konnten. Ab diesem Zeitpunkt stand für mich fest, dass ich einen wissenschaftlichen Beitrag zu diesem Thema leisten wollte, um an der Lösung des Problems mitzuwirken.

Mein ganz besonderer Dank geht an meine Eltern, Petra und Werner Tanger, sowie an meine Ehefrau Katarina. Meine Eltern unterstützten stets vorbehaltlos meinen Lebensweg und die Entscheidungen, die ich auf diesem getroffen habe. Sie ermöglichten mir meine Ausbildung, welche die Basis für meine persönliche und berufliche Entwicklung geworden ist. Nur durch Sie konnte ich mich immer auf meine Ziele konzentrieren, ohne mir Sorgen machen zu müssen. Meine Frau Katarina, welche seit dem Jahr 2016 an meiner Seite ist, gibt mir darüber hinaus in jeder Lebenslage ein unglaubliches Maß an Unterstützung, Rückhalt und Liebe. Ohne ihren Zuspruch und ihr Durchhaltevermögen hätte ich diese Arbeit wohl nicht vollenden können. All dies hat im wesentlichen Maße zum Gelingen dieser Arbeit beigetragen.

Paderborn, im November 2020

Daniel Wittig

Inhaltsverzeichnis

A. Einleitung	19
I. Die steigende Bedeutung von Software für Wirtschaft und Gesellschaft	20
II. Sicherheit von Software als gesamtgesellschaftliches deliktsrechtliches Problem	21
III. Die Herstellung fehlerfreier Software unter ökonomischen Gesichtspunkten	23
IV. Verkehrssicherungspflichten – Kernstück der Produzentenhaftung	26
B. Begriffsdefinitionen	28
I. Software und Softwareproduzent	28
II. Daten	29
III. IT-Sicherheit und Datensicherheit	30
1. Der Begriff der IT-Sicherheit	30
a) Schutzgüter der IT-Sicherheit	32
aa) Verfügbarkeit	32
bb) Vertraulichkeit	32
cc) Integrität	33
dd) Authentizität	33
ee) Interdependenz der Schutzgüter	33
b) Bedeutung der IT-Sicherheit	34
c) Abgrenzung der IT-Sicherheit von der Produktsicherheit	34
2. Ableitung des Begriffs der Datensicherheit	35
3. Abgrenzung der Datensicherheit vom Datenschutz	35
IV. Schwachstelle	36
C. Deliktische Haftungsrisiken und die Produzentenhaftung	39
I. Deliktisches Haftungsrisiko für Softwareproduzenten	39
1. Zweiseitige Inanspruchnahme möglich	39
2. Steigerung der Angriffsfläche durch IT und Vernetzung	41
3. Bisherige Tätigkeit des Gesetzgebers	41

II. Die Entwicklung der Produzentenhaftung	42
1. Die Schaffung der Produzentenhaftung	42
2. Verletzung einer Verkehrssicherungspflicht	44
3. Entwicklung von Beweiserleichterungen	45
III. Weiterhin offene Fragestellungen im Deliktsrecht	45
1. Entscheidungen autonomer Systeme	46
2. Daten als Rechtsgutverletzung	47
3. Kausalitätsprobleme	47
4. Verschuldensnachweis	48
5. Bestimmung von Verkehrssicherungspflichten	49
IV. Versicherbarkeit der Haftungsrisikos	50
V. Geltung der Produzentenhaftung für die Softwareerstellung	50
D. Die geltenden Verkehrssicherungspflichten	52
I. Konstruktionspflichten	55
1. Sicherheitserwartungen des Verkehrs	56
2. Erkennbarkeit des Fehlers bei einem Inverkehrbringen	57
3. Abgrenzung zum Entwicklungsfehler	57
II. Fabrikationspflichten	58
III. Instruktionspflichten	59
1. Erkennbarkeit der Gefahr bei einem Inverkehrbringen	60
2. Inhalt und Ausgestaltung der Instruktionspflichten	60
3. Umfang der Instruktionspflichten	61
IV. Produktbeobachtungspflichten	62
1. Haftungsgrund der Produktbeobachtungspflichten	64
2. Umfang der Produktbeobachtungspflicht	64
3. Zeitraum statt Zeitpunkt der Pflichterfüllung	65
4. Ende der Produktbeobachtungspflichten	65
5. Handlungspflicht bei Entdeckung eines Fehlers	66
a) Warnpflicht	67
b) Gefahrverdacht	67
c) Konstruktionsänderung	68
d) Rückrufverpflichtung	69
aa) Äquivalenz- vs. Integritätsinteresse	69
bb) Cheapest Cost Avoider	70
cc) Neben Rücknahme auch Reparatur	72
dd) Bestimmung im Einzelfall notwendig	73

ee) Kein subjektiver Anspruch des Nutzers	74
E. Die Ausgestaltung der Verkehrssicherungspflichten für Softwareproduzenten	75
I. Keine Entlastung bei Eingriffen durch Hacker	75
II. Maßstab für die Ermittlung der Verkehrssicherungspflichten	77
1. Einflussfaktor der Verkehrserwartung an Software	78
a) Einzelfallentscheidung – Kriterium des Absatzmarktes	79
b) Einzelfallentscheidung – Kriterium des Nutzerkreises	80
2. Einflussfaktor „aktueller Stand von Wissenschaft und Technik“	80
a) Abgrenzung vom Stand der Technik	80
aa) Verwendung unbestimmter Rechtsbegriffe	81
bb) Inhaltliche Unterscheidung	82
b) Heranziehung des Standes von Wissenschaft und Technik	83
c) Konkretisierung des Standes von Wissenschaft und Technik	85
d) Bildung des Standes von Wissenschaft und Technik in der IT-Branche	85
aa) Schwerpunkt Ausland	86
bb) Beeinflussung durch Marktführer	86
cc) Erweiterung der Einflussfaktoren	87
e) Einfluss technischer Standards und Zertifizierungen auf den aktuellen Stand von Wissenschaft und Technik	88
aa) Technische Standards	89
bb) Technische Standards sind keine verbindlichen Rechtsnormen	89
cc) Problem der Bestimmtheit eines technischen Standards	90
dd) Praktische Bedeutung von technischen Standards	91
(1) Keine Entlastung durch Einhaltung technischer Standards	91
(2) Technische Standards als Ansatzpunkt der Konkretisierung unbestimmter Rechtsbegriffe	92
f) Änderung des Standes von Wissenschaft und Technik nach einem Inverkehrbringen	93
g) Einhaltung des Standes von Wissenschaft und Technik durch Zertifizierung	93
aa) Begriff der Zertifizierung	94

bb) Keine Entlastung durch eine Zertifizierung	94
(1) Statische Zertifikate	94
(2) Ausnahme: Öffentliches Recht	95
h) Einfluss von Zertifizierungen auf Verkehrssicherungspflichten	95
i) Besonderheiten im IT-Sektor	95
aa) Common Criteria – aktueller Einfluss und zukünftige Möglichkeiten	97
(1) Ausgestaltung der Common Criteria	98
(2) Common Criteria als Selbstverpflichtung	99
(3) Vorteile der Common Criteria für die Produzenten	99
(4) Hürden für die Produzenten	100
(5) Zukünftige Möglichkeiten der Common Criteria	100
bb) Protection Profiles	101
(1) Aufbau der Protection Profiles	102
(2) Anwendungsbereich einzelner Protection Profiles	102
(3) Zukünftige Möglichkeiten der Protection Profiles	103
cc) Security Design Principles	103
dd) Branchenspezifische Standards (Beispiel: PCI-DSS)	104
ee) Secure Coding Guidelines	104
ff) Einfluss von unbekanntem Zertifikaten auf die Verkehrssicherungspflichten von Softwareproduzenten	105
(1) Bestehen unbekannter Zertifikate	105
(2) Auswirkungen von im Verkehr unbekanntem Zertifikaten	106
3. Einflussfaktor Cybersecurity Act	107
a) Allgemeine Ziele	107
b) Sicherheitsziele	108
aa) Harmonisierung	109
bb) Fragmentierung	110
cc) Konkrete Sicherheitsziele	110
c) Ausgestaltung des Zertifizierungsverfahrens	111
aa) Keine Einführung operativer Zertifizierungssysteme	111
bb) Rückgriff auf technische Normen	111
cc) Unterschiedliche Ansätze und Sicherheitsstufen für die Schemata	112
dd) Freiwilligkeit der Zertifizierung	112
d) Rolle der Cybersicherheitsbehörde ENISA	113

e) Auswirkungen der Verordnung auf Softwareproduzenten – Erhöhung des Sicherheitsstandards	114
4. Einflussfaktor DS-GVO auf die Verkehrssicherungspflicht	115
a) Datenschutz ungleich Datensicherheit	116
b) Anwendungsbereich der DS-GVO	116
aa) Sachlicher Anwendungsbereich	117
bb) Räumlicher Anwendungsbereich	118
cc) Persönlicher Anwendungsbereich	118
(1) Verantwortlicher	119
(2) Auftragsverarbeiter	119
(3) Hersteller – Softwareproduzenten	120
(4) Mittelbare Anwendbarkeit der DS-GVO auf Softwareproduzenten	121
(a) Hersteller als Adressat des Art. 25 DS-GVO – Datenschutz durch Technikgestaltung	122
(b) Hersteller als Adressat des Art. 32 DS-GVO – Sicherheit der Verarbeitung	124
(c) Hersteller als Adressat des Art. 42 DS-GVO – Zertifizierungsmaßnahmen	124
c) Bindung der Produzenten an die Pflichten der DS-GVO	125
aa) Konkretisierung der Verkehrssicherungspflichten	125
(1) Stand von Wissenschaft und Technik	126
(a) Diskrepanz: „Stand der Technik“ und „Stand von Wissenschaft und Technik“	127
(b) Umsetzungserschweris aufgrund fehlender verbindlicher Leitlinien	129
(c) Rückgriff auf bereits bestehende Normen unzureichend	129
(2) Erwartungshaltung der Verantwortlichen	131
(a) Notwendigkeit der präzisen Abgrenzung bei der Verkehrserwartung	132
(b) Keine direkte Verpflichtung über Verkehrssicherungspflichten möglich	132
(c) Indirekte Verpflichtung	133
(3) Zumutbarkeit	134
(4) Zwischenfazit zu den Auswirkungen der DS-GVO	135
(a) Verschiebung der Nachfrage	135
(b) Keine Bußgelder, aber Schäden	136
bb) Erste Schritte mit und durch die DS-GVO	137
(1) Beispiel: Produktbeobachtungspflicht	137

(2) Beispiel: Konstruktionspflichten	138
(3) Beispiel: Instruktionspflichten	138
cc) Möglichkeit der Einhaltung datenschutzrechtlicher Vorschriften	138
d) Die (mittelbaren) Pflichten der DS-GVO für Softwareproduzenten	139
aa) Grundsätze der Datenverarbeitung, Art. 5 DS-GVO	140
(1) Neuausrichtung der Grundsätze der Datenverarbeitung	140
(2) Bezugnahme auf die Sicherheit der Datenverarbeitung	141
bb) Datenschutz durch Technikgestaltung und Voreinstellung, Art. 25 DS-GVO	142
(1) Anforderungen der Norm	143
(a) Privacy by Design	143
(b) Privacy by Default	143
(c) Organisatorische Maßnahmen	144
(2) Umsetzung der Norm	144
(a) Frühe Berücksichtigung der Einzelmaßnahmen	145
(b) Die wirtschaftliche Komponente der Umsetzung	145
(c) Datenschutz durch Technikgestaltung	146
(d) Datenschutz durch Voreinstellung	148
cc) Sicherheit der Verarbeitung, Art. 32 DS-GVO	149
(1) Anforderungen der Norm	149
(2) Umsetzung der Norm	150
(a) Konkret benannte Maßnahmen	151
(b) Weitere – technisch unbenannte – Maßnahmen des Art. 32 DS-GVO	151
dd) Zertifizierungsmaßnahmen, Art. 42 DS-GVO	154
(1) Ziele der Zertifizierung	154
(2) Zertifikat für Produzenten	154
ee) Data Breach Notification, Art. 33 DS-GVO	156
5. Einflussfaktor „EU-Richtlinie über bestimmte vertragsrechtliche Aspekte der Bereitstellung digitaler Inhalte“	156
a) Anwendungsbereich	157
b) Updatepflicht – Mangelbegriff	158
c) Grenze der Unverhältnismäßigkeit	159

d) Gewährleistungsfrist	159
e) Mögliche Auswirkungen auf die Verkehrssicherungspflichten der Softwareproduzenten	160
III. Die Konstruktionspflichten der Softwareproduzenten	161
1. Schlüsselfrage: Sind Fehler bei einem Inverkehrbringen erkennbar?	164
a) Möglichkeit der fehlerfreien Erstellung von Software	164
aa) Falschaussage: Fehlerfreie Erstellung nicht möglich	165
bb) Gründe für die Fehlerhaftigkeit von Software	166
(1) Gestiegene Komplexität	167
(a) Mehr Code, mehr Fehlermöglichkeiten	167
(b) Erschwerung der Testbedingungen	168
(2) Kosten für Qualitätssicherung	168
(3) Missmanagement	169
(a) Qualitätssicherung als Geldverschwendung	169
(b) Fehlerhafte Zeit- und Kostenplanung	170
(c) Lückenhafte Dokumentation	171
(d) Wartungsverträge	171
(4) Marktgegebenheiten	171
(a) Nachgefragte Qualität	171
(b) Unklare Anforderungen an Software	173
(c) Verkürzte Produktzyklen	173
(d) Wettbewerbsnachteil Fehlerfreiheit	174
(5) Fehlende Cybersicherheitskenntnisse	174
(6) Agile Programmiermethoden	175
(7) Alterung der Software	176
b) Korrektur der These	177
2. Konstruktionspflichten der Softwareproduzenten	177
a) Keine Ausnahme von den Konstruktionspflichten	178
b) Abschwächung der Konstruktionspflichten durch Verkehrserwartung	179
c) Einzelne Konstruktionspflichten	180
aa) Konstruktive und analytische Qualitätssicherung	181
bb) Organisationspflichten im Arbeitsablauf – konstruktive Maßnahmen	182
(1) Organisation des Betriebs und des Arbeitsablaufs	182
(2) Einsatz eines Versionsverwaltungssystems	185
(3) Entwicklungs- und Testdokumentationen	185
(4) Absicherung der Entwicklungsumgebung	186
(5) Einzelfallentscheidung	186

cc) Analytische / Technische Prüfpflichten	187
(1) Automatisierte Prüfungstools (Debugging)	187
(a) Teilautomatisierte werkzeuggestützte Analyse	188
(b) Debugger	189
(c) Automatisierte Prüfungstools und Big Data	190
(d) Integrierte Entwicklungsumgebungen	191
(2) Softwaretests – Organisation und Grundsätze	191
(a) Testgrundsätze	192
(b) Testorganisation	193
(c) Testorganisation in verschiedenen Entwicklungsmodellen	195
(3) Softwaretests – Arten und Auswahl	197
(a) Statische und dynamische Testverfahren	198
(b) White- und Black-Box-Tests	198
(c) Überblick verschiedener Testarten	199
(d) Auswahl der Tests	201
(4) Manuelle Softwareprüfungen	203
(a) Formen der manuellen Softwareprüfungen	204
(b) Vorteile der manuellen Softwareprüfungen	205
(c) Manuelles Testen als Ergänzung, nicht als Alternative	206
(5) Nachweis der Fehlerfreiheit – Softwareverifizierung	207
dd) Weitere mögliche Konstruktionspflichten	207
(1) Programmierung redundanter Software	208
(2) Updatability by Design	208
(3) Behebung festgestellter Schwachstellen im Quellcode	210
(a) Anpassung in laufender Produktion	210
(b) Zeitrahmen zur Behebung der Schwachstelle	210
(4) Vorgaben an Zulieferer	212
3. Zusammenfassung der einzelnen Konstruktionspflichten	213
a) Organisatorische Pflichten	213
b) Technische / Analytische Pflichten	214
aa) Verpflichtende Testverfahren	215
bb) Testtechniken	216
c) Weitere Konstruktionspflichten	217
d) BSIMM-Studie	217
4. Besonderheit: OEM-Version	218

IV. Die Produktbeobachtungspflichten der Softwareproduzenten	218
1. Verschärfung der Produktbeobachtungspflicht bei (bewusst) fehlerhafter Software	219
2. Beobachtung der eigenen Software	220
3. Beobachtung von Fremdsoftware	220
4. Integrierte Produktbeobachtung	222
a) Vorteile der integrierten Produktbeobachtung	222
b) Integrierte Produktbeobachtung – Keine rein zukünftige Pflicht	223
c) Anwendbarkeit neben passiver und aktiver Produktbeobachtung	224
5. Handlungspflichten bei Entdeckung einer Gefahr	224
a) Warnung vor Schwachstellen	225
aa) Nutzerspezifische Warnung	226
(1) Art der Verbreitung der Warnung	227
(2) Mitteilungsquote	227
(3) Verständlichkeit der Warnung	228
bb) Unzulänglichkeit einer Warnung	228
cc) Warnung als Gefahrerhöhung	229
dd) Mitteilungspflicht von Sicherheitsstörungen beim BSI	230
b) Keine Herausgabe des Quellcodes	231
c) Softwarestilllegung mittels „Kill Switch“	231
aa) Stilllegung effektiver als Update?	232
bb) Grenzen der Stilllegung	232
(1) Beschränkung auf internetbasierte Software	233
(2) Mögliche Eigentumsverletzung durch Stilllegung	233
(3) Zwangsupdate als gleich effektives Mittel	233
(4) Bedrohung des Nutzers allein	234
cc) Kombination aus Kill Switch und Update	234
d) Erweiterte Rückruf- und Updatepflichten	234
aa) Ausgestaltung des Rückrufs bei Software	236
bb) Verschiebung der Zumutbarkeitserwägung	236
(1) Geringe Kosten für Update / Patch	237
(2) Dekompilierungsverbot	239
(a) Keine Umgehung des Dekompilierungsverbotes durch Reverse-Engineering	240
(b) Angewiesenheit auf den Produzenten	241
cc) Keine generelle Rückruf-/Updatepflicht	241
(1) Einklang mit dem Effizienzprinzip	243

(2) Nachhaltige Behebung	244
dd) Ende des Software-Supports	244
ee) Kein subjektiver Anspruch auf Updates	245
V. Weitere Verkehrssicherungspflichten für Softwareproduzenten	245
1. Fabrikationspflichten	246
2. Instruktionspflichten	247
a) Allgemeine Instruktionspflicht	247
b) Erneute Instruktionspflicht nach einem Update	248
F. Notwendigkeit zur Schaffung eines IT-Produktsicherheitsrechts oder von IT-Sicherheitsstandards	249
I. Schaffung einer IT-Sicherheit-Grundverordnung	251
1. Ausgestaltung und Bestandteile einer IT-Sicherheit- Grundverordnung	252
a) Einführung als Schutzgesetz i. S. d. § 823 Abs. 2 BGB	252
b) Einführung einer Meldepflicht	253
c) Sanktionierung unsicherer IT-Produkte	253
d) Updatepflicht	254
e) Prüfstelle für IT-Produkte	255
2. Notwendigkeit einer neuen Gesetzgebung	255
II. Schaffung von technischen Standards	256
1. Vorteile gegenüber Recht de lege ferenda	258
2. Ausgestaltung und Inhalt technischer Standards	259
a) Normungsorganisationen	260
b) Abstufung der Pflichten	261
c) Security by Design und Security by Default	262
d) Kontinuierliche Weiterentwicklung	264
e) Updatepflicht und Länge des Software-Supports	265
f) Mehrstufige Zertifizierungen	266
g) (Re-)Zertifizierungspflichten	266
h) Schwachstellen veröffentlichen	267
3. Schaffung von Standards allein genügt nicht	269
III. Initiative der Wirtschaft – Charter of Trust	270
1. Gründe für die Erstellung der Charter of Trust	271
2. Kernziele der Charter of Trust	271
3. Weitere Ziele und Maßnahmen der Charter of Trust	271
4. Fazit bezüglich der Charter of Trust	272

G. Fazit	274
Literaturverzeichnis	285

A. Einleitung

Software durchdringt und vernetzt immer mehr Bereiche unserer Wirtschaft und Gesellschaft. Sie verbreitet sich immer weiter und macht weder vor Alltagsgegenständen noch Industriegütern Halt. Die Bedeutung von Software hat in vielen Bereichen so sehr zugenommen, dass man bereits von einer Abhängigkeit sprechen kann.¹

Diese Entwicklung verwundert etwas, wenn man in Betracht zieht, dass es sich bei Software um ein Produkt handelt, zu dem sich bereits in den 1980er Jahren die These entwickelt hat, dass es nicht fehlerfrei hergestellt werden kann.² Diese These, auf die man auch gegenwärtig noch stößt, ist heutzutage schlichtweg falsch.³ Dennoch stimmt die Alltagserfahrung vieler Nutzer auch heute noch mit ihr überein. Jeder Softwarenutzer kennt Situationen, in denen er von Softwarefehlern geplagt wurde (sei es dadurch, dass ein Programm nicht mehr richtig reagiert, dass eine Fehlermeldung erscheint oder dass Daten verloren gehen). Auch gehört die Fehlerbehebung durch die Installation von Updates oder Upgrades zum Alltag der Softwarenutzer. Die Nutzer haben sich an diese Zustände schlichtweg gewöhnt.⁴

Aus rechtlicher Sicht stellt sich bei dieser Sachlage unmittelbar die Frage, welche Auswirkungen diese Gemengelage auf die Verkehrssicherungspflichten der Softwareproduzenten hat. Auf den ersten Blick ist also bereits fraglich, wie ein schnell entwickelnder Industriezweig, eine historische, aber mittlerweile falsche These, die Tatsache, dass Softwareproduzenten ihre Produkte trotz der Möglichkeit zur fehlerfreien Entwicklung nicht

1 BSI, Lagebericht 2019, S. 4; *Engel*, CR 1986, 702 (702); *Wiesemann/Mattheis/Wende*, MMR 2020, 139 (139); *Auer-Reinsdorff/Conrad/Conrad*, § 33, Rdnr. 2 f.

2 So auch: BSI, Lagebericht 2005, S. 15; BSI, Lagebericht 2015, S. 10; *Marly*, Rdnr. 1436; *Sassenberg/Faber/Wende*, Teil 2 C., Rdnr. 49; *Scholz*, S. 177; *Sodtalbers*, S. 145, Rdnr. 223; *Spindler*, Rdnr. 124; *Engel*, CR 1986, 702 (708); *Grünvogel/Dörrenbächer*, ZVertriebsR 2019, 87 (88); *Heussen*, CR 2004, 1 (1 f.); *Meier/Weblau*, CR 1990, 95 (96); *Müller-Hengstenberg/Kirn*, MMR 2014, 307 (310); *Raue*, NJW 2017, 1841 (1841); *Reusch*, BB 2019, 904 (908); *Spyra*, MPR 2015, 15 (17); *Taeger*, CR 1996, 257 (257); *Wiebe*, NJW 2019, 625 (625); *Wiesemann/Mattheis/Wende*, MMR 2020, 139 (139).

3 *Lenhard*, S. 27; *Sodtalbers*, S. 147, Rdnr. 226; siehe auch Abschnitt „Möglichkeit der fehlerfreien Erstellung von Software“, ab 164 dieser Arbeit.

4 *Faber*, DuD 2019, 393 (393).

A. Einleitung

fehlerfrei produzieren und die Gewöhnung der Nutzer daran, Einfluss auf unser Haftungssystem nehmen kann und ob es den Herausforderungen der Softwareindustrie noch gewachsen ist. Diese Arbeit soll einen Beitrag zur Aufklärung dieser Fragestellung leisten.

I. Die steigende Bedeutung von Software für Wirtschaft und Gesellschaft

Der Einsatz und die Bedeutung von Software nimmt unaufhörlich zu. In vielen Unternehmen ist Software bereits von solcher Wichtigkeit, dass sie deren Achillesferse darstellt.⁵ Durch die starke Verbreitung und die zunehmende Bedeutung von Software nimmt gleichzeitig aber auch das Gefahrenpotential für unsere Rechtsgüter zu.⁶

Die Cyberkriminalität wächst stetig. Auch Software wird immer häufiger das Ziel von Hackern.⁷ Das Wachstum der Kriminalität kann einerseits auf die vergrößerte Angriffsfläche durch die immer größere Anzahl der angreifbaren Software, andererseits aber auch auf eine Effizienzsteigerung der Angriffe zurückgeführt werden.⁸ Die Hacker können zum einen die mit Software betriebenen Geräte selbst attackieren, zum anderen können sie diese Geräte aber auch nutzen, um Geräte Dritter anzugreifen. Sobald neue Schwachstellen einer Software bekannt sind, werden diese innerhalb kürzester Zeit durch Hacker ausgenutzt.⁹ Da die Attacken mittels moderner Schadsoftware oftmals automatisiert erfolgen, können sie sich in rasender Geschwindigkeit ausbreiten. Daher drohen bei einem Softwareangriff schnell weltweite Schäden.¹⁰ Die Schadenshöhe ist dabei kaum begrenzt. So können Hacker Schwachstellen nutzen, um ganze Industrieproduktionen lahmzulegen oder um auf Kundendaten zuzugreifen. Gleichzeitig sind aber auch Verletzungen von hochwertigen Rechtsgütern zu befürchten (wie z. B. bei Angriffen auf autonom fahrende Pkw oder Medizinsyste-

5 BSI, Lagebericht 2019, S. 4; *Engel*, CR 1986, 702 (702); *Wiesemann/Mattheis/Wende*, MMR 2020, 139 (139); *Auer-Reinsdorff/Conrad/Conrad*, § 33, Rdnr. 2 f.

6 BSI, Lagebericht 2019, S. 4; *Engel*, CR 1986, 702 (702); *Wiesemann/Mattheis/Wende*, MMR 2020, 139 (139).

7 BSI, Lagebericht 2019, S. 4 f.; *Westermann/Witt*, S. 32; *Brisch/Rexin*, CR 2019, 606 (607).

8 *Brisch/Rexin*, CR 2019, 606 (607).

9 BSI, Lagebericht 2019, S. 8.

10 BSI, Lagebericht 2019, S. 4.

II. Sicherheit von Software als gesamtgesellschaftliches deliktsrechtliches Problem

me).¹¹ Die möglichen Bedrohungen sind vielfältig. Spiegelbildlich zu den Gefahren steigt somit die Bedeutung der Produzentenhaftung.¹²

Die meisten Angriffe auf Software sind nur deshalb möglich, weil Software sicherheitsrelevante Fehler (sog. Schwachstellen) enthält. Das Bundesamt für Sicherheit in der Informationstechnik hat in einer im Jahr 2016 durchgeführten Untersuchung der zehn meist genutzten Softwareprogramme über 717 Schwachstellen erfasst.¹³ Auch in seinem Lagebericht zur IT-Sicherheit aus dem Jahr 2019 kommt das BSI u. a. zu dem Ergebnis, dass Schadprogramme zu den größten Bedrohungen für die Nutzer von Software gehören.¹⁴ So werden durchschnittlich ca. 320.000 Schadprogramme pro Tag entwickelt.¹⁵

Trotz der kontinuierlich wachsenden Bedeutung von Software und der durch sie verursachten Gefahren fristet das IT-Sicherheitsrecht noch immer ein Nischendasein.¹⁶ Eine mögliche Ursache könnte darin liegen, dass es sich beim IT-Sicherheitsrecht um eine Querschnittsmaterie aus öffentlichem Recht, Strafrecht und Zivilrecht handelt.¹⁷ Das öffentlich-rechtliche IT-Sicherheitsrecht ist derzeit auf die Anwendung auf kritische Infrastrukturen beschränkt. Dadurch hat es jedoch nur Einfluss auf die Absicherung von Anlagen mit besonderer Bedeutung für das Gemeinwesen.¹⁸ Die Entwicklung des zivilrechtlichen IT-Sicherheitsrechts hat der Gesetzgeber hingegen allein der Rechtspflege überlassen, obwohl es einer klaren Haftungs-zuweisung bedarf, um Anreize für die Herstellung fehlerfreier Software zu schaffen.¹⁹

II. Sicherheit von Software als gesamtgesellschaftliches deliktsrechtliches Problem

Schwachstellen in einer Software setzen sowohl unsere Gesellschaft als auch unsere Wirtschaft einer massiven Gefährdung aus. Gleichzeitig stellen Softwareproduzenten weiter fehlerhafte Software her, obwohl eine feh-

11 BSI, Lagebericht 2019, S. 4.

12 *Etzkorn*, MMR 2020, 360 (360).

13 BSI, Lagebericht 2016, S. 8.

14 BSI, Lagebericht 2019, S. 7.

15 BSI, Lagebericht 2019, S. 11.

16 *Wischmeyer*, VERW 2017, 155 (155).

17 *Auer-Reinsdorff/Conrad/Conrad*, § 33, Rn. 9 f.; *Brisch/Rexin*, CR 2019, 606 (608); *Schneider*, MMR 2019, 485 (485).

18 *Raue*, CR 2018, 277 (277).

19 *Raue*, CR 2018, 277 (277); *Schneider*, MMR 2019, 485 (486).

lerfreie Herstellung möglich ist. Eine Ursache könnte darin liegen, dass die Haftung für Schwachstellen lange Zeit nicht im Fokus der Rechtsanwendung stand.²⁰ In den letzten Jahren hat die Rechtswissenschaft jedoch immer mehr die Dringlichkeit einer klaren Haftungszuweisung für Schwachstellen erkannt.²¹ Auch die Politik hat inzwischen einen entsprechenden Handlungsbedarf gesehen. So teilte die EU-Kommission in ihrem Bericht über die Gestaltung der digitalen Zukunft Europas mit, dass sie einen Anpassungsbedarf bei der wirksamen Anwendung und Durchsetzung von Haftungsvorschriften sieht.²²

Damit ist fraglich, ob das aktuelle Haftungsrecht den Anforderungen an eine immer vernetztere Welt, in der es maßgeblich auf die Sicherheit und Funktionsfähigkeit von Software ankommt, gewachsen ist. Der unzureichende Schutz marktüblicher Software spricht auf den ersten Blick dagegen.²³ Es bedarf jedoch einer eingehenden Analyse, um eine angemessene Antwort auf diese Frage zu finden.

Die Gefahren, die von Schwachstellen ausgehen, sind mannigfaltig. In den letzten Jahren sind insbesondere Datendiebstähle in den Fokus der Medien gerückt. Neben einem potentiellen wirtschaftlichen Schaden des bestohlenen Unternehmens geht ein Datendiebstahl stets mit einem schmerzlichen Vertrauensverlust der Nutzer einher, der auch das Firmenimage des Bestohlenen schädigt.²⁴ Cyberkriminalität stellt jedoch nicht bloß ein Risiko für die jeweils Betroffenen dar, sondern auch für die Gesellschaft insgesamt.²⁵ Es steht zu befürchten, dass die Gesellschaft langfristig ihr Vertrauen in Software und digitale Technologien verliert. In der Folge könnte man die Potentiale und Möglichkeiten dieser Technologien nicht mehr voll ausschöpfen, wodurch ein unkalkulierbarer gesamtgesellschaftlicher Schaden entstünde.²⁶ Schwachstellen und die Frage der Haftung für diese sind somit gesamtgesellschaftliche, d. h. deliktsrechtliche Probleme. Daher konzentriert sich diese Arbeit auf die deliktsrechtliche Produzentenhaftung der Softwareproduzenten.

20 *Chong*, in: *Cyber Insecurity*, 69 (69).

21 *Auer-Reinsdorff/Conrad/Conrad*, § 33, Rdnr. 2 f.

22 *EU-Kommission*, COM (2020) 67 final, S. 8;

So auch: *Jüngling*, MMR 2020, 440 (443).

23 *BSI*, Lagebericht 2019, S. 7 f.; *Westermann/Witt*, S. 32; *Brisch/Rexin*, CR 2019, 606 (607 f.); *Hackenjös/Mechler/Rill*, DuD 2018, 286 (287); *Wischmeyer*, VERW 2017, 155 (161).

24 *Schneiderei*, S. 30.

25 *Westermann/Witt*, S. 32; *Brisch/Rexin*, CR 2019, 606 (607).

26 *Brisch/Rexin*, CR 2019, 606 (607); *Westermann/Witt*, S. 32.

III. Die Herstellung fehlerfreier Software unter ökonomischen Gesichtspunkten

Die Produzentenhaftung besitzt in Bezug auf Software zudem besondere Relevanz, da die Hacker selbst für den Geschädigten oftmals kaum greifbar sind.²⁷ So verbleibt der Softwareproduzent, der die Attacke durch die Schwachstelle in seiner Software erst ermöglicht hat, für den Geschädigten als einziger Anspruchsgegner.

Die Bedeutung der Produzentenhaftung steigt darüber hinaus dadurch, dass die IT-Strukturen immer enger verflochten sind und die Lieferketten von Produkten immer komplexer werden. Als Konsequenz greift der Nutzer immer häufiger auf Softwareanwendungen zurück, mit deren Produzenten sie keine direkte Vertragsbeziehung pflegen.²⁸ Dementsprechend haben die Nutzer immer häufiger keine vertraglichen Haftungsansprüche gegen die Softwareproduzenten. Aber selbst dann, wenn ein Nutzer direkt mit dem Softwareproduzenten kontraktiert hat, werden vertragliche Haftungsansprüche oftmals aufgrund der begrenzten Gewährleistungsfrist von zwei Jahren ausscheiden. Dem Nutzer verbleiben sodann erneut nur deliktische Ansprüche.

Aus diesen Gründen ist die Frage der Haftung für Schwachstellen eine der Hauptherausforderungen der digitalen Revolution.²⁹ Außerdem stellt sie ein gesamtgesellschaftliches Thema dar, welches *inter partes* immer öfter deliktsrechtlich und nicht vertragsrechtlich gelöst werden muss.

III. Die Herstellung fehlerfreier Software unter ökonomischen Gesichtspunkten

Das Produzentenhaftungsrecht basiert auf der ökonomischen Annahme, dass ein haftungsauslösender Fehler anzunehmen ist, sobald eine alternative Konstruktion das Produkt in einem Maße sicherer gemacht hätte, welches die Mehrkosten der Maßnahme überwiegt.³⁰ Folglich kann die Produzentenhaftung von Softwareherstellern greifen, wenn eine Software Schwachstellen aufweist.³¹

Die Begründung einer deliktsrechtlichen Haftung von Softwareproduzenten für Schwachstellen kann nicht nur für einzelne Nutzer von Vorteil sein, sondern kann darüber hinaus auch wesentlichen Einfluss auf die wei-

27 *Schneiderei*, S. 30.

28 *Vásquez/Kroschwald*, MMR 217 (218 f.).

29 *Sassenberg/Faber/Henseler-Unger*, Teil 1, Rdnr. 18 f.

30 *Oppermann/Stender-Vorwachs*, S. 182, Rdnr. 10; *Wagner*, AcP 217 2017, 707 (732).

31 *Oppermann/Stender-Vorwachs*, S. 181, Rdnr. 9; *Wagner*, AcP 217 2017, 707 (726 f.).

tere Entwicklung der gesamten Softwareindustrie nehmen.³² Durch die größere Anzahl an potentiell Anspruchsberechtigten kann das deliktsrechtliche Haftungsrisiko der Softwareproduzenten wesentlich höher ausfallen als das Vertragliche. Im Gegensatz zu diesem besteht das deliktsrechtliche Haftungsregime nämlich nicht lediglich *inter partes*, sondern vielmehr multilateral, also Produzent zu einer unendlichen Anzahl Geschädigter.³³

In der Klärung der produzentenrechtlichen Haftungsfrage für Schwachstellen liegt sogleich eine weitere Antwort, nämlich die, wie die bewusste Entscheidung von Softwareproduzenten, keine fehlerfreie Software zu entwickeln, rechtlich und ökonomisch aufgefangen wird. Obwohl die These, dass Software nicht fehlerfrei programmiert werden kann, heutzutage schlichtweg falsch ist³⁴, erweckt der heutige Softwaremarkt einen anderen Eindruck. In der Praxis ist Software weiterhin oft fehlerbehaftet.

Sowohl die relative als auch die absolute Fehlerzahl im Code wird zukünftig nur dann sinken, wenn die Softwareproduzenten einen Anreiz zur Senkung der Fehleranzahl verspüren. Unter ökonomischen Gesichtspunkten bedeutet das, dass nur dann mit einer Senkung der Fehleranzahl zu rechnen ist, wenn es teurer ist Fehler zu machen als Fehler zu vermeiden. Genau an diesem Punkt kommt die Produzentenhaftung ins Spiel. Sie kann durch ihre Haftungsregeln Schwachstellen für Softwareproduzenten verteuern und so einen rechtlich-ökonomischen Anreiz für eine fehlerfreiere Softwareprogrammierung schaffen.³⁵

Die Produzentenhaftung trägt dem ökonomisch-juristischen Gedanken der Effizienz des Rechts Rechnung. Nach diesem sollte aus volkswirtschaftlichen Erwägungen derjenige das Risiko tragen, der es begründet, am günstigsten beheben und am besten vorhersehen kann. Mit anderen Worten soll der *cheapest risk / cost avoider* das Risiko tragen.³⁶ Dieser kann die Kosten der Risikotragung effizient auf seine Kunden umlegen. Zudem kann er sich gegen das Haftungsrisiko am effizientesten versichern, weshalb der *cheapest cost avoider* auch teilweise als *cheapest insurer* bezeichnet

32 *Döpke/Jülicher*, InTeR 2019, 16 (16).

33 *Ebring*, InTeR 2019, 70 (72).

34 *Lenhard*, S. 27; *Sodtalbers*, S. 147, Rdnr. 226.

35 *Schneier*, <https://www.schneier.com/crypto-gram/archives/2002/0415.html#6> (zuletzt abgerufen am 01.08.2020).

36 *Eidenmüller*, S. 402; *Rettenbeck*, S. 85; *Sassenberg/Faber/Wende*, Teil 2 C., Rdnr. 67; *Schäfer/Ott*, S. 252 f.; *Spindler*, Rdnr. 32; *Taeger*, S. 229; *Lehmann*, BB 1993, 1603 (1604); *Lehmann*, NJW 1992, 1721 (1722); *Raue*, CR 2018, 277 (281); *Spindler*, MMR 2008, 7 (8); *Spindler*, CR 2015, 766 (767).

net wird.³⁷ Die Versicherung wiederum kann durch die Höhe ihrer Prämien weitere Anreize dafür schaffen, dass der Softwareproduzent als cheapest cost avoider mehr in die Sicherheit seiner Produkte investiert.³⁸

Die ökonomische Analyse bietet die Möglichkeit, ein effizientes Haftungsrecht zu entwickeln, indem Schutzmaßnahmen danach bewertet und verteilt werden, ob sie einen gesamtgesellschaftlichen Vorteil bringen oder nicht. Durch die Haftungsverpflichtung kann also für den cheapest cost avoider der Anreiz geschaffen werden, Schutzmaßnahmen umzusetzen.³⁹ Dieser Anreiz besteht jedoch nur dann, wenn das Haftungsrisiko höher ist als die Kosten für die Umsetzung der Schutzmaßnahmen. Gleichzeitig ist zu beachten, dass ein zu hohes Haftungsrisiko das Entwicklungs- und Aktivitätsniveau der Produzenten ausbremsen und so das Gesamtwohl beeinträchtigen kann.⁴⁰ Nach der Learned-Hand-Formel ist das Sorgfaltsniveau, bei dessen Unterschreiten der Produzent haftet, stets so zu wählen, dass die Kosten weiterer Sorgfaltsmaßnahmen über dem Erwartungswert liegen, der sich aus der Schadenshöhe (S) multipliziert mit der Schadenswahrscheinlichkeit (q) berechnen lässt.⁴¹

Zwar wird die ökonomische Analyse des Rechts in den kontinentaleuropäischen Rechtsordnungen eher als eine Gesetzgebungstheorie eingeordnet, weil nur der Gesetzgeber in einer Demokratie ausreichend legitimiert sei, um sie zu einem Rechtsprinzip zu erheben.⁴² Aber dennoch kann sie auch für die Ebene der Rechtsanwendung fruchtbar gemacht werden. Das gilt insbesondere dann, wenn es um die Anwendung von Rechtsvorschriften geht, die bereits den Effizienzgedanken in sich tragen. Das beste Beispiel hierfür ist die richterliche Rechtsfortbildung der Produzentenhaftung, die auf der deliktischen Haftungsklausel des § 823 Abs. 1 BGB beruht. Sowohl die Haftungsvorschrift selbst als auch die Fortbildung der Produzentenhaftung folgen diesem Effizienzgedanken.⁴³

Die ökonomische Analyse des Rechts geht davon aus, dass der Mensch als *homo oeconomicus* auch auf Rechtsnormen nutzenmaximierend für sich

37 Eidenmüller, S. 402; Schäfer/Ott, S. 252 f.; Taeger, S. 229; Etzkorn, MMR 2020, 360 (364); Lehmann, BB 1993, 1603 (1604); Lehmann, NJW 1992, 1721 (1722); Raue, CR 2018, 277 (281).

38 Schneier, Liability and Security, <https://www.schneier.com/crypto-gram/archives/2002/0415.html#6> (zuletzt abgerufen am 01.08.2020).

39 Spindler, MMR 2008, 7 (8).

40 Eidenmüller, S. 3.

41 Spindler, MMR 2008, 7 (8).

42 Eidenmüller, S. 395.

43 Eidenmüller, S. 406; Spindler, MMR 2008, 7 (7).

selbst reagiert.⁴⁴ Dies zugrunde gelegt, deutet die aktuelle Situation der Softwareindustrie darauf hin, dass schlichtweg der haftungsrechtliche Anreiz für Softwareproduzenten nicht hoch genug ist, um fehlerfreie Software zu entwickeln. Die Ursache hierfür könnte darin liegen, dass die Regelungen zur Produzentenhaftung gegenüber Softwareproduzenten nicht schlagkräftig genug sind, um Haftungsanreize auszulösen. Sollte das tatsächlich der Fall sein, so ist im Softwarebereich nicht weniger als die Effizienz des Rechts gefährdet.

IV. Verkehrssicherungspflichten – Kernstück der Produzentenhaftung

Die Ursache für die Gefährdung der Effizienz des Rechts im Softwarebereich ist in den Verkehrssicherungspflichten der Softwareproduzenten zu suchen. Die Verkehrssicherungspflichten stellen das Kernstück der Produzentenhaftung dar. Sie sind die Schutzmaßnahmen, die so verteilt werden müssen, dass ein gesamtgesellschaftlicher Vorteil erzielt wird.

Zur produzentenrechtlichen Haftungsbegründung ist u. a. erforderlich, dass ein Produzent gegen seine Verkehrssicherungspflichten verstößt. Um einen solchen Verstoß jedoch überhaupt feststellen zu können, müssen zwangsläufig zunächst die konkreten Verkehrssicherungspflichten des Softwareproduzenten bestimmt werden. Genau dieser Schritt ist in Bezug auf Software jedoch höchst problematisch. Das ist u. a. deswegen der Fall, weil es bisher nur wenige technische Standards zur Softwareentwicklung gibt. Daher bildet die Untersuchung der Verkehrssicherungspflichten der Softwareproduzenten den Schwerpunkt dieser Arbeit – insbesondere die Frage, welche Pflichten bestehen, wenn es um die anfängliche Vermeidung und die nachträgliche Behebung von Schwachstellen in einer Software geht.⁴⁵

Sind die Verkehrssicherungspflichten für Softwareproduzenten nicht sicher zu bestimmen, so sind sowohl die Rechtsdurchsetzung als auch die Rechtswirkung der Produzentenhaftung gefährdet. Mit anderen Worten ist das Produzentenrecht in Bezug auf Software nicht mehr effizient und der ihr immanente Gedanke der Ökonomie des Rechts gefährdet.

In diesem Zusammenhang stellt sich daher ebenfalls die Frage, ob bereits ein Punkt erreicht ist, an dem sich das Recht *de lege ferenda* anpassen

⁴⁴ Eidenmüller, S. 3, 28 f.

⁴⁵ Chong, in: Cyber Insecurity, 69 (72).

muss, oder ob auch noch Entwicklungsmöglichkeiten *de lege lata* bestehen, um die Haftung der Softwareproduzenten effizient zu gestalten.

Dem Leser dieser Arbeit sollte stets bewusst sein, dass sich die Sicherheit von Software nie absolut erreichen lässt, wenn man den Effizienzgedanken zugrunde legt. Sie kann lediglich relativ erhöht werden, indem durch die Sicherheitsmaßnahmen die Kosten der Angriffe auf Software den Nutzen solcher Angriffe übersteigen.⁴⁶

Nach einer kurzen Begriffsdefinition soll im Folgenden zunächst die Entwicklung der Produzentenhaftung und die Ausgestaltung der allgemeinen Verkehrssicherungspflichten untersucht und dargestellt werden. Sodann richtet sich der Fokus der Arbeit auf die Besonderheiten der Bestimmung der Verkehrssicherungspflichten im Softwarebereich. Daran anschließend wird der Versuch unternommen, die Verkehrssicherungspflichten der Softwareproduzenten für Schwachstellen so konkret wie möglich zu bestimmen. Zum Schluss soll sodann die Frage diskutiert werden, ob die Einführung einer IT-Sicherheit-Grundverordnung bereits notwendig ist, oder ob ein Ansatz *de lege lata* ausreichend ist, um die Produzentenhaftung im Softwarebereich schlagkräftig auszugestalten.

46 Gaycken/Karger, MMR 2011, 3 (4).

B. Begriffsdefinitionen

Will man sich der Problematik der produzentenrechtlichen Verkehrssicherungspflichten von Softwareproduzenten annähern, so gilt es zunächst, die Bedeutung einzelner Begrifflichkeiten zu bestimmen. Denn obwohl die Fragestellung thematisch in der deliktischen Haftung angesiedelt ist, zählt sie tatsächlich und technisch zu den jüngeren Problemen der Rechtswissenschaft. Daher ist es umso wichtiger, dass eine gemeinsame Verständnisgrundlage für Leser und Verfasser geschaffen wird. Die folgenden Begriffsdefinitionen beanspruchen dabei keine allgemeine Geltung in dem Themenfeld der Informationstechnologie. Vielmehr handelt es sich um eine Klarstellung bestimmter Begrifflichkeiten, die lediglich Geltung für den Umfang dieser Arbeit beansprucht. Sie sollen helfen, Missverständnisse zwischen Leser und Verfasser zu vermeiden.

1. Software und Softwareproduzent

Zunächst ist für das Verständnis dieser Arbeit festzuhalten, dass der Begriff „Software“ im Sinne der ISO/IEC-Norm 24765 genutzt wird. Software ist demnach „[...] ein Programm oder eine Menge von Programmen, die dazu dienen, einen Computer zu betreiben“.⁴⁷ Als „Computerprogramm“ wird wiederum „Eine Folge von Befehlen, die nach Aufnahme in einen maschinenlesbaren Träger fähig sind zu bewirken, dass eine Maschine mit informationsverarbeitenden Fähigkeiten eine bestimmte Funktion oder Aufgabe oder ein bestimmtes Ergebnis anzeigt, ausführt oder erzielt“⁴⁸ verstanden. Der „Softwareproduzent“ im Sinne dieser Arbeit ist derjenige, der eine Software in eigener Verantwortung programmiert.⁴⁹

47 ISO/IEC-Norm 24765:2017.

48 Aus „Mustervorschriften für den Schutz von Computersoftware“ der WIPO, GRUR Int. 1978, 286-291 (290);

So auch: BGH, Beschluss vom 02.05.1985, Az. I ZB 8/84, GRUR 1985, 1055; *Engel*, CR 1986, 702 (706); *Sodtalbers*, S. 25, Rdnr. 6; *Etz Korn*, MMR 2020, 360 (361).

49 *Sodtalbers*, S. 329, Rdnr. 526.

II. Daten

Des Weiteren ist zu untersuchen und festzustellen, was unter dem Begriff der „Daten“ zu verstehen ist. Dabei gilt es sich bewusst zu machen, dass einzelne Begriffe verschiedene Bedeutungen in sich tragen können, je nachdem, in welchem Kontext sie genutzt werden. Vorliegend soll der Begriff der Daten in einem rechtlich-technischen Kontext bestimmt werden.

Die Begrifflichkeit der Daten ist in der DIN 44300 definiert worden als ein Gebilde aus Zeichen oder kontinuierlichen Funktionen, die aufgrund bekannter oder unterstellter Abmachungen Informationen darstellen, vorrangig zum Zwecke der Verarbeitung oder als deren Ergebnis.⁵⁰ Diese Definition wurde in leichter Abänderung auch in die VDI-Richtlinie 5002 übernommen. Hiernach sind Daten Zeichenfolgen, die in codierter Form mittels eines Systems zur Informationsverarbeitung gespeichert, dargestellt, übermittelt und ausgeführt werden können.⁵¹ Daten selbst kommt keine Steuerungsfunktion zu. Ihre Funktion liegt in der Bereitstellung von Ordnungs- und Mengeninformationen als Bezugsobjekt einer Datenverarbeitung.⁵²

Einfacher ausgedrückt kann man Daten als Zeichenfolgen, die von einem Computer verarbeitet werden, beschreiben.⁵³ Dabei sind Daten ein Träger dessen, was mittels naturgesetzlicher Strukturen beschreibbar ist.⁵⁴ So kann man Daten als Informationen in einem Zustand der (wertfreien)

50 Arbeitsgruppe „Digitaler Neustart“, S. 29.

51 Kilian/Heussen/Herchenbach-Canarius/Sommer, Kap. 1, Rdnr. 3; Lenckner/Winkelbauer, CR 1986, 483 (484); Welp, IuR 1988, 443 (444); Zech, CR 2015, 137 (138); Dierstein, <http://www.bayer.in.tum.de/lehre/WS2003/ITS-dierstein/DefDV03.pdf> (zuletzt abgerufen am 01.08.2020); <https://www.datenschutzbeauftragter-info.de/definition-und-unterscheidung-der-begriffe-daten-informationen-wissen/> (zuletzt abgerufen am 01.08.2020).

52 Marly, Rdnr. 25 f.; Becker, GRUR 2017, 346 (347); Dierstein, <http://www.bayer.in.tum.de/lehre/WS2003/ITS-dierstein/DefDV03.pdf> (zuletzt abgerufen am 01.08.2020).

53 Sassenberg/Faber/Sattler, Teil 2 A., Rdnr. 3; Börding/Jülicher/Röttgen/v. Schönfeld, CR 2017, 134 (134); <http://www.techfacts.de/ratgeber/was-sind-daten> (zuletzt abgerufen am 01.08.2020).

54 Behling/Abel/Giebichenstein/Schirp/Kölsch, Kap. 12, Rdnr. 6; <https://www.datenschutzbeauftragter-online.de/datenschutz-definition-was-sind-daten/6760/> (zuletzt abgerufen am 01.08.2020).