IT-Sicherheitsforschung und IT-Strafrecht

herausgegeben von

Sebastian Golla und
Dominik Brodowski

Mohr Siebeck

Sebastian Golla, geboren 1988; Juniorprofessor für Kriminologie, Strafrecht und Sicherheitsforschung im digitalen Zeitalter an der Ruhr-Universität Bochum.

Dominik Brodowski, geboren 1980; Universitätsprofessor für Strafrecht und Strafprozessrecht an der Universität des Saarlandes, Saarbrücken.
orcid.org/0000-0002-3711-4197

Die Veröffentlichung wurde finanziell gefördert durch das Center for Advanced Internet Studies (CAIS).

CAIS RESEARCH FOR THE DIGITAL AGE

ISBN 978-3-16-162179-6 / eISBN 978-3-16-162184-0
DOI 10.1628/978-3-16-162184-0

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliographische Daten sind im Internet über *https://dnb.de* abrufbar.

Das Buch wurde von Textservice Zink in Schwarzach gesetzt und von Beltz Grafische Betriebe in Bad Langensalza auf alterungsbeständiges Werkdruckpapier gedruckt und gebunden.

Printed in Germany.

Das Urheberrecht als Grenze der IT-Sicherheitsforschung

Linda Kuschel/Darius Rostam

Computerprogramme sind ein zentrales Objekt von IT-Sicherheitsforschung, zugleich aber auch Schutzgegenstand des Urheberrechts. Dass bestimmte Nutzungen deshalb Rechtsinhabern vorbehalten sind, kann IT-Sicherheitsforschung beeinträchtigen. Wo Konflikte zwischen Urheberrecht und IT-Sicherheitsforschung auftreten und wie sie unter dem geltenden Recht aufgelöst werden können, ist Gegenstand dieses Beitrags.

I. Einleitung

Das Urheberrecht soll durch sein Schutzversprechen literarische, wissenschaftliche und künstlerische Leistungen fördern. Indem der Schutz auf Ausdruck und konkrete Gestaltung begrenzt ist, wird verhindert, dass Ideen, Theorien und Informationen monopolisiert und der wissenschaftliche und gesellschaftliche Fortschritt behindert werden. Auch für Computerprogramme gilt: „Ideen und Grundsätze, die einem Element eines Computerprogramms zugrunde liegen […] sind nicht geschützt" (§ 69a Abs. 2 S. 2 UrhG). Und dennoch können sich aus dem Urheberrecht für Computerprogramme spürbare Einschränkungen für die IT-Sicherheitsforschung ergeben, weil das Testen und Untersuchen eines Computerprogramms häufig mit urheberrechtlich relevanten, teilweise strafbewehrten (vgl. §§ 106, 108a, 108b UrhG) Handlungen einhergehen. Es ist Kernaufgabe des Urheberrechts, die Interessen von Rechtsinhabern mit den Interessen der Allgemeinheit bestmöglich in Ausgleich zu bringen. Auch IT-Sicherheit ist ein solches Allgemeininteresse und muss daher in den Interessenausgleich einbezogen werden. Der Beitrag zeigt auf, wie das unter geltendem Recht geschieht. Zunächst werden die Besonderheiten von Computerprogrammen als Gegenstand des Urheberrechts und Objekt der IT-Sicherheitsforschung dargestellt (II.) und konkrete Maßnahmen, die in der IT-Sicherheitsforschung eingesetzt werden, urheberrechtlich eingeordnet (III.). Von entscheidender Bedeutung ist, unter welchen Umständen Maßnahmen der IT-Sicherheitsforschung durch urheberrechtliche Schranken privilegiert sind (IV.). Daneben haben sich in der Praxis ver-

schiedene Mechanismen etabliert, die negativen Auswirkungen des urheberrechtlichen Schutzes auf vertraglichem Wege begegnen (V.).

II. Gegenstand des Urheberrechts

Dass die Regelungen des Urheberrechts für die IT-Sicherheitsforschung überhaupt relevant sind, liegt vor allem daran, dass essentieller Bestandteil jeder IT-Infrastruktur Computerprogramme sind.[1] Sie sind urheberrechtlich geschützt, „wenn sie individuelle Werke in dem Sinne darstellen, daß sie das Ergebnis der eigenen geistigen Schöpfung ihres Urhebers sind" (§ 69a Abs. 3 S. 1 UrhG). Weitere Voraussetzungen, wie etwa ein ästhetischer Gehalt oder eine überragende handwerkliche Leistung des Programmierers[2], bestehen dafür nicht. Es gelten folglich keine höheren Schutzanforderungen als bei anderen Werkarten, so dass auch bei Computerprogrammen bereits die „kleine Münze", also jedes nicht ganz banale Programm, potentiell urheberrechtlich geschützt ist.[3] Und doch unterscheiden sich Computerprogramme in einigen Punkten wesentlich von anderen Schutzgegenständen: Sie sind nicht für die menschliche Wahrnehmung bestimmt, sondern schöpfen ihren Wert aus den Funktionen, die sie Computern im Wege einer Abfolge von Befehlen vermitteln. Zwar werden Computerprogramme in einer für Menschen verständlichen Programmiersprache als sog. Quellcode verfasst, sie müssen dann aber regelmäßig kompiliert, also in eine maschinenlesbare Form, den sog. Objektcode (oder: Maschinencode), übersetzt werden.[4] Obwohl dieser Objektcode für Menschen unverständlich ist, ist auch er geschützt, da sich das Urheber-

[1] Daneben können auch andere Werkarten betroffen sein, wenn etwa das Computerprogramm urheberrechtlich geschützte Grafiken oder Musikdateien enthält, vgl. hierzu *Kuschel/Rostam*, in: Ebers/Steinrötter (Hrsg.), Künstliche Intelligenz und smarte Robotik im IT-Sicherheitsrecht, 2021, S. 361, 362 f.

[2] So noch BGHZ 94, 276, 286, worin seinerzeit manche eine „grundsätzliche Bejahung und praktische Verneinung des Urheberrechtsschutzes für Computerprogramme" sahen, *Bauer* CR 1985, 5, 10. Mit Umsetzung der Computerprogramme-Richtlinie (2009/24/EG) ist dieses Erfordernis weggefallen.

[3] *BGH* GRUR 2013, 509, 510, Rn. 24; vgl. *Schack*, Urheber- und Urhebervertragsrecht, 10. Aufl. 2021, Rn. 208; Eingehend hierzu *Böcker*, Computerprogramme zwischen Werk und Erfindung, 2009, S. 144 ff.

[4] Teilweise werden Programme auch nur in eine maschinennahe „low-level" Zwischensprache kompiliert oder müssen gar nicht kompiliert werden, sondern werden im Augenblick der Ausführung von einem weiteren Programm interpretiert, vgl. *Franzen/Maier/Wagner* DuD 2020, 511, 513; *Böcker* (Fn. 3), S. 45 f.; *Ernst*, in: Hoeren/Sieber/Holznagel (Hrsg.), Handbuch Multimedia-Recht, 58. Aufl. 2022, Teil 7.1 Grundlagen des Urheberrechts Rn. 11.

recht auf Computerprogramme in jeder Ausdrucksform erstreckt (§ 69a Abs. 2 S. 1 UrhG).[5] Dadurch entsteht allerdings eine gewisse Diskrepanz zum urheberrechtlichen Grundsatz der Dichotomie von Ausdrucksform und Idee:[6] Allein erstere ist vom Schutz erfasst, während die in ihr enthaltene Idee – zumindest urheberrechtlich – stets frei bleibt. Auch die Ideen und Grundsätze, die in einem Computerprogramm niedergelegt sind, sind prinzipiell frei (§ 69a Abs. 2 S. 2 UrhG), soweit jedoch nur der Objektcode zur Verfügung steht, sind sie dem menschlichen Verständnis entzogen. Die Überprüfung des Programms auf Sicherheitslücken oder Fehler ist allein auf Grundlage des Objektcodes nur bedingt, deren Berichtigung gar nicht möglich. Anders als bei allen anderen Werkarten kann bei Computerprogrammen also die Nutzung und der Zugang zu den ungeschützten Ideen und Grundsätzen eines Programms eine urheberrechtlich geschützte Handlung erfordern, nämlich die Übersetzung des Objektcodes in eine für Menschen lesbare Form. Diese Besonderheiten von Computerprogrammen gilt es bei der Auslegung und Anwendung der urheberrechtlichen Regelungen zu berücksichtigen.[7]

III. Urheberrechtlich relevante Handlungen in der IT-Sicherheitsforschung

In der IT-Sicherheitsforschung kommen verschiedene Instrumente zur Untersuchung eines Computerprogramms auf Sicherheitslücken oder Fehler zum Einsatz. Beim sog. Black-Box-Testing etwa wird beobachtet und analysiert, wie ein Programm auf die Eingabe verschiedener Befehle

[5] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 36; *Schack* (Fn. 3), Rn. 209; *EuGH*, Urteil v. 2.5.2012 – C 406/10, *SAS Institute*, ECLI:EU:C:2012:259, Rn. 37 f.; *EuGH*, Urteil v. 22.12.2012 – C 393/09, *BSA*, ECLI:EU:C:2010:816, Rn. 33–35; *OLG Frankfurt a.M.* GRUR 2015, 784, 785; *Kaboth/Spies*, in: Ahlberg/Götting/Lauber-Rönsberg (Hrsg.), BeckOK UrhR, 35. Ed. 15.1.2022, § 69a Rn. 5; *Dreier*, in: Dreier/Schulze (Hrsg.), Urheberrechtsgesetz, 7. Auflage 2022, § 69a Rn. 19. Theoretisch ist freilich nicht ausgeschlossen, dass Fachleute bestimmten Objektcode auch verstehen können.

[6] Siehe hierzu *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 7 m.w.N. Siehe hierzu auch *Böcker* (Fn. 3), S. 136 ff.; *Grützmacher*, in: Wandtke/Bullinger (Hrsg.), Urheberrecht, 6. Aufl. 2022, § 69a Rn. 23, 28–33 sowie zum Hintergrund dieser Dichotomie und der Rolle der Gerichte bei der Grenzziehung zwischen Ausdruck und Idee *Spindler*, in: Schricker/Loewenheim (Hrsg.), Urheberrecht, 6. Aufl. 2020, § 69a Rn. 8 f.; Siehe zudem Begr. RegE BT-Drucks. 12/40224, 9.

[7] Ähnlich auch *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193 Rn. 8.

hin abläuft.[8] Die internen Strukturen der Software müssen dabei nicht offengelegt werden. Stattdessen erlauben der Programmablauf und seine Ergebnisse Rückschlüsse auf die Grundsätze des Programms oder decken bereits Fehlfunktionen auf. Auch das System Monitoring, bei dem die Kommunikation zwischen verschiedenen Programmen beobachtet und analysiert wird, ist von passiver Natur.[9] In beiden Fällen wird das zu untersuchende Programm lediglich im Zuge des Programmablaufs im Arbeitsspeicher vervielfältigt.[10]

Deutlich effektiver ist IT-Sicherheitsforschung, wenn sie auf Ebene des Softwarecodes selbst und nicht lediglich bei dessen Ausführung ansetzt. Um eine entsprechende Analyse des Softwarecodes vornehmen zu können, muss dieser in ein lesbares Format transferiert werden. Das kann im Wege der Dekompilierung, der Disassemblierung oder des Binary Lifting geschehen. Bei der Dekompilierung wird der Objektcode in einen Quellcode zurückübersetzt.[11] Das Ergebnis dieser Übersetzung ist allerdings nicht mit dem Originalquellcode des Programms identisch, weshalb bei einer Rückübersetzung auch von „Quasi-Quellcode" gesprochen wird.[12] Das Programm liegt danach also in einer weiteren Version vor. Für die Disassemblierung wird der nur maschinenlesbare Objektcode mithilfe eines Disassemblers interpretiert und als für Fachleute lesbares Assemblerprogramm ausgegeben.[13] Eine weitere Methode, die in der IT-Sicherheitsforschung eingesetzt wird, ist das Binary Lifting. Hier werden die im Objektcode vorhandenen Befehle „herausgehoben" und in eine für Menschen verständliche, aber immer noch maschinennahe (*low-level*) Sprache gebracht.[14]

---

[8] *Hoeren/Pinelli* CR 2019, 410, 411; *Schweyer*, Die rechtliche Bewertung des Reverse Engineering in Deutschland und den USA, 2012, S. 74; *Triebe* WRP 2018, 795, 796 Rn. 6; *Müller-Hengstenberger/Kirn* CR 2008, 755, 759.

[9] *Schweyer* (Fn. 8), S. 75.

[10] *Schweyer* (Fn. 8), S. 88 ff.

[11] Oder: „Rückentwicklung" ähnlich einem „reverse engineering", vgl. *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 40. Vgl. auch *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69e Rn. 4.

[12] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 37; *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 41; *Schmidt*, in: Auer-Reinsdorff/Conrad (Hrsg.), Handbuch IT- und Datenschutzrecht, 3. Aufl. 2019, § 1 Rn. 204; *Schneider*, in: Schneider (Hrsg.), Handbuch EDV-Recht, 5. Aufl. 2017, G. Urheberschutz für Software, Rn. 340.

[13] Vgl. *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69e Rn. 6; *Franzen/Maier/Wagner* DuD 2020, 511, 513.

[14] Vgl. hierzu *Franzen/Maier/Wagner* DuD 2020, 511, 513.

Dekompilierung, Disassemblierung und Binary Lifting gehen mit einer Vervielfältigung des Objektcodes im Arbeitsspeicher einher, die die Rechte des Urhebers gem. § 69c Nr. 1 UrhG berührt.[15] Darüber hinaus wird zumindest bei der Dekompilierung auch eine Übersetzung des Objektcodes in einen (neuen) Quellcode und dessen Vervielfältigung vorgenommen. Damit ist das Umarbeitungsrecht des Urhebers gem. § 69c Nr. 2 UrhG betroffen.[16] Bei Disassemblierung und Binary Lifting kommt es auf die Umstände des Einzelfalls an: Eine Umarbeitung setzt voraus, dass wesentliche Züge des Originalprogramms übernommen werden, wobei auch die Übernahme der Programmstruktur, des Programmablaufs sowie von Modulen oder Befehlsgruppen genügt.[17] Wenn also bei einer Disassemblierung oder einem Binary Lifting der gesamte Objektcode oder sämtliche darin enthaltenen Befehlsabfolgen übersetzt und in der neuen Version wiedergegeben werden, sind die Übernahmen so wesentlich, dass von einer Umarbeitung auszugehen ist. Werden hingegen nur einzelne Befehlssätze aufgegriffen und dargestellt, ist unwahrscheinlich, dass wesentliche individuelle Züge des Computerprogramms in der neuen Version übernommen wurden und eine Umarbeitung nach § 69c Nr. 2 UrhG scheidet aus. Das Umarbeitungsrecht für Computerprogramme ist bereits im Zeitpunkt der Herstellung einer veränderten Version betroffen und greift – anders als das Bearbeitungsrecht der Urheber anderer Werkarten nach § 23 UrhG – nicht erst mit ihrer Veröffentlichung.

Im Übrigen geht IT-Sicherheitsforschung häufig auch mit einer Veröffentlichung der Ergebnisse und des untersuchten Quellcodes einher. Idealerweise beherzigt eine solche Veröffentlichung die Grundsätze der Responsible Disclosure bzw. der Coordinated Vulnerability Disclosure[18] – aus urheberrechtlicher Perspektive stellt eine Veröffentlichung des Quellcodes allerdings eine öffentliche Zugänglichmachung nach § 69c Nr. 4 UrhG (bei Veröffentlichung im Internet) oder eine Verbreitung i.S.v. § 69c Nr. 3 UrhG (bei Veröffentlichung in einem Print-Medium) dar.

---

[15] *Schweyer* (Fn. 8), S. 88 ff.; vgl. *Hoeren/Pinelli* CR 2019, 410, 411; *Imhof*, in: Bisges (Hrsg.), Handbuch Urheberrecht, 1. Aufl. 2016, Rn. 202; *Kuschel/Rostam*, in: Ebers/Steinrötter (Fn. 1), S. 361, 367; vgl. *Strobel*, Reverse Engineering im Spannungsfeld der Sonderschutzrechte, 2022, S. 156–158; für die Dekompilierung siehe auch *Fiedler*, Der Computerprogrammschutz und die Schutzrechtskumulation von Urheber- und Patentrecht, 2013, S. 204 sowie *Wiebe* JIPITEC 2011, 89, 90 Rn. 9.

[16] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 39; *Vettermann/Wagner* InTer 2020, 126, 127 f.; *Wagner* DuD 2020, 111, 112; *Kuschel/Rostam*, in: Ebers/Steinrötter (Fn. 1), S. 361, 367; *Strobel* (Fn. 15), S. 159 f.

[17] *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69c Rn. 15, 17.

[18] Siehe hierzu *CEPS*, Software Vulnerability Disclosure in Europe, 2018, S. 5 ff.

Schließlich kommt in der IT-Sicherheitsforschung die sog. Code Emulation zum Einsatz, mithilfe derer Computerprogramme außerhalb der für sie eigentlich vorgesehenen Softwareumgebung ausgeführt und beobachtet werden können.[19] Beispielsweise lassen sich so die Betriebssysteme von Mobiltelefonen auf anderen Geräten virtualisieren und auf Schwachstellen überprüfen.[20] Ein Emulator ahmt dazu das Verhalten einer CPU (*central processing unit*) nach und interpretiert die Anweisungen des Objektcodes des jeweiligen Programms, was ausführlichere und spezifischere Analysen und Dokumentationen ermöglicht, als wenn das Programm auf der CPU des Herstellers abläuft.[21] Die Code Emulation setzt eine Vervielfältigung des Computerprogramms in der emulierten Umgebung voraus. Eine Umarbeitung findet demgegenüber nicht statt, denn das Computerprogramm selbst bleibt unverändert, nur die Umgebung ist eine andere.

Ist das jeweilige Computerprogramm mit technischen Schutzmaßnahmen versehen, kann eine Umgehung dieser Maßnahmen erforderlich sein, um das Programm zu untersuchen, insbesondere zu dekompilieren, disassemblieren oder ein Binary Lifting durchzuführen. Technische Schutzmaßnahmen sind bei Computerprogrammen weit verbreitet, insbesondere bei auf Endnutzergeräten gespeicherter proprietärer Software. Sie verhindern den Zugriff auf bestimmte Systembereiche, den Quellcode oder Veränderungen des Codes. Die Umgehung der technischen Schutzmaßnahmen geht zum einen mit einer urheberrechtlich relevanten Umarbeitung gem. § 69c Nr. 2 UrhG einher. Zum anderen ist sie nach § 95a Abs. 1 UrhG ausdrücklich verboten, wenn die Schutzmaßnahme zugleich andere urheberrechtlich geschützte Gegenstände als das Computerprogramm selbst schützt, etwa im Programm enthaltene Grafiken.[22]

IV. Gesetzliche Privilegierungen der IT-Sicherheitsforschung

Nicht jeder Umgang mit einem urheberrechtlichen Schutzgegenstand greift auch in die Ausschließlichkeitsrechte ein. Zum einen sind Handlungen nach § 69d Abs. 1 UrhG vom Ausschließlichkeitsrecht ausgenommen, wenn sie für eine bestimmungsgemäße Benutzung des Computerprogramms notwendig sind (1.). Zum anderen dürfen Berechtigte gemäß

[19] Vgl. hierzu *Franzen/Maier/Wagner* DuD 2020, 511, 513.

[20] Siehe dazu auch das Verfahren von Apple gegen den Virtualisierungsdienstleister Corellium, *Apple Inc. v. Corellium, LLC,* 510 F. Supp. 3d 1269 (S.D. Fla. 2020).

[21] Vgl. hierzu *Franzen/Maier/Wagner* DuD 2020, 511, 513.

[22] *Kuschel/Rostam*, in: Ebers/Steinrötter (Fn. 1), S. 361, 364 ff.

§ 69d Abs. 3 UrhG analysieren, wie ein Programm funktioniert, um die zugrundeliegenden Ideen oder Grundsätze zu ermitteln (2.). Zu klären ist, in welchem Umfang IT-Sicherheitsforschung durch diese Ausnahmen privilegiert ist.

1. Anforderungen des § 69d Abs. 1 UrhG

a) Bestimmungsgemäße Nutzung

§ 69d Abs. 1 UrhG stellt Vervielfältigungen und Umarbeitungen frei, die für die bestimmungsgemäße Nutzung eines Computerprogramms durch einen zur Verwendung Berechtigten notwendig sind. Dazu gehört auch die Fehlerberichtigung. Im Folgenden soll zunächst betrachtet werden, welche Handlungen im Zusammenhang mit IT-Sicherheitsforschung einen bestimmungsgemäßen Gebrauch des Programms darstellen (aa)). Anschließend werden die Fehlerbehebung und Fehlersuche in den Blick genommen (bb)). Schließlich ist das Verhältnis zur Dekompilierung gemäß § 69e UrhG zu klären (cc)).

aa) Für die bestimmungsgemäße Nutzung notwendige Handlungen

Die oben genannten Techniken der IT-Sicherheitsforschung gehen mit Vervielfältigungen und Umarbeitungen von Software einher. § 69d Abs. 1 UrhG erlaubt diese Handlungen unter der Bedingung, dass sie für den bestimmungsgemäßen Gebrauch einer Software erforderlich sind. Zum zulässigen Normalgebrauch gehört etwa, die Software in den Arbeitsspeicher zu laden und sie ablaufen zu lassen.[23] Deshalb sind passive Analysetechniken wie das Black-Box-Testing oder System Monitoring, die sich an den normalen Ablauf einer Software heften, nach § 69d Abs. 1 UrhG zulässig. Techniken wie die Dekompilierung, Disassemblierung, Code Emulation, das Binary Lifting und die Responsible Disclosure führen dagegen zu Vervielfältigungen oder Umarbeitungen, die meist nicht für die bloße bestimmungsgemäße Nutzung notwendig sind.[24] Sie können möglicherweise aber unter die Fehlerbehebung gefasst werden.

bb) Fehlerbehebung

Im Kontext der IT-Sicherheitsforschung stellt sich zunächst die Frage, ob ein mangelndes Sicherheitsniveau ein Fehler im Sinne von § 69d Abs. 1

[23] *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 9.
[24] Zum Begriff der bestimmungsgemäßen Nutzung sogleich unter bb) (1).

UrhG ist (1). Anschließend ist zu klären, welche Maßnahmen zur Fehlerbehebung von § 69d Abs. 1 UrhG gedeckt sind, wobei neben die Fehlerberichtigung (2) die Fehlersuche (3) tritt.

(1) Mangelndes Sicherheitsniveau als Fehler

Nach dem Gesetz zählt zur bestimmungsgemäßen Verwendung auch die Berichtigung von Fehlern. Den Fehlerbegriff hat der *EuGH* für den zugrundeliegenden Art. 5 Abs. 1 Computerprogramm-RL näher bestimmt: Fehler sind danach alle Defekte in einem Computerprogramm, die zu einer Fehlfunktion führen und die Möglichkeit zur bestimmungsgemäßen Benutzung beeinträchtigen.[25] Was einen Fehler konstituiert, steht folglich immer in Bezug dazu, wie das Computerprogramm benutzt werden soll.

Bisher geht die herrschende Ansicht davon aus, dass sich die bestimmungsgemäße Nutzung nach dem Überlassungszweck und sonstigen vertraglichen Umständen zwischen Rechtsinhaber und Erwerber richtet.[26] Nach teilweise vertretener Ansicht soll dazu sogar allein die vom Rechtsinhaber festgelegte Bestimmung entscheidend sein.[27] Folgt man diesem subjektiven Verständnis können nur Beeinträchtigungen des vereinbarten Programmablaufs einen Fehler konstituieren. Damit ein mangelndes IT-Sicherheitsniveau einen berücksichtigungsfähigen Fehler darstellt, müsste es also die vertraglich vorgesehene Benutzung des Programms beeinträchtigen. Eine solche Beeinträchtigung können etwa Viren oder trojanische Pferde darstellen.[28] Sie sind auch dann Fehler, wenn sie im Programm nicht selbst angelegt sind, sondern erst nach Erstellung von außen wirken.[29] Auch wenn Sicherheitslücken einer Software einen rechtskonformen Einsatz nicht mehr erlauben, ist die bestimmungsgemäße Nutzung für gewöhnlich beeinträchtigt.[30] Grenzen für einen konformen Einsatz bilden

[25] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 58 ff.

[26] *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 7; *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 75.

[27] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 75; *Lehmann*, in: Festschrift für Schricker zum 60. Geburtstag, 1995, S. 543, 560, 568; *LG Düsseldorf* CR 1996, 737, 738.

[28] Vgl. *Dreier*, in: Dreier/Schulze (Fn. 5), § 69d Rn. 9; *Kaboth/Spies*, in: BeckOK UrhG (Fn. 5), § 69d Rn. 7; *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 10; *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 21.

[29] *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 10; *Dreier*, in: Dreier/Schulze (Fn. 5), § 69d Rn. 9.

[30] Vgl. *Halder* jurisPR-ITR 6/2022 Anm. 3, D.IV; *Vettermann/Wagner* InTeR 2020, 126, 133; vgl. auch *Bodden/Rasthofer/Richter/Roßnagel* DuD 2013, 720, 725.

etwa Art. 32 Abs. 1 lit. d DSGVO, § 8c BSIG und Sorgfaltspflichten im Rahmen von § 823 Abs. 1 BGB. Kein Fehler ist es hingegen nach dem subjektiven Begriff, wenn eine Software veraltet ist – Bezugspunkt ist nämlich stets der ursprüngliche Bestimmungszweck.[31]

Allerdings könnte auch ein objektiver Bestimmungsbegriff greifen: Der EuGH hat sich in *Top System* gerade nicht dem Vorschlag des Generalanwalts angeschlossen, auf die vom Urheber festgelegte oder zwischen den Parteien bestimmte Vereinbarung abzustellen.[32] Dass er stattdessen nur von einer „Fehlfunktion" spricht, deutet an, dass auch objektive Umstände für die bestimmungsgemäße Nutzung zu berücksichtigen sein können. Das stünde im Einklang mit den jüngsten europäischen Reformen des Mängelgewährleistungsrechts, die an einen objektiven Mangelbegriff anknüpfen und die Sicherheit von Kaufgegenständen zum Leistungsmerkmal erklären (vgl. etwa Art. 8 Abs. 1 lit. b DI-RL, Art. 7 Abs. 1 lit. d WK-RL). Danach begründet ein unübliches und vom Nutzer nicht zu erwartendes Sicherheitsniveau einer Software einen Mangel (vgl. §§ 434 Abs. 3 S. 1 Nr. 2, S. 2, 327e Abs. 3 S. 1 Nr. 2, 475b Abs. 4 Nr. 1 BGB). § 69d Abs. 1 UrhG könnte nun parallel zum kaufrechtlichen Mängelgewährleistungsrecht auszulegen sein:[33] Wenn das Sicherheitsniveau objektiv vertragswidrig ist, besteht danach ein Fehler, der auf Grundlage von § 69d Abs. 1 UrhG berichtigt werden kann. Nach diesem objektiven Verständnis kann auch eine technische Veralterung ein Fehler sein, soweit sie den objektiven Anforderungen widerspricht (vgl. § 327e Abs. 3 S. 1 Nr. 5, 327f, 475b Abs. 4 Nr. 2 BGB). Der Kreis berichtigungsfähiger Fehler erweitert sich dann um Beeinträchtigungen, die in den Vereinbarungen zwischen Hersteller und Nutzer keine Berücksichtigung fanden, Nutzer aber dennoch objektiv im Gebrauch stören. Ein Interesse des Verkäufers, den Fehler selbst zu beheben, kann über das Merkmal der Notwendigkeit Berücksichtigung finden.[34]

(2) Fehlerbeseitigung

Wenn die bestimmungsgemäße Benutzung durch eine Sicherheitslücke beeinträchtigt ist, stellt sich die Frage, welche Handlungsmöglichkeiten das

[31] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 76 f.

[32] Vgl. *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 75.

[33] *Gülker* InTeR 2022, 22, 26; in diese Richtung auch *Schack* (Fn. 3), Rn. 473.

[34] *Gülker* InTeR 2022, 22, 24 f., dazu sogleich unter IV.1.c).

Urheberrecht IT-Sicherheitsforschern gewährt. § 69d Abs. 1 UrhG erlaubt die Vervielfältigung, Übersetzung, Bearbeitung, das Arrangement und andere Umarbeitungen des Computerprogramms, sofern es für die bestimmungsgemäße Benutzung notwendig ist. Dazu gehört ausdrücklich auch die *Fehlerberichtigung*. Erlaubt sind damit Eingriffe in die Software, um Programmfehler zu entfernen oder zu umgehen und Funktionsstörungen zu beseitigen.[35] Soweit IT-Sicherheitslücken einen Fehler begründen, dürfen sie also beseitigt werden. Das kann etwa umfassen, ein zusätzlich erstelltes Softwaremodul einzubauen oder ein Antiviren-Programm einzusetzen.[36] Dagegen gehören Verbesserungen und Anpassungen an veränderte gesetzliche oder technische Anforderungen nach der Literatur nicht zur Fehlerbeseitigung.[37] Das kann nach dem Wortlaut allerdings nur gelten, soweit sie über den bestimmungsgemäßen Gebrauch hinausgehen.[38] Alle Handlungen, die für einen bestimmungsgemäßen Gebrauch notwendig sind, dürfen nämlich auch ohne Zustimmung des Rechtsinhabers erfolgen. Entscheidend für das Fehlerbeseitigungsrecht ist damit, ob eine Sicherheitslücke überhaupt ein Fehler ist – wenn das der Fall ist, sind auch Berichtigungshandlungen erlaubt.

(3) Fehlersuche

IT-Sicherheitsforschung betrifft nicht nur die Beseitigung bekannter Schwachstellen, sondern auch deren Aufdeckung. Sie widmet sich insbesondere Sicherheitslücken, die noch nicht in Erscheinung getreten sind. Infrage steht deshalb, ob nicht nur die Fehler*berichtigung*, sondern auch schon die Fehler*suche* von § 69d Abs. 1 UrhG gedeckt sein kann.

Der Wortlaut der Norm beschränkt sich zwar auf die Fehlerbeseitigung. Der Begriff kann aber auch so verstanden werden, dass die vorgelagerte Suche in der Fehlerbeseitigung als ein Minus enthalten ist. Wer Fehler sogar beseitigen darf, muss sie auch überhaupt erst feststellen können. Denn es können nur Fehler beseitigt werden, die auch bekannt sind.[39] Mit Blick auf die Interessen des Rechtsinhabers liegt deshalb ein Argumentum a fortiori nahe: Wenn § 69d Abs. 1 schon die eingriffsintensivere Feh-

[35] Vgl. *Wiebe*, in: Spindler/Schuster (Hrsg.), Recht der elektronischen Medien, 4. Aufl. 2019, § 69d Rn. 15; *Grützmacher* in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 21.

[36] Vgl. BGH GRUR 2008, 866, 868; *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 10, *Kaboth/Spies*, in: BeckOK UrhG (Fn. 5), § 69d Rn. 7.

[37] Vgl. *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 10; *Kaboth/Spies*, in: BeckOK UrhG (Fn. 5), § 69d Rn. 7; *Dreier*, in: Dreier/Schulze (Fn. 5), Rn. 9.

[38] So auch *Dreier*, in: Dreier/Schulze (Fn. 5), § 69d Rn. 9.

[39] Vgl. *Vettermann/Wagner* InTeR 2020, 126, 128.

lerbeseitigung erlauben soll, muss die bloße Fehlersuche erst recht zulässig sein.[40]

Dafür spricht auch das Telos der Vorschrift. § 69d UrhG soll die besonderen Charakteristika von Software als Schutzgegenstand des Urheberrechts ausgleichen.[41] Bereits der bestimmungsgemäße Gebrauch erfordert eine Vervielfältigung, fällt also unter das Verbotsrecht des Rechtsinhabers (vgl. § 69c Nr. 1 UrhG). Der Rechtsinhaber gewinnt dadurch eine dem Urheberrecht fremde Monopolmacht über die zugrundeliegende Idee.[42] § 69d UrhG soll deshalb die Ausschließlichkeitsrechte zugunsten berechtigter Nutzer beschränken, indem Handlungen für einen bestimmungsgemäßen Gebrauch zustimmungsfrei zulässig sind.[43] Klar ist daher: Wenn ein Fehler sich zeigt und der bestimmungsgemäße Gebrauch bereits beeinträchtigt ist, kann für seine Korrektur keine Erlaubnis erforderlich sein. Denn anderenfalls würden sich technische Besonderheiten von Software gerade „artfremd"[44] zulasten berechtigter Nutzer auswirken. Diese Gefahr besteht aber auch, wenn die Fehler*suche* von der Zustimmung des Rechtsinhabers abhängt. Denn Software hat die Besonderheit, dass ihr Inhalt im Wege einer „einfachen sensorischen Analyse" normalerweise nicht zur Kenntnis genommen werden kann.[45] Computerprogramme können deshalb nicht hinreichend auf Mängel untersucht werden, ohne zugleich in das Ausschließlichkeitsrecht des Urhebers einzugreifen. Wenn sich ein Fehler nun erst bemerkbar machen müsste und nicht proaktiv aufgefunden werden dürfte, würde diese Besonderheit[46] von Software für berechtigte Nutzer Nachteile erzeugen. Nach § 377 Abs. 1, 2, 381 Abs. 2 HGB haben gewerbliche Käufer eine Ware unverzüglich zu untersuchen und über Mängel eine Anzeige zu machen – anderenfalls gilt die Ware als genehmigt.[47] Wenn

[40] *Hoeren/Pinelli* CR 2019, 410, 413; *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 21.

[41] Vgl. *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 1: Begrenzung der „aus den technischen Gegebenheiten von Software resultierenden, dem Urheberrecht aber artfremde (sic) Befugnisse des Rechtsinhabers".

[42] Vgl. *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 1.

[43] Vgl. *Dreier*, in: Dreier/Schulze (Fn. 5), § 69d Rn. 1, *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 1; *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 1.

[44] Vgl. *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 1.

[45] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021: 193, Rn. 6 f.

[46] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021: 193, Rn. 6: „ungewöhnlich".

[47] Vgl. zur Anwendung auf Software *BGH* NJW 2000, 1415.

die Fehlersuche von § 69d Abs. 1 UrhG nicht gedeckt wäre, müssten Käufer darauf warten, dass sich ein Mangel zeigt, bevor sie tätig werden können. Dann können Gewährleistungsansprüche gemäß §§ 434 Abs. 1 Nr. 3, 327j Abs. 1 S. 1, 634a Abs. 1 Nr. 1 BGB allerdings bereits verjährt sein. Nutzer drohen also Mängelrechte zu verlieren, wenn sie die Software nicht auf Fehler überprüfen dürfen.[48]

Außerdem überzeugt die Differenzierung zwischen Fehlerberichtigung und Fehlersuche nicht, weil § 69d Abs. 1 UrhG jede Handlung freistellt, die *für eine bestimmungsgemäße Nutzung notwendig* ist. Die Norm stellt nicht auf einen subjektiven, sondern einen objektiven Fehlerbegriff ab; nicht nur bekannte, sondern auch unbekannte Fehler sind deshalb berichtigungsfähig – entscheidend ist, dass die bestimmungsgemäße Nutzung es erfordert. Unbekannte Fehler wie etwa Sicherheitslücken beeinträchtigen die „normale Benutzung" genauso wie bekannte Programmfehler.[49] Ob Angreifer Fehler der Software im Geheimen ausnutzen können oder das in Kenntnis der Nutzer tun: stets ist die bestimmungsgemäße Benutzung der Software eingeschränkt oder sogar ausgeschlossen. Nach wohl einhelliger Ansicht in der Literatur ist beispielsweise ein berichtigungsfähiger Fehler gegeben, wenn eine Software mit einem Virus oder Trojaner infiziert ist.[50] Der Fehler muss nicht erst hervortreten, indem die Software falsche Ergebnisse erzeugt oder den Dienst einstellt, damit eine Fehlerberichtigung nach § 69d Abs. 1 UrhG zulässig ist. Deshalb ist auch die Fehlersuche erlaubt, wenn sie der Aufdeckung eines noch unbekannten Fehlers dient.

Problematisch ist aber, dass zum Zeitpunkt der Fehlersuche nie sicher ist, ob es einen unbekannten Fehler überhaupt gibt. Denkbar wäre daher, jede Fehlersuche zuzulassen. Dadurch würde sich aber ein Missbrauchspotential eröffnen, wenn unter dem Deckmantel der Fehlersuche ein Reverse Engineering stattfindet. Idealerweise ist eine Fehlersuche und -berichtigung also nur dann erlaubt, wenn ein Fehler vorliegt – aufgrund der Unwissenheit über den Fehler könnte man sich allerdings nie sicher sein, dass diese Voraussetzung erfüllt ist. Deswegen darf die Erlaubnis

[48] *Hoeren/Pinelli* CR 2019, 410, 413. Auch *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 21 sieht deshalb die Fehlersuche umfasst.

[49] Vgl. *Vettermann/Wagner* InTeR 2020, 126, 128; ähnlich *Halder* jurisPR-ITR 6/2022 Anm. 3, D.IV.

[50] *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 10, *Kaboth/Spies*, in: BeckOK UrhG (Fn. 5), § 69d Rn. 7; *Dreier*, in: Dreier/Schulze (Fn. 5), § 69d Rn. 9; *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 21; *Triebe* WRP 2018, 795, 798 Rn. 26.

nicht von einem tatsächlich vorhandenen Fehler abhängen, sondern muss bereits dann eingreifen, wenn *die Umstände des Einzelfalls einen verdeckten Fehler nahelegen*.[51] Das kann z.B. der Fall sein, wenn Sicherheitslücken in einer weit verbreiteten Programmbibliothek bekannt werden, die auch in der vom Berechtigten verwendeten Software eingesetzt sein könnte[52] oder eine besonders angreifbare Methode der Kryptographie implementiert ist.[53] Anlass zur Fehlersuche kann schließlich auch bestehen, wenn bekannt wird, dass der Softwarehersteller in anderen Zusammenhängen fehlerhaft gearbeitet hat.

Dieses Auslegungsergebnis steht auch im Einklang mit Art. 13 und 17 Abs. 2 GrCh. Das durch Art. 17 Abs. 2 GrCh geschützte Partizipationsinteresse des Rechtsinhabers ist realisiert, indem § 69d Abs. 1 UrhG nur zur Verwendung Berechtigte privilegiert und die Norm zum Teil unter dem Vorbehalt besonderer vertraglicher Bestimmungen steht.[54] Die Wissenschaftsfreiheit nach Art. 13 GrCh, der die IT-Sicherheitsforschung ggf. unterfällt, könnte sich demgegenüber nicht entfalten, wenn Forschung grundsätzlich verboten und nur unter dem Vorbehalt der Zustimmung des Rechtsinhabers erlaubt wäre – jedenfalls, soweit es dafür kein anerkennenswertes Interesse des Rechtsinhabers gibt.[55]

cc) Verhältnis zu § 69e UrhG

§ 69d Abs. 1 UrhG nimmt auf alle in § 69c Nr. 1 und 2 UrhG genannten Handlungen Bezug. Gedeckt wäre nach dem Wortlaut deshalb auch das Dekompilieren, weil es eine Vervielfältigung und Übersetzung der Codeform bedeutet. Allerdings besteht mit § 69e UrhG eine speziellere Regelung, die die Dekompilierung nur zur Herstellung von Interoperabilität mit anderen Programmen unter restriktiven Bedingungen erlaubt. In der Vergangenheit ist die herrschende Ansicht deshalb davon ausgegangen,

[51] Daneben begrenzt auch das Merkmal der Notwendigkeit den Kreis zulässiger Handlungen, dazu sogleich unter c).

[52] S. jüngst etwa die Sicherheitslücke in der weitverbreiteten Java-Logging-Bibliothek *Log4j*, vgl. https://www.heise.de/news/Kritische-Zero-Day-Luecke-in-log4j-gefaehrdet-zahlreiche-Server-und-Apps-6291653.html, archiviert unter perma.cc/FM5H-3YQM.

[53] Vgl. dazu die Schilderung eines Falls von *Franzen/Vettermann/Wagner* DuD 2020, 511, 512.

[54] Dazu sogleich IV.1.d.

[55] Vgl. zum Eingriff *Jarass*, in: Jarass (Hrsg.), Charta der Grundrechte der EU, 4. Aufl. 2021, Art. 13 Rn. 11; vgl. auch zur Perspektive nach deutschem Verfassungsrecht *Vettermann/Wagner* InTeR 2020, 126, 131.

dass eine Dekompilierung für eine bestimmungsgemäße Benutzung auf Grundlage von § 69d Abs. 1 UrhG nicht zulässig sein kann.[56]

Diese Ansicht ist mit der Entscheidung des *EuGH* in der Sache *Top System* überholt. Überzeugend begründet der *EuGH*, dass die Dekompilierungsausnahme in Art. 6 der Computerprogramm-RL[57] die Anwendung von Art. 5 Abs. 1 der RL nicht sperrt. Die Normen sind nahezu wortgleich in §§ 69d, 69e UrhG umgesetzt. Der Wortlaut gibt jeweils keinen Anhaltspunkt für eine Sperrwirkung.[58] Wenn die Dekompilierung von Art. 5 der RL ausgenommen wäre, müsste sie auch von den Ausschließlichkeitsrechten des Art. 4 der RL ausgenommen sein – dann gäbe es absurderweise aber keine Vorschrift, die die Dekompilierung untersagen würde.[59] Auch aus der Gesetzeshistorie folgt nichts anderes, denn Art. 6 der RL sollte von vornherein nur die Dekompilierung außerhalb der gewöhnlichen Benutzung regeln. Dass die Vorschrift im Gesetzgebungsprozess eingeschränkt wurde, lag nur daran, dass von Art. 6 der RL – anders als von Art. 5 – keine vertragliche Abweichung möglich war.[60] Entscheidend aber ist für den *EuGH*, dass beide Vorschriften unterschiedliche Zwecke verfolgen: Während Art. 6 der RL die Interoperabilität sicherstellen und damit Wettbewerb begünstigen soll, dient Art. 5 der RL der Gewährleistung der bestimmungsgemäßen Nutzung und richtet sich damit an den Nutzer.[61] Es wäre außerdem der praktischen Wirksamkeit der Fehlerberichtigung abträglich, wenn kein Zugriff auf den Quellcode bestünde.[62] Aus diesem Grund kann die Dekompilierung auch nach § 69d Abs. 1 UrhG ohne Zustimmung des Urhebers zulässig sein.

---

[56] *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 26; *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 3; für eine wissenschaftsfreundliche Auslegung aber schon *Vettermann/Wagner* InTeR 2020, 126, 133.

[57] RL 91/250/EWG des Rates vom 14.5.1991 über den Rechtsschutz von Computerprogrammen.

[58] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 54.

[59] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 60.

[60] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 59, 63.

[61] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 49; *Gülker* InTeR 2022, 22, 23.

[62] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 52.

b) Privilegierter Personenkreis

Der Personenkreis, der von § 69d Abs. 1 UrhG profitieren kann, ist begrenzt. Nach Art. 5 Abs. 1 Computerprogramm-RL sind lediglich Handlungen für die bestimmungsgemäße Nutzung von „rechtmäßigen Erwerber[n]" ausgenommen. § 69d Abs. 1 UrhG stellt dagegen auf „jeden zur Verwendung eines Vervielfältigungsstücks des Programms Berechtigten" ab. Die deutsche Umsetzung korrigiert dadurch ein Redaktionsversehen in der Richtlinie, die Lizenznehmer nicht vom Anwendungsbereich ausschließen wollte.[63]

Berechtigt sind IT-Sicherheitsforscher also zunächst, wenn sie als Ersterwerber bzw. Lizenznehmer vom Rechtsinhaber Nutzungsbefugnisse erworben haben. Sie können aber auch als Zweiterwerber berechtigt sein, wenn der Ersterwerber die Software an sie überlassen durfte.[64] Eine geschlossene Kette an Nutzungsrechtseinräumungen zum Rechtsinhaber ist dagegen nicht erforderlich. § 69d Abs. 1 UrhG wirkt insofern wie eine gesetzliche Lizenz, die zu den genannten Handlungen selbstständig berechtigt.[65] Es reicht deshalb aus, wenn ein Nutzer ein Vervielfältigungsstück des Computerprogramms rechtmäßig erworben hat, der Erwerb also nicht gegen urheberrechtliche Vorschriften verstoßen hat.[66] Auch Personen, die keine eigenen Nutzungsbefugnisse erworben haben, können durch eine Gestattung des Berechtigten in den Anwendungsbereich fallen; dazu gehören etwa Familienmitglieder, Freunde und Angestellte.[67]

Der Wortlaut der Norm erzwingt es im Übrigen nicht, dass eine berechtigte Person die freizustellenden Handlungen selbst vornimmt. Die Handlungen müssen lediglich für eine bestimmungsgemäße Benutzung durch einen Berechtigten notwendig sein. Berechtigt muss also lediglich die Person sein, die von den Handlungen profitiert – nicht die Person, die die Handlungen vornimmt. Es ist deshalb auch möglich, dass Dritte das Recht zur Fehlerbeseitigung ausüben, solange die weiteren Voraussetzungen vorliegen.[68] IT-Sicherheitsforscher können also von Berechtigten beauf-

[63] *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 28.

[64] *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 4a; *Dreier*, in: Dreier/Schulze (Fn. 5), § 69d Rn. 6.

[65] *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 30.

[66] *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 30; *Wiebe*, in: Spindler/Schuster (Fn. 35), § 69d Rn. 8.

[67] *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 4b, *Wiebe*, in: Spindler/Schuster (Fn. 35), § 69d Rn. 10.

[68] *BGH* GRUR 2000, 866, 868; *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 5; *Gülker* InTeR 2022, 22, 24.

tragt werden, um die zur bestimmungsgemäßen Benutzung notwendigen Handlungen vorzunehmen.

c) Notwendigkeit

Eingeschränkt wird § 69d Abs. 1 UrhG dadurch, dass die betroffenen Handlungen für die bestimmungsgemäße Benutzung *notwendig* sein müssen.

Für IT-Sicherheitsforschungsmaßnahmen wie die Fehlerberichtigung und Fehlersuche ist häufig der Zugriff auf den (Quasi-)Quellcode erforderlich, weil der Objektcode zu diesem Zweck nicht ausreichend lesbar ist.[69] Handlungen wie das Dekompilieren, die den Quellcode zugänglich machen, sind daher im Sinne von § 69d Abs. 1 UrhG erforderlich – es sei denn, der Quellcode ist auf anderem Wege rechtlich oder vertraglich[70] zugänglich. Regelmäßig ist dann auch ein Zugang zum *gesamten* Quellcode notwendig. Zwar erfordert die Fehlerbeseitigung Änderungen normalerweise lediglich in einem kleinen Teil des Quellcodes, allerdings kann der Fehler nur beseitigt werden, wenn die zu berichtigenden Stellen zunächst durch die Analyse ausfindig gemacht werden.[71]

Im Rahmen von § 69d Abs. 1 UrhG sind wiederum Wertungen aus dem Kaufrecht zu berücksichtigen: Soweit aufgrund eines Fehlers ein Nacherfüllungsanspruch besteht, ist der Zugang zum Quellcode nicht erforderlich – Handlungen zur Herstellung des bestimmungsgemäßen Gebrauchs sind dann erst notwendig, wenn Vertragspartner die Mängelbeseitigung verweigern.[72] Außerdem kann der Zugang zum Quellcode nicht erforderlich sein, wenn Hersteller kostenlose Sicherheitsupdates bereitstellen.[73]

Für die IT-Sicherheitsforschung spielt außerdem der weitere Umgang mit den gewonnenen Erkenntnissen eine große Rolle. Anders als § 69e Abs. 2 UrhG enthält § 69d UrhG keine ausdrücklichen Vorgaben, wie mit dem Quasi-Quellcode umzugehen ist. Allerdings wirkt hier die Erforderlichkeit erneut als Grenze: Vervielfältigungen des Quellcodes dürfen nur

[69] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 62; *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 79.

[70] Zur Rolle besonderer vertraglicher Bestimmungen sogleich.

[71] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 85.

[72] *Gülker* InTeR 2022, 22, 24 f.

[73] Vgl. *Gülker* InTeR 2022, 22, 25.

dann an Dritte gegeben werden, wenn dies für die bestimmungsgemäße Benutzung der Software notwendig ist.[74] Das führt zu Schwierigkeiten, wenn Forscher die Existenz und Schließung von Sicherheitslücken im Rahmen einer *coordinated vulnerability disclosure* bekannt geben möchten.[75] Notwendig kann die Weitergabe gegenüber dem Hersteller sein (sog. *limited disclosure*[76]), wenn dessen Mitwirkung bei der Wiederherstellung der bestimmungsgemäßen Nutzbarkeit erforderlich ist. Die Veröffentlichung des vulnerablen Quellcodes gegenüber der Öffentlichkeit (sog. *full disclosure*[77]) hingegen wird für die bestimmungsgemäße Nutzung eines Programms nie notwendig sein – sie erfordert daher die Erlaubnis des Rechtsinhabers (vgl. § 69c Nr. 3 S. 1 UrhG). Davon sind aber die Ideen und Grundsätze nicht erfasst, auf denen eine Software und ihre Sicherheitslücken basieren. Diese unterfallen von vornherein nicht dem urheberrechtlichen Schutz, können also frei kommuniziert werden.

d) Rolle besonderer vertraglicher Bestimmungen

§ 69d Abs. 1 UrhG ist nur anwendbar, soweit keine „besonderen vertraglichen Bestimmungen" bestehen. Das bedeutet allerdings nicht, dass der Rechtsinhaber die Erlaubnis zur Fehlerbeseitigung und -suche schlicht vertraglich ausschließen kann. Denn nach der Rechtsprechung des *EuGH* darf die Berichtigung von Fehlern, die die Funktion beeinträchtigen, nicht vollständig untersagt werden.[78] Die Parteien dürfen § 69d Abs. 1 UrhG lediglich insofern einschränken, dass primär der Rechtsinhaber für die Fehlerbeseitigung zuständig ist.[79] § 69d Abs. 1 UrhG hat also einen abredefesten Kern, nämlich eine prinzipielle Möglichkeit der Fehlerbeseitigung; nicht zwingend ist hingegen, dass gerade *der Nutzer* den bestimmungsgemäßen Zustand wiederherstellen kann. Hat der Rechtsinhaber sich die Fehlerbeseitigung vertraglich vorbehalten, darf der Nutzer sie selbst dann nicht eigenhändig vornehmen, wenn der Rechtsinhaber untätig bleibt. In diesem Fall bleibt dem Nutzer nur die Möglichkeit, die Fehlerbeseitigung

[74] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 69 ff.

[75] Zum Prozess s. *CEPS*, Software Vulnerability Disclosure in Europe (Fn. 18), S. 5 ff.

[76] S. dazu *Balaban u.a.,* Whitepaper zur Rechtslage der IT-Sicherheitsforschung, 2021, abrufbar unter https://sec4research.de/assets/Whitepaper.pdf, S. 30.

[77] S. dazu *Balaban u.a.*, Whitepaper (Fn. 76), S. 29.

[78] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 65 f.; vgl. auch schon *BGH* GRUR 2000, 866, 868.

[79] Vgl. *EuGH*, Urteil v. 6.10.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:811, Rn. 67.

gerichtlich zu erzwingen.[80] Die Fehlersuche hingegen kann vertraglich nicht ausgeschlossen werden, weil anderenfalls der Berechtigte seine Mängelrechte nicht ausüben könnte.[81]

2. Anforderungen des § 69d Abs. 3 UrhG

Auf den ersten Blick relevant für die IT-Sicherheitsforschung scheint § 69d Abs. 3 UrhG. Die Norm stellt allerdings lediglich klar, dass keine Zustimmung des Rechtsinhabers erforderlich ist, um Ideen und Grundsätze eines Programms zu gewinnen, indem ein Berechtigter[82] es beobachtet, untersucht oder testet, denn Ideen und Grundsätze sind nach § 69a Abs. 2 S. 2 UrhG ohnehin vom Schutz ausgenommen. § 69d Abs. 3 UrhG beschränkt die zulässigen Handlungen zudem auf das Laden, Anzeigen, Ablaufen, Übertragen oder Speichern des Programms. Zu diesen Handlungen sind Nutzer bereits gemäß § 69d Abs. 1 UrhG berechtigt.[83]

Der Erklärungsgehalt der Norm liegt deshalb weniger in einer Freistellung, sondern zum einen darin, den Kreis zulässiger Handlungen zur Programmanalyse *nicht* zu erweitern. Zulässig für Programmtests sind also keine Eingriffe in den Code wie Vervielfältigungen, Änderungen oder Übersetzungen. Das Dekompilieren kann damit nicht auf § 69d Abs. 3 UrhG gestützt werden.[84] Erlaubt sind dagegen IT-Sicherheitsforschungsmaßnahmen passiver Art, wie das Black-Box-Testing, Speicherabzüge/Memory Dumps, System Monitoring, Debugging oder Line Tracing.[85] Die zu gewinnenden Ideen und Grundsätze können dabei beliebigen Zwecken, also auch der IT-Sicherheitsforschung dienen.[86]

In Verbindung mit § 69g Abs. 2 UrhG stellt § 69d Abs. 3 UrhG zum anderen klar, dass vertragliche Abweichungen insofern nichtig sind. Die dem Programm zugrundeliegenden Ideen und Grundsätze können deshalb nicht durch einen Lizenzvertrag geschützt werden.[87] Die Norm berück-

[80] *Gülker* InTeR 2022, 22, 25.
[81] S. dazu bereits IV.1.a)bb)(1).
[82] S. dazu schon oben IV.1.b).
[83] Vgl. *EuGH*, Urteil v. 2.5.2012 – C 406/10, *SAS Institute*, ECLI:EU:C:2012:259, Rn. 59; *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 23.
[84] *Czychowski*, in: Fromm/Nordemann (Hrsg.), Urheberrecht, 12. Aufl. 2018, § 69d Rn. 28.
[85] Vgl. *Czychowski*, in: Fromm/Nordemann (Fn. 84), § 69d Rn. 29; *Dreier*, in: Dreier/Schulze (Fn. 5), § 69d Rn. 22; *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 76 f. mit Beschränkung auf das „normale Maß".
[86] Vgl. *Spindler*, in: Schricker/Loewenheim (Fn. 6), § 69d Rn. 22.
[87] *EuGH*, Urteil v. 2.5.2012 – C 406/10, *SAS Institute*, ECLI:EU:C:2012:259, Rn. 51.

sichtigt dadurch die Besonderheit von Computerprogrammen, deren Ausdrucksform für Menschen nicht wahrnehmbar ist, wodurch sich zugrundeliegende Ideen leicht verbergen ließen.[88] Problematisch ist in dieser Hinsicht, dass § 69d Abs. 3 UrhG nur für Rechte an Computerprogrammen gilt. Eingriffe in Rechte an anderen Werken deckt die Vorschrift nicht unmittelbar ab.[89] Programmtests von softwarebasierten hybriden Werken können in diesen Fällen aber nach einer teleologischen Auslegung von §§ 44a, 57 UrhG zulässig sein.[90]

3. Zwischenergebnis

Auch wenn IT-Sicherheitsforschung die Ausschließlichkeitsrechte des Urhebers berührt, sind bestimmte Handlungen ohne Zustimmung des Rechtsinhabers zulässig: Erlaubt ist zunächst die passive Beobachtung von Programmen, soweit sie dem bestimmungsgemäßen Gebrauch dient oder Ideen und Grundsätze gewonnen werden sollen. Aber auch Eingriffe in den Code durch Umarbeitungen einschließlich der Dekompilierung sind zulässig, wenn sie für die Berichtigung eines Fehlers erforderlich sind. Schließlich ist auch die Fehlersuche unter den Voraussetzungen des § 69d Abs. 1 UrhG erlaubt, wenn die Umstände des Einzelfalls einen verdeckten Fehler nahelegen. Dagegen erfordern anlasslose Eingriffe in den Code, die weder der Fehlerberichtigung dienen noch für den bestimmungsgemäßen Gebrauch notwendig sind, eine Erlaubnis der Rechtsinhaber.

V. Rechtsgeschäftliche Erlaubnis von IT-Sicherheitsforschung

IT-Sicherheitsforschung abseits der gesetzlichen Ausnahmen hängt grundsätzlich von der Erlaubnis der Rechtsinhaber ab. Häufig haben auch sie ein Interesse daran, Sicherheitslücken aufzudecken und zu schließen. In der Praxis sind deshalb Wege entstanden, IT-Sicherheitsforschung vertraglich zu erlauben.

---

[88] *GA Szpunar*, Schlussantrag v. 10.3.2021 – C 13/20, *Top System*, ECLI:EU:C:2021:193, Rn. 7.

[89] Vgl. *BGH* GRUR 2017, 266, 273 Rn. 65, 67; *Grützmacher*, in: Wandtke/Bullinger (Fn. 6), § 69d Rn. 75; *Vettermann/Wagner* InTeR 2020, 126, 129.

[90] Vgl. dazu *Grützmacher* ZGE 9 (2017), 423, 439.

1. Open Source Software

Zunächst können Rechtsinhaber den Nutzern ihrer Software den Quellcode offenlegen und sie mit Nutzungsrechten ausstatten, die IT-Sicherheitsforschung ermöglichen. Das ist insbesondere bei Open Source Software der Fall, wenn also das Computerprogramm nicht proprietär vermarktet wird, sondern quelloffen zugänglich ist. Für Rechtsinhaber bietet das neben einem höheren Verbreitungsgrad auch Vorteile für die IT-Sicherheit. Die Suche nach Fehlern und ihre Behebung stehen einer großen Menge von IT-Sicherheitsforschern und Entwicklern offen. Da jeder den Code auf seine Funktionsweise und Sicherheitslücken überprüfen kann, steigt das Vertrauen in die Software. Gleichzeitig können Schwachstellen sofort bekannt gegeben werden, ohne dass Vertraulichkeitsvereinbarungen der Hersteller berücksichtigt werden müssen.

Für Rechtsinhaber bietet es sich insofern an, auf bestehende Open Source Software-Lizenzen zurückzugreifen. Nahezu alle gängigen Lizenzen erlauben die für die IT-Sicherheitsforschung erforderliche Vervielfältigung und die Bearbeitung des Quellcodes. Beispielsweise gestattet die weitverbreitete GNU General Public License (GPL), die Software zu vervielfältigen, zu verbreiten und zu bearbeiten (Version 2) oder zu modifizieren und zu propagieren (Version 3). Die MIT License stellt sogar jede Nutzung der lizenzierten Software frei.[91] IT-Sicherheitsforscher müssen dann lediglich die weiteren Lizenzbedingungen beachten: im Falle der MIT License etwa, dem lizenzierten Material stets die Lizenz und einen Hinweis darauf beizufügen; im Fall der GPL, Veränderungen hervorzuheben und unter derselben Lizenz zugänglich zu machen. Bei einem Verstoß gegen die Bedingungen entfällt die Nutzungsrechtseinräumung.[92]

2. Vulnerability Disclosure Policies und Bug-Bounty-Programme

Hersteller von Software können Sicherheitsforschung auch im Rahmen sog. Vulnerability Disclosure Policies (kurz VDP, auch Responsible Disclosure Policies genannt) erlauben. VDP werden vom Hersteller einer Software zugänglich gemacht und beschreiben den Meldeweg für Sicherheitsforscher, die auf eine Sicherheitslücke gestoßen sind. Sie benennen einen Kontakt für die Meldung und geben einen Zeitplan für eine Emp-

[91] https://opensource.org/licenses/mit-license.php, archiviert unter https://perma.cc/5X8Z-9TCN.

[92] Vgl. *OLG Hamm* GRUR-RR 2017, 421, 424.

fangsbestätigung und weitere Statusupdates vor.[93] Sie dienen dem Ausgleich zwischen dem Interesse der Allgemeinheit, vor Sicherheitslücken schnellstmöglich gewarnt zu werden und dem Interesse der Hersteller, Zeit zur Behebung zu gewinnen, damit Nutzer nicht gefährdet werden.[94] Daneben können Hersteller auch sog. Bug-Bounty-Programme auflegen. Sie verfolgen den Zweck, Sicherheitsforscher zur Analyse von Software anzureizen, indem sie Prämien (*bounties*) für aufgefundene Sicherheitslücken ausloben. Gleichzeitig legen die Programmbedingungen fest, unter welchen Umständen die Sicherheitsforschung erfolgen darf.

VDP begründen meist keine Privilegierung für IT-Sicherheitsforschung.[95] Sie skizzieren lediglich den idealtypischen Meldeprozess, treffen aber keine Regelungen zum Auffinden einer Lücke. Sie erfassen einen wichtigen Teil von IT-Sicherheitsforschung deshalb nicht. Bug-Bounty-Programme dagegen regeln für gewöhnlich, welche Handlungen vorgenommen werden dürfen. Sie bezwecken es gerade, IT-Sicherheitsforschern Rechtssicherheit zu verschaffen. Nicht selten adressieren die Bedingungen auch explizit Eingriffe in das Urheberrecht des Software-Herstellers. Nach den Bedingungen der „Apple Security Bounty" etwa werden Handlungen von Teilnehmern des Programmes nicht als Verletzung der Lizenzbedingungen von Apple-Software angesehen, wenn drei Bedingungen erfüllt sind: Es muss sich um redliche Sicherheitsforschung mit dem Ziel einer Responsible Disclosure handeln, die Aktivitäten müssen während der Teilnahme am Bug-Bounty-Programm erfolgt sein und die übrigen Bedingungen des Programms müssen eingehalten worden sein.[96] Zu diesem Zweck stellt Apple einigen Sicherheitsforschern Geräte ohne übliche Schutzmaßnahmen zur Verfügung.[97] Das Bug-Bounty-Programm von Meta legt fest, dass das Unternehmen keine Zivilverfahren gegen Teilnehmer anstrengen wird, wenn die Bestimmungen des Programms gewahrt sind.[98] Dazu gehören ausdrücklich auch Forderungen nach dem Digital Millenium Copyright Act (DMCA) aufgrund der Umgehung technischer Schutzmaßnahmen. Das Bug-Bounty-Programm von Microsoft schließlich eröffnet einen Safe Harbor, indem es Sicherheitsforschung im Ein-

[93] Vgl. Provision 5.2–1 der ETSI EN 303 645 V2.1.1 (2020–06) Cyber Security for Consumer Internet of Things: Baseline Requirements.

[94] *Balaban u.a.*, Whitepaper (Fn. 76), S. 54.

[95] Vgl. *Wagner* DuD 2020, 111, 119: „reine Selbstverpflichtung".

[96] https://developer.apple.com/security-bounty/requirements/, archiviert unter perma.cc/R3TF-D3KA.

[97] Apple Security Research Device Program, https://developer.apple.com/programs/security-research-device/, archiviert unter perma.cc/G9VK-ST9J.

[98] https://de-de.facebook.com/whitehat, archiviert unter perma.cc/9CHH-R92L.

klang mit den Bedingungen als autorisiert ansieht und auf Ansprüche nach dem DMCA verzichtet.[99]

Die Erlaubnis zur IT-Sicherheitsforschung durch die Programme erlaubt auch Eingriffe in das Software-Urheberrecht. Sie können urheberrechtlich unterschiedlich gewürdigt werden: Widersprüche ergeben sich etwa, wenn Unternehmen einerseits auf Ansprüche verzichten wollen, andererseits aber die Einstellung des Programmes in das eigene Ermessen stellen. Die Auslegung nach dem Übertragungszweck (vgl. § 31 Abs. 5 S. 2 UrhG) legt dann nahe, dass kein dauerhafter Anspruchsverzicht im Sinne eines Erlassvertrages (§ 397 BGB)[100] gemeint ist. Vielmehr kann man in diesen Fällen von einer Einräumung einfacher Nutzungsrechte ausgehen. Dafür spricht, dass die Unternehmen häufig hohe Geldbeträge ausloben und so Sicherheitsforscher zu Investitionen in die Ermittlung von Lücken motivieren. Folglich besteht ein legitimes Interesse der Forscher an einer verlässlichen Rechtsposition.

3. Konkludente Erlaubnis

Wenn Rechtsinhaber und Nutzer keine ausdrücklichen vertraglichen Abreden treffen, ist an eine konkludente Erlaubnis für die IT-Sicherheitsforschung zu denken. Bei normalen Software-Lizenzverträgen ist dazu auf die Umstände des Einzelfalls, insbesondere den Vertragszweck und die vorangegangene Vertragspraxis sowie die Branchenübung abzustellen.[101] Zwar werden Sicherheitstests von Software in mehr und mehr Branchen üblich. Nach dem Übertragungszweckgedanken (vgl. §§ 69a Abs. 4, 31 Abs. 5 UrhG) erwirbt der Nutzer im Zweifel aber nur die Rechte, die zur Erreichung des Vertragszwecks erforderlich sind. Häufig sind die Vertragswerke im Softwareurheberrecht so ausdifferenziert, dass sie wenig Raum für eine konkludente Erlaubnis lassen.[102] Denkbar ist eine konkludente Erlaubnis also nur dann, wenn IT-Sicherheitsforschungsmaßnahmen an der Software nach den Umständen des Einzelfalls wie der Branchenüblichkeit zulässig sein sollten und die expliziten Abreden nicht als abschließend betrachtet werden können.

Mehr Raum für konkludente Einwilligungen bieten Bug-Bounty-Programme. Enthalten deren Bedingungen keine urheberrechtliche Regelung,

[99] https://www.microsoft.com/en-us/msrc/bounty-safe-harbor, archiviert unter perma.cc/M3WM-2F5P.

[100] Vgl. dazu *Schack* (Fn. 3), Rn. 348.

[101] *OLG Frankfurt a.M.* MMR 2014, 661, 662.

[102] *Hoeren/Pinelli* CR 2019, 410, 412.

kann man von einer konkludenten Erlaubnis im Wege einer schlichten Einwilligung ausgehen.[103] Es wäre widersprüchlich, wenn ein Rechtsinhaber einerseits dazu anreizt, Lücken aufzuspüren und davon profitiert, andererseits dafür erforderliche Nutzungshandlungen aber nicht billigen möchte. Die schlichte Einwilligung bezieht sich dann allerdings lediglich auf Nutzungshandlungen, die mit den Bedingungen des Programms im Einklang stehen und für die Sicherheitsforschung notwendig sind.

VI. Fazit

Im Kern ist IT-Sicherheitsforschung keine Tätigkeit, die mit dem exklusiven Schutzbereich des Urheberrechts konfligiert, weil sie sich mit der Funktionalität und den Ideen und Grundsätzen von Computerprogrammen beschäftigt. Sie berührt das Urheberrecht aber reflexhaft, weil ihre Forschungsmethoden mit urheberrechtlich relevanten Handlungen, insbesondere der Vervielfältigung und Umarbeitung, einhergehen. Einen Interessenausgleich zwischen IT-Sicherheitsforschern und Rechtsinhabern stellen, wie auch sonst im Urheberrecht, die urheberrechtlichen Schranken her. Die gesetzliche Erlaubnis für Handlungen zur bestimmungsgemäßen Benutzung stellt dabei nicht nur die normale Ausführung des Programms und die Behebung von Fehlern frei, sondern ermöglicht richtigerweise auch die anlassbezogene Fehlersuche. In einer kürzlich ergangenen Entscheidung hat der *EuGH* zudem eine weitere Hürde für die IT-Sicherheitsforschung abgebaut, indem er auch die Dekompilierung zur bestimmungsgemäßen Nutzung zählt und außerhalb der Herstellung von Interoperabilität zulässt. Abseits der gesetzlichen Schranken sind vertragliche Regelungen in Gestalt von Open Source-Lizenzen, Vulnerability Disclosure Policies oder Bug-Bounty-Programmen maßgeblich – sie liegen auch im Interesse der Hersteller. Weder diese vertraglichen Modelle noch die gesetzlichen Schrankenbestimmungen vermögen es allerdings, IT-Sicherheitsforschung einen idealen urheberrechtlichen Boden zu bereiten. Dies wird dem großen gesellschaftlichen Interesse an IT-Sicherheit nicht gerecht. Wünschenswert wäre *de lege ferenda* daher eine explizite Privilegierung der IT-Sicherheitsforschung im Rahmen der urheberrechtlichen Regelungen für Computerprogramme.

[103] So auch *Halder* jurisPR-ITR 6/2022 Anm. 3, D.IV.