»Der Angriff weiß, was er will. Die Verteidigung befindet sich in dem Zustand der Ungewissheit.«

Helmuth Graf von Moltke

Zur Sicherheit softwarebasierter Produkte

Status Quo, Ausblick und FAQ zu Entwicklung und Betrieb softwarebasierter Produkte

www.bitkom.org

bitkom

Herausgeber

Bitkom e. V.
Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V.
Albrechtstraße 10 | 10117 Berlin

Ansprechpartner

Dr. Frank Termer | Bereichsleiter Software
T 030 27576-232 | f.termer@bitkom.org

Verantwortliches Bitkom-Gremium

AK Quality Management

Projektleitung

Dr. Frank Termer | Bitkom e.V.

Inhaltsverzeichnis

Einführung und Motivation

Mit zunehmender Digitalisierung im Berufs- und Privatleben und der damit entstehenden Omnipräsenz softwarebasierter Produkte rückt das Thema Sicherheit in das allgemeine Bewusstsein. Ob der heimische Fernseher, das genutzte Smartphone oder das moderne Auto: Software ist allgegenwärtig. Sie ist ein zentrales Element der digitalen Transformation und ermöglicht in vielen Fällen erst die Digitalisierung ganzer Branchen. Gleichzeitig ergeben sich durch den steigenden Softwaredurchdringungsgrad in Produkten Gefahren und Bedrohungsszenarien, die auf die entsprechende Software abzielen. Damit erhält Software eine zunehmende Bedeutung bei der Betrachtung und Bewertung der Sicherheit von Produkten insgesamt, und eine erhöhte Wahrnehmung bei den Anwendern softwarebasierter Produkte. Die Akzeptanz von Produkten und Dienstleistungen, die durch einen nicht unerheblichen Teil durch Software definiert werden, hängt in besonderem Maße von deren Sicherheit ab. Damit wird Sicherheit zu einem zunehmenden Erfolgsfaktor für Unternehmen, um langfristig am Markt agieren zu können.

Bitkom, und die darin verankerten softwarebezogenen Gremien, erreichen zunehmend Fragen, warum das Thema Sicherheit von softwarebasierten Produkten scheinbar nicht in den Griff zu bekommen ist. In diesem Dokument sollen einige dieser Fragen exemplarisch aufgegriffen und möglichst allgemein verständlich erläutert werden. Die Fragen haben dabei keinerlei Anspruch auf Vollständigkeit, werden aber so oder in leicht modifizierter Form immer wieder gestellt. Gleichzeitig ist es dem Bitkom ein wichtiges Anliegen in diesem Dokument darzustellen, auf welche vielschichtige Art und Weise die Anbieter softwarebasierter Produkte und Dienstleistungen sich um die Sicherheit ihrer Produkte kümmern, und welchen Aufwand sie betreiben, um den diesbezüglich berechtigten Erwartungen der Anwender gerecht zu werden. Ebenso werden aber auch die Grenzen des derzeitig Machbaren dargestellt und an einigen Stellen erläutert, warum manches von Anbietern softwarebasierter Produkte eben nicht gemacht wird.

Im vorliegenden Dokument werden nun aufgeworfene Fragen einzeln adressiert und in jeweils einem separaten Kapitel betrachtet. Dabei werden auch unterschiedliche Sichtweisen und Aspekte berücksichtigt, die bei der Produktauswahl für den Anwender von Bedeutung sein könnten (z. B. Preis, die Zeit, Leistungsfähigkeit oder die Erwartungshaltung des Konsumenten). Am Ende des Dokuments fasst zudem eine FAQ-Sammlung diese Fragen und kurze Antworten dazu zusammen.

Wir hoffen, mit diesem Dokument dem Thema Sicherheit von softwarebasierten Produkten allgemeinverständlich und mit hohem fachlichen Anspruch gerecht zu werden. Es ist explizites Ziel, dieses Dokument laufend fortzuschreiben, zu aktualisieren und um weitere Fragen zu ergänzen. Wir laden daher herzlich ein, sich an der Weiterentwicklung des Dokuments zu beteiligen. Ihr Feedback nehmen wir gern entgegen! Für Fragen, aber auch für kritische Anmerkungen zu unseren Antworten, stehen wir ebenfalls gerne jederzeit zur Verfügung.

- Susanne Dehmel, Bitkom e.V.
- Marc Fliehe, Bitkom e.V.
- Stefan Luckhaus, PASS Consulting Group
- Dr. Frank Simon, BLUECARAT AG
- Dr. Frank Termer, Bitkom e.V.

1 Patches und Updates

Unsere Welt verändert sich permanent. Täglich entstehen neue Technologien, werden Prozesse angepasst und Organisationen umstrukturiert. Software als ein wesentlicher Bestandteil dieser Lebensbereiche ist folglich einem ständigen Änderungs- und Anpassungsprozess unterworfen. Diese Veränderung von Software wird für den (End-) Anwender immer dann sichtbar, wenn Aktualisierungen in Form von Patches und Updates bereitgestellt werden und eine Software aktualisiert werden muss. Ein wesentlicher Anteil der Aktualisierungen adressiert die Sicherheit von Software. So kann folglich die Frage gestellt werden, **warum Softwarehersteller nicht gleich sichere Produkte ausliefern anstatt ständig Patches und Updates nachzuliefern.**

»Softwarebasierte Produkte brauchen Patches und Updates.«

Zum einen ist es schlicht nicht möglich, alle Einsatzszenarien einer Software im Vorfeld der Nutzung zu kennen, und Programme bspw. gegen mögliche Angriffe umfänglich abzusichern. Sowohl Einsatzszenarien als auch Angriffsmethoden verändern sich ebenfalls fortwährend. Ohne bereitgestellte Patches und Updates »veraltet« ein Produkt sehr schnell und entspricht im Laufe der Zeit nicht mehr dem technologischen Stand. Selbst wenn ein System einen festgelegten Anwendungskontext hat, so steigt das Bedrohungspotenzial für Software, in dem Angreifer ihre Möglichkeiten, Methoden und Werkzeuge weiterentwickeln. Ein softwarebasiertes Produkt, welches zum Zeitpunkt seiner Auslieferung dem aktuellen technologischen Standard und Sicherheitsniveau entspricht, wird mit zunehmender Einsatzdauer unsicherer. Was vor einem Jahr noch als sicher galt, muss heute nicht automatisch unverändert sicher sein. Insofern ist es notwendig, softwarebasierte Produkte regelmäßig mit Patches und Updates zu versorgen, um hierdurch mindestens ein aktuell bestehendes Niveau an Sicherheit zu halten.

Ein zweiter Aspekt zur Beantwortung der Frage, warum Softwarehersteller permanent Updates und Patches ausliefern ist darin zu sehen, dass Qualität, und hierzu gehört auch der Parameter der Sicherheit, nur ein Produktmerkmal von Vielen darstellt. Insbesondere der Preis, aber auch Leistungsumfang, Zeitpunkt der Lieferung und die Marke / der Brand eines Produktes sind wesentliche Entscheidungskriterien bei der Auswahl softwarebasierter Produkte. In der ganzheitlichen Betrachtung aller Produktmerkmale kann festgestellt werden, dass die Qualität und damit die Sicherheit nicht in jedem Fall das entscheidende Kaufargument darstellt. Wenn für den Markt Sicherheitsprobleme so dominant wären, wie in der Frage suggeriert ist, würde der Markt fehlerfreiere Produkte herstellen. Allerdings ist Qualität nur ein Kaufparameter neben anderen: So dominiert häufig der Preis (»Geiz ist geil«) und der Zeitpunkt (»Trendsetter«), beides Parameter, die partiell als konkurrierend zum Attribut Qualität aufgefasst werden. Auch die Marke und das von ihr ggfs. in der Werbung geworfene Bild hat Einfluss auf das Marktverhalten, was ggfs. andere Aspekte kompensiert.

2 Qualitätssicherung

Das Bereitstellen von Patches und Updates hat zum einen das Ziel, neue Funktionalitäten bereitzustellen. Zum anderen werden mit der Zeit des Einsatzes aber auch Schwachstellen und Sicherheitslücken bekannt, die ebenfalls durch Aktualisierungen beseitigt oder zumindest abgemildert werden können. Bei Nutzern von softwarebasierten Produkten kann hier allerdings der Eindruck entstehen, dass eine Vielzahl sogenannter Bugs und sicherheitsrelevanter Fehler in diesen Produkten enthalten ist. Somit stellt sich die Frage, **ob einfach zu wenig in die Qualitätssicherung bei der Softwareentwicklung investiert wird?**

»Aufwände für Qualitätssicherung können sich je nach Kontext stark unterscheiden.«

Diese Frage kann auf drei Ebenen beantwortet werden. Zunächst ist dabei der »Kann-Teil« zu betrachten, der beschreibt, wie der Stand-der-Technik in der Qualitätssicherung von softwarebasierten Produkten heute prinzipiell aussieht. Bereits seit einiger Zeit lassen sich gerade Security-Schwächen durch ein sicheres Systemdesign konzeptuell reduzieren. Security-by-Design und Security-by -Default sind Ansätze, die einige Unternehmen schon sehr lange praktizieren, um ihren Kunden ein Höchstmaß an Sicherheit bieten zu können. Dabei sind sowohl das Vorhandensein von Sicherheitsfunktionalitäten im Produkt, als auch die Sicherheit des Softwareproduktes selbst Gegenstand der Betrachtung. In besonders sicherheitsrelevanten Systemen lässt sich die Korrektheit zudem formal sogar beweisen, wobei allerdings lediglich die Korrektheit zu einer Spezifikation bewiesen werden kann, nicht der Wert eines Systems zum Erreichen eines Zieles.

Die zweite Ebene adressiert den »Soll-Teil«, der beschreibt, wie ein langfristig agierendes Unternehmen mit dem Thema Qualität umgehen sollte. Der Softwaremarkt ist ähnlich vielfältig wie andere Bereiche und Branchen. So gibt es auch hier Unternehmen, die an einem längerfristigen Marktauftritt interessiert sind: Etwaige Kosteneinsparungen durch Vernachlässigung der Sicherheit können solche Hersteller finanziell durch die Kosten notwendiger Nachbesserungen sowie maßgeblich durch Reputationsverlust schädigen, so dass solche Unternehmen gegensteuern und bewusst auf ein hohes Maß an Qualität und Sicherheit setzen. Auf der anderen Seite sind prinzipiell aber am Softwaremarkt auch Hersteller denkbar, deren Businessmodell nur einmal ein kurzes, aber dafür evtl. besonders großes Geschäft ausmacht. Diese Unternehmen nehmen daher womöglich bewusst geringere Qualitäts- und Sicherheitsniveaus in Kauf. Während in anderen Bereichen, wie z. B. der Automobilbranche, ein relativ klares Verständnis existiert, welcher dieser beiden Strategien ein Unternehmen folgt, ist diese Unternehmensmarkteinschätzung im Softwarebereich noch nicht etabliert.

Eine letzte Ebene betrifft den »Stand-der-Praxis«, der angibt, wie das Thema Qualität am Markt wahrgenommen wird. Ein Markt, der sich typischerweise im Spannungsfeld zwischen Angebot und Nachfrage entwickelt, reagiert insbesondere auf verändertes Konsumentenverhalten. Wenn also Kunden softwarebasierte Produkte trotz auftretender Qualitäts- oder Sicherheitsprobleme nicht ablehnen, haben auch solche Hersteller am Markt eine Chance, deren Produkte eben solche Probleme aufweisen. Entsprechend besteht hier kein marktseitiger Druck, mehr in Qualitätssicherung im Bereich der Softwareentwicklung zu investieren. Würde der Markt empfindlicher auf Qualitätsprobleme reagieren, wären Hersteller gezwungen, den Aspekten Qualität und Sicherheit mehr Aufmerksamkeit zu widmen. Kunden können daher durch ihre Macht als Nachfrager selbst mitbestimmen, in welchem Umfang Hersteller softwarebasierter Produkte in Qualität und Sicherheit investieren.

3 Fehleranfälligkeit

Ebenfalls durch die häufige Bereitstellung von Updates und Patches für softwarebasierte Produkte, aber auch durch eine in der Vergangenheit medienwirksame Bekanntmachung von Fehlern in Software , ist in der allgemeinen Wahrnehmung mittlerweile der Eindruck entstanden, dass Software nie zuverlässig, sicher und korrekt funktioniert. Somit kann die Frage gestellt werden, warum **Software denn nie fehlerfrei ist**?

»Es gibt keine absolute Sicherheit bei softwarebasierten Produkten.«

Auch wenn jeder Mensch ein intuitives Verständnis von Zuverlässigkeit, Sicherheit und Korrektheit im Zusammenhang mit Software hat, so gibt es einheitliche Definitionen, die sich insbesondere damit auseinandersetzen, wann bei einer Software von einem Fehler gesprochen werden kann. Je nach Definition ist Software tatsächlich nie fehlerfrei.

a) Ein Fehler ist, wenn das System intern anders als geplant arbeitet.

In Systemen treten üblicherweise interne Fehler auf, ohne dass dies eine Fehlerwirkung hat, also dem Anwender tatsächlich bewusst wird. Einem Autofahrer wird es bspw. kaum auffallen, wenn die Software zur Vergasersteuerung völlig unbenutzten Code enthält, d. h. Programmteile, die nie zur Ausführung kommen. Dies ist nicht geplant und kann von daher als Fehler gesehen werden. Es kann daher nicht nur dann von einem Fehler gesprochen werden, wenn durch eben jenen, eine (unerwünschte) Außenwirkung auftritt. Softwaresysteme haben meist sehr viele interne Fehler, die allerdings durch geschickte Absicherungen keine Außenwirkung haben.

b) Ein Fehler ist, wenn das System in der Außenwirkung von der Spezifikation abweicht.

Die engste Definition eines Fehlers ist die als Abweichung von einer Spezifikation. Diesem Verständnis folgend kann durchaus fehlerfreie Software hergestellt werden. Hierfür existieren viele formale Methoden, auf deren Basis Programmierung kein kreativer Akt mehr ist, sondern lediglich die Transformation eines Formalismus in einen anderen. Einige autonom fahrende Fahrzeuge (z. B. U-Bahnen) sowie Kernkraftwerkssteuerungen sind so erstellt. Die formale Beschreibung beschränkt allerdings naturgemäß die Vielfältigkeit von Eingaben und Darstellungen. Graphische Benutzungsoberflächen, interaktive Dialoge und multimediale Darstellungen sind daher für formal spezifizierte Systeme kaum realisierbar. Da aber immer mehr Systeme und auch softwarebasierte Produkte keiner ausschließlichen formalen Beschreibung folgen und für unterschiedliche, im Vorfeld nicht immer bekannte Nutzungskontexte entworfen werden, steigt die Wahrscheinlichkeit, dass Software in der Außenwirkung fehlerhaft arbeitet.

c) Ein Fehler ist, wenn das System dem Nutzer bei der Bearbeitung einer Aufgabe nicht wie erwartet hilft.

Die weitreichendste Definition eines Fehlers als Abweichung von einer Erwartungshaltung unterstreicht wiederum die Hypothese grundsätzlich nicht fehlerfreier Software. Da jeder Nutzer i.d.R. eigene Erwartungen hat, kann ein System entlang dieser Definition kaum fehlerfrei sein. So resultiert der bekannte Spruch »It's not a bug, it's a feature« aus einer Sicht, bei der das System sich anders verhält, als erwartet. Es arbeitet aber noch, stellt kein Sicherheitsproblem dar und lässt sich meist auch weiter verwenden. Je mehr Menschen mit einem solchen System arbeiten, desto mehr Fehler entlang dieser Definition werden gefunden.

4 Regelmäßige Aktualisierungen

Aus der Annahme, dass Updates und Patches ein notwendiger Bestandteil in der aktuellen Entwicklung und Nutzung von Software sind, könnte geschlussfolgert werden, dass zunächst jedes softwarebasierte Produkt in einem unfertigen Zustand bereitgestellt wird. So wird in der Öffentlichkeit oft die Meinung kommuniziert, dass Softwarehersteller ihre Software häufig beim Kunden reifen lassen. So würden bewusst unfertige Softwareprodukte ausgeliefert und lediglich punktuell und schrittweise verbessert. Sind die vielen Bugfixes und Updates nur dadurch zu erklären, dass **Software erst beim Kunden reift**?

»Die Sicherheit von Software nimmt ohne regelmäßige Aktualisierungen kontinuierlich ab.«

Der Eindruck, dass Softwarehersteller bewusst vermeintlich unfertige, aber zumindest unausgereifte, Software absichtlich auf den Markt bringen, und damit auch deren Anwender wissentlich Gefahren aussetzen, welche auf die »unreife« Software zurückzuführen sind, hat seine Ursache wenigstens in den folgenden Aspekten.

a) Der Markt als wichtige Steuerungsgröße und als Korrektiv der Softwarebranche bestraft Fehler und Unzulänglichkeiten in softwarebasierten Produkten nur unzureichend.

Die Softwarebranche agiert in allergrößten Teilen entlang des freien Marktes: Demnach geht es für ein Softwareunternehmen darum, Produkte an den Kunden zu verkaufen und die Kunden nachhaltig an sich zu binden. Ist der Kunde mit einem Produkt nicht zufrieden, bricht diese Kette und das Softwareunternehmen verliert Marktanteile. Die Vergangenheit hat nun gezeigt, dass Produkte gekauft und auch wieder gekauft werden, selbst wenn Fehler nachträglich korrigiert und Patches sukzessive aufgespielt werden (müssen). Dabei ist es häufig unerheblich, wie gravierend diese Fehler sind; in keinem Fall kann durch diese Probleme ein signifikanter Marktverlust der produzierenden Unternehmen festgestellt werden. Es scheint demnach am Markt aktuell eine Erwartungshaltung zu geben, die sich mit einem minimalen Niveau an Sicherheit und Qualität begnügt, welchem die Industrie dann naturgemäß entspricht.

b) Durch die erhöhte Sichtbarkeit von Software wird zunehmend auch über deren Fehler und Fehlverhalten berichtet, so dass in der öffentlichen Wahrnehmung negative Darstellungen über Software dominieren und Normalverhalten ignoriert wird.

Heute werden in der Presse und damit in der Öffentlichkeit primär die vielen Security-Probleme und Patch-Days hervorgehoben. Die Vielzahl dieser identifizierten Probleme korreliert aber in keinster Weise mit der Vielzahl der heute eingesetzten und zu großen Teilen jahrelang völlig problemlos laufenden Softwaresysteme.

c) Software wird heute in immer kürzeren Intervallen verändert und aktualisiert, wobei diese Veränderungen fast immer augenblicklich dem Anwender bereitgestellt werden, weil dieser das erwartet.

Die Vergangenheit hat gezeigt, dass derjenige ein Marktsegment erobert, der zuerst präsent ist (first-mover-advantage). Die langen Warteschlangen bei neuen Smartphone-Releases, bei denen bis dahin nur die Hersteller die neuen Features angepriesen haben und noch kein unabhängiger Test erfolgt, belegen dies eindrucksvoll. Dieser Tradeoff zwischen »Früh live gehen, aber vermutlich mit einigen Fehlern« und »den Markt anderen überlassen, dafür aber dann fehlerfrei rausgehen« muss jedes Unternehmen selbst beantworten; es spricht aber aktuell einiges dafür, dass wenigstens einige Kunden die Umsetzung der Variante 1 ebenfalls honorieren.

d) Software wird zunehmend in verschiedenen Kontexten eingesetzt. Diese Bereiche sind im Vorfeld des Einsatzes nicht vollständig bekannt und Software wird zudem in Bereichen eingesetzt, für die diese nicht konzipiert war.

Während die Erfüllung funktionaler Anforderungen an eine Software mit Hilfe konstruktiver und analytischer Qualitätssicherung nahezu vollständig verifiziert werden kann, ist die Überprüfung des Qualitätsmerkmals Sicherheit nur bezogen auf einen bestimmten Zeitpunkt und einen bestimmten Anwendungskontext möglich. Gerade heute ist der Anwendungskontext aber beliebig vielfältig. Dies schließt den Anwendungskreis (fast jeder Mensch ist heute in irgendeiner Form mit Software konfrontiert), ebenso ein wie den Einsatzort, der vom Desktop-Rechner, über das Smartphone über das Auto bis zum Smart-Home reichen kann. Eine vollständige Abdeckung aller nur denkbaren Nutzungskontexte kann hier immer weniger erreicht werden.

e) Software ist einer steigenden Zahl von Bedrohungsszenarien ausgesetzt und kann in vielen Fällen höchstens mit der aktuellen Gefährdungslage mithalten, nur selten aber zukünftigen Angriffsvektoren vorbeugen.

Angriffspunkte und Angriffsmethoden bei Software entwickeln sich permanent weiter. Somit steigen auch die Anzahl und der Umfang möglicher Bedrohungen weiter an. Hinsichtlich des Merkmals der Sicherheit kann Software daher nicht »reifen«, im Sinne irgendwann das höchste Maß an Sicherheit erreicht zu haben. Sie kann nur versuchen, Schritt zu halten mit der Weiterentwicklung seitens der Angreifer und proaktive Maßnahmen zu ergreifen, bspw. durch eine Risikoreduktion durch Security-by-Design, um zukünftige Bedrohungsszenarien zu adressieren.

5 Herstellerland unabhängig

Wie in den Fragen zuvor dargestellt, agieren Softwarehersteller nach marktwirtschaftlichen Prinzipien, was grundsätzlich eine globale Entwicklung softwarebasierter Produkte mit einschließt. Wie in anderen Branchen und bei anderen Produkten üblich, findet auch Softwareentwicklung über geographische und politische Grenzen hinweg statt. Dabei sind Formen des Outsourcings, wie Near- und Offshoring mittlerweile gängig. In der gesellschaftlichen Wahrnehmung hat sich allerdings das Gütesiegel »Made in Germany« als besondere Auszeichnung qualitativ hochwertiger Produkte etabliert, was im Umkehrschluss zuweilen den Eindruck entstehen lässt, dass Software umso unsicherer ist, je mehr davon in Offshore-Ländern umgesetzt wird. Damit geht auch die Frage einher: **Wenn die Hersteller alles in Deutschland machen würden, wäre dann nicht die Sicherheit von softwarebasierten Produkten besser?**

»Die Sicherheit softwarebasierter Produkte hängt nicht vom Herstellerland ab.«

Die Verknüpfung der Bezeichnung »Made in Germany« mit hoher Qualität sollte in zweierlei Hinsicht differenziert betrachtet werden. Zum einen muss bei der Betrachtung des Zusammenhangs zwischen Made in Germany und der Produktqualität gesagt werden, dass es keine belastbaren Studien gibt, die besagen, dass Software, die in Deutschland entwickelt wird, »besser« (im Sinne von besserer Qualität oder Sicherheit) ist als Software, die Offshore entwickelt wurde. Softwareunternehmen bedienen sich aus unterschiedlichen Gründen global agierender Ressourcen. Dies mag vordergründig aus Kostengründen heraus geschehen. Speziell in Deutschland kommt heute in jedem Fall der Fachkräftemangel hinzu. Es gibt keine Studie, dass Software, an der globale Entwickler mit gearbeitet haben, mehr Fehler aufweist, als solche, die nur in Deutschland entwickelt wird.

Zum anderen inkludiert das Verständnis des klassischen »Made in Germany« fälschlicherweise, dass Offshore bei Produkten mit diesem Siegel nicht zum Einsatz kommt, was falsch ist. Der Begriff Made-in-Germany wird zumeist mit dem deutschen Industriebereich und hier vor allen Dingen der Automobilbranche assoziiert. Durch ein geschicktes Marketing ist es gelungen, das »Made in Germany« in der Öffentlichkeit als ein »ohne Outsourcing« zu verstehen. In der Praxis ist dies jedoch bei weitem nicht so. Zahlreiche Unternehmen, bspw. gerade auch im Automobilbau, haben nur noch eine sehr geringe Wertschöpfungstiefe und eine Vielzahl an zugelieferten Komponenten und Bauteilen wird durch Unternehmen aus unterschiedlichen Ländern beigesteuert. Durch eine global vernetzte Welt werden in nahezu keinem Bereich alle Materialien und Arbeitsstunden in einem Land zu einem Produkt vereint. Insbesondere unter Berücksichtigung marktwirtschaftlicher Prinzipien ist eine Herstellung softwarebasierter Produkte »Made in Germany« kaum noch vorstellbar.

6 Produkthaftung

Durch die bisherigen Darstellungen (Software braucht Updates und Patches, Software ist nie fehlerfrei, Softwareentwicklung orientiert sich an marktwirtschaftlichen Vorgehensweisen) wird deutlich, dass es im Regelfall keine Garantien für das einwandfreie Funktionieren einer Software geben kann, insbesondere, wenn der Kontext der zukünftigen Verwendung nicht klar definiert werden kann, und sich Software zunehmend nicht vorhersehbaren Bedrohungen gegenübersieht. Dennoch stellt sich die Frage nach den Konsequenzen, wenn softwarebasierte Produkte ursächlich für eingetretene Schäden sind. Wer kann zur Verantwortung gezogen werden? Ein Softwarenutzer kann Sicherheitslücken oder Qualitätsmängel an einer Software nicht selbst beheben, sondern er ist auf Sicherheitsupdates durch den Hersteller angewiesen. Somit liegt der Gedanke nahe, dass insbesondere wenn durch Sicherheitslücken in softwarebasierten Produkten Schäden entstehen, grundsätzlich die Haftung durch den Hersteller möglich sein sollte. Dies ist aktuell nicht der Fall. Daher stellt sich die Frage, **warum es keine Produkthaftung für softwarebasierte Produkte gibt bzw. was aktuell gegen Produkthaftung für Software spricht?**

»Eine Produkthaftung für softwarebasierte Produkte muss Betrieb und Regulatorien berücksichtigen.«

Die Beantwortung dieser Frage muss zumindest von zwei unterschiedlichen Standpunkten aus erfolgen:

a) Vom Standpunkt der vertraglichen Einhaltung: Hier existiert bereits für vertraglich klar festgelegte Punkte eine Produkthaftung, was allerdings keine beliebige Produkthaftung bedeutet.

Einem Softwarehersteller kann nur dann ein Vorwurf gemacht werden, wenn er vertragliche Vereinbarungen, Gesetze oder Standards, zu deren Einhaltung er sich verpflichtet hat oder damit wirbt, verletzt hat. In diesem Zusammenhang kommt Industriestandards, wie beispielsweise PCI DSS (Payment Card Industry Data Security Standard), eine große Bedeutung zu: Dem Hersteller einer Software, die Kreditkartendaten verarbeitet, kann Fahrlässigkeit vorgeworfen werden, wenn er diesen Standard der Kartengesellschaften nicht einhält. Hält er diesen sehr strengen Standard ein und unterzieht seine Software regelmäßig den geforderten Tests, sind Sicherheitslücken zwar nicht völlig auszuschließen, er hat aber angemessene Maßnahmen ergriffen und Vorwürfe sind ungerechtfertigt. Sobald ein Fehler auftaucht, der darin begründet ist, dass er den Standard nicht eingehalten hat, greift die Haftung.

b) Vom Standpunkt des Betriebs: Hier sind deutlich mehr Komponenten beteiligt, als nur das System auf das sich die Produkthaftung beziehen würde.

Die Norm ISO/IEC 25010 definiert die Sicherheit eines Softwareprodukts wie folgt: »Grad, in dem ein Produkt oder System Informationen und Daten schützt, so dass Personen sowie andere Produkte oder Systeme Datenzugriffe in dem Ausmaß haben, das ihrer Art und Berechtigungsstufe angemessen ist«. Eine Software kann also grundsätzlich erst dann »unsicher« werden, wenn sie in einer Ablaufumgebung betrieben wird (unabhängig davon, ob es sich dabei um ein Smartphone oder ein Rechenzentrum handelt) und wenn sie Daten verarbeitet oder speichert. Somit ist eine isolierte statische Betrachtung der Sicherheit von »Software« nicht sinnvoll und greift insbesondere für eine Produkthaftung völlig ins Leere.

Ein auf einem Datenträger gespeichertes Programm ist immer sicher, solange es nicht ausgeführt wird. Ein Softwarehersteller kann nicht dafür zur Verantwortung gezogen werden, wenn seine Software in einer unsicheren Umgebung betrieben wird oder die Anwender Daten erfassen, die hinsichtlich ihrer Kritikalität nicht zur Erfassung vorgesehen sind. Softwarehersteller müssten im Fall einer Produkthaftung ähnlich einem »Beipackzettel« auf die vorgesehene Anwendung ihrer Software und auf Risiken und Nebenwirkungen hinweisen.

Neu sind bezogen auf diesen Punkt Cloudlösungen, da hier ein Großteil der Laufzeitumgebung ebenfalls vom Lieferanten bereitgestellt wird. Passenderweise sind bzgl. solcher Lösungen heute auch schon sehr viel restriktivere Service Level Agreements, also vertraglich festgelegte Eigenschaften, üblich, als dies bei On-Premise-Lösungen jemals der Fall war. Eine Produkthaftung kann folglich umso besser funktionieren, je mehr die Herstellerempfehlungen hinsichtlich der Ablaufumgebung und der Nutzung dokumentiert sind und ihre Einhaltung nachvollziehbar ist, wie dies z. B. bei Cloudlösungen der Fall ist.

7 Sicherheitsrisiken managen

Es fällt auf, dass Sicherheitsprobleme vor allen Dingen im IT-Bereich zu dominieren scheinen. In anderen komplexen Produkten wie Fahrzeugen und Flugzeugen scheint Sicherheit beherrschbar zu sein. Daher stellt sich die Frage, **warum Sicherheit in anderen Bereichen scheinbar besser funktioniert als im Softwarebereich.**

»Das Management von Sicherheitsrisiken ist auch außerhalb des IT-Bereichs schwierig.«

Bei der Betrachtung dieses Vergleichs vermischen sich mindestens drei Aspekte, die zu diesem verzerrten Bild führen:

a) Für Produkte wie Flugzeuge oder Roboter existieren heute starke Regularien, die ein Vielfaches an Qualitätssicherungsmaßnahmen (und -kosten) erzwingen.

Die Mechanismen des freien Marktes, die es jedem Softwarehersteller letztlich freistellen, wieviel Aufwand in die Qualitätssicherung investiert wird, sind in solchen IT-Systemen, die physischen Schaden bewirken können, durch entsprechende Regularien massiv eingeschränkt. In der IT wird diese Unterscheidung durch die Begriffe Security, die die Verfügbarkeit, Integrität und Vertraulichkeit von Informationen adressiert, und Safety, die die physische Unversehrtheit adressiert, realisiert. So muss Software, die dazu beiträgt, dass ein Flugzeug in der Luft bleibt, einigen Regularien genügen (z. B. STANAG 4671, DO-178-B, DO-178C oder DO-254). Diese erzwingen teilweise sehr klare Qualitätssicherungsmaßnahmen wie z. B. Pair-Review, wonach jede Zeile Code von mindestens einem weiteren Programmierer freigegeben werden muss, Testcode-Abdeckungen, wonach so intensiv getestet werden muss, dass wenigstens jede Anweisung und jede Bedingung einmal nachweislich durchlaufen werden muss, oder auch das Verbieten besonders moderner Programmierung (z. B. Polymorphie in objektorientierten Sprachen), da diese nicht transparent genug sind. Dies führt dazu, dass üblicherweise in solchen Systemen mehr als 70 Prozent des gesamten Projektbudgets in Qualitätssicherung investiert wird, wodurch potentiell mehr Fehler früher erkannt werden können. In softwarebasierten Produkten, die nicht solchen Regularien unterworfen sind, liegt die Quote des in Qualitätssicherung investierten Projektbudgets deutlich unterhalb von 70 Prozent.

b) In der Vergangenheit wurden Produkte häufig für einen »standalone«-Betrieb entwickelt. Die fehlende Vernetzung dieser Produkte mit anderen verhinderte in der Vergangenheit viele Einbruchszenarien. Dies ändert sich gerade.

Während der klassische EDV-Arbeitsplatz in den frühen 80-iger Jahren noch nicht vernetzt war und damit auch kaum Möglichkeiten für Security-Vorfälle bot, hat sich die IT heute dramatisch vernetzt: Und dieser Trend hält weiterhin an: Studien zufolge hatte ein typisches IT-System 2012 schon 33 Schnittstellen zu anderen Systemen, die damit prinzipiell eine Security-Gefahr bedeuten.[1] Diese Zahl ist 2015 bereits auf 54 angewachsen. Trends wie das Internet-der-Dinge oder Industrie 4.0 werden diese Entwicklung noch weiter unterstützen. Sollten sich die Parameter der Softwareentwicklung (Märkte, Möglichkeiten etc.) nicht signifikant ändern, ist leider davon auszugehen, dass die Anzahl von Sicherheitsvorfällen auch in nicht typischen IT-Domänen steigt.

1 vgl. Theresa Lanowitz, Lisa Dronzek: »Market Snapshot Report: Service Virtualization«, January 2015

c) Auch andere Produkte haben Sicherheitsprobleme.

Natürlich haben auch schon heute viele klassische Systeme, in denen IT verbaut ist, Sicherheitsprobleme. Stuxnet ist hier ebenso ein Beispiel wie die Vorfälle rund um Keyless-driven Cars und Connected Cars. Auch die Cyberattacken auf unterschiedliche Industrien, in Deutschland hier vor allen Dingen der ferngesteuerte Fernofen, belegen, dass das Thema Security auch dort relevant ist.[2]

2 vgl. http://www.heise.de/security/meldung/BSI-Sicherheitsbericht-Erfolgreiche-Cyber-Attacke-auf-deutsches-Stahlwerk-2498990.html

8 Gesetzgeber muss mitspielen

Es wird deutlich, dass die Entwicklung und der Betrieb von Software eine zunehmend komplexer werdende Herausforderung darstellt. **Welche Rolle spielt dann der Staat, wenn es um die Sicherheit von Software geht und welche gesetzlichen Rahmen existieren bzgl. der Sicherheit von Software?**

Auch bei dieser Frage kann eine Antwort nur differenziert erfolgen.

a) Zum Schutze der Gesellschaft und jedes einzelnen Menschen müssen alle in Umlauf gebrachten Produkte Mindestkriterien erfüllen. Für den Softwarebereich ist die Entwicklung entsprechender Kriterien noch im Fluss.

»Der Gesetzgeber kann zur Sicherheit softwarebasierter Produkte entscheidend beitragen.«

Es ist die Pflicht der Gesetzgeber, der IT-Industrie eindeutige und angemessene Vorgaben hinsichtlich Sicherheitstechnik und Sicherheitsmanagement zu machen, wo anderenfalls der Gesellschaft und/oder dem einzelnen Bürger eine große Gefahr droht. In anderen Bereichen ist dies über Mindestanforderungen wie z. B. das GS-Prüfzeichen seit Jahren etabliert. Ohne solche Vorgaben ist bspw. auch eine Produkthaftung nicht möglich.

Das bedeutet für das geforderte Regulativ im Bereich softwarebasierter Produkte, dass hier der Gesetzgeber einen dringenden Nachholbedarf hat. Erst wenn er hier seinen Pflichten nachkommt und mit der Geschwindigkeit des technischen Fortschritts Schritt hält, wird auch »Sicherheit« vergleichbar, prüfbar und kann bei Vernachlässigung durch Hersteller oder Betreiber von Software geahndet werden.

Um die IT-Sicherheit zu erhöhen, hat der Gesetzgeber bereits einige Sicherheitsgesetze auf den Weg gebracht. So wurde im Jahr 2015 das erste ↗ IT-Sicherheitsgesetz verabschiedet, welches neben einer Meldepflicht für Sicherheitsvorfälle auch die Implementierung von branchenspezifischen Sicherheitsstandards nach Stand der Technik bei den Betreibern Kritischer Infrastrukturen vorsieht. Auch die ↗ Betreiber von Telemediendiensten haben ihre Angebote nach Stand der Technik abzusichern.

Auf europäischer Ebene verfolgt ↗ die Richtlinie zur Erhöhung der Netzwerk- und Informationssicherheit (NIS) ein ähnliches Ziel, zu der das deutsche IT-Sicherheitsgesetz als ↗ nationale Umsetzung der Richtlinie verstanden werden kann.

Die Sicherheit von Software und Softwareentwicklung selbst wird jedoch von beiden genannten nicht direkt adressiert.

Einfluss auf die Softwareentwicklung könnte die neue ↗ EU-Datenschutz-Grundverordnung (DS-GVO) haben. Sie ist nach jahrelangen Verhandlungen in Brüssel im Mai 2016 in Kraft getreten und wird nach einer Übergangsfrist von zwei Jahren ab dem 25. Mai 2018 für alle Unternehmen gelten, die in Europa Produkte und Dienste anbieten und dazu Daten europäischer Bürger verarbeiten.

In der Verordnung wurde das Prinzip »Data Protection by Design and by Default« (Art. 25 DS-GVO) verankert. Dieses schreibt vor, dass die Datenschutzgrundsätze wie z. B. Datenminimierung durch geeignete technische und organisatorische Maßnahmen bei der Konzipierung von Datenverarbeitungsprozessen umgesetzt und sichergestellt werden müssen. Damit gibt es eine konkrete »technische« Umsetzungvorgabe, für die schon bisher bekannten Datenschutzprinzipien. Das kann zum Beispiel bedeuten, dass Maßnahmen zur Sicherung vor unbefugtem Zugriff ergriffen werden oder bestimmte Funktionen einer Software zunächst einmal standardmäßig ausgeschaltet sind und nur zur Freischaltung für den Nutzer vorgesehen werden.

Eine weitere Neuerung ist die Einführung der verpflichtenden Datenschutzfolgenabschätzung (Art. 35 DS-GVO) sowie ein eigener Artikel zur Sicherheit der Datenverarbeitung (Art. 32 DS-GVO). Beide verfolgen einen risikobezogenen Ansatz. Art. 32 verpflichtet den für die Datenverarbeitung Verantwortlichen geeignete technische und organistorische Maßnahmen zu treffen, um ein dem Risiko angemessenes Schutzniveau zu gewährleisten. Art. 35 verpflichte den Verantwortlichen bei besonders risikobehafteten Datenverarbeitungen (wie z. B. der umfangreichen Verarbeitung von Gesundheitsdaten) bereits vor Durchführung der Datenverarbeitung eine umfassende Analyse der Risiken für die Rechte und Freiheiten der Betroffenen, die durch die Verarbeitung entstehen könnten, durchzuführen, zu dokumentieren und geeignete Maßnahmen zu ergreifen, um diese Risiken zu minimieren.

> b) Für eine Chancengleichheit auf dem gerade in der IT internationalen Markt darf keine Wirtschaft benachteiligt sein. Hier entscheidet häufig noch das schwächste Glied der Wertschöpfungskette über das Gesamtregulativ.

Jegliche Rahmen durch die Politik müssen gerade in der IT berücksichtigen, dass es einen globalen Markt gibt: Strengere Vorgaben ausschließlich in Deutschland würden das große Risiko bergen, dass diese Produkte durch günstigere Produkte aus dem Ausland, in dem diese Vorgaben nicht gelten, substituiert würden. Die Möglichkeit, Einfuhrbeschränkungen für solche Produkte zu erlassen, ist über Clouddienste, bei denen die Produkte faktisch nicht eingeführt werden, effektfrei. An dieser Stelle können neue Regularien nur international eingeführt werden. Selbst rein europäische Vorgaben laufen hier Gefahr, die Vormachtstellung der US-Software-Industrie weiter auszubauen. Hier gilt es also weiter, auf internationalem Parkett einen Minimalkatalog von Mindestanforderungen zu erarbeiten.

Zudem darf es keine Pflicht zur Implementierung staatlicher Backdoors geben. Die Hersteller haben ein starkes Interesse daran, dass ihre Produkte vertrauenswürdig sind. In diesem Sinne dürfen Softwarehersteller nicht zum Erfüllungsgehilfen staatlicher Sicherheitsinteressen gemacht werden.

9 Zusammenfassung – Q & A

1. Warum erfährt fast jedes softwarebasierte Produkt dauernd Patches?

Zum einen dienen Patches nicht nur zum Korrigieren von Sicherheitslücken, sondern es werden häufig auch neue Funktionen ergänzt. Durch Updates und Patches werden Produkte an sich verändernde Anforderungen angepasst und Produkte können zunächst so angeboten werden, dass sie einen guten Kompromiss aus Funktionalität, Preis und schneller Verfügbarkeit darstellen.

2. Wird zu wenig in Qualitätssicherung bei der Softwareentwicklung investiert?

Sicherheitslücken sind nicht zwingend ein Ausdruck mangelnder Qualitätssicherung: Neue Einsatzzwecke und neue Betriebsumgebungen bringen häufig neue Sicherheitsanforderungen mit sich, die beim ursprünglich intendierten Einsatzszenario aber nicht absehbar waren. Mit Konzepten wie Security-by-Design und Security-by-Default wird der Aspekt der Sicherheit bereits von Anfang an bei der Entwicklung von softwarebasierten Produkten berücksichtigt. Allerdings kann Sicherheit auch als Kostenfaktor verstanden werden, so dass Hersteller zwischen verschiedenen Aspekten wie Qualität, Sicherheit, Kosten und langfristiger Marktteilnahme abwägen müssen. Zudem orientieren sich Hersteller an der Marktnachfrage, welche in Teilen ein geringes Maß an Qualität und Sicherheit in Kauf zu nehmen scheint, so dass nur in notwendigem Umfang in Qualitätssicherung bei der Softwareentwicklung investiert wird. Qualitätssicherung ist daher auch ein Kostentreiber. Hier kann der Kunde aber durch sein Kaufverhalten mitbestimmen, welchen Kostenaufwand die Hersteller für diese Produkte betreiben.

3. Warum ist Software denn nie fehlerfrei?

Je nach Verständnis des Begriffes »Fehler« trifft es zu, dass Software nie fehlerfrei ist. Insbesondere wenn eine nicht spezifizierte Außenwirkung auftritt oder gar eine (implizite) Erwartungshaltung nicht erfüllt wird und dies als Fehler verstanden wird, wird Software nie fehlerfrei sein. Zudem lassen sich menschliche Fehler nicht vermeiden und können sich somit bspw. auch im Design, der Architektur oder im Quellcode einer Software wiederfinden. Ziel ist es jedoch immer, diese Fehler zu erkennen und deren Anzahl zu reduzieren sowie deren Ausmaß zu begrenzen.

4. Reift Software erst beim Kunden?

Durch immer kürzere Veränderungszyklen bei Software, durch den zunehmenden Einsatz von Software auch in unbekanntem Nutzungskontext sowie durch eine sich verschärfende Bedrohungslage kann der Eindruck entstehen, dass Software erst beim Kunden reift. Regelmäßige Anpassungen sind notwendig, um dem Softwarenutzer ein bestmögliches Produkt anzubieten, allerdings orientieren sich Hersteller dabei an der Marktnachfrage nach Sicherheit.

5. Gibt es keine »Security made in Germany«?

In einer global agierenden Welt kann kein softwarebasiertes Produkt vollständig in Deutschland entwickelt werden. Es gibt keine Belege dafür, dass in Deutschland entwickelte Software »besser« wäre, als Offshore entwickelte Software. In diesem Sinne würde »Security made in Germany« keine bessere Qualität oder Sicherheit im Sinne eines Automatismus liefern. Auch bei Produkten »Made in Germany« findet ein Teil der Wertschöpfung nicht in Deutschland statt, so dass grundsätzlich ein äquivalentes Siegel »Security Made in Germany« nicht mit »ohne Outsourcing hergestellt« gleichgesetzt werden darf.

6. Warum gibt es keine Produkthaftung für softwarebasierte Produkte?

Für vertraglich festgelegte Merkmale existiert bereits jetzt eine Produkthaftung, die allerdings nicht beliebig erweitert werden kann. Zudem greift eine Haftung bei Fehlern, die durch das Nichteinhalten von Standards begründet sind. Ein wesentlicher Faktor bei der Sicherheit softwarebasierter Produkte ist ein sicherer Betrieb, für den ein Softwarehersteller nicht zur Verantwortung gezogen werden kann, da eine Software mit vielen weiteren Komponenten zusammen agiert und der Hersteller keinen Einfluss auf die Gestaltung der tatsächlichen Betriebsumgebung einer Software hat. Bei Cloudlösungen ist dies anders, da hier ein Großteil der Laufzeitumgebung ebenfalls vom Lieferanten bereitgestellt wird.

7. Warum funktioniert Sicherheit in anderen Bereichen besser als im Softwarebereich?

Diese Wahrnehmung ist zum einen für die Bereiche zutreffend, wo eine Vielzahl an Regularien Security und Safety sicherstellen sollen. Dies bewirkt allerdings, dass ein überdurchschnittlicher Anteil von Projektbudgets in die Qualitätssicherung von Software investiert wird. Zudem wird immer weniger Software standalone betrieben und mit jeder Schnittstelle zu anderen Systemen steigt potenziell auch die Gefährdung durch Software. Zuletzt ist damit zu rechnen, dass durch die zunehmende Durchdringung aller Lebensbereiche mit IT und Software auch dort tendenziell Sicherheitsvorfälle zu erwarten sind.

8. Welche Rolle spielt der Staat, wenn es um die Sicherheit von Software geht?

Der Staat sollte eindeutige und angemessene Vorgaben hinsichtlich Sicherheitstechnik und Sicherheitsmanagement für softwarebasierte Produkte erlassen. Allerdings ist eine nationale Lösung nicht zielführend, da Softwareentwicklung im internationalen Raum stattfindet. Es muss daher das Ziel sein auf internationaler Ebene einen Katalog von Mindestanforderungen zu erarbeiten.

Bitkom vertritt mehr als 2.400 Unternehmen der digitalen Wirtschaft, davon 1.600 Direktmitglieder. Sie erzielen mit 700.000 Beschäftigten jährlich Inlandsumsätze von 140 Milliarden Euro und stehen für Exporte von weiteren 50 Milliarden Euro. Zu den Mitgliedern zählen 1.000 Mittelständler, mehr als 300 Start-ups und nahezu alle Global Player. Sie bieten Software, IT-Services, Telekommunikations- oder Internetdienste an, stellen Hardware oder Consumer Electronics her, sind im Bereich der digitalen Medien oder der Netzwirtschaft tätig oder in anderer Weise Teil der digitalen Wirtschaft. 79 Prozent der Unternehmen haben ihren Hauptsitz in Deutschland, weitere 9 Prozent kommen aus Europa, 8 Prozent aus den USA. 4 Prozent stammen aus Asien, davon die meisten aus Japan. Bitkom fördert die digitale Transformation der deutschen Wirtschaft und setzt sich insbesondere für eine innovative Wirtschaftspolitik, eine Modernisierung des Bildungssystems und eine zukunftsorientierte Netzpolitik ein.

Bundesverband Informationswirtschaft,
Telekommunikation und neue Medien e.V.

Albrechtstraße 10
10117 Berlin
T 030 27576-0
F 030 27576-400
bitkom@bitkom.org
www.bitkom.org