Kernkompetenzzentrum
Finanz- & Informationsmanagement

Fraunhofer FIT
Projektgruppe
Wirtschaftsinformatik

Mehr Sicherheit durch Open Source – Irrweg oder Zielgerade?

von

Hans Ulrich Buhl, Jochen Dzienziol

2003

in: Wirtschaftsinformatik, 45, 4, 2003, p. 474-482

Universität Augsburg, D-86135 Augsburg
Besucher: Universitätsstr. 12, 86159 Augsburg
Telefon: +49 821 598-4801 (Fax: -4899)

Universität Bayreuth, D-95440 Bayreuth
Besucher: Wittelsbacherring 10, 95444 Bayreuth
Telefon: +49 921 55-4710 (Fax: -844710)

www.fim-rc.de

WI-872

Meinung/Dialog

Mehr Sicherheit durch *Open Source* – Irrweg oder Zielgerade?

Open-Source-Software (OSS) – Software, deren Entstehung sich durch die weltweit freiwillige Mitwirkung einer Vielzahl von Programmierern deutlich von der Entwicklung proprietärer Software unterscheidet – ist in den letzten Jahren ein Thema geworden, mit dem sich Fachzeitschriften, Tagungen und wissenschaftliche Veröffentlichungen vermehrt beschäftigen. Dies steht insbesondere mit der Erfolgsgeschichte von OSS-Vertretern wie dem Apache-Webserver oder dem Betriebssystem Linux im Zusammenhang. Dabei beschreibt Open Source nicht nur die natürlich wichtige Tatsache, dass der Quellcode der betreffenden Programme zur Einsicht und Durchführung von Veränderungen offen liegt, gemäß der Open-Source-Initiative gehören auch die Freiheit zur beliebigen Nutzung und zur unveränderten oder veränderten Weitergabe der Software zu ihren Charakteristika.

Doch selbst wenn OSS insbesondere erst in den letzten 5 Jahren beständig steigende Aufmerksamkeit erfuhr, handelt es sich dabei keinesfalls um eine neue Bewegung. Vielmehr war der freie Austausch von Software sowie das gegenseitige Profitieren von Neuerungen in den Ursprüngen der Softwareentwicklung üblich. Im Zuge der rasant steigenden Bedeutung der Datenverarbeitung wurden jedoch die Entwicklung und der Verkauf von Software schnell zu einem lukrativen Geschäftsfeld, wobei in den Lizenzen die Nutzung, der Zugriff und die Weitergabe der erworbenen Software beschränkt wurden. Als bewusste Gegenbewegung und zur Wahrung der ursprünglichen Kultur entstand – insbesondere forciert durch die Bemühungen von Richard Stallman, ehemaliger Wissenschaftler am MIT und Gründer der *Free Software Foundation* – Mitte der achtziger Jahre eine Gruppierung von Programmierern, die sich mit der gemeinschaftlichen Entwicklung von „free software“ befasste. Hieraus ging 1998 – hauptsächlich wohl aus Marketinggründen gegenüber Vertretern aus der Unternehmenswelt, für welche der Begriff „freie Software“ negative Assoziationen hervorrufen könnte – die Bezeichnung „Open Source“ hervor [Krog03]. Um zu verhindern, dass in der weiteren Entwicklung der Zugriff und die Verwendung von bestimmter OSS durch die Umwandlung in proprietäre Software restringiert wird, wurde eine Lizenz geschaffen, welche die mit OSS verbundenen Freiheiten schützt: die General Public License, die aufgrund Ihrer Zielsetzung häufig als „copyleft“ anstatt als copyright bezeichnet wird [Ljun00].

Mittlerweile ist OSS in vielen Unternehmen nicht mehr wegzudenken. So arbeiten bereits 36 % der deutschen Unternehmen mit Linux [Hutt03]. Große Hardwarehersteller wie IBM, Sun und Hewlett Packard nahmen Open Source durch den Start umfangreicher Initiativen in ihre Geschäftsstrategie mit auf, und sogar in vielen öffentlichen Verwaltungen wie in Norwegen, Frankreich und auch in Deutschland wird die Verbreitung von Linux aktiv gefördert. Als einer der Gründe für den Einsatz von Linux wird in einer Befragung der Meta Group von IT-Managern – neben der Hoffnung auf Kosteneinsparungen – das Thema Sicherheit angeführt [oV02]. Interessant ist beispielsweise auch, dass die neue Sichere Inter-Netzwerk-Architektur (SINA), welche die deutschen Auslandsvertretungen miteinander verbinden wird, auf einer angepassten Linux-Plattform basiert und der Sicherheitsdienstleister VeriSign seine Produkte auf Linux migrieren will [oV03]. Auf den ersten Blick mag es verwundern, wie in einem Softwareentwicklungsprozess in der Linux-Gemeinde von freiwilligen Programmierern, welche *Eric S. Raymond* als „großer, wild durcheinander plappernder Basar von verschiedenen Zielsetzungen und Ansätzen“ [Raym98] beschreibt, Software entstehen soll, welche in Bezug auf ihre Sicherheit proprietärer Software, deren Entwicklung vermutlich eher mit dem sorgfältig geplanten Bau einer Kathedrale vergleichbar ist, überlegen sein kann. Und tatsächlich ist die Frage, ob die Sicherheit von OSS wie bspw. Linux höher als die vergleichbarer, proprietärer Softwareprodukte ist, trotz (oder gerade wegen) des Vorteils des einsehbaren und veränderbaren Quellcodes, der niedrigen Bug-Dichte und der früher seltenen, nun aber zunehmenden Anzahl an Hackerangriffen nach wie vor umstritten. Für die ausgewogene Beurteilung spielt insbesondere das Begriffsverständnis von Sicherheit eine große Rolle.

Der Begriff Sicherheit umfasst vier Bestandteile [StGr97]: Erstens erfordert die Sicherheit die grundsätzliche Betriebsbereitschaft (availability). Zweitens muss auch die Funktionsbereitschaft (reliability) gewährleistet sein, die im Gegensatz zur Betriebsbereitschaft nicht nur die grundlegende Funktionstüchtigkeit voraussetzt, sondern auch die Sicherstellung korrekter Ergebnisse. Die Forderung der Vertraulichkeit (confidentiality) stellt den dritten Aspekt des Begriffs „Sicherheit“ dar, die der Integrität (integrity), also des Datenschutzes und der Datensicherheit, schließlich den vierten.

Unter diesen Kriterien können alle Arten von Sicherheitsrisiken eingeordnet werden. Beispiele wie Schäden durch höhere Gewalt oder durch Havarien (z. B. Brand), die auch die Betriebsbereitschaft gefährden, sind zwar kaum durch die eingesetzte Software beeinflussbar, jedoch werden andere Aspekte, wie z. B. Vandalismus (durch einen Cracker) sehr wohl durch die Art der eingesetzten Software beeinflusst.

Betrachtet man die Sicherstellung der vier Aspekte über einen längeren Zeitraum, so lassen sich ebenfalls strategische/betriebswirtschaftliche Anforderungen wie Planungssicherheit und Investitionssicherheit unter dem Sicherheitsbegriff subsumieren. Beispielsweise ist für ein Unternehmen die Verfügbarkeit neuer Softwareversionen, die diese Sicherheitskriterien auch in Zukunft erfüllen, enorm wichtig.

Insgesamt bleibt zu bemerken, dass nahezu jedes Sicherheitsniveau prinzipiell erreichbar, allerdings mit einem finanziellen Trade-off in Form von höheren Kosten für Installation, Wartung und spezifische Anpassungen verbunden ist. Die Höhe dieser Kosten wird ebenfalls durch die eingesetzte Software beeinflusst. Einerseits bietet hier OSS einem Unternehmen prinzipiell die Möglichkeit, den Quelltext durch die eigene IT-Abteilung mit beliebig skalierbarem Aufwand auf Schwachstellen überprüfen zu lassen. Gegebenenfalls ist hier auch eine kostengünstige Zusammenarbeit mit den Entwicklern der OSS möglich, die für die Hilfe bei der Verbesserung ihrer Software dankbar sind. Andererseits können mit einem Anbieter kommerzieller Software Haftungs- und Wartungsverpflichtungen vertraglich vereinbart werden, während OSS-Entwickler prinzipiell zwar für ihre Hilfe bezahlt werden können, Haftung allerdings in der Regel ausschließen. Zwischen diesen beiden Extremen existieren am Markt noch weitere Alternativen, wie der professionelle Support für OSS durch spezialisierte IT-Dienstleister oder aber beispielsweise das Shared-Source-Angebot von Microsoft, das Kunden gegen Entgelt Einsicht in die Quellen der Produkte gewährt. Welche dieser Alternativen im Einzelfall vorzuziehen ist, ist von den konkreten Gegebenheiten abhängig. Allerdings ist bei OSS der Kunde in keinem Fall von einem einzigen Anbieter abhängig.

Ein Hauptgrund für die strittige Beurteilung der Sicherheitsfrage im Zusammenhang mit Open Source ist die Tatsache, dass einerseits die angesprochenen Aspekte des Sicherheitsbegriffs je nach Anwendungsgebiet unterschiedliche Bedeutung besitzen und anderer-

seits OSS und proprietäre Software unterschiedliche Eigenschaften hinsichtlich dieser Aspekte aufweisen. Dies soll in der heutigen Ausgabe von Meinung/Dialog näher diskutiert und von verschiedenen Seiten beleuchtet werden. Hierbei haben wir versucht, durch die gewonnenen Autoren eine möglichst differenzierte, von emotionalen Glaubensbekenntnissen hinsichtlich OSS oder proprietärer Software losgelöste Betrachtung zu motivieren. Lesen Sie im Folgenden die Beiträge von:

- Hermann-Josef Lamberti, Chief Operating Officer bei der Deutsche Bank AG, verantwortlich für Kosten- und Infrastrukturmanagement, Informationstechnologie, Operations, Gebäude- und Flächenmanagement sowie Einkauf,
- Dr. Udo Helmbrecht, Präsident des Bundesamtes für Sicherheit in der Informationstechnik,
- Harald Lieder, Accenture GmbH, Leiter des Bereichs Global Architecture & Core Technologies im deutschsprachigen Raum (ASG),
- Dr. Thomas Mangel, Chief Technology Officer, Postbank Systems AG und
- Richard Seibt, Chief Executive Officer, SuSE Linux AG.

Wenn auch Sie zu diesem Thema oder einem Artikel der Zeitschrift Wirtschaftsinformatik Stellung nehmen möchten, dann senden Sie Ihre Stellungnahme (max. 2 DIN A4 Seiten, gerne auch als E-Mail) bitte an den Hauptherausgeber, Prof. Dr. Wolfgang König, Universität Frankfurt am Main, E-Mail: koenig@wiwi.uni-frankfurt.de

Literatur (Auswahl)

[Raym98] *Raymond, Eric S.:* The Cathedral and the Bazaar. http://www.phone-soft.com/RaymondCathedralBazaar/catb_g.0.html, 1998-08-08, Abruf am 2003-05-17.

[Ljun00] *Ljungberg, Jan:* Open Source Movements as a Model for Organizing. In*: Bichler, M.; Hansen, H.-R.; Mahrer, H.* (Hrsg.): Proceedings of the 8th European Conference on Information Systems (ECIS) 2000. Gabler, Wiesbaden 2000, S. 501–508.

[Krog03] *von Krogh, Georg:* Open-Source Software Development. In: MIT Sloan Management Review 44 (2003) 3, S. 14–18.

[oV02] *ohne Verfasser:* Linux profitiert von der Branchenkrise. In: Computerwoche (2002) 41, S. 54.

[StGr97] Stichwort Sicherheitsmanagement. In: *Stickel, Eberhard; Groffmann, Hans-Dieter; Rau, Karl-Heinz* (Hrsg.): Gabler Wirtschaftsinformatiklexikon. Gabler, Wiesbaden 1997, S. 645–646.

[oV03] *ohne Verfasser:* Deutsche Botschaften stellen weltweit auf Linux und VPN-Tunelling um. http://www.de.internet.com/index.html?id=2018845, 2003-01-21, Abruf am 2003-05-17.

[Hutt03] *Huttenloher, Rainer:* Kostenersparnis öffnet Linux die Türen. In: Computer Zeitung 34 (2003) 6.

Prof. Dr. Hans Ulrich Buhl,
Dipl.-Kfm. Jochen Dzienziol, M.Sc.,
Lehrstuhl für Betriebswirtschaftslehre,
Wirtschaftsinformatik
& Financial Engineering,
Kernkompetenzzentrum
IT & Finanzdienstleistungen,
Universität Augsburg

Open Source – eine Alternative zu kommerziell lizenzierter Software?
von Hermann-Josef Lamberti

Open-Source-Software erobert immer breitere Einsatzbereiche in Wirtschaft und öffentlicher Verwaltung. Insbesondere das freie Betriebssystem Linux fand in den vergangenen Jahren eine zunehmende Verbreitung außerhalb seiner angestammten Einsatzgebiete im akademischen Umfeld. Aber auch andere Open-Source-Anwendungen werden teils schon seit Jahren in der Wirtschaft erfolgreich eingesetzt – allen voran der Webserver Apache mit einer weltweiten Marktdurchdringung von über 60 %. Gleiches gilt für die in der breiten Öffentlichkeit weniger bekannten Werkzeuge wie sendmail, bind oder PERL.

Wie man in den letzten Jahren beobachten konnte, entstammen die industriellen Linux-Anwender ganz unterschiedlichen Bereichen der Wirtschaft: Die Pixar Animation Studios, eine Hollywoodgröße im Bereich der Computeranimation, schafft Filmwelten auf Linux-Rechnern. VeriSign, Anbieter von sicheren Kommunikationslösungen, migriert Verschlüsselungsdienstleistungen auf das freie Betriebssystem. Und Verbrauchsgüterriese Unilever will bis zum Jahr 2006 seine IT-Landschaft im Serverbereich auf Linux umstellen – um Kosten zu sparen und die Flexibilität seiner IT-Systeme zu steigern. In den vergangenen zwei Jahren hat sich Linux auch verstärkt bei geschäftskritischen Systemen in der Finanzdienstleistungsindustrie etabliert, insbesondere im traditionell Unix-lastigen Investmentbankinggeschäft. Der Treiber für diese Entwicklung sind die geringeren Gesamtkosten, die sich (abhängig vom Einsatzgebiet) mit Linux erzielen lassen.

Diese Entwicklung spiegelt sich nicht zuletzt in den Umsätzen der IT-Branche im Serverbereich wider. Nach Zahlen von Gartner Dataquest wurden beispielsweise im vierten Quartal 2002 in den USA mit Linux-Serversystemen knapp 385 Millionen US-Dollar umgesetzt – gut 90 % mehr als im Vorjahresquartal. Der Gesamtumsatz mit Servern stieg in den USA im gleichen Zeitraum um lediglich 5 %. Allein IBM und Hewlett-Packard setzten 2002 mit Software, Hardware und Dienstleistungen für Linux weltweit 1,5 bzw. 2 Milliarden US-Dollar um. Insgesamt ist der Marktanteil installierter Linux-Systeme gegenüber Microsoft-Servern jedoch nach wie vor klein. Gleichwohl geht die Butler Group davon aus, dass sich bis 2009 ein zwischenzeitlich standardisiertes Linux als dominantes Serverbetriebssystem etabliert haben wird [BlDa02].

Im Lichte dieser Entwicklung betrachtet, ist Linux wohl weniger nur eine Alternative zu kommerziellen Betriebssystemen, als vielmehr eine klassische disruptive Technologie im Sinne von Clayton Christensen, Professor an der Harvard Business School [Clay97]. Christensen argumentiert, dass sukzessive Wellen dominanter Technologien unsere Gesellschaft durchdringen: Die jeweils nächste Welle baut auf die vorherigen auf, und scheinbar langlebige Technologien werden plötzlich durch einen neuen Herausforderer entthront. Diese disruptiven Technologien zeichnen sich zwar durch eine echte Neuerung aus, sind dabei aber anfangs gerade gut genug, um sich in einer Nische zu etablieren. Im Falle von Linux lag diese im akademischen Umfeld. In der anfänglichen Randexistenz werden disruptive Technologien jedoch kontinuierlich weiterentwickelt, wodurch sie sich dann auf breiter Ebene am Markt durchsetzen können. Dabei verdrängen sie schließlich nicht nur angestammte Konkurrenten, sie gestalten vielmehr ihren gesamten Markt in einer schöpferischen Zerstörung um. Dies betrifft insbesondere die Preismodelle und Profitmargen, die auf den betroffenen Märkten realisiert werden können. Die echte Neuerung, die mit Linux am Betriebssystemmarkt eingeführt wurde, ist die Quelloffenheit der Software. Damit wurde ein Unix-artiges Betriebssystem für verschiedenste Hardwareplattformen verfügbar. Dies gilt insbesondere für die massenhaft verbreiteten x86-Systeme, mit ihren bekannten Kostenvorteilen. Für den disruptiven Charakter von Linux spricht zudem die Tatsache, dass Industriegrößen wie IBM oder HP schon vor einigen Jahren ihr Geschäftsmodell angepasst und eine Linux-Strategie implementiert haben.

Für den Einsatz quelloffener Software hat die Deutsche Bank eine pragmatische Strategie verfolgt. Dabei kann man speziell im Falle von Linux grob in drei Phasen unterscheiden. Bis Ende 1998 wurde dieses Betriebssystem in unserer Bank hauptsächlich in wenig kritischen Bereichen eingesetzt. Dies waren beispielsweise Webserver im In-

tranet, Terminalserver in der Systemadministration oder die Datei-, Druck- oder Faxserver einzelner Abteilungen. In dieser ersten Diffusionsphase wurden wertvolle Erfahrungen im Umgang mit Linux gewonnen. Diese Erfahrungen – insbesondere im Hinblick auf Sicherheit – ermöglichten es in der zweiten Phase zwischen 1999 und 2001, Linux auch in kritischeren Bereichen einzusetzen. Hierzu zählen insbesondere Systemmanagementwerkzeuge im Rechenzentrumsumfeld, die teils eingekauft, teils selbst entwickelt wurden. Zudem wurde in dieser zweiten Phase Linux verstärkt als Plattform für den Aufbau von Firewalls zur Absicherung der Unternehmensnetze benutzt. In der zweiten Phase wurde Linux also auch für sicherheitskritische Anwendungen eingesetzt, die allerdings einen klaren Bezug zur IT-Infrastruktur haben. Man könnte daher von einer Infrastrukturphase sprechen. Dies änderte sich in der dritten Phase, die seit Ende 2001 bis heute anhält und die Diffusion von Linux in Systeme des Kerngeschäfts der Bank markiert. Seither wird Linux im Grunde für beliebige Einsatzgebiete verwendet, insbesondere auch als Plattform für geschäftskritische Anwendungen. Hierzu zählen beispielsweise die globale Risiko- und P&L Analyse für Kreditderivate [Golt03].

Der Verbreitungsprozess von Linux in der Deutschen Bank spiegelt also den jeweiligen Reifegrad und damit die erzielbare Sicherheit dieser Technologie wider. Er ist das Ergebnis eines sorgfältig geplanten Vorgehens, in dem sehr früh die besten internen und externen Linux-Projekte identifiziert und analysiert wurden. Aus dieser Analyse ergaben sich dann die strategischen Vorgaben für den weiteren Einsatz dieses Betriebssystems. Heute ist Linux eine strategische Plattform für Unixfunktionalitäten, die bislang teuren proprietären Systemen vorbehalten war. Dies umfasst einen breiten Anwendungsbereich, wie Systeme im Hochleistungsrechenbereich, die bekannten N-Schicht-Konfigurationen typischer Webapplikationen oder Anwendungen im Sicherheitsbereich.

Tatsächlich ist nach unseren Erfahrungen Linux gerade aus der Sicherheitsperspektive interessant. In Bezug auf die zugrundeliegende Sicherheitsphilosophie unterscheidet sich Linux nicht wesentlich von kommerziellen Unixderivaten. Zudem etabliert sich dieses Betriebssystem zunehmend als eine Standardplattform im Sicherheitsumfeld, da es aufgrund seiner Modularität leicht an spezielle Anforderungen angepasst werden kann. Dies gilt beispielsweise im Firewall- und VPN-Bereich. Bei der internen Sicherheitsabnahme in der Deutschen Bank wurde Linux nicht anders behandelt als kommerzielle Unixderivate. Dies betrifft insbesondere die zugrundegelegten Sicherheits- und Betriebsstandards. Im Gegensatz zu einigen anderen Betriebssystemen erfüllte Linux unsere anspruchsvollen Standards größtenteils schon ohne Modifikation der ursprünglichen Konfiguration.

Allerdings ist Sicherheit ein Prozess und kein Zustand. Deshalb ist auch die Sicherheit von Banksystemen viel weniger von den eingesetzten Plattformen wie Linux abhängig als von der Sicherheitsorganisation, die deren Einsatz steuert. Gerade im Hinblick auf die oben skizzierte zunehmende wirtschaftliche Bedeutung des Open-Source-Betriebssystems auch für unsere Bank waren die frühe Adaption von Linux und der Lernprozess, der für unsere IT-Organisation daraus resultierte, notwendige Schritte.

Neben den klassischen Sicherheitszielen wie Verfügbarkeit, Integrität und Authentizität von IT-Systemen ist im betrieblichen Umfeld zudem noch die Investitionssicherheit von besonderer Bedeutung. Diese bezieht sich sowohl auf den Schutz bereits getätigter Investitionen als auch auf die Zukunftssicherheit aktueller Investitionsentscheidungen. Im Hinblick auf die getätigten Investitionen sollte man daher nicht erwarten, dass Linux beispielsweise die mainframebasierten Systeme im Finanzsektor in naher Zukunft ersetzt. Ebenso wenig sollte man erwarten, dass sich Linux in naher Zukunft zum dominanten Desktopbetriebssystem im Finanzsektor entwickelt. Zu groß wäre der Aufwand bei der Mitarbeiterschulung für das neue Betriebssystem. In Bezug auf künftige Investitionen in Linux-basierte Systeme kann man inzwischen darauf bauen, dass dieses Betriebssystem bei einer Vielzahl von namhaften Herstellern dauerhaft verankert ist. Im Hinblick auf die bestehende (und zunehmende) Verfügbarkeit von Software, Dienstleistungen und Komplettlösungen im Linux-Umfeld erscheint es in der Tat nicht unwahrscheinlich, dass sich Linux zu einem dominanten Serverbetriebssystem entwickelt, wie die Butler Group dies prognostiziert. Aus dem gleichen Grund erscheint die Prognose von db-research nicht übertrieben, dass sich Open-Source-Kompetenz zu einem wichtigen Kriterium in Unternehmensbewertungen und Investitionsentscheidungen entwickeln könnte [Hofm02].

Ob sich diese Prognosen im vollen Umfang erfüllen, ist dabei für die Bedeutung von Linux für die Industrie von erheblicher Wichtigkeit. Allerdings hat schon die alleinige Existenz einer leistungsfähigen Alternative zu proprietären Systemen einen mäßigenden Einfluss auf die Lizenzkosten. Darüber hinaus erlaubt der Einsatz kostengünstiger Hardware unter Linux einen wichtigen Beitrag zur Kostensenkung in Unternehmen.

Literatur (Auswahl)

[BlDa02] *Blowers, Mark; Davis, Mike; Holt, Maxine:* Server Operating Systems Report – Winners & Losers in the Open/Proprietary OS Market. Butler Direct Limited, Hull, 2002.

[Clay97] *Clayton Christensen:* The Innovator's Dilemma. Harvard Business School Press, Harvard 1997.

[Golt03] *Goltzsch, Patrick:* Schneller Rechnen. In: CIO Magazin (2003) 4.

[Hofm02] *Hofmann, Jan:* Free software, big business? In: Deutsche Bank Research, E-conomics (2002) 32.

Hermann-Josef Lamberti,
Chief Operating Officer,
Deutsche Bank AG

Freie Software in der Sicherheitsstrategie der Behörden von Udo Helmbrecht

Ereignisse wie der Terroranschlag am 11. September 2001, Katastrophen wie das Elbehochwasser 2002 und erfolgreiche Angriffe wie *Loveletter* zeigen immer wieder, wie abhängig die Gesellschaft von einer reibungslos funktionierenden Informationstechnik (IT) ist. Um die Verletzlichkeit der Informationsgesellschaft zu reduzieren, gilt es, die zum Einsatz kommende IT zu diversifizieren. Es darf in Zukunft nicht möglich sein, dass durch ungewollte oder vorsätzliche Angriffe auf die IT aufgrund der globalen Vernetzung flächendeckende Ausfälle und Schäden verursacht werden können.

Dem der Biologie entlehnten Prinzip der Vermeidung von Monokulturen folgend gilt es, im Sinne der IT-Sicherheit auch die Vielfalt der Informationstechnik zu fördern. Dazu unterstützt das Bundesamt für Sicherheit in der Informationstechnik (BSI) das Bundesministerium des Innern (BMI) in dem Bemühen, den Einsatz Freier Software insbesondere im Behördenbereich zu fördern. Ziel ist es, die Nutzung der am Markt verfügbaren proprietären und Freien Software in einem integrierten, heterogenen Umfeld zu ermöglichen.

Bei der Frage, wie die IT-Sicherheit im Open-Source-Umfeld derzeit einzuschätzen ist, müssen sowohl technische als auch strategische Aspekte berücksichtigt werden.

Technische Aspekte

Technisch gesehen ist die Aufgabe jeder IT-Sicherheitsmaßnahme die Gewährleistung der Verfügbarkeit, Vertraulichkeit und Integrität der IT-Systeme und Daten. Welche Möglichkeiten bietet hier die Verwendung von Freier Software und welche nicht? Dies wird im Folgenden am Beispiel einiger zentraler Aspekte beleuchtet.

Sicherheitsprüfung

Der Zugang zum Quellcode ist eine notwendige, wenn auch keine hinreichende Voraussetzung für die Sicherheitsüberprüfung von Software. Der dafür vorgesehene internationale Standard (*Common Criteria*) zur Prüfung (Zertifizierung) von Softwareprodukten fordert z. B. für eine Reihe der vorgesehenen Prüfschritte die Offenlegung der Quellen.

Viele Softwareprodukte und einige Betriebssysteme sind bereits oder werden demnächst nach *Common Criteria* zertifiziert. Zur Zeit findet auch für das Betriebssystem Linux eine solche Prüfung statt.

Auch bei nicht zertifizierter Software können Aussagen über die Codequalität und den Sicherheitswert des Softwareprodukts getroffen werden: Die Offenlegung des Quellcodes bei Freier Software ermöglicht es einer breiten Basis von Testern, einen Blick ins „Software-Innere“ zu werfen. Darüber hinaus besteht auch die Möglichkeit, Sicherheitsanalysen auf Source-Code-Basis unabhängig vom Entwickler in Auftrag zu geben.

Sicherheitslücken

Die vielfach geäußerte Vermutung, durch die Geheimhaltung des Quellcodes von Software (Security by Obscurity) Sicherheitslücken schwerer auffindbar zu machen, hat sich leider in der Realität nicht bewahrheitet, wie die täglich in der Fachpresse veröffentlichten Beispiele zeigen. Ein sicherheitstechnisch bedenklicher Code wird nicht allein dadurch besser, dass der Quellcode nicht direkt zugänglich ist. Denn auch übersetzter Quellcode kann insbesondere durch Profis in vielen Fällen mit genügend hohem Aufwand analysiert werden.

Das Vorhandensein von Sicherheitslücken und Softwarefehlern ist ein Sicherheitsrisiko, das durch den Einsatz von sogenannten Sicherheitsanalysen (Audits) im Quelltext am einfachsten minimiert werden kann.

Ein weiterer wichtiger Aspekt nach dem Auffinden von Sicherheitslücken ist eine kurze Reaktionszeit bis zur Bereitstellung einer Fehlerbehebung. Hersteller proprietärer Software sowie Entwickler Freier Software bemühen sich, Sicherheitsupdates zeitnah zur Verfügung zu stellen.

Mit Freier Software ist die Anwendergemeinde zudem in der Lage, Fehler selbst zu beheben – ein Vorgehen, dass bei proprietärer Software aus lizenzrechtlichen Gründen meist untersagt ist.

Selbst wenn der öffentlichen Verwaltung der Einblick in proprietäre Software von den Herstellern relativ leicht gewährt wird, so ist dies ein Privileg, das der Wirtschaft, vor allem dem Mittelstand, oft nicht zuteil wird.

Härtung des Systems

Je „minimaler“ und übersichtlicher ein System aufgebaut ist, desto sicherer kann es gemacht und umso besser kontrolliert werden. Dies gilt auch für Software. Der zentrale Aspekt ist hier die Modularität: Sie ermöglicht eine Minimierung der Funktionen und reduziert so die Fläche für potentielle Angriffe. Gleichzeitig wird die Ausfallsicherheit erhöht.

Freie Software wird durch den Entwicklungsprozess sehr modular entwickelt und kann daher meist leicht minimiert werden. Das Gleiche gilt für einen Teil der proprietären Software.

Ein Beispiel für die Modularisierung und die damit mögliche Minimierung bietet der Linux-Kernel. Das BSI hat diese Möglichkeiten genutzt, um eine Sichere-Inter-Netzwerk-Architektur (SINA) zu schaffen. SINA ermöglicht PC-Anwendern die verschlüsselte Übertragung von Informationen mit sehr hohem Schutzbedarf über öffentliche, ungeschützte Netze und so die exklusive private Kommunikation zwischen verschiedenen Zugangspunkten. Zum Einsatz im SINA Projekt wurde der Linux-Kern auf die wesentlichen, absolut notwendigen Teile reduziert, überprüft, verbessert und gesichert. Das SINA-System dient unter anderem zur Vernetzung der deutschen Botschaften und Auslandsvertretungen.

Softwarevielfalt

Ein wichtiger Aspekt in einer großen IT-Umgebung ist der Aspekt der Softwarevielfalt. Ein homogenes IT-System ist genauso angreifbar durch Viren wie eine Monokultur durch Schädlinge. Nur so konnte es passieren, dass durch den *Loveletter*-Virus ein derart großer Schaden angerichtet wurde.

Freie Software schützt zwar nicht automatisch vor Viren, zur Zeit sind jedoch meist proprietäre Systeme Ziel der Virenattacken. Dies könnte sich in Zukunft mit der Verbreitung Freier Software auf dem Desktop relativieren.

Insgesamt wird durch Softwarevielfalt die Sicherheit in der Informationstechnik erhöht, gleichzeitig aber auch die Komplexität des Systems. Zudem muss die Kommunikation der verschiedenen Systeme sichergestellt werden.

Strategische Aspekte

Interoperabilität

Wichtig bei der Auswahl der IT-Systeme ist die Interoperabilität der Systeme. Um die Kommunikation der Softwarekomponenten untereinander und mit anderen Systemen zu gewährleisten, ist die Verwendung offener Standards und Schnittstellen unabdingbar. Um vielfältige Programme einsetzen zu können und die Kompatibilität sicherzustellen, sollten Standards frei zugänglich dokumentiert und einsetzbar sein – wobei nur von offenen Standards gesprochen werden kann, wenn unabhängige Implementierungen realisiert werden können.

Erst offene Standards und Schnittstellen ermöglichen, dass eine Produktpalette im Sinne der Diversifizierung zur Verfügung steht – sowohl als Freie Software als auch als kompatible proprietäre Software.

Die erreichbare Marktbreite von Entwicklungsfirmen, die die offenen Schnittstellen nutzen können, erhöht darüber hinaus die Verfügbarkeit der notwendigen Produkte. Letztlich fördert die Stabilität eines offenen Standards auch die Qualität der Software.

Support

Support für Freie Software kann genau wie für proprietäre Software bei qualifizierten Unternehmen eingekauft werden. Ein Unterschied ist der Umfang des Supports für Freie Software.

Auf der einen Seite ist der Support für Installation und Wartung der Software bei Freier und proprietärer Software gleich. Man kann beliebige Unternehmen damit beauftragen oder den Support selbst übernehmen.

Anders sieht es bei der Behebung von Sicherheitslücken aus. Hier ist man bei proprietärer Software auf den Hersteller angewiesen und darauf, dass dieser die Lücken schnell behebt. Bei Freier Software kann die Behebung der Sicherheitslücken direkt durch die Softwareentwickler geschehen. Der Anwender hat aber auch die Möglichkeit, diese bei einem beliebigen qualifizierten Softwareunternehmen zu beauftragen. Beispiele haben gezeigt, dass beide Verfahren gut in der Praxis funktionieren können, wobei hervorzuheben ist, dass auch bei proprietärer Software in der Regel kein An-

spruch auf die Behebung von Sicherheitslücken besteht.

Das Gleiche gilt für die Anpassung und Weiterentwicklung von Software: Bei proprietären Produkten übernimmt dies der Hersteller. Dies sichert meistens regelmäßige Releasezyklen mit neuen Funktionalitäten. Der Anwender hat begrenzt Einfluss darauf. Auch Freie Software hat Releasezyklen, in denen neue Funktionalitäten in die Software einfließen. Weiterentwicklungen, die nicht durch die Entwickler aufgegriffen werden, aber den Einsatz im eigenen Unternehmen verbessern, können jedoch bei unabhängigen Softwareunternehmen in Auftrag gegeben werden.

Investitionssicherheit
Ein wichtiges Kriterium bei der Auswahl von Software ist die Herstellerunabhängigkeit. Die Insolvenz eines Softwareherstellers bedeutet für den Nutzer von dessen Software in der Regel den Kauf und die Einführung eines neuen Softwareproduktes. Wie die Praxis zeigt, müssen heutzutage nicht nur kleine und mittelständische Unternehmen Insolvenz anmelden, sodass Investitionen in deren Software verloren sind.

Um herstellerunabhängig zu bleiben, gibt es mehrere Möglichkeiten, so z. B. den Einsatz Freier Software oder die Sicherstellung des Zugriffes auf den Quellcode proprietärer Software im Insolvenzfall.

Eine Möglichkeit, die immer häufiger eingesetzt wird, ist die Hinterlegung des Quellcodes proprietärer Software bei einem Notar. Bei Insolvenz wird dem Lizenznehmer voller Zugriff auf die Quellen gewährt. Dies erfordert allerdings die Einwilligung des Herstellers. Die Lizenzen Freier Software sichern den Zugang zum Quellcode sowie das Recht, diesen zu verändern und einzusetzen. Mit dem Besitz des Quellcodes und dem Recht, diesen zu ändern und in der veränderten Form auch einzusetzen, ist ein Investitionsschutz erreicht.

Planungssicherheit
Über das hinaus, was bereits im Abschnitt Investitionssicherheit gesagt wurde, ist ein zusätzlicher Aspekt der Planungssicherheit die Verfügbarkeit abwärtskompatibler Versionen von Software. Die meisten Hersteller achten darauf, dass ihre Software abwärtskompatibel ist. Dies gelingt meistens solange, bis ein Redesign stattfindet oder wesentliche Funktionen geändert werden. Das ist auch bei Freier Software der Fall.

Die Wartung von älteren Versionen entfällt meistens nach einiger Zeit. Bei Freier Software lässt sie sich im Bedarfsfall noch über Supportverträge einkaufen. Das Gleiche gilt für die Weiterentwicklung von älteren Versionen.

Fazit: Sicherheit ist ein Prozess.
Der erfolgreiche Erhalt von IT-Sicherheit bedingt die genaue Kenntnis des IT-Systems sowie eine regelmäßige Wartung und eine schnelle Behebung von Sicherheitslücken. Der Einsatz Freier Software hat in diesem Prozess einige strategische Vorteile, bietet jedoch keine Gewähr für ein sicheres System. Das notwendige IT-Sicherheits-Know-how sowie gute, an die jeweiligen Anforderungen angepasste Lösungen sind unumgänglich.

Unabhängigkeit und Softwarevielfalt sowie die Verwendung offener Standards sind eine Basis für IT-Sicherheit. Diese sollten als zentrale Aspekte in den Auswahlprozess für Softwareprogramme eingehen.

Dr. Udo Helmbrecht,
Präsident des Bundesamtes für Sicherheit
in der Informationstechnik

Die Sicherheit von Software wird nicht allein durch deren Quellcode bestimmt
von Harald Lieder

Vielfach wird behauptet, Open-Source-Software (OSS) sei grundsätzlich sicherer als proprietäre Software. Als Beleg hierfür werden Statistiken angeführt, die zeigen, dass z. B. Microsofts Webserver IIS im Verhältnis öfter erfolgreich attackiert wurde als das Open-Source-Gegenstück Apache. Aus meiner Sicht ist grundsätzlich fraglich, ob solche Statistiken tatsächlich den direkten Rückschluss auf die Sicherheit des Produkts erlauben. Zum einen wird mit wachsender Verbreitung von OSS im kommerziellen Bereich diese auch für Hacker immer interessanter, zum anderen können die Ursachen erfolgreicher Angriffe nicht ausschließlich auf das Entwicklungsmodell (Open/Closed Source) der Software reduziert werden. Zudem sollte das Thema „Sicherheit“ bei Software nicht nur unter dem Gesichtspunkt einer Offenlegung des Quellcodes diskutiert werden.

Weder das Offenlegen des Quellcodes alleine noch dessen Geheimhaltung schafft mehr Sicherheit
In den Diskussionen über den Zusammenhang von frei zugänglichem Quellcode und der Sicherheit von Software tauchen immer wieder die beiden folgenden Argumente für und wider Open Source auf: Auf der einen Seite ist zu hören, dass OSS sicherer sei, da eine Vielzahl von Entwicklern den Quellcode auf Sicherheitslücken hin untersuchen. Auf der anderen Seite wird genau diese Offenheit als Argument gegen OSS verwendet: Es wird argumentiert, dass Hacker im Quellcode nach sicherheitsrelevanten Schwachstellen suchen und diese gezielt ausnutzen könnten. Aus meiner Sicht müssen beide Argumente relativiert werden:

Die Tatsache, dass der Quellcode eines Programms einseh- und veränderbar ist und von einer Vielzahl von Entwicklern geprüft werden kann, macht ein Programm noch lange nicht sicher. Es kann sogar dazu führen, dass sich ein falsches Sicherheitsgefühl einstellt. Das „Viele-Augen-Prinzip“ hat nur Erfolg, wenn tatsächlich Security-Reviews durchgeführt werden und die Entwickler, die sich den Code ansehen, entsprechend fundierte technische Kenntnisse im Bereich Softwaresicherheit besitzen. Dass dies nicht selbstverständlich ist, zeigen Open-Source-Programme wie GNU Mailman und wu-ftp, bei denen gravierende Sicherheitslücken auch über eine lange Zeit hinweg trotz diverser Reviews Bestand hatten. Mitwirkende in Open-Source-Projekten untersuchen zudem einen fremden Code meist aus eigenem Interesse und weniger aus altruistischen Beweggründen. Deshalb haben Programme im Open-Source-Bereich, deren Quellcode nur schwer entwirrbar ist oder in einer unpopulären Programmiersprache geschrieben wurde, oft nur eine geringe Anzahl von Reviewern. Dies sollten Anwender von OSS bei der Softwareauswahl beachten, da bei zu wenigen Reviewern ein sicheres Kontrollverfahren nicht garantiert ist.

Auf der anderen Seite wird Software jedoch nicht automatisch dadurch sicherer, dass der Quellcode unveröffentlicht bleibt. Durch Reverse Engineering ist es heute jedem Hacker möglich, die Binärdateien in der einen oder anderen Form lesbar zu machen und nach Mustern für Sicherheitslücken zu durchsuchen.

Ein Problem von Closed-Source-Software ist sicherlich die potenzielle Möglichkeit der Programmierer, sich Hintertürchen für administrative oder andere Zwecke offen zu lassen und damit in Bezug auf die Sicherheit der Software Lücken zu schaffen. Bei in einer Versionskontrolle befindlichen Open-Source-Projekten ist dies eher unwahrscheinlich. Die Gefahr, dass andere Entwickler solche „Kuckuckseier“ entdecken und dass der entsprechende Programmierer seinen in der Open-Source-Gemeinschaft so wichtigen eigenen guten Ruf verliert, ist sehr hoch. Deshalb tauchen in der Praxis auch fast ausschließlich in proprietärer Software solche für den Anwender sicherheitsrelevanten Disfunktionalitäten auf. Aus eben diesem

Grund versuchen in letzter Zeit die Hersteller proprietärer Software das Misstrauen, das vor allem Regierungsinstitutionen dem Closed-Source-Modell entgegenbringen, durch teilweise oder komplette Offenlegung des Quellcodes abzubauen. Ein Beispiel ist die Shared-Source-Initiative von Microsoft. Nachdem Microsoft bereits seit zehn Jahren akademischen Institutionen den Windows-Quellcode für Forschungszwecke zur Verfügung gestellt hat, richtet sich diese neue Initiative nun hauptsächlich an Großkunden, Regierungsinstitutionen und Systemintegratoren mit dem Ziel, eine größere Transparenz bezüglich des Quellcodes zu erreichen.

Eine Ungewissheit über mögliche Sicherheitslücken im Quellcode besteht damit sowohl bei Open als auch bei Closed Source. Werden an eine Software durch den Nutzer extreme Sicherheitsanforderungen gestellt und traut der Nutzer weder den Reviewprozessen der Open-Source-Entwickler noch den Standards kommerzieller Softwareanbieter, so bleibt als Ausweg nur das selbstständige Durchsuchen des Codes auf mögliche Schwachstellen.

Kleinere Unternehmen oder Konzerne, die die Softwareauswahl den einzelnen Abteilungen überlassen, können sich aber den Aufwand, Tausende von Zeilen Quellcode auf Herz und Nieren zu prüfen, aus Mangel an Zeit, Geld und Ressourcen oft nicht leisten. In diesem Fall ist es letzten Endes eine Frage des Vertrauens, ob man die Überprüfung des Codes vielen, unabhängigen Entwicklern der Open-Source-Gemeinde oder aber wenigen, dafür auf Sicherheitsfragen spezialisierten Mitarbeitern eines Softwareherstellers überlässt.

Viele Angriffe auf IT-Systeme waren letztendlich nur deshalb erfolgreich, weil versäumt wurde, rechtzeitig die aktuellsten Servicepacks einzuspielen

Beurteilt man Software unter Sicherheitsgesichtspunkten, so sollte man sich aber grundsätzlich nicht nur mit der Frage der Öffentlichkeit des Programmcodes beschäftigen. Genauso wichtig ist es zu überprüfen, wie viele entdeckte Sicherheitslücken veröffentlicht wurden, wie lange es dann gedauert hat, bis die entsprechenden Korrekturen zur Verfügung standen, und wann diese schließlich von Systemadministratoren eingespielt wurden.

Open-Source-Projekte reagieren auf öffentlich bekannt gewordene Sicherheitslücken im Allgemeinen schneller als Hersteller proprietärer Software; die anschließende Verteilung der Software in den Unternehmen läuft dann aber meist nicht mehr so problemlos. Grund hierfür ist, dass die Existenz neuer Patches meist über Mailinglisten oder Userforen bekannt gegeben wird. Dies erfordert ein ständiges, aktives Überwachen dieser Kommunikationskanäle.

Dagegen brauchen Hersteller kommerzieller Software mit ihrer Reaktion meist etwas länger, allerdings haben einige von ihnen schon Verfahren, bei denen die aktuellsten Servicepacks und Korrekturen sofort nach deren Freigabe automatisch heruntergeladen werden können.

Schwer zu konfigurierende Software, unzureichende Dokumentation und fehlende Schulung bergen weitere Sicherheitsrisiken

Nicht selten ist aber auch eine fehlerhafte Konfiguration eines Programms für Sicherheitslücken in der Software eines Unternehmens verantwortlich. Der Grund hierfür ist die oft unzureichende Schulung der Administratoren und eine mangelhafte Dokumentation. Hier hat OSS eindeutig Nachteile, denn oft ist die Dokumentation von OSS nur für Experten verständlich. Gerade für eher unerfahrene Entwickler und Administratoren ist das Verständnis über Funktionsweise und Konfiguration sicherheitsrelevanter Systeme aber von grundlegender Bedeutung. Auch Schulung bzw. Training wird oft nur für populäre OSS angeboten und dann meist von lokal tätigen Anbietern. Für globale Unternehmen, die eine einheitliche Ausbildung ihrer Administratoren anstreben sollten, kann dies keine befriedigende Lösung sein.

Investitionssicherheit ist weitgehend unabhängig von der Frage, ob es sich um Open-Source- oder proprietäre Software handelt

Neben den sicherheitsrelevanten Punkten technischer Natur muss die Frage nach der Investitionssicherheit des Produkts geklärt werden. Aufgrund des Marktdrucks sind die Hersteller proprietärer Software gezwungen, in immer kürzeren Zeitzyklen neue Versionen ihrer Programme auf den Markt zu bringen. Sind für ein Unternehmen die Kosten der Umstellung auf eine neuere Version zu hoch, leidet es dann in der Folge unter auslaufenden Supportverträgen oder höheren Wartungskosten für ältere Versionen. Das gleiche Problem stellt sich auch für Nutzer von Open-Source-Produkten, sogar noch in einer verschärften Form. Bei großen Open-Source-Projekten werden vor allem in der Anfangsphase neue Versionen in sehr kurzen Zeitabständen zur Verfügung gestellt. Entscheidet man sich nun aus Kosten-, Stabilitäts- oder sonstigen Gründen dafür, ein Release länger beizubehalten, sieht man sich mit einem ständig sinkenden Interesse der Entwicklergemeinde konfrontiert, sich Problemen zurückliegender Releases anzunehmen und dafür passende Lösungen anzubieten.

Die Angst, die Weiterentwicklung von Open-Source-Produkten sei nicht gesichert, ist allerdings für populäre OSS mit großer, aktiver Entwicklergemeinde unbegründet. Zusätzlich versuchen Konzerne wie IBM, HP oder Sun durch eigene oder gesponserte Open-Source-Projekte Investitionssicherheit zu gewährleisten.

Fazit

Die Frage nach der Sicherheit von Software kann nicht pauschal durch die Empfehlung des Open- oder Closed-Source-Modells beantwortet werden. Ob der Anwender von offenem und veränderbarem Quellcode überhaupt profitieren kann, richtet sich nach seinem Sicherheitsbedürfnis und nach der Möglichkeit, den Code in einem ökonomisch vertretbaren Rahmen selbst zu überprüfen. Daneben sind noch andere Punkte bei der Entscheidungsfindung für Software unter Sicherheitsaspekten wichtig, die zum Teil unabhängig vom Entwicklungsmodell sind. Hierzu gehören vor allem folgende Punkte:

- Welche formalen Securityreviews für die Software wurden durchgeführt?
- Wer hat diese Reviews durchgeführt?
- Auf welchem Weg soll die Versorgung mit aktuellen Patches bzw. Servicepacks erfolgen?
- Mit welchem Aufwand und mit welchen Kenntnissen ist eine sichere Konfiguration und Handhabung der Software möglich?
- Ob und in welchem Umfang sind Dokumentation und Schulungen für das entsprechende Produkt verfügbar?
- Ist der Support auch älterer Releases der eingesetzten Software in Zukunft gesichert?

Jeder dieser Punkte sollte sowohl nach Relevanz für den Anwender als auch nach dessen zur Verfügung stehenden Möglichkeiten, wie z. B. Zeit, Budget und Expertise der eigenen Mitarbeiter bewertet werden. Erst dann kann die Auswahl für ein passendes Produkt getroffen werden – ohne vorher grundsätzlich pro oder contra Open Source entschieden zu haben.

Harald Lieder,
Leiter des Bereichs Global
Architecture & Core Technologies ASG,
Accenture GmbH

Offen, transparent . . . sicher?
von Thomas Mangel

Die aktuellen Diskussionen um Open-Source- versus Closed-Source-Systemen scheinen stark von subjektiven Argumenten geleitet und führen in einigen Bereichen zu einer gewissen Zurückhaltung gegenüber der euphorischen Begrüßung von Open-Source-Systemen in einer breiten Anwenderschicht. Dennoch stützen Veröffentlichungen wie [BaRe00] die Erfahrung, dass Open-Source-Systeme in Bezug auf geschäftsorientierte Werte eines Unternehmens, wie z. B. Wirtschaftlichkeit, Verfügbarkeit und nicht zuletzt Sicherheit mehr zu bieten haben, als nur ein politisches Statement.

Als Bank sind wir dem Vertrauen unserer Kunden besonders verpflichtet, als Multikanalbank hat die Verfügbarkeit unserer Leistungen eine immense Bedeutung, als Internetbank müssen unsere elektronischen Transaktionen mit entfernten Geschäftspartnern absolut integer sein, und diese Partner müssen wir als solche auch sicher identifizieren können. Somit lassen sich die Grundwerte der IT-Sicherheit für die Postbank festhalten als Vertraulichkeit, Integrität, Authentizität, Verbindlichkeit und Verfügbarkeit.

Bezieht man diese Grundwerte in die Diskussion um die Sicherheit in Open-Source-Systemen ein, ergibt sich ein Bild, aus dem man die objektiven Werte für die Bank sehr wohl herauslesen kann. Und so sind Open-Source-Produkte bereits ein unverzichtbarer Bestandteil der IT-Infrastruktur der Postbank. Auch in sicherheitssensiblen Bereichen werden Open-Source-Produkte gezielt eingesetzt.

Institutionen mit besonders hohen Sicherheitsanforderungen haben seit längerem erkannt, dass die Möglichkeit, die Funktions- und Verarbeitungsabläufe in einem System einsehen zu können, Grundlage zur Beurteilung der Sicherheit von Software ist [BaRe00; BSIT03; BMWT01]. Als Kunde eines Softwareprodukts ist dies naturgemäß nur bei Open Source der Fall. Die gleichzeitig geführte Argumentation, dass dann natürlich auch Hacker den Code verändern können und nicht jeder den Code versteht, um die Sicherheit beurteilen zu können, wurde durch die faktische Erfahrung der letzten Jahre im wesentlichen entschärft.

Eine Offenlegung des Quellcodes einer Software ist aber kein Garant für die Sicherheit und Qualität dieser Software. Denn die Erkenntnisse über Fehler und Verbesserungen der Software müssen wieder kontrolliert in das System einfließen. Auch wenn „tausende Augen“ jederzeit die Software bis auf die Fundamente einsehen und daran arbeiten, ist im Gegensatz zu oft zitierten Aussagen keine Sicherheit für das Vertrauen in diese Software ableitbar [Raym98].

Daher schützen viele Open-Source-Projekte ihre Programmentwicklung inzwischen durch die Nutzung zentraler Entwicklungsplattformen. Dort können viele Entwickler, die meist namentlich bekannt und persönlich erreichbar sind, gemeinsam arbeiten, die Veränderung werden immer automatisch sichtbar und Kollisionen in der Entwicklung können vermieden werden. Bei diesen Systemen führt der Synergieeffekt hunderter bis tausender Expertisen zu ungeheueren Innovationsschüben auch in Punkten der Sicherheit.

Im Gegensatz dazu lässt sich diese Transparenz bei der Entwicklung und Einführung von Closed-Source-Systemen nicht aufweisen. Hier führen Abhängigkeiten von Wiederverwertbarkeit, Marktanteilen, Imagewerten und Zeithorizonten der Partnerschaften zu komplexeren Vertrauensstrukturen die meist langjährig erarbeitet werden. Qualität und Sicherheit wird den Erfordernissen entsprechend aus beiderseitigen Kosten- und Nutzenrechnungen eingestellt. Steuerungsmittel sind die definierten Vertragsverhältnisse.

Da es diese Vertragsverhältnisse bei Open-Source-Systemen nicht gibt, scheint ein nicht berechenbarer Zustand vorzuliegen. Hierzu existieren jedoch interessante Erhebungen, in denen z. B. aufgezeigt wird, wie die Anzahl der Fehler in einer Software während der Programmentwicklung durch Alpha- und Betatests reduziert wird und vergleicht diesen Prozess bei offenen und proprietären Systemen. Das Ergebnis ist erstaunlich: Während in der Theorie dieser Prozess für beide Systeme von unterschiedlichen Ausgangspunkten auf das gleiche Niveau läuft, führen langjährige Analysen realer Projekte zu einer Begünstigung von Open-Source-Systemen.

Die Ursachen hierfür sind äußerst vielschichtig und daher in kaum einem Modell wiederzufinden. Stark vereinfacht lässt sich sagen, dass bei Closed-Source-Systemen Kunde und Hersteller aufgrund ihrer Interessenlage oftmals gegensätzliche Standpunkte beziehen müssen und die Entwicklung quasi ein Verhandlungsprozess ist. Im Gegensatz dazu stehen bei Open-Source-Systemen Kunde und Entwickler nicht nur auf der selben Seite, sie sind ebenso oft identisch. Der sich daraus ergebende Aspekt zur Beurteilung der IT-Sicherheitsrisiken muss daher die Interessen der Beteiligten einbeziehen. Grundvoraussetzung und somit Limitierung des Open-Source-Systems ist selbstverständlich das Vorhandensein eines breiten Interesses der gebotenen Funktionalität. „Kundenspezifische“ Anforderungen und Lösungen sind nicht im Modell vorgesehen.

Als Argument gegen den Einsatz von Open-Source-Software wird häufig der Investitionsschutz angeführt. Hierbei wird angenommen, dass ein Open-Source-Produkt nur solange weiterentwickelt wird, solange es „Hype“ ist und dann der Anwender eine evtl. fehlerbehaftete veraltete Anwendung betreiben muss, für die er keinen Support mehr bekommt, wohingegen Closed-Source-Software durch Pflegeverträge, in denen die Hersteller sich zur Weiterentwicklung verpflichten, gegen solche Unwägbarkeiten abgesichert sei. Die Vergangenheit hat aber leider gezeigt, das auch Closed-Source-Produkte dieses bei Open-Source-Systemen befürchtete Schicksal erleiden. Denn ist eine Software – gerade für ein Großunternehmen mit breitem Portfolio – unrentabel, dann wird sie häufig nicht mehr weiterentwickelt. Bei einigen Produkten hat dies zu Firmenausgründungen geführt, die dann die Betreuung der „Altkunden“ übernommen haben.

Im Open-Source-Bereich werden bestimmte Produkte auch „abgekündigt“, dann jedoch häufig durch andere Entwicklergruppen übernommen. Wegen der Open-Source-Lizenzen ist eine Übernahme und Betreuung durch andere Gruppen bzw. eigenes Knowhow auch wesentlich leichter möglich. Ein besonderes Beispiel ist hier die Weiterentwicklung des CERN-HTTP-Servers. Als dieses Projekt stockte, wurde kurzerhand ein neues Projekt geschaffen: Apache. Heute ist der Apache-HTTP-Server der im Internet am weitesten verbreitete und weltweit am häufigsten eingesetzte Webserver.

Will man die IT-Ressourcen des Unternehmens langfristig effektiv nutzen und gleichzeitig die Geschäftsprozesse bezüglich der Grundwerte der IT-Sicherheit zuverlässig schützen, ist man auf eine umfassende Kontrolle über die eingesetzten IT-Werkzeuge angewiesen. Bei Programmen z. B. zur Wahrung der Vertraulichkeit ist es seit einigen Jahrzehnten schon so, dass ausschließlich offene Standards eingesetzt werden; in vielen Fällen, wie z. B. bei Behörden, ist der Einsatz gar zwingend.

Um die Sicherheit der IT eines Unternehmens zu gewährleisten, werden auch in Zukunft immer neue Herausforderungen zu bewältigen sein. Open-Source-Systeme bieten mit hoher Innovationsrate und naturgegebener Flexibilität eine Chance, die zwar ihren Preis hat, aber sicher kein „Irrweg“ im

Streben nach verlässlicheren IT-Systemen ist.

In sicherheitssensiblen Bereichen können Open-Source-Systeme durchaus mit Vorteilen gegenüber Closed-Source-Systemen aufwarten. Sie fördern offene Standards und vermeiden Softwaremonokulturen. Die Entscheidung zwischen funktionsgleichen Open-Source- und Closed-Source-Produkten ist jedoch in jedem Einzelfall auf Basis von Analysen der Wirtschaftlichkeit, der Absicherung der betrieblichen Verfügbarkeit und des angestrebten IT-Sicherheitsniveaus zu treffen.

Open-Source-Systeme werden schon heute vielfach in der Postbank-IT eingesetzt, die Tendenz ist steigend. Der Skill unserer IT-Mitarbeiter im Open-Source-Bereich ist inzwischen beeindruckend. Somit liegen alle Vorraussetzungen vor, sich des Themas Open Source innerhalb der Postbank geschäftsorientiert anzunehmen, eine Integration dieser Themen in unsere Organisationsstrukturen zur Bündelung von Kompetenzen und Verantwortungen findet statt.

Literatur (Auswahl)

[BaRe00] *Bahr, Rudolf E.; Reiländer, Ralf; Troles, Egon:* Open Source Software in der Bundesverwaltung. In: KBSt. Brief Nr. 2/2000.

[BSIT03] Bundesamt für Sicherheit in der Informationstechnik: Open Source Software. http://www.bsi-fuer-buerger.de/11/, Abruf am 2003-05-21.

[BMWT01] Bundesministerium für Wirtschaft und Technologie: Open Source Software – Leitfaden für kleine und mittlere Unternehmen.

[Raym98] *Raymond, Eric S.:* The Cathedral and the Bazaar. http://tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar.html, 1998-08-08, Abruf am 2003-05-14.

[Möll01] *Möller, Erik:* Die Reformation zum Anfassen: GNU/Linux und Open Source. http://www.heise.de/tp/deutsch/inhalt/te/9786/1.html, 2001-10-12, Abruf am 2003-05-21.

[Ande02] *Anderson, Ross:* Security in Open versus Closed Systems The Dance of Boltzmann, Coase and Moore. http://www.ftp.cl.cam.ac.uk/ftp/users/rja14/toulouse.pdf, 2002-06-18, Abruf am 2003-05-21.

[NSA03] *National Security Agency:* Security-Enhanced Linux Frequently Asked Questions. http://www.nsa.gov/selinux/faq.html, Abruf am 2003-05-21.

[Pesc03] *Pescatore, John:* Nimda Worm Shows You Can't Always Patch Fast Enough. http://www3.gartner.com/resources/101000/101034/101034.pdf, 2001-09-19, Abruf am 2003-05-22.

Dr. Thomas Mangel,
Chief Technology Officer,
Postbank Systems AG

Transparenz schafft Sicherheit
von Richard Seibt

Software – gleich, ob lizenziert, gekauft oder selbst entwickelt – verhält sich sicherheitstechnisch ungleich anders als anfassbare Maschinen, Werkzeuge, Büromöbel oder Verbrauchsmaterialien. Wer sich an Werbeaussagen der EDV-Frühzeit erinnert: „Dieser Schnellrechner erledigt in einer Minute die gleiche Arbeit, für die 5000 Mathematiker 13 Jahre bräuchten!", bekommt ein Gefühl für das Problem.

Natürlich erledigt der Schnellrechner überhaupt nichts, die Software tut's. Und wenn wir sicher sein wollen, dass sie genau das tut, was sie soll? Mit 5000 Mathematikern hätten wir es in 13 Jahren nachgerechnet. Oder sollen wir Testsoftware einsetzen, die selbst nur ziemlich fehlerfrei ist? Es ist eine nichttriviale und unexakte Wissenschaft, Software noch während ihrer Lebenszeit auf korrekte Funktion zu testen; Softwaretesting stellt eine eigene Industrie dar.

Wirtschaftlich sinnvolles Softwaretesting bringt zwar fehlerarme, doch kaum fehlerfreie Produkte auf den Markt. Dabei geht es nicht nur um unbeabsichtigte Fehler. Bei genügend kriminell untermauertem Marktmachtstreben könnte beispielsweise ein Unternehmen der Kosmetikindustrie ein Softwarepaket zur molekularen Modellierung von Faltenglättungssubstanzen anbieten, das garantiert nur Stoffe entwirft, die auf der Haut einen ausgeprägten Faltenwurf erzeugen. Bis der Wettbewerber dieses bemerkt, ist der Anbieter des quasi trojanischen Stoffpferdes längst Marktführer.

Mit dem industriellen Einsatz des Internets nun bringen Softwarefehler ein ganz neues Schadenspotenzial mit sich. Denn solch ein Fehler kann

- Beeinträchtigungen auf Fremdrechnern hervorrufen, für die wir zumindest zivilrechtlich haftbar sind,
- Informationen, deren Vertraulichkeit geschäftsentscheidend ist, über das Netz kompromittieren, bis wir wettbewerbsunfähig gemacht sind.

Es ist nicht Paranoia, an kriminelle Softwarefehler zu denken. Die National Security Agency (NSA) in den USA betreibt seit Jahren versteckte und direkte Finanzierungen von Unternehmen im Bereich Computersicherheit. Zuweilen dringen solche Bemühungen bis an die Oberfläche der Öffentlichkeit, wenn beispielsweise Autoren bestimmter Verschlüsselungsverfahren gerichtlich oder anderweitig bedroht oder größere Schlüssellängen einfach per Gesetz verboten werden. In Frankreich dürfen Unternehmen ihre Daten nur dann verschlüsselt übertragen, wenn Regierungsbehörden einen Zweitschlüssel erhalten. Und in Hongkong (wie in der restlichen Chinesischen Volksrepublik) ist Datenverschlüsselung unter Androhung drakonischer Strafen generell untersagt; einige Unternehmen haben deswegen ihre Hauptverwaltungen aus dem chinesischen Einflussbereich heraus verlagert.

Software auf (absichtlich oder unabsichtlich) versteckte Fehler zu untersuchen, gestaltet sich ungleich schwieriger, wenn der Quellcode nicht zu Verfügung steht und der übersetzte Binärcode analysiert werden muss. Das Ausmaß dieser Schwierigkeiten kann ermessen, wer als Veterinär Krankheiten eines Lebewesens diagnostizieren muss, dessen Anatomie ihm unbekannt ist.

Die traditionelle Softwareindustrie hat – ursprünglich, um geistiges Eigentum zu schützen – ein Übriges getan, Softwarepflege und Schutz vor Softwarefehlern weitgehend zu erschweren. Denn Software wird in aller Regel nicht verkauft, sondern eingeschränkte Nutzungsrechte werden lizenziert. Im Kleingedruckten der Lizenzierungsverträge werden bestimmte Rechte ausdrücklich ausgenommen, insbesondere das Recht, den Quellcode aus dem Binärcode zu restaurieren. Der Kunde verpflichtet sich also zu Schadenersatz, wenn er herausfinden möchte, was eine Software so alles bei ihm anstellt. Im selben Vertrag stellt sich der Softwareanbieter von der Haftung aus Fehlfunktionen frei.

Die Machtlosigkeit des traditionellen Softwarekunden zeigt sich besonders drastisch, wenn bei Online-Update-Prozeduren ein Fenster aufscheint: „Der Anbieter überträgt keine privaten Daten währen des Updates." Woher weiß denn der Anbieter, welche Daten privat sind? Hat er uns gefragt? Oder schon ausspioniert und vormundlich für uns entschieden, was wir als privat bezeichnen dürfen?

Das Minimum, was ein Kunde aus seinem eigenen Sicherheitsbedürfnis heraus von der Softwareindustrie erwarten darf, ja fordern muss:

- Der Anbieter muss Hilfsmittel mitliefern, die Fehlfunktionen aufspüren und beseitigen helfen. Das schlichtweg wichtigste Hilfsmittel ist der Quellcode.
- Will der Softwareanbieter den Quellcode nicht offen legen, so kann er die Haftung für Softwarefehler nicht auf den Kunden abwälzen. Die Lizenzverträge müssen das widerspiegeln.

Open Source als Geschäftsprinzip bestärkt somit die Kundenrechte deutlich. Der Kunde erhält uneingeschränkten Zugang zum Quellcode, kann ihn selbst nach Gutdünken analysieren und weiterverwenden. Open Source bringt als erwünschten Nebeneffekt durch das Prinzip, eine Vielzahl von Programmierern weltweit in jeder Projektphase Einblick in den Quellcode nehmen und die Programme schon in embryonaler Phase testen zu lassen, in aller Regel stabilere und weniger fehlerbehaftetere Programme. Bereits in den Anfangsjahren der Linux-Entwicklung wiesen deshalb Linux-Server legendäre unterbrechungsfreie Einsatzzeiten auf; Systeme, die in der traditionellen Softwarewelt auf Neustarts im Stunden- oder Tagesrhythmus angewiesen waren, liefen mit Linux Monate, sogar Jahre ohne softwarebedingte Unterbrechungen.

Sicherheit rund um Software erfordert Grundsatzentscheidungen. Auf die Auswahl der Lieferanten kommt es an. Die richtigen Lieferanten können folgende Fragen zufriedenstellend beantworten:

- Hast Du schon während der Softwareentwicklung eine breite, internationale Öffentlichkeit zum Testen eingesetzt?
- Darf ich deine Antworten nachprüfen (lassen) und mir deine Software im Quellcode ansehen?

Damit die Rechte der Kunden – insbesondere die Sicherheitsbedürfnisse – respektiert werden, hat sich SuSE unwiderruflich für Open Source als Geschäftsgrundlage entschieden. Mit Binärcode allein ist der Sicherheit kaum zu dienen.

Richard Seibt,
Chief Executive Officer,
SuSE Linux AG

Aus den Hochschulen

Prof. Harald Eichsteller, Jahrgang 1961, der als Manager E-Business und Geschäftsführer der Aral Online GmbH tätig war und nach der Fusion mit BP das Online-Business verantwortete, hat in der Hochschule der Medien (HdM) in Stuttgart eine Professur für Internationales Medienmanagement übernommen. Seine Arbeitsschwerpunkte sind Medienproduktion, Online-Marketing und E-Business.

Mario Jeckle, Jahrgang 1974, der bislang als Forschungsgruppenleiter bei DaimlerChrysler in Ulm beschäftigt war, hat an der Fachhochschule Furtwangen im Fachbereich Wirtschaftsinformatik den Ruf auf eine Professur für Software-Engineering angenommen. Seine Forschungsschwerpunkte sind Datendarstellung und -modellierung, insbesondere objektorientierte Modellierung mit UML sowie Weiterentwicklung und Anwendung der Metasprache XML (http://www.jeckle.de).

Das E-Finance Lab Frankfurt a. M., ein von der Wirtschaftspraxis (z. B. Accenture, Deutsche Bank, Deutsche Postbank, Microsoft, Siemens, T-Systems) für zunächst drei Jahre finanziertes und gemeinsam von den Universitäten Frankfurt a. M. und Darmstadt getragenes Forschungsprogramm, wurde mit einem Symposium, in dessen Rahmen der Hessische Ministerpräsident Koch und weitere Wissenschaftler und Praktiker das Wort ergriffen haben, eröffnet. Ziel ist die Entwicklung und Erprobung von Verfahren, um Finanzdienstleistungsunternehmen bei der Industrialisierung ihres Geschäfts zu unterstützen, z. B. hinsichtlich des Aufbrechens von bislang weitgehend hausintern realisierten Wertschöpfungsketten und deren neuartigen Zusammensetzung unter Einbezug von qualitätsgesicherten Zulieferungen. Seitens der Universitäten Frankfurt und Darmstadt tragen die **Professoren König**, **Skiera**, **Wahrenburg** und **Steinmetz** die Verantwortung (http://www.efinancelab.de).

Dr. sc. nat. Christopher Lueg, Jahrgang 1966, der seit seiner Promotion an der Universität Zürich als Senior Lecturer in Information Systems an der University of Technology, Sydney, Australien, tätig ist, hat einen Ruf auf den Stiftungslehrstuhl E-Business (Chair in E-Business, der von Computer Sciences Corp. (CSC) unterstützt wird) an der in Gründung befindlichen Charles Darwin University in Darwin, Australien, erhalten.

Prof. Dr. Thomas Myrach, Jahrgang 1961, vorher als Lehrstuhlvertreter an der RWTH Aachen tätig, hat einen Ruf auf eine ordentliche Professur für Wirtschaftsinformatik an der Universität Bern angenommen und ist seit Ende 2002 Mitdirektor des dortigen Instituts für Wirtschaftsinformatik. Seine Forschungsschwerpunkte sind Informationsmanagement mit dem Fokus auf Datenbanksystemen sowie die betriebliche Nutzung von Internettechnologien im Zuge des E-Business (http://www.im.iwi.unibe.ch/).

Prof. em. Dr. Dieter Preßmar, Jahrgang 1936, der an der Universität Hamburg bis 2002 das Institut für Wirtschaftsinformatik leitete, wurde von der Fakultät für Wirtschaftswissenschaften der Universität Siegen die Würde eines Doktors ehrenhalber verliehen. Die Laudatio betonte seine besonderen Leistungen bei Gründung und Aufbau des Fachs Wirtschaftsinformatik in der Bundesrepublik Deutschland.

Prof. Dr.-Ing. Rainer Schmidt, Jahrgang 1965, hat einen Ruf an die Fachhochschule Aalen auf eine Professur für Wirtschaftsinformatik angenommen. Zuvor war er in einer Unternehmensberatung sowie als Professor an der Berufsakademie Lörrach tätig. Seine Forschungsgebiete sind unternehmensübergreifende Geschäftsprozesse und deren Optimierung sowie IT-Management.

Prof. Dr. Eberhard Stickel, Jahrgang 1958, der an der Wirtschaftswissenschaftlichen Fakultät der Europa-Universität Frankfurt (Oder) die Professur für Allgemeine Betriebswirtschaftslehre, insbesondere Wirtschaftsinformatik, Finanz- und Bankwirtschaft, bekleidet, hat als Gründungsrektor die Leitung der Hochschule der Sparkassen-Finanzgruppe – University of Applied Sciences – in Bonn übernommen. Prof. Stickel wurde für seine neue Tätigkeit beurlaubt. Informationen über die Hochschule der Sparkassen-Finanzgruppe, die im Juli 2003 mit den ersten zwei Bachelor-Studiengängen, darunter ein Studiengang „Bachelor of Financial Information Systems“, startet, findet man unter http://www.s-hochschule.de.