DuD
Datenschutz und Datensicherheit

Öffnungszeiten

Dirk Fox

Die Diskussion um „offene Software“ dauert an – nicht nur, weil sie das Geschäftsmodell eines der größten, schillerndsten und profitabelsten Unternehmen der Welt bedroht, sondern weil sie die Existenzberechtigung einer der (wenigen) verbliebenen Branchen mit zweistelligen Wachstumsraten und wirtschaftlichen Hoffnungsträger moderner Wissensgesellschaften auf den Prüfstand stellt.

Die Diskussion ist alt. Freie Software mit offen gelegtem Code gibt es seit mehr als 20 Jahren. Die profilierteste Initiative ist das von Richard Stallman 1984 initiierte GNU[1] Projekt zur Entwicklung eines Unix-artigen, freien Betriebssystems, das seit 1985 von der gemeinnützigen „Free Software Foundation“ (FSF) getragen wird. Heute zählt der Index der FSF mehr als 4.700 unter der GNU-Lizenz („Copyleft“) entwickelte Softwarepakete – darunter das wohl bekannteste Open-Source-Programm Linux, aber auch GnuPG.

Tatsächlich ist die Diskussion weit vielschichtiger, als auf den ersten Blick zu erkennen. Denn es geht nicht allein um urheberrechtliche Grundsatzfragen und wirtschaftliche Interessen. In die Diskussion mischen sich auch gänzlich andere Motive und Motivationen: der Ehrgeiz guter Programmierer mit genügend Zeit- und finanziellen Ressourcen, die Lust, an einer großen Herausforderung unter hehren Zielen mitzuarbeiten, die Wut über ignorantes Verhalten zahlreicher Hersteller gegenüber ihrer von der Funktionsfähigkeit und dem Support der Software oft abhängigen Kundschaft und vielleicht auch hier und da ein Quäntchen revolutionäre Grundstimmung. Jenseits der unterschiedlichen Argumente gibt es sicherlich zwei Grundwahrheiten:

- Die Entwicklung freier Software hat ein Mindestmaß an wirtschaftlicher Prosperität zur Voraussetzung – wer seine Zeit für die Finanzierung seines Lebensunterhalts benötigt, der kann nur in sehr begrenztem Umfang an einem solchen Projekt mitwirken.

Etwas befremdlich wirkt es da, wenn die Bundesregierung mit Steuergeldern – auch von Software-Herstellern – Open-Source-Entwicklungen finanziert: man sollte tunlichst nicht an dem Ast sägen, auf dem man sitzt.

- Auch wenn ihm die Ehre als Entgelt für seinen Einsatz genügt, steht auch für den Entwickler in einem Open-Source-Projekt die Urheberschaft seiner Leistung außer Frage.

Daher darf man „offenen Code“ nicht mit „freiem Code“ verwechseln, auch wenn es der eine oder andere Hersteller damit nicht so genau nimmt – eine riskante Strategie.

Zunehmend ist die Sicherheitsfrage ins Zentrum der Diskussion um Open-Source-Software gerutscht. Für beide Seiten ein beliebtes „Killerargument“ – wer wollte schon prinzipiell unsicherere Software einsetzen? Bei der Einschätzung, welches Modell wohl die sicherere Software hervorbringt, gehen die Meinungen allerdings stark auseinander.[2] Ohne Frage ermöglicht offen gelegter Code jedermann ein Review. Allein die Möglichkeit zum Review garantiert jedoch noch nicht das gute Ergebnis – wenn niemand den Code prüft, bleiben sicherheitskritische Schwachstellen unentdeckt. Und „Closed Source“ ist zweifellos nicht dem Prinzip „Security by Obscurity“ gleich zu setzten, wie gerne unterstellt wird. Denn der Verzicht auf unbeschränkte Offenlegung des Codes bedeutet nicht, dass der Hersteller keine standardisierten Verfahren einsetzt oder auf eine Qualitätssicherung des Codes verzichtet. Schließlich gibt es gute Argumente dafür, Fehler nicht durch Code-Review, sondern durch intensives Testen aufzudecken – ein Verfahren, zu dem die Kenntnis des Source Code nicht erforderlich ist.

Umgekehrt ist es allerdings auch nicht in jedem Softwareunternehmen mit der Qualitätssicherungs- und Testdisziplin sehr gut bestellt: Die zahlreichen sicherheitskritischen Bugs, die immer wieder gefunden werden, belegen den Nachholbedarf.

Tatsächlich gibt es bislang keinen stichhaltigen Beleg für eine der beiden Behauptungen. Als „analytische Annäherung“ wird daher mit Metriken gearbeitet: Über feste Zeiträume werden für vergleichbare Produkte die aufgedeckten sicherheitskritischen Fehler und der Umgang damit (Bugfixes, Reaktionszeiten etc.) beobachtet.[3]

Auch wenn die Wogen gelegentlich hoch schlagen, kann man den Fortgang der Diskussion getrost mit einer gewissen Gelassenheit verfolgen.

Denn erstens bleibt ein zentrales Support-Problem: Viele Open-Source-Projekte leben von und mit dem Elan des Initiators – tritt er ab, hängt der Fortgang des Projekts davon ab, ob sich ein vergleichbar engagierter Nachfolger findet. Das ist nur bei Projekten mit großen Nutzerzahlen verlässlich zu erwarten – ein kritischer Punkt für den Investitionsschutz derjenigen, der sich auf die Open-Source-Lösung eingelassen haben.

Zweitens belebt die Konkurrenz das Geschäft: Ohne den Frontalangriff von Linux wäre Bill Gates „Sympathieoffensive“ und sein Schwenk zu sicherer Softwareentwicklung möglicherweise gar nicht, sicherlich aber später gekommen. So gesehen konnte Microsoft kaum etwas Besseres passieren als Linus Torvalds. Auch mit Rom ging es nach der Zerstörung Karthagos bergab – lieber ein starker Gegner als gar keiner. Linux als Retter von Microsoft – dass hatte Torvalds sicher nicht geplant.

Und drittens bringt die Open Source-Szene eine Vielzahl hoch kompetenter, erprobter und erfahrener Entwickler hervor, die die Phase der technischen Detailverliebtheit hinter sich gelassen haben, „das Ganze“ nicht aus dem Blick verlieren und es gewohnt sind, ihre Leistung am Feedback der Nutzer zu messen – eine große Chance für die Sicherheit zukünftiger kommerzieller Produkte, denn viele dieser Entwickler werden irgendwann ihr Hobby zum Beruf machen.

[1] “GNU is not Unix”; www.gnu.org

[2] Siehe die lesenswerte Übersicht von *Rolf Oppliger*, DuD 11/2003, S. 669-675.

[3] Siehe *Stefan Hunziker* und *Simon Rihs*, in diesem Heft.