Open-Source in der Verwaltung

Michail Bachmann
michail.bachmann@cms.hu-berlin.de

Michael Bell
michael.bell@cms.hu-berlin.de

Warum wird heute in vielen Bereichen, die die Sicherheit der Universitätsverwaltung berühren, so strikt Wert auf Open Source gelegt? Was unterscheidet Open-Source-Software (OSS) von »normaler« Software? Der folgende Artikel versucht, auf diese Fragen eine Antwort zu geben.

Open Source im Alltag

(am Beispiel E-Mail-Verschlüsselung der Personalabteilung)

Aus Benutzersicht:

Max Mustermann erscheint wie jeden Tag pünktlich um acht Uhr, startet sein E-Mail-Programm und liest die erste Mail. Pflichtbewusst antwortet er sofort mit einer verschlüsselten E-Mail, da es sich um Personaldaten handelt.

Aus Open-Source-Sicht:

1. Starten des E-Mail-Programms und Login
Mozilla, früher auch unter dem Namen Netscape bekannt, sucht sich seine Hilfsprogramme zusammen und startet – selbstverständlich ohne registriert werden zu müssen. Nach dem Laden der Einstellungen sucht Mozilla sich seinen Weg zum ausliefernden Mailserver der Verwaltung. Auf dem Weg dorthin wird es mit Adressen (DNS-Server Bind), Verschlüsselung (OpenSSL) und einem IP-Filter (OpenBSD, ipf, carp ...) versorgt. Endlich auf dem ausliefernden Mailserver (Cyrus IMAP-Server) angekommen, muss Max Mustermann sein Login und sein Passwort angeben. Dieses wird über einen LDAP-Server (OpenLDAP, PADL PAM-Modul) verifiziert.

2. Lesen einer E-Mail
Bevor Herr Mustermann nun glücklich seine E-Mail lesen kann, muss sie erst einmal empfangen werden. Dies erledigt Sendmail. Selbstverständlich wird vorher der Spam durch SpamAssassin erkannt und vorsortiert.

3. Senden einer E-Mail
Die Beantwortung der E-Mail erfordert »leider« eine Verschlüsselung, da es sich um eine E-Mail mit personenbezogenen Daten handelt. Dies wird durch Mozilla erledigt, welcher zuvor Zertifikate von der universitätseigenen PKI erhalten hat (OpenCA). Selbstverständlich sind diese Zertifikate via WWW (Apache) oder LDAP (OpenLDAP) herunterladbar. Das Senden übernimmt dann wieder der nette Postfix-Server von nebenan.

Alle bisher erwähnten Produkte sind Open-Source-Software und stehen damit kostenlos zur Verfügung. Jetzt stellen Sie sich mal vor, Sie müssten für all das bezahlen bzw. die Universität müsste dafür Geld ausgeben.

Mit freundlichen Grüßen
Ihre Heinzelmännchen

Was ist Open-Source-Software?

Dem Namen nach bedeutet Open Source (dt. quelloffen) erst einmal nur, dass der Benutzer der Software Zugriff auf den Quelltext dieser Software hat.

Es gibt z. B. von Microsoft die Shared Source Initiative [1], die es ausgewählten Benutzern erlaubt, auf Teile des Quelltextes nach Maßgabe von Microsoft zuzugreifen. Eine Verwendung, Weitergabe oder Veränderung des Quellcodes für eigene kommerzielle Entwicklungen ist nicht erlaubt [2]. Ist Windows damit Open-Source-Software?

Manche Hersteller – wie z. B. die HIS GmbH mit QISPOS oder die PGP Corporation mit dem Kryptografieprogramm PGP – liefern den Quelltext ihrer Software mit aus. Obwohl Weitergabe oder Veränderung nicht erlaubt sind, sieht sich die PGP Corporation explizit als Hersteller von Open-Source-Software [3].

Bei anderer Software – wie z. B. dem Web-Browser Mozilla von der Mozilla Foundation oder dem Betriebssystem Linux – wird der Quelltext der Software mitgeliefert und darf von jedem verändert und auch weitergegeben werden.

Offensichtlich reicht die Verfügbarkeit des Quelltextes als Unterscheidungsmerkmal allein nicht aus, um das Besondere an Open-Source-Software zu erkennen. Dieses Problem wurde auch von der Open Source Initiative (OSI) [4] erkannt. Sie hat Abgrenzungsmerkmale gefunden, die die Besonderheiten von Open-Source-Software widerspiegeln. Wenn ein Softwarelizenzvertrag folgende Bedingungen erfüllt, kann die Software als Open-Source-Software im engeren Sinne bezeichnet werden:

- Die Software (d. h. deren Quelltext) liegt in einer für den Menschen lesbaren und verständlichen Form vor.
- Die Software darf beliebig kopiert, verbreitet und genutzt werden.
- Die Software darf verändert und in dieser veränderten Form weitergegeben werden.

Es gibt für Open-Source-Software keine Nutzungsbeschränkungen, weder bezüglich der Anzahl der Benutzer, noch bezüglich der Anzahl der Installationen. Mit der Vervielfältigung und der Verbreitung von Open-Source-Software sind auch keine Zahlungsverpflichtungen gegenüber einem Lizenzgeber verbunden. Durch den offen gelegten Quelltext ist Verändern ohne weiteren Aufwand für jeden möglich. Man erkennt, dass von den oben angeführten Beispielen nur die letzte Gruppe alle Bedingungen erfüllt, um als Open-Source-Software bezeichnet zu werden.

Besonderheiten bei der Softwareentwicklung

Durch die Rechte, die eine Open-Source-Lizenz den Anwendern der Software verleiht, unterscheiden sich die Entwicklungsmethoden von OSS deutlich von denen herkömmlicher Software. Eric S. Raymond hat den Unterschied 1997 in seinem Essay »The Cathedral and the Bazaar« (Die Kathedrale und der Basar) [5] sehr bildlich beschrieben: Die herkömmliche Softwareentwicklung gleicht dem Bau einer Kathedrale »entworfen von einzelnen, erleuchteten Künstlern oder einer Handvoll auserwählter Baumeister hinter gut verschlossenen Türen« [6]. Im Gegensatz dazu gleicht die Entwicklung von Open-Source-Software »einem großen plappernden Basar mit verschiedenen Tagesabläufen und Ansätzen«[6] und mit einem »Maintainer«, dem »Wächter über das Marktrecht« [7]. Probleme werden gelöst, wenn sie auftauchen; es gilt das Prinzip »Given enough eyeballs, all bugs are shallow« (Hat man genügend hilfreiche Köpfe, sind alle Fehler offensichtlich.). Der Maintainer koordiniert das Projekt und sorgt dafür, dass Regeln eingehalten werden.

Diese spezifische Art der Softwareentwicklung ist für folgende charakteristische Eigenschaften von Open-Source-Software verantwortlich:

- Investitionssicherheit,
- Qualitätssicherung,
- Support,
- Überprüfbarkeit und
- Anpassbarkeit.

Investitionssicherheit

Die Investitionssicherheit steht nicht ohne Grund an erster Stelle. In vielen Bereichen wurden mit kommerziellen Firmen schlechte Erfahrungen gemacht. Hierbei sind nicht nur die Branchenführer ein Problem, die nicht in der Lage sind, auf Kundenanforderungen angemessen zu reagieren, sondern auch kleine und mittelgroße Firmen. Neben der oft mangelhaften Fähigkeit, auf neue Probleme und Herausforderungen in angemessener Zeit zu reagieren und Lösungen bieten zu können, werden auch immer wieder Produkte einfach eingestellt (Banyan-VINES-Mail, F-Secure VPN+, SuSE Mailserver oder auch Kerios Personal Firewall), Wartungsverträge gekündigt (F-Secure VPN+), Firmen verschwinden vom Markt (Banyan) oder die Lebenszyklen der Software sind einfach zu kurz im Verhältnis zur Laufzeit der Systeme (SuSE Mailserver oder die diversen Microsoft Update Services). Dies trägt nicht wirklich dazu bei, auf kommerzielle Anbieter für sicherheitsrelevante Software zu vertrauen.Wer möchte schon gern seine persönlichen Daten im Klartext durch das normale Universitätsnetz oder gar Internet wandern sehen, weil der Firewall- oder VPN-Anbieter aus irgendeinem Grund die Lust am Produkt verloren hat?

Qualitätssicherung

Wenn jemand zum ersten Mal mit der Idee von verteilter Softwareentwicklung und Open Source konfrontiert wird, gibt es immer wieder Bedenken bezüglich der Qualität der Software. Wenn jeder einfach so die Software verändern kann, wer garantiert dann die Funktionsfähigkeit?

Solche Bedenken resultieren meist aus einem Missverständnis bezüglich des Entwicklungsmodells von Open-Source-Software.

Die Qualitätssicherung wird bei Open-Source-Software vom Maintainer übernommen. Obwohl jeder die Software ändern darf und diese Änderungen auch weitergeben kann, sind diese jedoch erst einmal unabhängig von der Software – als Patches – verfügbar. Erst durch die Aufnahme der Patches in das Projekt durch den Maintainer werden diese ein vollwertiger Bestandteil des Projektes.

Support

Der Support seitens des Herstellers ist ein oft angeführtes Argument für kommerzielle Software. Die letzten Jahre haben aber eher das Gegenteil aufgezeigt. So kann man jahrelang für Datenbanksupport bezahlen, aber wenn die Datenbank tatsächlich abstürzt, wird man darauf verwiesen, dass man mit dem Supportvertrag auch das Recht erworben hat, auf die nächste Version zu upgraden und mit dieser alten Variante kennt sich eigentlich auch niemand mehr aus. Falls die Software zusätzlich über eher unterdurchschnittliche Debuggingfähigkeiten und auch eher nichts sagende Fehlermeldungen verfügt, ist das Dilemma perfekt. Ein Umstieg auf eine neue Variante ist auch keine Sache von Minuten, da die Hersteller sich meistens nur auf die Aussage einlassen, dass es grundsätzlich gehen müsste. Diverse, nicht mehr laufende Skripte oder kleinere Änderungen an der Datenbank-Engine können halt mal passieren. Dies sollte allerdings keinesfalls den Eindruck erwecken, dass Datenbankhersteller einen schlechten Support bieten. Im Gegenteil, man bekommt wenigstens noch Antworten und im Ernstfall bei der nötigen Umstellung auch Hilfe (wenn auch gegen weiteres Geld). Andere Hersteller ignorieren Anfragen oder man bekommt Aussagen der Preisklasse »Das ist halt so!«. Fehlermeldungen werden zwar aufgenommen, wann sie aber behoben und nicht nur umgangen werden, ist oftmals genauso wenig bekannt wie der Weg einer Fehlermeldung zu den eigentlich Entwicklern. Solange der Support sich auf Standardfehler durch den Anwender bezieht, ist er im Allgemeinen gut. Treten aber schwerwiegende Probleme auf, bei denen die Systeme an ihre Grenzen geraten, ist von gutem Support oft nicht viel zu spüren. Dabei spielt es keine Rolle, ob es um Windows-Probleme oder abstürzende Unix-Datenbanken und Unix-Filesysteme geht. Dieses Problem nur auf Microsoft zu beziehen, geht glatt am Kern des Problems vorbei. Auch bei Open Source ist nicht alles eitel Sonnenschein. Hier gibt es mit alten Versionen schon mal Probleme, weil die Entwickler nicht mehr da sind (wie bei kommerzieller Software) oder keiner mehr Interesse daran hat oder die Entwickler so ein System vor zwei oder drei Jahren das letzte Mal angefasst haben. Bei diffizilen Problemen bieten Mailinglisten fast immer Abhilfe. Die Reaktionszeit ist typischerweise sehr kurz. Größere Probleme kann es schon eher mal mit exotischen Konfi-

gurationen oder Problemen geben, wenn sich niemand auf einer Mailingliste dafür interessiert. Der grundsätzliche Vorteil von Open Source liegt darin, dass man kommerziellen Support kaufen kann aber nicht muss, und im Zweifelsfall ist der Code für jeden einsehbar, was aber eigenes qualifiziertes Personal voraussetzt. Dieses wird zwar auch für kommerzielle Software benötigt, diese Erkenntnis hat sich allerdings noch nicht überall herumgesprochen.

Überprüfbarkeit

Die Überprüfbarkeit von Software wird oft auch unter solchen Schlagworten wie Auditing oder Reviews vermarktet. Kommerzielle Anbieter versuchen dies durch Zertifizierungen gemäß verschiedenen Standards nachzuweisen. Bei großen Systemen wie hochkomplexen Betriebssystemen oder auch Firewalls wird im Regelfall nur eine einzige Konfiguration zertifiziert. Diese entspricht so gut wie nie den in der Praxis eingesetzten Konfigurationen, da die zertifizierten Systeme natürlich extrem auf Sicherheit getrimmt werden. In solchen Fällen kann es schon mal vorkommen, dass wesentliche Peripheriegeräte einfach fehlen. Ein weiteres großes Problem ist die Wartbarkeit solcher Systeme, weil streng genommen jeder Patch neu zertifiziert werden muss. Dieser Aufwand ist natürlich nicht zu leisten, so dass die Zertifizierung nur für ein immer weiter eingeschränktes System gilt und oftmals das zertifizierte System erhebliche Schwachstellen aufweist, da die Zertifizierung nicht aktualisiert wird. Es gibt heute mehrere Ansätze, diese Probleme zu beheben – bis hin zur beschränkten Weitergabe des Codes. Allerdings sind Non-Disclosure Agreements (NDAs) ein eigenes Thema.

Open Source geht hier, wie der Name schon sagt, einen anderen Weg. Die Quellen stehen jedermann zur Verfügung. Nicht nur potentielle Angreifer, sondern auch Entwickler, interessierte Anwender und – nicht zu vergessen – Forscher und Lernende, wie zum Beispiel Studenten, können sich den Code anschauen. Dabei können sie nicht nur den Code und die Software verstehen lernen, sondern die meisten Open-Source-Lizenzen erlauben auch die Anpassung und Weitergabe des geänderten Codes. Damit ist der Kreis der Personen, die den Code überprüfen können, wesentlich größer und man ist nicht auf die Kooperation des Herstellers angewiesen, wenn man einem Verdacht oder Problem nachgehen möchte. Ein besonders schönes Beispiel hierfür ist sicher OpenSSL, das von vielen Forschern zu Tests benutzt wird und wodurch schon mehrere Schwachstellen in einem standardisierten Protokoll oder einem Best-Practice-Model entdeckt wurden.

Anpassbarkeit

Der letzte Punkt, die Anpassbarkeit, ist sicher die Paradedisziplin von Open Source. Hier hat de facto kein aktuelles kommerzielles Lizenzmodell etwas Vergleichbares zu bieten. Hierbei ist zu beachten, dass der einzelne Nutzer nicht nur die Software ändern kann, sondern er kann diese Änderungen auch weiter- und vor allem zurückgeben. Der wesentliche Punkt ist hierbei sicher auch eine Art Gentleman-Agreement im Open-Source-Bereich. Die freie Software und der offen zugängliche Code steigern die Motivation, Änderungen dem jeweiligen Projekt/Hersteller zur Weiterverwendung zu überlassen, erheblich. Es kommen sicher nur die wenigsten auf die Idee, eine Änderung an kommerzieller, bezahlter Software dem Anbieter kostenlos zur Verfügung zu stellen. Dies führt dann dazu, dass jeder Anwender für sich eine Lösung schaffen muss, zumal viele Lizenzen und auch NDAs hier noch zusätzliche Hürden aufbauen. Es gibt sogar Hersteller, die an die Nutzer einer User-Mailingliste eine unfreundliche E-Mail verschicken, wenn ein Problem zuerst auf der Liste diskutiert wird – wodurch alle Mit-Leser von dem Problem (und auch von der potentiellen Lösung oder Nicht-Lösung) wissen – und nicht der Hersteller die Chance bekommt, das Problem vorher stillschweigend mit dem nächsten Release zu lösen.

Open-Source-Software lebt von der aktiven Beteiligung der Anwender an der Entwicklung. Diese Beteiligung muss nicht nur durch das Schreiben von Quellcode geschehen, schließlich kann nicht jeder programmieren. Hinweise zu Programmfehlern oder auch Verbesserungsvorschläge zur Optik der Programme sind hilfreich.

Anwendungen in der Verwaltung

Im Bereich der Abteilung DV in der Verwaltung kommt Open-Source-Software in vier Schwerpunktgebieten zum Einsatz – Mail, Web, Firewall und PKI.

Das Thema Mail wird hier nicht weiter betrachtet, da die entsprechende Technik nicht von dieser Abteilung betrieben wird, sondern die Dienste zwar als separate Server laufen, aber komplett von den Mitarbeitern betrieben werden, die auch die zentralen Mailserver betreuen. Dies ist im Rahmen der Kompetenzbündelung und somit bester technischer Betreuung auch gar nicht anders möglich.

Auf allen Webservern des CMS wird der Apache als Serversoftware eingesetzt. Dieser, innerhalb der Apache Software Foundation entwickelte Server, wird auf ca. 70% aller Webserver im Internet eingesetzt. Dynamische Inhalte werden überwiegend mit Hilfe einer der Open-Source-Programmiersprachen PHP, Perl oder Python erzeugt. Der Zugriff auf das WWW erfolgt zumeist mit Open-Source-Software, nämlich dem Mozilla- oder dem Firefox-Browser.

Ein schon fast traditioneller Schwerpunkt unserer sicherheitsrelevanten Arbeit ist die Firewall. Alle Verwaltungsmitarbeiter, die sich »hinter« ihr befinden, kennen sie und haben sie auch schon mindestens einmal in ihrer Dienstzeit verflucht. Nichtsdestotrotz ist die Akzeptanz derselbigen für ein DV-System sehr hoch. Was steht also technisch hinter diesem abstrakten Begriff und was leistet das System? Technisch besteht die komplette Firewall aus Open-Source-Komponenten. Dies beginnt bei der Wahl von OpenBSD als Betriebsystem, das nicht nur sehr schlank ist, sondern auch von Grund auf für maximale Sicherheit entwickelt und nicht erst mit Features voll gestopft wurde und danach die Sicherheit angebaut bekam. Dieses Betriebssystem beinhaltet zum

Beispiel auch schon jene Filter (pf, carp, pfsync), die nicht nur Verbindungen zu gewissen Peer-To-Peer-Netzen unterbinden, sondern auch diverse Einfallstore für Viren und Würmer zuverlässig geschlossen halten. Neben diesem an sich schon sehr effektivem Schutzmechanismus verfügt die Firewall noch über diverse so genannte Application-Gateways. Das sind Server, die ankommende und abgehende Datenströme von und für Anwendungen verstehen und interpretieren können. Im Zweifelsfall werden solche Datenströme zusätzlich gesichert, eingeschränkt oder im Extremfall gekappt. Beispiele hierfür sind Filter für Web-Inhalte (squid). Die meisten Anwender nörgelten an der JavaScript-Filterung nur solange herum, bis eine kurze Demo die Kompromittierung ihres normalen Arbeitsplatzrechners mit einer ungefilterten Seite demonstrierte. Dies ist wahrscheinlich auch einer der wichtigsten Punkte für die hohe Akzeptanz eines Systems, welches streng genommen die Freiheiten der Mitarbeiter einschränkt. Zum einen ist trotz allem der Zugang zum Internet möglich und zum anderen ist es bisher noch zu keinem einzigen Wurmbefall innerhalb des geschützten Bereiches gekommen. Wenn also wieder einmal ein Wurm den Totalausfall ganzer Firmen und sogar Ministerien nach sich zog, konnte die Verwaltung hinter der Firewall recht ungestört arbeiten – mit Ausnahme der in diesem Fall korrekten Fehlermeldung, dass das Internet nicht funktioniert bzw. kaputt ist.

Ein weiterer Schwerpunkt für Open-Source-Sicherheitssoftware ist unsere Public Key Infrastructure (PKI). Das Trustcenter als Grundlage der PKI basiert auf OpenCA. In diversen Artikeln haben wir diese Entscheidung und die Hintergründe dafür schon ausführlich dargelegt und bisher kann man zusammenfassend sagen: Wir haben es nicht bereut. Mittlerweile können wir sogar stolz feststellen, dass auch andere Einrichtungen wie der DFN-Verein sich in diesem Bereich engagieren, so dass die im Rahmen des DFN-Projektes »UVsec« [8] investierten Mittel sicher gut angelegt waren und sind. Eine Infrastruktur besteht aber nicht nur als zentrale Komponente, die zum Selbstzweck Zertifikate und eventuell Smartcards ausstellt. In gewissen Fällen spielen auch Anwendungen eine Rolle. Hier kommen dann zum Beispiel Mail- und Web-Server aber auch Klienten wie Mozilla zum Einsatz. Wenn also Mitarbeiter oder Studierende ihre Mails über eine verschlüsselte Verbindung zum Mailserver lesen, damit nicht alle anderen WLAN-Nutzer mitlesen können, dann kommt dafür Open Source zum Einsatz. Dasselbe passiert, wenn Studierende sich zur Prüfung anmelden möchten. Die Applikation selbst stammt von der HIS GmbH, aber alle sicherheitsrelevanten Teile, die direkten Kontakt zum Internet haben – wie z. B. Web- (Apache und OpenSSL) und Applikations-Server (Tomcat) – basieren auf Open Source. Dies zeigt, dass selbst bei Firmen die Einsicht herrscht, dass es nicht ganz so schlecht ist, sich auf Open Source zu verlassen. Eine etwas herausragende Rolle spielen Systeme, die besonders kritisch sind, wie zum Beispiel die Verschlüsselung sämtlicher Mails mit personenbezogenen Daten in der Personalabteilung. In solchen Fällen kommen oftmals reine Open-Source-Lösungen zum Einsatz, da zum einen der Betrieb unbedingt sichergestellt werden und zum anderen die Möglichkeit der vollständigen Problemsuche auch für uns selbst gegeben sein muss.

Fazit

Auch wenn als Erstes immer der Kostenaspekt von Open-Source-Software ins Auge fällt, so ist doch die mit dem Einsatz gewonnene Freiheit nicht zu unterschätzen.

Im Zusammenhang mit Vergleichen von Open Source und Closed Source werden oft Kritikpunkte wie Betreuungsaufwand, Anpassung oder Verantwortlichkeit bei Fehlfunktionen als Nachteile von Open Source angeführt. Häufig wird hierbei übersehen, dass diese Kritik Open Source und Closed Source Software gleichermaßen betrifft.

Es ist möglich, Open Source Software aus Sicht der Systemverwalter genauso zu behandeln wie Closed Source Software. Bei beiden Lizenzmodellen ist es möglich, Support extern einzukaufen.

Bei Closed Source Software ist dieser Zukauf jedoch die einzige Möglichkeit zur Anpassung der Software beim Auftreten größerer Probleme oder Änderungswünschen. Im Gegensatz dazu ist es bei Open Source Software zusätzlich möglich, eigenes Know-How zu nutzen. Welche Möglichkeit kostengünstiger ist, hängt von den lokalen Gegebenheiten, Risiken und vorhandenen Kompetenzen ab.

Die Vorteile von Open Source können am besten bei Infrastruktur- oder auch bei Massensoftware ausgespielt werden. Wenn die Anforderungen an die Software hinreichend ähnlich sind, dann können individuelle Anstrengungen der Allgemeinheit zugute kommen. Durch Open Source wird sichergestellt, dass jedes Problem – idealerweise – nur einmal gelöst werden muss. Diese Lösung kann dann von allen anderen verwendet werden.

Dieser Mechanismus greift jedoch nicht bei Individualsoftware, d. h. Software, die speziell auf einen Kunden zugeschnitten ist. Aber auch da wäre es wegen der Investitionssicherheit wünschenswert, wenn die Software im Quelltext vorliegen würde.

Literatur

[1] http://www.microsoft.com/resources/sharedsource/default.mspx
[2] http://www.microsoft.com/resources/sharedsource/Licensing/Enterprise.mspx
[3] http://www.pgp.com/company/letterceo.html
[4] http://www.opensource.org/
[5] http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/
[6] http://www.linux-magazin.de/Artikel/ausgabe/1997/08/Basar/basar.html
[7] http://de.wikipedia.org/wiki/Die_Kathedrale_und_der_Basar
[8] http://www.cms.hu-berlin.de/ueberblick/projekte/uvsec/