LEGAL REVOLUTIONary
Rechtsmagazin der digitalen Wirtschaft

Automatisiertes Open Source Lizenzmanagement

Open Source Software erfassen, rechtlich prüfen und rechtssicher einsetzen – so automatisiert wie möglich

Dan-Alexander Levien | Rechtsanwalt | Syndikusrechtsanwalt | Mediator | Systemischer Coach | Leiter Rechtservice, Audi Electronics Venture GmbH

22. Mai 2020

LR 2020, Seiten 156 bis 160 (insgesamt 5 Seiten)

Die Nutzung von Open Source Software ist inzwischen Standard in der modernen Softwareentwicklung. Ihr Einsatz erfordert zwingend die Einhaltung der Lizenzbedingungen, um Rechtsverletzungen zu vermeiden. Denn im Falle von Lizenzverstößen drohen dem Verwender Konsequenzen, von Schadensersatz über Produktions- und Verkaufsausfälle bis hin zu Reputationsverlusten. Die Lösung ist ein professionelles Lizenzmanagement von Open Source Softwarebestandteilen und den jeweiligen Lizenzen. 1

I. Einleitung

Was ist eigentlich Open Source Software und warum ist das Thema für Sie in Ihrem Alltag von Relevanz? 2

Sehr verkürzt beschrieben lässt sich Open Source Software (OSS) wie folgt charakterisieren:

- sie ist in einer von Menschen lesbaren Form (Quelltext/Sourcecode) verfügbar;
- sie darf beliebig kopiert, verbreitet und genutzt werden;
- sie darf frei verändert und in der veränderten Form weitergegeben werden.[1]

Ein Grundgedanke von OSS ist, Verwender zur aktiven Beteiligung an der (Weiter-) Entwicklung einzuladen. OSS lebt davon, dass Verwender daran mitarbeiten, sie zu verbessern. 3

OSS finden wir inzwischen überall, wo Software eingesetzt wird, z.B. im Auto, Smartphone, Fernseher oder Staubsauger bis hin zur Küchenmaschine. Die Nutzung von OSS unterliegt hierbei Spielregeln, die in Lizenzbedingungen zu finden sind. Ein 4

[1] Vgl. die von der Open Source Initiative angewendeten Charakteristika, zu finden unter https://opensource.org/osd.

professionelles Lizenzmanagement ist ein Beitrag zur Einhaltung der Lizenzbedingungen und zum aktiven Risikomanagement, um das Risiko von Rechtsverletzungen zu vermeiden.

Das ist schnell gesagt, aber wie leicht lässt sich das in der operativen Softwareentwicklung umsetzen? 5

II. Problemstellung

OSS ist einfach zugänglich. Eine kurze Suche im Internet eröffnet mannigfaltige Möglichkeiten[2], um bestehende hervorragende Lösungen zu finden und zu nutzen. Für Softwareentwickler ist es heutzutage daher selbstverständlich, in ihren Projekten OSS zu nutzen. Beispielhaft hat die Nutzung von OSS folgende Vorteile: 6

- Herstellerunabhängigkeit: OSS ist typischerweise nicht von einer bestimmten Herstellerfirma abhängig, der Quelltext ist frei zugänglich[3];
- Schwarmintelligenz: In die Entwicklung / Weiterentwicklung bringen sich viele Entwickler ein, sodass sich OSS stetig weiterentwickelt;
- OSS ist einfach zugänglich, der Anwender kann sie einfach herunterladen, und kann sperrige Beschaffungsprozesse inklusive komplizierter Budgetdiskussionen vermeiden. Dies kann jedoch in Konflikt zu Unternehmensprozessen (z.B. IT-Security) stehen.

Mit den genannten Vorteilen und der etablierten Nutzung von OSS geht jedoch die Verpflichtung zur Einhaltung von Lizenzbedingungen einher, die der Rechtsinhaber der OSS für seine Komponente ausgewählt hat. Die Einhaltung dieser Lizenzbedingungen kann kompliziert werden, z.B., weil 7

- bisweilen eine hohe Anzahl von OSS-Lizenzbedingungen erfasst, eingeordnet, geprüft und administriert werden müssen;
- strenge und liberale Lizenzbedingungen in einem Paket verbaut sein können, woraus Inkompatibilitäten und somit Lizenzkonflikte resultieren können;
- die fortlaufenden Änderungen an den OSS-Pakete zu erneuten zeitaufwendigen Prüfungen führen können.

III. Markttrends

Die Lösung der oben beschriebenen Problemstellung beschäftigt viele in der Open Source Community und in Unternehmen. Viele bauen sich kleine Tools, Skripte und sonstige Lösungen, um die Komplexität in den Griff zu bekommen. Hilfestellungen und 8

[2] U.a. ist www.github.com eine beliebte Plattform mit professionellem Ökosystem für OSS Entwickler.
[3] Vgl. https://de.wikipedia.org/wiki/Open_Source.

Lösungsansätze entwickeln seit vielen Jahren schon Organisationen wie z.B. die Open Source Initiative[4] oder die Open Source Automation Development Lab eG[5].

Verschiedene Hersteller bieten auch Softwarelösungen[6] an, um beim Thema OSS-Lizenz-Compliance zu unterstützen. Der Anwender kann auf diese Weise Quelltexte untersuchen (scannen)[7], Regeln erstellen und hinterlegen, Hinweise auf Vulnerability-Risiken erhalten und mehr. Zumeist sind diese Lösungen aber eher für den Entwickler als für Rechtsanwälte gedacht und bieten oft nur wenig Unterstützung bei der fallbasierten Bewertung der Rechtslage. 9

Es gibt auch weitere Hilfestellungen, um Lizenztexte schneller und besser zu verstehen, die einer übersichtlichen Darstellung von Rechten und Pflichten dienen[8] oder die dabei helfen, geeignete Lizenztexte zu finden, wenn ein Softwareentwickler seine Softwarekomponente als OSS zur Verfügung stellen möchte[9]. 10

Allen Anbietern ist jedoch Eines gemeinsam: sie vermeiden es typischerweise rechtlich verbindliche Aussagen zu treffen, was aus Gründen der Haftung und den gesetzlichen Anforderungen des deutschen Berufsrechts für Rechtsanwälte jedoch leicht nachvollziehbar ist. 11

IV. Lösung: Open Source Management

Ein Lösungsweg kann sein, ein professionelles OSS-Management aufzubauen und einzuführen. Das Vorgehen bei vielen Unternehmen, die ihren Blickwinkel auf OSS lenken, zeigt typische Herangehensweisen. Einige dieser Ansätze sind: 12

- Die Nutzung von OSS wird kategorisch verboten - bis akzeptiert wird, dass es sich faktisch nicht sinnvoll verbieten lässt.
- Tabellen werden zur Bestandsaufnahme von OSS genutzt, die jedoch nie vollständig und daher im besten Fall als Momentaufnahme valide sind.
- Nutzer werden durch Fragenkataloge gequält, die in einem Massengeschäft kaum abgearbeitet werden können – und zwar weder von den Verwendern noch von den rechtlichen Prüfern / Freigebenden.

[4] Vgl. https://opensource.org.
[5] Vgl. https://www.osadl.org/.
[6] Beispielsweise https://fossa.com/features/.
[7] Vgl. https://www.blackducksoftware.com/; https://www.whitesourcesoftware.com/; https://fossid.com/.
[8] Vgl. https://tldrlegal.com/.
[9] Vgl. https://choosealicense.com/.

Fortschrittliche und moderne Ansätze erfassen die Fallprüfungen datenbankmäßig und bieten den Nutzern einen Fragenkatalog mit Antwortmöglichkeiten in Form von Auswahlmenüs, idealerweise in einem Webtool. 13

Ziel muss es sein, den Beteiligten einen soweit möglich einfachen Prozess anzubieten, um die OSS-Compliance sicherzustellen. Anforderungen dafür sind z.B.: 14

1. Automatisierung bei der Falleingabe

Für den Entwickler muss es so einfach wie möglich sein, seinen Anwendungsfall mit den prüfungsrelevanten Angaben im Tool zur Verfügung zu stellen. Dies kann idealerweise durch eine Einbindung in die jeweils bestehende Entwicklungs- sowie Continuous Integration / Continuous Delivery-Umgebung erreicht werden. 15

2. Anbindung unterschiedlichster Repositories

Zur Automatisierung der Falleingabe sollten prüfungsrelevante Informationen auch direkt aus den Repositories bezogen werden, in denen die Quellcodes und Binärartefakte liegen. 16

3. Mit der Zeit skalierbare automatisierte juristische Prüfung der eingegebenen Fälle

Juristische Prüfungsprozesse sollten so modelliert und mit Lizenzen und Fallbeschreibungen verknüpft werden, dass sie dadurch automatisiert ausgeführt werden können. Hier sollte zunächst mit einfachen Fallkonstellationen begonnen werden, um dann Schritt für Schritt immer komplexer werdende juristische Prüfungskonstellationen einzubeziehen, die dann automatisiert bearbeitet werden können. Das System sollte dabei für zukünftige Eingaben ständig mitlernen. 17

4. Anbindung von Source Code Scannern

Weiter sollten die jeweils besten verfügbaren Open Source Scanner flexibel angebunden werden können, um effizient Scans des Quellcodes durchzuführen. Hier macht es Sinn auf bestehende Lösungen aufzusetzen, und den Markt laufend zu beobachten, um die Stärken und Schwächen der verschiedenen Hersteller zu kennen. 18

5. Abgleich mit Vulnerability Datenbanken

Weiter sollten Datenbanken[10] angebunden werden, die über Sicherheitslücken bei Software informieren, damit Entwickler automatisch über sicherheitsrelevante Themen informiert werden und sehr frühzeitig Gegenmaßnahmen ergreifen können. Insbesondere bei stark inhomogener Verteilung von Programmiersprachen, angefangen von Webentwicklung mit JavaScript bis hin zur Entwicklung von embedded Code für Steuergeräte, ist es wichtig, eine zentrale Stelle für eine Vulnerabilityübersicht zu haben und zu nutzen. 19

V. Fazit

In einer idealen Welt könnte man allen Beteiligten viel Aufwand ersparen und Rechtssicherheit gewinnen, wenn die Nutzer lediglich wenige, anerkannte und weitverbreitete freie Lizenzbedingungen nutzen würden. So könnte OSS rechtssicher genutzt werden, ohne das ein Jurist jeden Einzelfall prüft. Eine zugegebenermaßen sehr idealisierte Vision könnte sein, dass von den Anwendern exakt zwei Lizenztypen genutzt werden: eine liberale[11] und eine strenge Lizenzbedingung[12], von denen je nach Intention und Strategie eine ausgewählt würde. Dies führte zu weniger Komplexität, Fehleranfälligkeit und Aufwand für alle Beteiligten und könnten dabei den eigentlichen OSS-Gedanken dank maximaler Transparenz unterstützen. 20

Die Realität wird freilich von Vielfalt geprägt bleiben - mit einem Trend zu etablierten, jedoch zahlreichen Lizenzmodellen. Und damit führt aus meiner Sicht kein Weg an professionellen Lizenzmanagementsystemen vorbei, die Technik und Recht integrieren, um die Menge unterschiedlichster Lizenzbedingungen zu prüfen und dadurch eine vertragsgerechte Umsetzung in der Praxis sicherstellen. 21

[10] z.B. https://vuldb.com/de/ oder https://nvd.nist.gov/.
[11] für OS-Experten: ohne Copyleft-Effekt wie z.B. MIT, https://opensource.org/licenses/MIT.
[12] für OS-Experten: mit strengem Copyleft-Effekt wie z.B. GNU General Puplic License (GPL V3), https://www.gnu.org/licenses/old-licenses/gpl-2.0.