Über Trojanische Pferde in Übersetzerimplementierungen

Wolfgang Goerigk

Institut für Informatik und Prakt. Mathematik, Christian-Albrechts-Universität zu Kiel
Preußerstraße 1-9, D-24105 Kiel, Germany, Email: wg@informatik.uni-kiel.de

Zusammenfassung

Bereits geraume Zeit ist bekannt (Ken Thompson 1984, L.M. Chrica, D.F. Martin 1986, J S. Moore 1988), daß Untersuchungen auf Quellcodeebene allein nicht ausreichen, um zuverlässiges und sicheres Funktionieren ablaufender Programme zu garantieren. Die harte Zusicherung gewünschter Eigenschaften ablaufender Maschinenprogramme ist in sicherheitskritischen Hardware/Softwaresystemen (Informatiksystemen) wichtig, aber gerade im Bereich der *security*, der Sicherheit z.B. in weltweiten Rechnernetzen, ist sie entscheidend. Hier jedoch hängen gängige Sicherheitstechniken allesamt davon ab und verlassen sich darauf, daß auf lokaler Maschine schließlich vertrauenswürdige, korrekte Implementierungen ablaufen und nötige Überprüfungen zuverlässig durchführen, z.B. virtuelle Maschinen, Übersetzer, just-in-time-Compiler oder Plug-in's.

Dieser Beitrag macht deutlich, daß es sich bei dem angedeuteten Risiko um ein reales handelt. Wir greifen Andeutungen von Ken Thompson (aus seiner Turing-Award-Lecture 1984) auf und machen klar, daß und wie es gelingt, in einer binären Übersetzerimplementierung ein Virus (ein Trojanisches Pferd) so zu verstecken, daß es sich, obwohl es im Quellcode nicht auftaucht, sogar beim Bootstrapping des Übersetzers mit sich selbst reproduziert. Ein solch absichtlich infizierter Übersetzer wird alle klassischen Verfahren der Validierung überstehen, den Bootstrap-Test, jedwede Quellcodeinspektion oder gar -verifikation, und auch eine Zertifizierung nach heutigem Stand der Technik.

Ken Thompson schloß damals mit dem Fazit: *„You can't trust code that you did not totally create yourself. (Especially code from companies that employ people like me.) No amount of source-level verification or scrutiny will protect you from using untrusted code.“* – einem Fazit, das leider bis heute uneingeschränkt richtig ist. Erst volle und rigorose Verifikation auch der binären Maschinenimplementierungen von Übersetzern und anderer kritischer Systemsoftwareteile kann diese Situation entscheidend verbessern. Solange dies nicht mit Sorgfalt durchgeführt wird, müssen wir weiterhin davon ausgehen, daß jedes von uns benutzte Programm bereits heute potentiell infiziert ist, irgendwann einmal mehr oder weniger katastrophale Auswirkungen zeigt.