xz-Sicherheitslücke: Desaster mit Ansage

Das war knapp: Um ein Haar wäre es Angreifern gelungen, eine im Betrieb kaum auffindbare Hintertür in OpenSSH einzuschleusen. Das hätte ihnen Zugriff auf rund 20 Millionen Linux- und Unix-Server im Internet gegeben, inklusive etlicher VMs in der Cloud, wie eine schnelle Shodan-Suche zeigt. Und fast jeder Linux-Server in Firmennetzen wäre ein leichtes Opfer gewesen, wenn man erst mal ins Netz eingedrungen ist.

Auf Seite 26 erklärt Jürgen Schmidt von heise Security die Details des in vielerlei Hinsicht raffinierten Angriffs. So richtete er sich nicht direkt gegen OpenSSH, eine Software, die wegen ihrer hohen Bedeutung für die Sicherheit von Linux-Servern unter sorgfältiger Beobachtung steht, sondern gegen eine harmlose Kompressionsbibliothek, die niemand für sonderlich sicherheitsrelevant gehalten hätte. Der Schadcode zur Ausnutzung der Hintertür kommt in Form eines SSH-Schlüssels und damit als legitimer Netzwerkverkehr auf kompromittierte Systeme – unentdeckbar für Netzwerkscanner. Die Angreifer haben einen Weg gefunden, wie aus absolut sauberem Quellcode im Repository ein Binary mit Hintertür entsteht. Sie hatten ausreichend langen Atem, um die Attacke über Jahre vorzubereiten, und waren skrupellos genug, Menschen so zu manipulieren, dass sie Durchgriff auf das xz-Projekt erhielten. Am Ende mussten eine ganze Menge glücklicher Zufälle zusammenkommen, damit Andres Freund die Hintertür entdeckte, wie er auf Mastodon beschreibt (siehe ix.de/zvbp).

Hohe technische Kompetenz und ausgeklügeltes Social Engineering bei Cyberangriffen sind nichts Neues, aber diese Supply-Chain-Attacke erreicht ein neues Level. „Weiter so“ ist jetzt keine Option mehr: Die aktuelle Praxis der Softwareentwicklung muss sich ändern. In Open-Source-Projekten wie in Unternehmen. Und nicht nur bei exponierten Anwendungen wie OpenSSH, sondern bei jeder Software, die auf einem potenziellen Angriffsziel installiert ist.

Die Mittel für eine sichere Softwareentwicklung sind schon lange bekannt: Mehr-Augen-Prinzip, rigorose Code-Reviews, umfassende Tests, gründliche Dokumentation, SBOMs, möglichst wenige externe Abhängigkeiten. Das ist mühsam, kostet Zeit und macht weniger Spaß, als drauflos zu programmieren. Aber dieser Angriff hat gezeigt, dass jeder Code, der auf einer vernetzten Maschine läuft, ein Sicherheitsrisiko ist und genauso gründlich geprüft werden muss wie Patches für OpenSSH oder den Linux-Kernel.

Das sind schlechte Nachrichten für die vielen Open-Source-Projekte, die als Hobby betrieben werden, für kleine Entwicklerbuden mit knappen Ressourcen – und auch für große Unternehmen, denen neue Features wichtiger sind als Sicherheit. Genau deswegen hat die CISA, die US-Behörde für Cybersicherheit, Microsoft gerade massiv kritisiert (siehe Seite 25).

Aber entweder ist uns eine Software die Mühe einer sicheren Entwicklung wert – oder sie taugt nicht mehr für den produktiven Einsatz. Nur weil wir diesmal mit viel Glück davongekommen sind, heißt das nicht, dass die nächste Attacke genauso glimpflich ausgeht. Oder dass nicht schon längst ähnlich ausgeklügelte Hintertüren in der Produktion angekommen sind. Es ist höchste Zeit zu handeln.

OLIVER DIEDRICH