Wie das Internet fast vergiftet wurde

Obskure Akteure, die wohl im Auftrag eines Staates handelten, wollten eine wichtige Komponente des weltweiten Internets unterwandern. Nur durch Glück scheiterte der Angriff. Das hat auch mit einem Kernprinzip der IT zu tun: Freiwilligenarbeit.

Von Basil Schöni, 23.04.2024

Die Anschlussbuchse als Tor zur Welt – und umgekehrt. Aus der Arbeit «Das Internet als Ort» von Heinrich Holtgreve/Ostkreuz

Es war Karfreitag und das Internet brannte.

Zumindest der Teil des Internets, in dem sich die IT-Sicherheits-Community rumtreibt. Der Grund dafür war ein Ereignis, das auf den ersten Blick vor allem für Sicherheitsforscherinnen interessant schien, auf den zweiten aber eher ein geopolitischer Schachzug eines staatlichen Geheimdienstes war: In einer sehr weit verbreiteten Softwarekomponente hatte ein Softwareingenieur eine schwerwiegende Sicherheitslücke entdeckt, die ein unbekannter Akteur absichtlich dort platziert hatte – eine sogenannte Hintertür.

Es war ein Angriff in der Breite. Wäre die Hintertür in der Softwarekomponente nicht frühzeitig entdeckt worden, wäre die Sicherheit eines bedeutenden Teils des Internets drastisch geschwächt gewesen – jedenfalls gegen Angriffe des Akteurs, der die Hintertür gebaut hat. Dieser Akteur – da ist sich die IT-Sicherheits-Community einig – war sehr wahrscheinlich ein Staat oder eine Gruppe mit staatlicher Unterstützung.

Dafür spricht einerseits, dass die Hintertür sehr raffiniert gebaut und gut versteckt war. Und andererseits, dass die ganze Aktion, an deren Ende die angreifbare Sicherheitslücke stand, über einen Zeitraum von drei Jahren durchgeführt wurde. Ein derart langer Atem, eine solche Expertise und ein so grosser Ressourceneinsatz – das passt nicht zu gewöhnlichen Cyberkriminellen. Aber es passt zu einem Geheimdienst, der die geopolitischen Interessen eines Staates verfolgt.

Um welchen Staat es sich dabei handeln könnte und was das konkrete Ziel der Operation war, lässt sich kaum beurteilen. Klarer ist aber, welche Möglichkeiten der Angreifer mit dieser Hintertür gehabt hätte.

Ein denkbares Szenario wäre etwa gewesen, dass der Angreifer die kritische Infrastruktur eines anderen Landes in der Breite lahmlegt. Also gleichzeitig in die IT-Infrastruktur einbricht von beispielsweise Energieunternehmen, Spitälern, Banken, Post, Verwaltung, Parlament, Grossverteilern, Medien, Flughäfen, Militär, Polizei et cetera. Und dann die Informatiksysteme all dieser Institutionen auf einen Schlag ausser Betrieb setzt.

Ein solcher Angriff wäre aufwendig gewesen. Die entdeckte Sicherheitslücke alleine hätte dafür nicht ausgereicht. Aber sie hätte das Fundament einer derartigen Operation werden können.

Öffentlich vs. geheim

Um zu verstehen, wie der Angreifer eine Hintertür in eine so zentrale Komponente des globalen Internets einbauen konnte, muss man wissen, wie ein grosser Teil der Software auf diesem Planeten entwickelt wird. Und was Software überhaupt ist.

Ein Computerprogramm ist im Kern eine Reihe von Anweisungen an den Computer – der sogenannte Programmcode. Diese Anweisungen sind in einer Programmiersprache geschrieben und lassen sich mit einem einfachen Textverarbeitungsprogramm anschauen und verändern. Aus technischen Gründen, die auszuführen den Rahmen dieses Artikels sprengen würde, ist es möglich, dass ein Endanwender ein Programm auf seinem Computer ausführt, ohne im Besitz des Programmcodes zu sein. Aber nur wer Zugriff auf den Code hat, kann verstehen, wie eine Software im Detail funktioniert, und sie verändern und weiterentwickeln.

Man kann nun zwei Arten von Software unterscheiden: jene, deren Programmcode öffentlich ist, und jene, deren Code geheim ist. Im ersten Fall spricht man von quelloffener oder auch Open-Source-Software. Die zweite Art bezeichnet man als proprietäre Software.

Proprietäre Software ist vor allem bei gewinnorientierten Firmen verbreitet. Sie betrachten den Code ihrer Programme als Geschäftsgeheimnisse, deren Bekanntwerden den wirtschaftlichen Erfolg der Firma gefährden würde. Auf dem Konsumentenmarkt ist proprietäre Software weitverbreitet. Wenn Sie etwa einen Apple- oder Windows-Computer benutzen oder an ihrem Arbeitsplatz beispielsweise ein Programm für die Spesenerfassung verwenden, nutzen Sie zu grossen Teilen proprietäre Software.

Anders sieht es bei der Infrastruktur des Internets aus. Das Internet ist vereinfacht gesagt ein Netzwerk aus sehr vielen Computern – Server genannt –, die jeweils eine spezifische Aufgabe erfüllen. Die Software, die auf diesen Servern läuft, ist zu grössten Teilen Open Source. Ein wichtiges Beispiel ist hier das quelloffene Betriebssystem Linux, das fast alle Server des Internets am Laufen hält.

Während proprietäre Software innerhalb einer Firma von deren Angestellten entwickelt wird, steht bei quelloffenen Programmen die freie Zusammenarbeit beliebiger Menschen, also auch Hobby-Programmiererinnen, im Zentrum. Wer ein Open-Source-Projekt lanciert, kann seinen Code auf verschiedenen Entwicklerplattformen hochladen. Dort können andere Menschen den Code herunterladen, Änderungen anbringen und diese Änderungen als Vorschläge wieder hochladen. Die Person, die das Projekt

verwaltet – Maintainer genannt – entscheidet dann, ob der Änderungsvorschlag übernommen oder verworfen wird. Der wirtschaftliche Aspekt ist dabei kaum relevant: Wer zu einem Open-Source-Projekt beiträgt, wird vom Maintainer in der Regel nicht bezahlt, sondern tut dies unentgeltlich.

Dieses Konzept der freien Zusammenarbeit hat sich als sehr effektiv erwiesen. Viele gute Ideen sind als kleines Hobbyprojekt einer Einzelperson gestartet und mit der Zeit zu komplexen Grossprojekten mit einer umfangreichen Community gewachsen. Auch das oben erwähnte Linux, das man mit nur leichter Übertreibung als das Betriebssystem des Internets bezeichnen könnte, hat als Freizeitprojekt eines einzelnen Informatikstudenten in Finnland begonnen.

Freie Kollaboration

Aus der riesigen Menge an Open-Source-Projekten, die über die Jahrzehnte lanciert wurden, hat sich mit der Zeit eine Sammlung an ausgeklügelten Programmen herauskristallisiert, die als optimierte Lösungen für bestimmte Probleme gelten und von vielen Entwicklerinnen auf der ganzen Welt verwendet werden. Eine Art geteilter, globaler Werkzeugkasten. Frei verfügbar für alle. Ohne Gebühren und ohne zentrale Kontrollinstanz.

Die Diversität in diesem Werkzeugkasten ist gross. Es gibt viele kleine Programme, die sehr spezifische Dinge tun, wie etwa eine Datei komprimieren (so wie es das bekannte Programm «Zip» tut). Solche Projekte werden häufig von wenigen oder gar einzelnen Personen verwaltet, oft auch unbezahlt in deren Freizeit.

Daneben gibt es auch mittlere und grosse Projekte wie den Browser Firefox oder das Betriebssystem Linux. Um diese Projekte haben sich meist grosse Communitys gebildet, die einen Teil der Wartung und Entwicklung finanzieren. So gibt es etwa die Mozilla Foundation, die sich unter anderem um Firefox kümmert, oder die Linux Foundation, deren Hauptaufgabe das Betriebssystem Linux ist. Solche Organisationen bezahlen Softwareentwicklerinnen, die an den jeweiligen Projekten arbeiten. Das betrifft aber nur einen Kern an Mitarbeitenden. Auch bei derartigen Grossprojekten tragen viele Freiwillige unentgeltlich zur Weiterentwicklung bei.

Linus Torvalds, der finnische Informatikstudent, der das Linux-Projekt gestartet hat, nannte kürzlich Zahlen hierzu: An jeder Version des Linux-Betriebssystems arbeiten etwa 1000 Personen mit. Rund die Hälfte davon steuert einmalig ein kleines Stück Code bei und ist ansonsten in keiner Weise an dem Projekt beteiligt. Im Jahr 2015 steckten in den Projekten der Linux Foundation geschätzt 41'192 Personenjahre an Entwicklungszeit. Ein grosser Teil davon dürfte unbezahlte Freiwilligenarbeit sein.

Auch gewinnorientierte Firmen tragen längst zum globalen Open-Source-Werkzeugkasten bei. Sie haben realisiert, dass die freie Zusammenarbeit über Länder- und Firmengrenzen hinweg ein Potenzial freisetzen kann, das innerhalb einer Firma kaum zu erreichen ist. Viele grosse Tech-Konzerne finanzieren darum nicht nur Non-Profit-Organisationen wie die Linux Foundation, sondern stellen auch den Code ihrer eigenen Projekte zur freien Verfügung ins Internet.

Das Internet würde in seiner heutigen Form nicht existieren, hätte sich die Idee der freien Zusammenarbeit, losgelöst von wirtschaftlichen Interessen, nicht durchgesetzt. Diese Idee, die man gleichzeitig als die erfolgreichste wie auch unsichtbarste Errungenschaft der Counter-Culture-Bewegungen

des späten 20. Jahrhunderts bezeichnen könnte, ist heute ein Fundament der globalen technischen Infrastruktur geworden.

Doch das Konzept der freien Zusammenarbeit basiert auf der Annahme, dass jede Mitarbeiterin gute Absichten hat. Dass niemand den geteilten Werkzeugkasten vergiften will.

Und genau das ist nun passiert.

Angriff auf die Lieferkette

Der Angriff, der am Karfreitag aufgedeckt wurde, war eine sogenannte Supply-Chain-Attacke, also ein Angriff auf die Lieferkette.

Software funktioniert in dieser Hinsicht ähnlich wie jede andere Industrie. Wer ein Auto bauen will, fertigt nicht jede Komponente des Autos selber an. Stattdessen bezieht man etwa den Motor, das Chassis und die Elektronik von jeweils separaten Zulieferern. Die Fabrik, wo der Motor hergestellt wird, bezieht wiederum die Einspritzvorrichtung von einem anderen Hersteller. Und dieser lässt etwa Schläuche und Dichtungen von einem weiteren Zulieferer produzieren. So ergibt sich eine Lieferkette, an deren Anfang kleine, wenig komplexe Einzelteile stehen. Und am Ende ein ganzes Auto.

Software ist gleich aufgebaut. Kleinere, weniger komplexe Programme werden in etwas grössere Programme eingebaut. Und diese wiederum in noch grössere Programme. Und so weiter. So ergibt sich eine Softwarelieferkette.

Die Softwarekomponente, die nun von dem mutmasslich staatlichen Akteur infiziert wurde, ist ein Fernwartungsprogramm mittlerer Komplexität namens OpenSSH, mit dem man sich über das Internet auf einem Server einloggen kann. Doch der eigentliche Angriff fand nicht in OpenSSH statt, sondern weiter hinten in der Lieferkette: Bei einem relativ simplen Programm im globalen Werkzeugkasten, das Dateien komprimieren kann. Der Angreifer schaffte es, diesem Projekt schädlichen Code hinzuzufügen. Und weil das Programm von der Fernwartungssoftware OpenSSH verwendet wird, landete der Schadcode schliesslich auch dort.

Wie ein manipulierter Einspritzschlauch, der erst Schaden anrichtet, wenn er in einen Motor eingebaut wird. Und dort auf Kommando den Motor zum Explodieren bringt.

Das kleine Programm, in dem der schädliche Code platziert wurde, heisst xz. Es wurde lange Zeit von einem einzelnen Entwickler in dessen Freizeit verwaltet. Sein Name ist Lasse Collin und er kommt aus Schweden. Collin war bis 2022 der alleinige Maintainer des Projektes und hatte somit die Kontrolle darüber, welche Änderungen in den Programmcode übernommen werden.

Doch Lasse Collin war überarbeitet. Das machte sich der Angreifer zunutze.

Der Account, mit dem der Angreifer im Februar dieses Jahres den schädlichen Code in das Programm xz einbaute, wurde drei Jahre zuvor, Anfang 2021 erstellt. Er läuft unter dem Namen Jia Tan. Im ersten Jahr hatten Jia Tans Beiträge noch nichts mit xz zu tun. Er schrieb Code für verschiedene andere Projekte.

Angriff auf den Menschen

Im Frühling 2022 begann dann der Social-Engineering-Angriff auf Lasse Collin. Social Engineering bezeichnet ein Vorgehen, mit dem eine Zielperson manipuliert und zu einem bestimmten Verhalten gebracht werden soll. Ziel des Social-Engineering-Angriffs war es, dass Lasse Collin Vertrauen zu Jia Tan aufbaut und ihn zu einem Maintainer des Projektes macht – ihm also die Berechtigung gibt, selbstständig Änderungen am Programmcode einzubauen.

Im April 2022 machte Jia Tan einen (harmlosen) Änderungsvorschlag für das Projekt xz. Kurz darauf traten zwei weitere Person erstmals auf, die sich Jigar Kumar und Dennis Ens nannten. Beide waren zuvor nirgends öffentlich in Erscheinung getreten. Zu ihren E-Mail-Adressen gibt es keinerlei Spuren im Internet. Sicherheitsforscher gehen davon aus, dass sie alle Teil der selben Operation wie Jia Tan waren.

Kumar und Ens begannen, Druck auf Lasse Collin auszuüben.

«Wird xz für Java noch gewartet?», fragte etwa Dennis Ens auf einer Mailingliste des xz-Projekts. «Ich habe vor einer Woche eine Frage gestellt und noch keine Antwort erhalten. Das Projekt wurde seit über einem Jahr nicht upgedatet.»

Diesen Ball nahm Jigar Kumar auf: «Es wird keinen Fortschritt geben, bis es einen neuen Maintainer gibt. (...) Hier Änderungsvorschläge zu machen, hat keinen Sinn im Moment. Der aktuelle Maintainer hat das Interesse verloren. Es ist traurig, das bei so einem Projekt zu sehen.»

Darauf reagierte Lasse Collin: «Ich habe das Interesse nicht verloren. Aber ich konnte mich nur sehr begrenzt um das Projekt kümmern, vor allem wegen längerfristigen Problemen mit meiner psychischen Gesundheit, aber auch wegen ein paar anderen Dingen.»

In seiner Antwort zeigte sich auch, dass Jia Tan langsam Lasse Collins Vertrauen gewann: «Ich habe in letzter Zeit ein bisschen mit Jia Tan an den xz-Werkzeugen gearbeitet. Vielleicht wird er in der Zukunft eine wichtigere Rolle spielen. Wir werden sehen.»

Und an Kumar gerichtet: «Es ist auch gut, im Kopf zu behalten, dass das ein unbezahltes Hobbyprojekt ist.»

Doch Jigar Kumar hielt den Druck hoch: «Mit deiner aktuellen Geschwindigkeit bezweifle ich, dass wir die nächste Version dieses Jahr noch sehen.- (...) Du ignorierst die vielen Änderungsvorschläge, die auf dieser Mailingliste vor sich hin rotten. Du erstickst dein Projekt. Warum bis zur nächsten Version warten, um den Maintainer zu wechseln? Warum hinauszögern, was dein Projekt braucht?»

Und gleich darauf Dennis Ens: «Es tut mir leid, dasss du Probleme mit deiner psychischen Gesundheit hast, aber es ist wichtig, die eigenen Grenzen zu kennen. Ich verstehe, dass das ein Hobbyprojekt ist für alle Beteiligten, aber die Community verlangt mehr.»

Gemeinsame Verantwortung

Die Druckversuche schienen erfolgreich zu sein. Kurz nach diesen Mails tauchen erste Änderungen im Code von xz auf, deren Autor Jia Tan ist. Of-

fenbar beginnt Lasse Collin, immer mehr auf die Mithilfe von Tan abzustützen.

Spätestens im Januar 2023 scheint Jia Tan dann selbstständig Änderungen einbauen zu können. Im darauffolgenden Jahr vollzieht Jia Tan einige Vorbereitungsschritte. Im Februar 2024 schliesslich baut er die Hintertür in das xz-Projekt ein.

Damit war der grösste Teil des Angriffs durchgeführt. Doch damit tatsächliche Server in der freien Wildbahn verwundbar sind, musste die neuste Version von xz zuerst noch auf diesen landen. Im Frühjahr 2024 war das nur noch eine Frage der Zeit. Mit den Wochen und Monaten hätten die Projekte, die weiter vorne in der Lieferkette sind, die neue xz-Version bei sich integriert. So hätte sich die Hintertür langsam, aber sicher auf Millionen von Servern im ganzen Internet verteilt. Und der mutmasslich staatliche Akteur hätte den Angriff auf seine eigentlichen Ziele durchführen können.

Doch ein Softwareentwickler namens Andres Freund machte ihnen einen Strich durch die Rechnung. Per Zufall hatte er eine Testumgebung bei sich am Laufen, die die infizierte Version schon installiert hatte. Und weil er sein System aus einem anderen Grund gerade sehr gut anschaute, bemerkte er, dass die infizierte Fernwartungssoftware langsamer war als gewohnt.

Er suchte den Grund für diese Verlangsamung. Und entdeckte den schädlichen Code. Am Karfreitag machte er seinen Fund publik. Damit konnte die Hintertür beseitigt werden, noch bevor sie grosse Verbreitung in der freien Wildbahn fand.

Der globale Werkzeugkasten entging also knapp einem Unterwanderungsversuch von ziemlich grossem Ausmass.

Nun ist es nicht so, dass solche Angriffe nur bei quelloffener Software passieren können. 2020 wurde etwa die Firma Solar Winds, die proprietäre Software herstellt, Opfer einer Supply-Chain-Attacke. Und auch grössere Projekte können in dieser Weise angegriffen werden, wie der Entdecker der Hintertür, Andres Freund, anmerkte.

Trotzdem zeigt sich an xz eine spezifische Verwundbarkeit von kleinen, quelloffenen Projekten, auf denen grössere und wichtigere Komponenten aufbauen. Wenn wenige oder gar einzelne Personen in Freiwilligenarbeit für Projekte zuständig sind, die zum Fundament der weltweiten IT-Infrastruktur gehören, können bösartige Akteure relativ leicht den geteilten Werkzeugkasten vergiften.

Eine einfache Lösung für dieses Problem gibt es nicht. Verschiedene Massnahmen könnten aber das Risiko minimieren, dass solche Angriffe unentdeckt bleiben.

Am Anfang müsste dafür die Erkenntnis stehen, dass der globale Werkzeugkasten auch in der Verantwortung des Gemeinwesens steht. Einzelne Maintainer von wichtigen Softwarekomponenten müssen unterstützt werden. Diese Unterstützung kann einerseits finanzieller Natur sein. Projekte wie der Sovereign Tech Fund übernehmen diese Aufgabe mit Unterstützung des deutschen Staates. Ein ähnliches Projekt könnte auch die offizielle Schweiz ins Leben rufen oder unterstützen.

Andererseits könnten öffentlich geförderte Teststellen eine wichtige Rolle spielen. In der Schweiz gibt es etwa das Nationale Testinstitut für Cybersicherheit. Solche Institutionen brauchen aber viele Ressourcen, denn das

Überprüfen von Software auf Sicherheitslücken ist eine teure und nie endende Aufgabe.

Schliesslich stehen aber nicht nur der Staat und das Gemeinwesen in der Verantwortung. Viele grosse Firmen profitieren ganz konkret und finanziell von quelloffener Software. Auch diese Firmen müssten in Open-Source-Projekte investieren, um deren Integrität besser zu schützen. Entweder freiwillig, oder indem sie dazu verpflichtet werden.

Denn am Ende bedient sich die ganze Welt an diesem geteilten Werkzeugkasten, aus dem die IT-Infrastruktur des Planeten gebaut ist. Diesen gegen Unterwanderung zu schützen, dürfte mit den zunehmenden geopolitischen Spannungen noch wichtiger werden.