Gunter Bitz

Integrität der Software Lieferkette

Supply Chain Integrity

Unter der Überschrift „Supply Chain Integrity“ definiert „SAFECode“¹ eine mögliche Herangehensweise an das Thema Integrität von Softwareprodukten. Darunter ist ein Konzept zur Absicherung gegen bewusst herbeigeführte aber unerwünschte Änderungen von Software zu verstehen. Insbesondere die Einkaufsorganisationen sicherheitssensibler Bereiche der öffentlichen Hand wie z.B. Militär und Nachrichtendienste stellen verstärkt die Frage des Nachweises der Manipulationsfreiheit von Software.

Fragestellung

In einer Welt, die sich vermehrt abhängig von Informationstechnologie macht, fokussieren sich dem Trend folgend auch Angriffe gegen dieselbe. Während früher die größten Betriebs- oder Staatsgeheimnisse in Form von Akten in einem Safe aufbewahrt wurden, liegen die Informationen heute binär codiert auf einem Speichermedium und sind über dedizierte Terminals abrufbar. Die Herausforderung von heute liegt darin, die Informationssysteme gegen Angriffe zu schützen.

Die meisten Angriffe gegen IT Systeme basieren darauf, dass ein Angreifer eine bereits bekannte aber nicht behobene oder eine unbekannte Schwäche in einem System nutzt. Bei unbekannten Schwachstellen hat der Angreifer ein kleineres Entdeckungsrisiko und sehr große Erfolgsaussichten, da die Verteidiger sich noch nicht gegen die Schwachstelle schützen können.

Daher hat sich ein recht lukrativer Untergrund-Markt für bisher unbekannte Schwachstellen, sogenannte „Zero-Day Exploits“ entwickelt.

Statt als Angreifer mühevoll nach Schwachstellen zu suchen oder diese zu kaufen, was mit dem Manko der Nicht-Exklusivität behaftet ist, da die Betreiber der Schwachstellenbörse ihre Produkte oft mehrfach verkaufen, könnte ein Angriff auch bewusst eine Schwachstelle bereits im Fertigungsprozess der Software einschleusen. Diese wäre nur einem sehr kleinen Personenkreis bekannt. Sie kann sehr leicht ausgenutzt werden, sobald die Software beim Anwender in Betrieb ist.

Man unterscheidet zwei mögliche Szenarien:

1. Den zielgerichteten Angriff, der genau für einen Endanwender vorbereitet wird. Typischerweise findet sich dies bei Auftragsarbeiten, wo ein Softwareprodukt quasi als Einzelstück nach individuellen Wünschen gefertigt wird.
2. Vom nicht zielgerichteten Angriff spricht man, wenn eine Schwachstelle in eine Software eingebaut wird, die bei vielen Endanwendern zum Einsatz kommt, und in allen Kopien der Software vorhanden ist. Dies ist vor allem bei Software, die für den Massenmarkt vorgesehen ist und eine hohe Verbreitung besitzt, gegeben.

SAFECode

Außer den Forderungen des Nachweises manipulationsfreier Software von Einkaufsorganisationen der öffentlichen Hand gab es bisher keine Veröffentlichungen der Software Industrie zu diesem Thema.

Unter der Schirmherrschaft von „SAFECode“ arbeiten Vertreter der großen Softwarefirmen wie z.B. Microsoft, SAP und Adobe an den Fragestellungen rund um das Thema „Supply Chain Integrity“.

Dabei wird die Sicherheit der gesamten Software Lieferkette betrachtet. Ein Soft-

wareprodukt kann aus verschiedenen Komponenten bestehen, die alle von verschiedenen Lieferanten hergestellt oder bezogen werden. Je nach Unternehmensstrategie kann auch Open Source Software in einem Produkt integriert sein. Für alle Komponenten müssen alle als relevant definierten Sicherheitsanforderungen erfüllt sein.

Kontrollen

SAFECode hat mit Erscheinen dieses DuD-Heftes gerade eine Beschreibung der wichtigen Kontrollen zur Sicherung der Integrität von Softwareprodukten publiziert. Neben den technischen Kontrollen der sicheren Software Entwicklung wie Testen, Code Review, Paar Programmierung, Design Review, usw. sind auch organisatorische Maßnahmen unverzichtbar. Dazu gehören:

- ♦ Verträge mit Lieferanten
- ♦ Haftung der Lieferanten für Fehler
- ♦ Ausgangstests (bei Auslieferung)
- ♦ Abnahmetests (beim Eingang)
- ♦ Backgroundchecks des Personals (so weit es rechtliche Rahmenbedingungen erlauben)
- ♦ Sicherheit des Source Code Repositoriums und der Entwicklungsumgebung
- ♦ Protokollierung jeder Änderung am Source Code (beim Check-in)
- ♦ Nicht-Abstreitbarkeit der Identität des Entwicklers (Eindeutige Zuordnung einer Änderung zu einer Person)

Darüber hinaus lassen sich vorsätzlich eingebaute Schwachstellen zu einem gewissen Grad auch durch automatische Source Code Tests erkennen. Der Beitrag von Wysopal/Eng/Shields in diesem Heft widmet sich diesem Thema.

¹ <http://www.safecode.org>