Secure Boot und Secure Remote Update mit TPM 2.0:

Sicher von Anfang an

(Bild: Orla | Shutterstock)

Secure Boot und Secure Remote Update sind zentrale Komponenten für jedes aktuelle Embedded-Gerät. Im Rahmen eines internen Projektes wurden bei Mixed Mode die Grundlagen für eine plattformunabhängige Lösung mit Unterstützung für TPM-2.0-Module geschaffen.

Von Kai Che und Markus Wamser

Meldungen über Sicherheitslücken in IoT-Geräten sind leider alltäglich. Gerade günstige vernetzte Kameras stehen dabei oft im Fokus, da ein unberechtigter Zugriff auf den Videostream die Folgen einer Sicherheitslücke deutlich vor Augen führt. Für den Endanwender weniger deutlich, aber vom Effekt schwerwiegender ist der Missbrauch von IoT-Geräten als Teil von Bot-Netzen. Dabei wurden einzelne Angriffe beobachtet, bei denen Hunderttausende von kleinen Devices große Internetseiten und sogar Teile der globalen Internet-Infrastruktur in die Knie zwangen. Das wirft zwei Problemstellungen auf:

- Wie lässt sich verhindern, dass unautorisierte Software (z. B. Malware) auf einem Gerät ausgeführt wird und sich dort einnistet?
- Wie können Updates sicher über das Internet auf IoT-Geräte ausgebracht werden?

Die Lösung für das erste Problem ist ein „Secure Boot"-Prozess. Hierbei wird jeder Schritt des herkömmlichen Bootvorgangs um eine Sicherheitsprüfung des *nachfolgenden* Schrittes erweitert (**Bild 1**).

Sicherheit ab dem „Power On"

Für einen typischen Systemstart bedeutet dies, dass nach dem Einschalten der initiale ROM-Code des Prozessors die Integrität des nachfolgenden First-Stage-Loaders gegen eine fest in den Prozessor eingebrannte Prüfsumme oder einen öffentlichen Schlüssel prüft. Nur im Falle einer erfolgreichen Prüfung wird der First-Stage-Loader ausgeführt. Dieser prüft wiederum den eigentlichen Bootloader. Der eigentliche Bootloader kann als minimales Betriebssystem dann komplexere Bedingungen prüfen, flexibel mit mehreren Signaturarten umgehen und auch auf ein Trusted Platform Modul (TPM) als Hardware-Vertrauensanker einbinden.
Ab diesem Punkt ist auch eine vollständige Abstrahierung der Hardware möglich, sodass der im nachfolgenden vorgestellte Ansatz vollkommen generisch implementiert werden konnte.

Ist ein System in einem vertrauenswürdigen Zustand, so kann dieser mit Hilfe eines TPM-Moduls gegenüber dem eigenen Backend attestiert werden. Das wiederum erlaubt sicherzustellen, dass Updates an das korrekte System ausgeliefert werden und gleichzeitig, dass nur vorgesehene Upgrade-Pfade (und eventuell Downgrade-Pfade) beschritten werden.

Hardware-Abstraktion erhöht Wiederverwendbarkeit

Aus der täglichen Praxis ergab sich der Wunsch nach einem plattformunabhängigen und wiederverwendbaren Basiskonzept für Secure Boot und Secure Update. Ergänzend sollte die Integration von TPM-2.0-Funktionen in den Bootloader „Barebox" geprüft werden. Letzteres war unter Verwendung von „wolfTPM" und eigenem Code möglich, wodurch Barebox hinsichtlich TPM-2.0-Unterstützung mit seinem großen Bruder „U-Boot" gleichzieht. Die bessere Hardware-Abstraktion durch Barebox erlaubte es dabei, eine komplett platt-

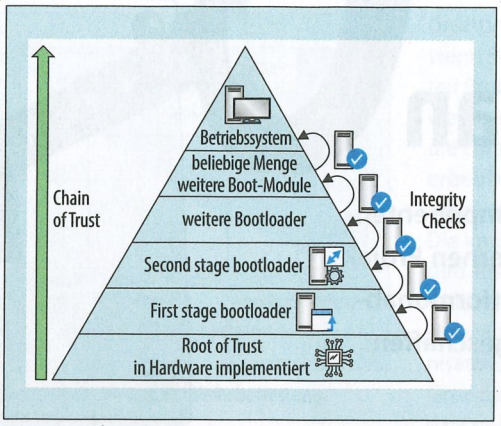

Bild 1. Ablauf eines Secure-Boot-Vorgangs, ausgehend von der „Root of Trust" unten über die beiden Bootloader im ersten und zweiten Schritt. Darüber können individuell noch zusätzliche Bootloader sowie andere bootfähige Module eingeschaltet werden. Der Integritycheck wird von Schritt zu Schritt ausgeführt – und nur dann, wenn alle diese Schritte positiv durchgeführt worden sind, ist die Chain of Trust etabliert. Andernfalls wird der Bootvorgang abgebrochen. Seiteneinstiege sind damit praktisch nicht möglich. (Bilder: Mised Mode)

formunabhängige Lösung zu entwickeln. Während die Lösung initial für einen Raspberry Pi (Broadcom SoC) entwickelt wurde, war für die Nutzung auf einem Beaglebone-Board (TI Sitara SoC) lediglich ein Neucompilieren der Software notwendig.

Exemplarisch wurde ein Software/Produkt-Lebenszyklus in drei Stufen angenommen (**Bild 2**): Die erste Stufe (Provisioning Phase) umfasst die Produktion des IoT-Gerätes durch einen Original Equipment Manufacturer. Üblicherweise besorgt dieser OEM auch die initiale Provisionierung des kryptographischen Vertrauensankers (TPM) mit Schlüsselmaterial des Produktherstellers.

Im konkreten Beispiel sind dies ein öffentlicher Schlüssel (OEM Public Key) zur Überprüfung von Signaturen, für den das TPM als integerer Speicher dient, sowie ein geheimer Schlüssel für symmetrische Verschlüsselung und ein geheimer Schlüssel für Signaturen. Der symmetrische Schlüssel ist stellvertretend für einen Schlüssel zur Freischaltung von Lizenzen zu sehen. Der asymmetrische Schlüssel bietet im Beispiel nahezu denselben Funktionsumfang wie die vom TPM bereitgestellten „Endorsement Keys" und damit unterzeichnete Signaturschlüssel und soll im Wesentlichen die Möglichkeit zur sanften Migration von Schlüsselmaterial und -konzepten in eine sichere Umgebung darstellen.

Zweite Stufe: Die auszuführende Software muss nun vor der Auslieferung mit dem passenden Schlüssel (OEM Private Key) signiert werden. Im Gegensatz zur ersten Stufe, die einmalig durchgeführt wird, kann diese zweite Stufe (Software Distribution Phase) beliebig oft wiederholt werden.

Dritte Stufe: Besonderer Clou der Secure-Boot-Lösung ist die Nutzung von TPM-2.0-Policies. Damit lässt sich in der dritten Stufe (Boot Phase) die Überprüfung des Betriebsystems mit der Zugriffsautorisierung für die privaten Schlüssel kombinieren. Konkret wird bei der Speicherung der geheimen Schlüssel im TPM als Zugriffsregel (Policy) hinterlegt, dass eine weitere Policy, die mit dem privaten Schlüssel des Produktherstellers signiert ist, vorgelegt werden muss. Diese Policy muss dann erfüllt werden, um die gespeicherten Geheimnisse verwenden zu können.

Im Kontext von Secure Boot fordert diese Policy lediglich, dass eine bestimmte kryptographische Prüfsumme für das Betriebsystem ermittelt und in ein spezielles Platform Configuration Register (PCR) des TPM geschrieben wurde. Diese Anforderung kann aber beliebig erweitert werden, etwa um die Präsenz von Hardware-Dongles, Kenntnis von Passwörtern oder Protokolle der vorhergehenen Bootschritte. Die erforderlichen Daten und Signaturen können dabei im Backend mit einem beliebigen TPM-2.0-Modul erzeugt werden.

Updates mit Autorisierung

Wurde der Secure Boot erfolgreich durchgeführt, kann das IoT-Gerät aus dem bereits erwähnten PCR und optional weiteren Daten (Lizenzschlüssel, angeschlossene Geräte) vom TPM einen signierten Fingerabdruck erstellen lassen und dem Backend einen definierten Systemzustand nachweisen (Remote Attestation, **Bild 3**). Nach positiver Prüfung stellt das Backend ein verschlüsseltes Update oder eine ergänzende Softwarebibliothek bereit, das vom IoT-Device mit Hilfe des vom

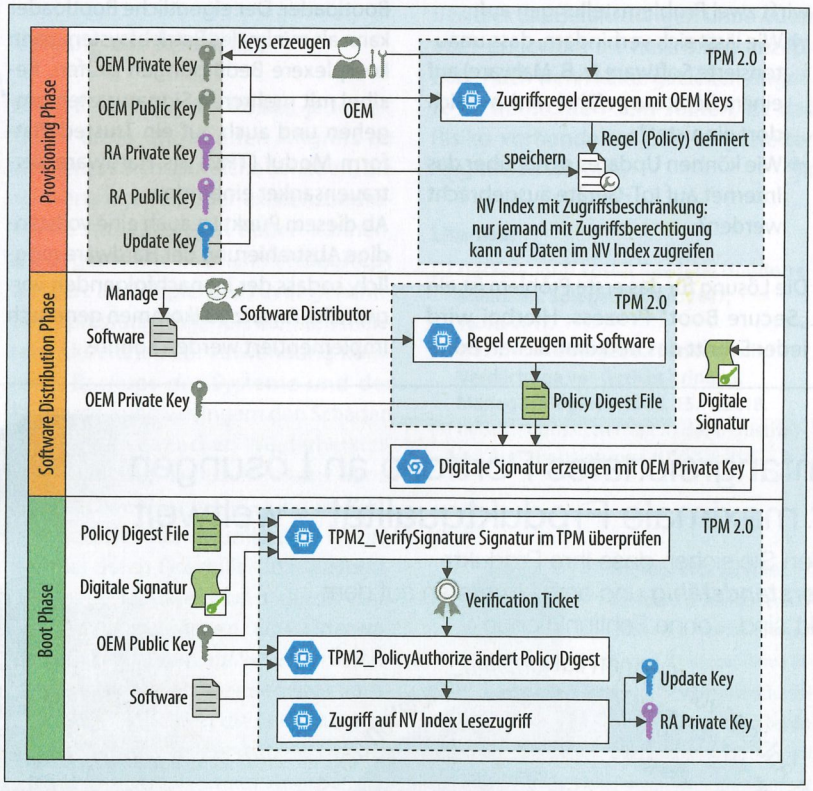

Bild 2. Die drei zentralen Stufen im Lebenszyklus eines Secure-Boot-Konzeptes: Zuoberst die Provisioning Phase, die nur einmal ausgeführt wird. In der Mitte die Software Distribution Phase, die beliebig oft durchlaufen wird, und zuunterst die Boot Phase, die die TPM-2.0-Policies einsetzt.

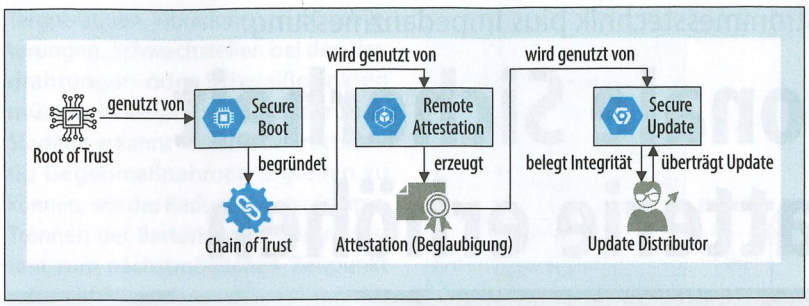

Bild 3. Zusammenhang zwischen Secure Boot, Remote Attestation und Secure Remote Update: Das Secure Boot erzeugt ausgehend von der „Root of Trust" die „Chain of Trust", die wiederum Ausgangspunkt für die „Remote Attestation" ist. Erst wenn diese vorhanden ist, kann der Update Distributor das Secure Update ausführen.

TPM freigeschalteten symmetrischen Schlüssels wieder entschlüsselt werden kann. Dieses Update enthält aktualisierte Signaturen und Secure-Boot-Policies, die fortan bei jedem Systemstart geprüft werden und den Schutz der Software-Integrität sowie die Einhaltung von Zugriffsbeschränkungen sicherstellen.

Secure Boot und Secure Update sind natürlich nicht auf IoT-Geräte beschränkt. Zwar sind diese mit ihrer permanenten Vernetzung besonders gefährdet und erfordern eine vordringliche Betrachtung, gleichzeitig bieten sie aber auch die Chance, sich mit diesen Sicherheitskonzepten vertraut zu machen und Erfahrungen zu sammeln. Diese Erfahrungen können dann gewinnbringend auch in safetykritischen Systemen wie im Bereich der Medizintechnik oder dem Anlagenbau eingesetzt werden, um die Anwendersicherheit zu erhöhen und auch ein Haftungsrisiko des Herstellers durch versteckte Manipulation zu reduzieren.

Markus Wamser
ist seit 2017 als Expert Embedded Security, Entwickler und Referent bei Mixed Mode in Gräfelfing tätig. Zuvor hat er am Lehstuhl für Sicherheit in der Informationstechnik der Technischen Universität München unterrichtet und promoviert. Seine beiden Diplome in Mathematik und Informatik erwarb er an der Universität Augsburg.

Kai Che
war Masterand bei Mixed Mode und am Lehrstuhl für Sicherheit in der Informationstechnik der TU München, wo er auch seinen Bachelor of Science Elektrotechnik und Informationstechnik erwarb.

Zu guter Letzt bietet ein vertrauenswürdiges System mit TPM 2.0 auch eine ideale Basis für ein Lizenz- und Rechtemanagement von klassischen Softwarelizenzen bis zu Pay-per-Use und Manufacturing-as-a-Service im Industrie 4.0-Umfeld. *jk*