# IoT-Software-Updates unterhalb der Edge

**Eigentlich sollte man davon ausgehen, dass erkannte und veröffentlichte Sicherheitslücken in IoT-Produkten umgehend durch Updates geschlossen werden. Mangels technischer Möglichkeiten ist das aber nur relativ selten der Fall. Für industrielle Anwendungen ist diese Handlungsoption völlig inakzeptabel.**

Auf Grund der zahlreichen Software-Schwachstellen in den Millionen (vielleicht sogar Milliarden) IoT-Geräten, die inzwischen weltweit im Einsatz sind, müssen wir uns wohl damit abfinden, dass unzählige Sicherheitslücken (beispielsweise in den TCP/IP-Stacks [1]) auf Grund fehlender technischer Voraussetzungen überhaupt nicht durch Updates beseitigt werden. Zumindest sind am Verhalten der Anbieter, Anwender und für die Cybersicherheit zuständigen Behörden keine proaktiven Reaktionen bzw. Aktionen erkennbar, die diese fragwürde Sicherheitslage kurz- oder mittelfristig verändern würden.

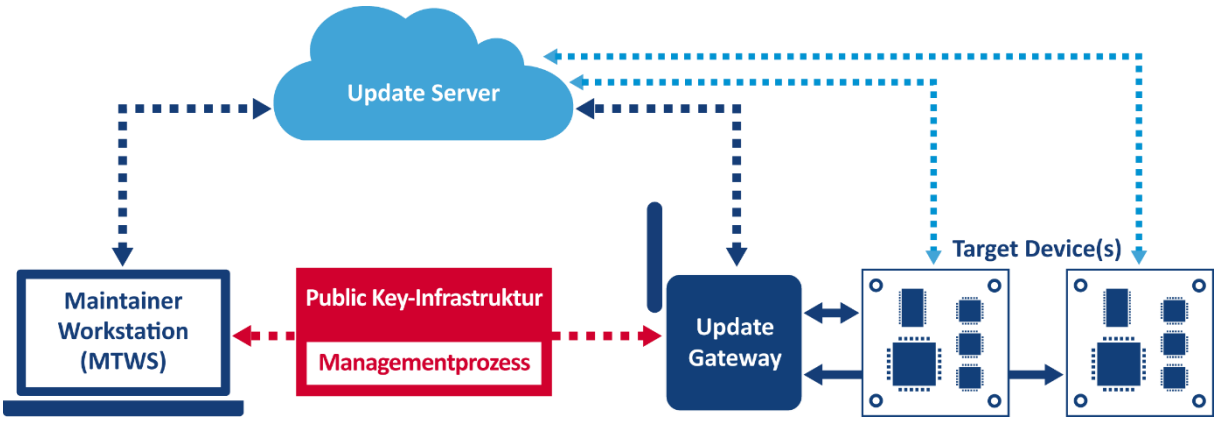


**Abbildung 1:** Ein automatisches Software-Updatesystem für IoT Devices besteht im Prinzip aus drei Bausteinen: Die Maintainer Workstation als Update-Quelle, das Target Device als Ziel des Updates und dem Update Server als funktionales Bindeglied. Die erforderliche IT-Sicherheit wird durch eine Public Key Infrastructure (PKI) gewährleistet. Lässt sich das Target Device aus technischen Gründen nicht direkt mit dem Server verbinden, wird ein Update Gateway als Agent dazwischengeschaltet.

Die Ursachen für die unzähligen Schwachstellen in den IoT-Produkten sind außerordentlich vielfältig. Ein sehr wichtiger Grund sind sicherlich fehlende Normen zur Cybersecurity für Hersteller, aber auch für die Betreiber von IoT-Geräten. Im Vergleich zur elektrischen Produktsicherheit – hier existieren hochentwickelte und überwiegend praxistaugliche Industrienormen – sind die Normierungsgremien beim Thema IoT-Sicherheit bisher sehr inaktiv. Einige wenige Ausnahmen aus dem vergangenen Jahr sind das ETSI-Grundlagendokument EN 303 645 [2] und das Standard-Dokument NISTIR 8259 des US-amerikanischen *National Institute of Standard and Technology*. Es adressiert primär die IoT-Device-Hersteller und trägt den Titel *Foundational Cybersecurity Activities for IoT Device Manufacturers* [3]. Dieses Dokument enthält zwei hilfreiche Kapitel. Das eine befasst sich mit den Aspekten vor der Auslieferung eines IoT Device (Pre-Market-Phase). Das andere geht auf die Phase nach der Produktauslieferung ein (Post-Market-Phase). Hier findet man unter *4.2.4 Software Updates* z. B. durchaus praxistaugliche Anregungen, mit denen sich IoT-Update-Probleme vermeiden lassen. Da die Automatisierung identische Probleme hat, ist das NISTIR 8259-Dokument auch für die Smart Factory geeignet.

## Lokale Schnittstellen vorhanden

In der Automatisierungstechnik haben wir inzwischen für die gesamte Operation Technology (OT)-Ebene mit Hilfe verschiedener Schnittstellen und Protokolle einen sehr hohen Vernetzungsgrad erreicht. An der „Edge“ zur IT sind sogar IP-basierte Koppler (die sogenannten Edge-Gateways) zu

finden. Auch eine unternehmensweite Gesamtvernetzung ist daher mittlerweile durchgängig existent. Sie reicht vielfach vom letzten IO-Link-Sensorhub in einer Fertigungszelle über alle Steuerungen mehr oder weniger direkt bis in die IT-Ebene und somit indirekt auf jeden Fall bis ins Internet. Insofern müsste man eigentlich davon ausgehen, dass auch für alle vernetzten OT-Baugruppen ein durchgängiges „Device Management“, inklusive der Möglichkeit bei Bedarf Software-Updates einzuspielen, existiert. Die Frage, die sich jeder OT-Technikverantwortliche eigentlich stellen müsste, ist: Wie gehen mein IO-Link-Sensoriklieferant, mein Steuerungsanbieter, der Frequenzumrichter- und Edge-Gateway-Hersteller sowie all die anderen Device-Hersteller eigentlich mit Software-Updates um, damit wir in unserer Smart Factory nicht die gleichen Probleme wie in der IoT-Welt bekommen (siehe Tabelle mit Fragenliste)? Die Antworten sind häufig nicht zufriedenstellend.

Sehr viele Baugruppen in der OT-Ebene sind nur über sehr spezielle (hardwarenahe) Schnittstellen und Protokolle durch einen Spezialisten vor Ort Update-fähig. Zur Schnittstellenvielfalt gehören z. B. USB, JTAG, UART (TTL, RS-232, RS-485), CAN oder andere herstellerspezifische In-System-Programming-(ISP) Varianten. In einigen Fällen sind diese Schnittstellen nicht frei zugänglich. Sie befinden sich im inneren der Baugruppe (z. B. JTAG). Teilweise ist für den Software-Update-Vorgang eine besondere Handlungssequenz erforderlich (z. B. erst die Taste X drücken, dann innerhalb von Y Sekunden die PC-Software Z starten usw.).

Auf Grund dieser Details ist es in der Praxis daher nicht ohne weiteres möglich, OT-Baugruppen mit einem zentralen Update-Server zu verbinden und bei Bedarf voll- oder halbautomatisch mit einer neuen Softwareversion auszustatten, wie wir das beispielsweise von unseren Smartphones, Tablets, IP-TVs und vernetzten PKWs kennen. Unmöglich ist es aber auch nicht (siehe Abbildung 1).

## Software-Update-Systeme als Retrofit

Um die notwendige Cybersecurity in einer vernetzten Smart Factory zu gewährleisten, wären natürlich durchgängige Software-Update-Prozesse mit einem möglichst hohen Automatisierungsgrad für alle OT-Geräte wünschenswert. Dafür muss nicht die gesamte Gerätebasis ausgetauscht werden, da sich moderne Software-Update-Lösungen inzwischen auch per Retrofit nachrüsten lassen. Baugruppen-hersteller und Smart-Factory-Betreiber können solche Prozesse aber nur gemeinsam entwickeln. Geeignete Branchenverbände, die so etwas über Arbeitsgruppen auf den Weg bringen könnten, gibt es im deutschsprachigen Raum. Die entsprechende Software-Update-Fragenliste der NISTIR 8259 eignet sich hier durchaus als roter Faden für die ersten Schritte.

Spätestens, wenn digitale Zwillinge und KI-basierte Edge-Anwendungen mit vortrainierten Machine-Learning-Modellen auf breiter Front Einzug in die Automatisierungslandschaften halten, geht es nicht mehr ohne automatisierte Konfigurations- und Software-Updates – siehe die sogenannten DevOps in der IT-Welt (DevOp = Team-basierter Lösungsansatz aus Development und Operation. Gemeint sind damit die kontinuierlichen Weiterentwicklungen und fortlaufende Integration der neuen Software-komponenten in den operativen Betrieb).

## Quellen:


[1] https://www.heise.de/news/Amnesia-33-ein-Ripple20-Deja-vu-im-Open-Source-Gewand-4982063.html

[2] https://www.etsi.org/deliver/etsi_en/303600_303699/303645/02.01.01_60/en_303645v020101p.pdf

[3] https://nvlpubs.nist.gov/nistpubs/ir/2020/NIST.IR.8259.pdf

## Tabelle:

| |
|---|
| Werden vom Hersteller Software-Updates zur Verfügung gestellt? Wenn ja, wann werden sie veröffentlicht? Wenn die Nutzer (Anwender) beispielsweise wissen, ob Aktualisierungen nach einem festgelegten Zeitplan oder sporadisch bereitgestellt werden, können sie das Update der einzelnen Devices besser planen. |
| Aus welchen Gründen werden Updates veröffentlicht? Beispiele hierfür sind Bugfixes für fehlerhafte Software und die Korrektur einer zuvor unbekannten Sicherheitsschwachstelle in einem Kommunikationsprotokoll oder einer Systembibliothek. |
| Wie werden Updates zur Verfügung gestellt oder geliefert? Gibt es hinsichtlich der Update-Verfügbarkeit gezielte Benachrichtigungen vom Anbieter an die Nutzer? Beispielsweise können Nutzer den Einsatz eines Software-Updates besser planen, wenn sie wissen, dass ab einem bestimmten Termin ein Update in einem Portal zur Verfügung steht. Nutzer profitieren von einer Benachrichtigung auch dann, wenn der gesamte Update-Prozess vollständig automatisch abläuft und der Nutzer selbst keinerlei manuelle Aktivitäten durchführen muss. |
| Welches Unternehmen bzw. welche Instanz ist für die Durchführung von Aktualisierungen verantwortlich (beispielsweise Nutzer, Hersteller, Dritter)? Kann der Nutzer selbst bestimmen, wer verantwortlich ist (z. B. Updates werden immer automatisch vom Hersteller durchgeführt)? Für den Anwender ist es hilfreich, wenn auch ein Dritter (Third Party als Serviceanbieter) Updates zur Verfügung stellen kann und die Verantwortlichkeiten den Erfordernissen entsprechend geregelt werden können. |
| Wie können Nutzer die Updates selbst überprüfen und authentifizieren? Beispiele sind der kryptografische Hash-Vergleich, die Validierung der Codesignatur und das Vertrauen in die vom Hersteller bereitgestellte Software, die automatisch die Überprüfung und Authentifizierung von Updates durchführt. |
| Welche Informationen sollten bei jedem einzelnen Update mitgeteilt werden? Beispiele sind die Art des Updates (z. B. Korrekturen von Fehlern, geänderte oder neue Funktionen) und mögliche Auswirkungen einer Update-Installation auf die vorhandenen Konfigurationseinstellungen eines Anwenders. |

**Tabelle 1:** Hinsichtlich der Software-Updates für IoT Devices sollten Anwender und Anbieter unbedingt einige grundsätzliche Fragen klären. Die Software-Update-Fragenliste im Dokument NISTIR 8259 des US-amerikanischen National Institute of Standard and Technology eignet sich dabei als Orientierungshilfe.

**© SSV Software Systems GmbH 2021 / Alle Rechte vorbehalten / 0.4 / 11.02.2021**