

Entwicklung sicherer Systeme

# Zuverlässig?

Im Englischen unterscheidet man »Safety« und »Security«, ins Deutsche übersetzt sich beides mit »Sicherheit«. Selbstverständlich sollten Embedded Systeme sowohl »safe« als auch »secure« sein – doch wie stellt man das am besten an?

Timothy Stapko

Es gibt keine dummen Fragen, so heißt es. Dieses Axiom hält jedoch weder den Adressaten der Frage noch den Hilfesuchenden davon ab, manches als überflüssig zu betrachten. Letzteres ist das schlimmere Problem, denn wer es nicht wagt, Fragen zu stellen, bekommt mit Sicherheit auch keine Antwort. Zum Thema »Sicherheit von Embedded Systemen« erläutern wir im Folgenden zehn Punkte, die Entwickler wissen sollten, aber im Allgemeinen nicht zu fragen wagen.

## Offensichtliches

Vermutlich die erste Frage, die sich ein Ingenieur stellt, wenn er ein Sicherheitskonzept für ein Embedded System realisieren soll, ist »Welche(s) Security Package(s) brauche ich?«. Unglücklicherweise gibt es unzählige Security-Packages, und ein Entwickler, der sich zum ersten Mal mit dem Thema Sicherheit befasst, betrachtet dieses Thema möglicherweise nur unter dem Aspekt von Kryptografie und Virenschutz. Kryptografie steigert zwar die Sicherheit, und Virenscanner tragen aus technischer Sicht zu einem besseren Sicherheitsgefühl bei, aber beide

sind nicht immer das, was man wirklich braucht. Praktischerweise sind Security-Funktionen heute in vielen Applikationen bereits integriert, und die Pakete, die der Ingenieur zusätzlich benötigt, werden in der Regel durch das diktiert, was an den Schnittstellen der Applikation hängen soll. Ist die Applikation Web-basiert, greift ein Entwickler vermutlich auf SSL/TLS (Secure Sockets Layer bzw. Transport Layer Security) zurück. Andere Anwendungen setzen auf Techniken wie IPSEC (Internet Protocol Security) oder CCMP (WPA2-WLAN-Verschlüsselung). Die Bedeu-



tung dieser Akronyme zu kennen ist weniger wichtig als zu wissen, welche Protokolle wie unterstützt werden. Und erst wenn das geklärt ist, kann die Einkaufstour beginnen. Im Anschluss folgt dann »Wie implementiere ich ein Security-Package für meine Applikation?«. Je nach Anwendung kann die Lösung ganz einfach

sein, beispielsweise der Einsatz einer Programmdatei mit aktivierten Sicherheitsfunktionen. Dient beispielsweise Embedded-Linux oder WindowsCE als Betriebssystem, dann gibt es vermutlich Programme, die schon eine ganze Menge Security-Funktionen mitbringen, etwa einen SSH-Client (Secure Shell). Steht kein Binärcode zur Verfügung, so kann es gut sein, dass Quellcode erhältlich ist. Open-Source-Pakete wie »OpenSSL« und »OpenSSH« zählen zu den besten Implementierungen unter den verfügbaren Security-Protokollen; und das Beste daran ist, sie sind kostenlos. Wer sich nicht zu den absoluten Kryptografieexperten zählen kann, sollte es unbedingt vermeiden, ein eigenes Sicherheitsprotokoll zu realisieren – andernfalls stehen die Chancen gut, dass das System verletzlich ist. Ist schließlich die Applikation realisiert und alle Security-Protokolle eingebunden, liegt die Frage nahe: »Wie sicher ist mein System?«. Hier hilft es, wie ein Krimineller zu denken oder jemanden zu beauftragen, der das kann (die Beschäf-

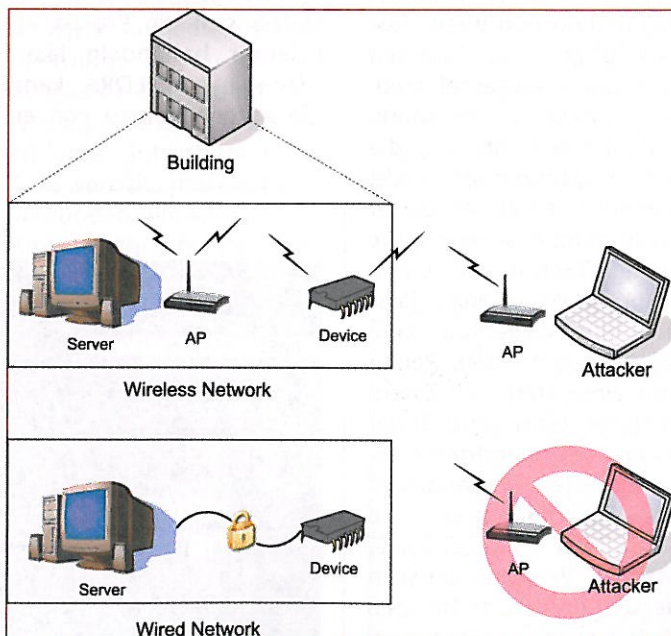


Bild 1: Vergleich der Sicherheit von drahtgebundenen und drahtlosen Systemen

tigung wirklicher Krimineller wird aber hier nicht empfohlen). Grundsätzlich verwerten Angreifer alles, solange der Gewinn aussichtsreich genug erscheint. Der Sinn der Security-Funktionen liegt darin, die Kosten einer erfolgreichen Attacke höher zu gestalten als den Profit, den der Angreifer daraus ziehen könnte. Moderne Kryptografie basiert auf mathematischen Verfahren, für die moderne Computerhardware Tausende von Jahren benötigen würde, um sie auszuhebeln. Unglücklicherweise kann jedes Protokoll und jeder Algorithmus eine nicht bekannte Schwachstelle aufweisen, über die sie sich schließlich aushebeln lässt. Außerdem steigt die Leistungsfähigkeit der Hardware rapide, was diese »Tausende von Jahren« sehr schnell relativiert.

### Weniger Offensichtliches

Selten gestellt, doch extrem wichtig: »In welchem Umfeld soll die Applikation eingesetzt werden?«. Will der Entwickler den Umfang der zu ergreifenden Sicherheitsmaßnahmen bestimmen, so ist der Einsatzort ebenso wichtig wie alle anderen Faktoren. Embedded-Anwendungen stehen oft an Orten, an denen ein Angreifer uneingeschränkter Zugriff auf die Hardware hat. Hat ein Angreifer physikalisch Zugriff auf das System, so versagen softwarebasierte Sicherheitsmechanismen, und auch Hardwaremechanismen schneiden nicht viel besser ab. Wer Security-Funktionen auf höchstem Niveau implementiert, der stelle bitte auch sicher, dass der physische Schutz des Systems mindestens dem in der Applikation angewandten Sicherheitsniveau entspricht. Dazu passt auch eine Analyse der Frage: »Wer sind die potenziellen Angreifer?«. Um eine Liste potenzieller Angreifer zu generieren, gilt es zu überlegen, wer von einem

Einbruch in das System profitieren könnte. Diese Liste könnte Konkurrenten, Terroristen, Geheimdienste oder auch einfach nur gelangweilte Teenager umfassen. Die Leute, die offensichtlich am meisten von einem Angriff auf ein System profitieren würden, sind normalerweise auch die, die hinter einem Angriff stecken. Aber der Angreifer ist möglicherweise gar nicht an dem interessiert, was den Entwickler am meisten beunruhigt, und das führt zu der nächsten Frage: »Welche Informationen sind für etwaige Angreifer am wertvollsten?«. Kleiner Tipp: Möglicherweise ist es gar nicht das, was man denken würde. Dies ist eine wirklich trickreiche Frage, weil es einem Angreifer möglicherweise gar nicht auf irgendwelche Daten ankommt. Vielleicht reicht es ihm schon aus, die Applikation zum Absturz zu bringen, so wie beispielsweise ein frustrierter Stromkunde seinen elektronisch vernetzten Stromzähler manipuliert, um kostenlos Strom zu zapfen. Vielleicht ist der Angreifer auch nur daran interessiert, die Hardware unter Kontrolle zu bekommen. Es kommt schon vor, dass PCs infiltriert und so in Zombies verwandelt werden, welche die Internet-Präsenz eines anderen Opfers mit hohem Verkehr bombardieren oder ganze Systeme mit E-Mails überfluten und so außer Betrieb setzen – und die Angreifer verdienen möglicherweise daran. Aktuell verklärt der »Conficker«-Wurm in immer neuen Varianten unzählige Rechner auf der ganzen Welt. Immer mehr Geräte sind miteinander vernetzt, und so ist es nur wahrscheinlich, dass jemand diese als Hardwareressource ansieht, die es auszubeuten gilt. Auch die Anbindung entscheidet: »Wie unterscheidet sich die Sicherheit von drahtlosen und drahtgebundenen Systemen?«. Drahtlosnetze bieten im Vergleich mit drahtgebundenen Netzwerken eine

zusätzliche Ebene der Verwundbarkeit – das physikalische Übertragungsmedium. In einem drahtgebundenen Netzwerk kommt eine Leitung als Übertragungsmedium zum Einsatz. Ein Abhören der Kommunikation setzt ein physikalisches Anzapfen der Leitung oder zumindest physikalische Nähe voraus. Leitungen können durch gesicherte Gebäude, unterirdisch, auf Telegrafmasten oder durch Beton verlaufen, was ein Anzapfen von vornherein erschwert. Ein Funknetz verwendet den freien Raum als Übertragungsmedium. Sobald ein Drahtlosgerät seine Daten in alle Richtungen abstrahlt, benötigt ein Angreifer lediglich eine Antenne, um sich Zugang zu verschaffen. Aus diesem Grunde warten die meisten Funkprotokolle bereits mit integrierten Verschlüsselungsmechanismen auf (Bild 1).

### Per Definition sicher

»Inwieweit beeinflussen die gewählte Hard- und Software die Sicherheit?«. Manche Systeme sind schon standardmäßig sicherer, was auf höherwertige Software oder spezielle Sicherheitsmaßnahmen zurückzuführen ist. Es ist immer hilfreich, sich mit anderen auszutauschen, die ähnliche Systeme bereits realisiert haben, und so herauszufinden versuchen, welche Applikationen auf diesem Gebiet schon eingesetzt wurden. Auch für die Hardware gibt es Security-Lösungen, die sich schon bewährt haben. Unter diesem Gesichtspunkt ist es natürlich hilfreich zu wissen, »welche Angriffe auf die benutzte Sicherheitstechnik bekannt sind«. Wer ein hohes Maß an Sicherheit gewährleisten will, muss sicherheitstechnisch auf dem Laufenden sein. Jeden Tag arbeiten Tausende von Hackern und Forschern daran, die vorhandenen Sicherheitsmechanismen auszuhebeln. Die Erfolgreichen werden berühmt (oder berüch-

tigt), und das alleine ist schon genügend Motivation, sich neue Angriffe auf existierende Systeme auszudenken. Der aktuelle Stand der eingesetzten Verfahren sollte bekannt sein, der Entwickler sollte alles über bekannte Angriffe wissen und auch alle verfügbaren Nachrichten auf diesem Gebiet verfolgen, um sicher zu sein, dass keine neuen Angriffsflächen entdeckt worden sind. Die letzte Frage beleuchtet Sicherheit unter einem anderen Aspekt: »Braucht mein System wirklich das Höchstmaß an Sicherheit?«. Man kann sehr leicht der Philosophie verfallen, dass man absolut das Beste, Robusteste und Leistungsfähigste in punkto Sicherheit aufbieten muss, aber in Wahrheit ist das vielleicht der absolute Overkill. Man denke einfach nur an das Beispiel mit dem Stromzähler: Macht es wirklich etwas aus, wenn jemand die vom Stromzähler übertragenen Daten anzapft und abhört? Bei den alten Zählern muss auch nur jemand einen Blick auf den Zählerstand werfen, und deshalb spielt es wohl auch bei der vernetzten Version weniger eine Rolle, ob die besten Verschlüsselungsmechanismen zum Einsatz kommen oder nicht. Hier kommt es in erster Linie darauf an, den Verbrauch korrekt aufzuzeichnen und die Daten unmanipuliert an den Versorger zu liefern. Es gibt preisgünstigere Methoden, um dieses Ergebnis auch ohne ausgefeilte Security-Funktionen zu erreichen. Bei der Analyse jener Funktionen gilt es darüber nachzudenken, wie viel Sicherheit wirklich nötig ist. Bei den Hardwarekosten lassen sich Geld und Entwicklungszeit sparen, indem der Entwickler unnötige Funktionen von vornherein von der Liste streicht. (mc)

#### Timothy Stapko

ist Lead Software Engineer und Projektmanager bei **Digi International**  
Telefon 02 31/97 47 0  
[www.digi.de](http://www.digi.de)