Unser Gerät kommt ins Internet - Herausforderungen in der Praxis

Willi Flühmann
Senior Software Engineer / Architect
5. Juni 2018
Bestehendes Gerät
Bestehendes Gerät mit mehr Konnektivität
Oder sind die wahren Verhältnisse eher so?
Frage:
Eine Internetanbindung ist gefordert. Wie sieht das aus Sicht eines Embedded-Systems aus?
Vielfältige Auswirkungen

**Schnelligkeit**
- Protokolle und Services ändern sich laufend
- Firmware-Update unverzichtbar

**Kompatibilität**
- Alte und neue Geräte gemischt
- Update auch für ältere Geräte

**Globale Datenaggregation**
- Zeitabweichung zwischen Geräten
- Verschiedene Zeitzonen

**Komplexität**
- Schnellere Hardware, mehr Software
- Ständig erreichbar: Höherer Stromverbrauch

**Autonomie**
- Was funktioniert noch ohne Internet?

**Fernzugriff**
- Niemand vor Ort, um einzuzugreifen

**Neue Gefährdungen der Stabilität**
- Mehr Fehler- und Störquellen
- Hacking, DoS-Attacken
Agenda

1. Plattform
2. Kommunikation
3. Cloud
4. Firmware-Update
5. Datenschutz und Security
Anforderungen an die Plattform

- Einfache Binärformate
- USB
- RS-232
- Feldbus
- Bluetooth
- Ethernet
- TCP/IP
- WLAN
- CoAP
- MQTT
- Web-UI
- TLS
- REST
- JSON
- XML

Bild: Wikipedia / Konstantin Lanzet, CC BY 3.0
## Anforderungen an die Plattform

<table>
<thead>
<tr>
<th>Prozessor</th>
<th>Flash/RAM</th>
<th>Betriebssystem</th>
<th>Machbare Netzwerkverbindung</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cortex M oder 8/16 Bit</td>
<td>bis xx kB</td>
<td>Ohne</td>
<td>(nur mit Zusatzprozessor)</td>
</tr>
<tr>
<td>Cortex M</td>
<td>ab xxx kB</td>
<td>RTOS</td>
<td>Einfache TCP/IP-Verbindungen + TLS, mehrheitlich proprietär</td>
</tr>
<tr>
<td>Cortex A</td>
<td>ab xx MB</td>
<td>Embedded-Linux (Yocto)</td>
<td>Verschiedene Standardprotokolle durch fertige Software-Komponenten</td>
</tr>
</tbody>
</table>
## Anforderungen an die Plattform

<table>
<thead>
<tr>
<th>Prozessor</th>
<th>Flash/RAM</th>
<th>Betriebssystem</th>
<th>Machbare Netzwerkverbindung</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cortex M oder 8/16 Bit</td>
<td>bis xx kB</td>
<td>Ohne</td>
<td>(nur mit Zusatzprozessor)</td>
</tr>
<tr>
<td>Cortex M</td>
<td>ab xxx kB</td>
<td>RTOS</td>
<td>Einfache TCP/IP-Verbindungen + TLS, mehrheitlich proprietär</td>
</tr>
<tr>
<td>Cortex A</td>
<td>ab xx MB</td>
<td>Embedded-Linux (Yocto)</td>
<td>Verschiedene Standardprotokolle durch fertile Software-Komponenten</td>
</tr>
</tbody>
</table>

Wechsel auf ein Embedded-Linux drängt sich auf
Wechsel von RTOS auf Embedded-Linux

- Ressourcenbedarf / Softwareumfang um Größenordnungen höher
- Riesiges Ökosystem mit Software-Komponenten:
  - Komponenten übernehmen statt selbst bauen
  - Konfiguration und Scripting
- Open Source:
  - GPL-Lizenz beachten
  - Achtung: Secure Boot inkompatibel mit GPL v3
- Längere Aufstartzeit, Echtzeitverhalten schwieriger
- Know-How: Andere Entwickler nötig?
Anspruchsvollere Software-Entwicklung

• Größeres Projekt
• Interdisziplinäres Team
  – Embedded
  – Cloud
  – Web
  – Mobile
• Entwicklungsprozess wird wichtiger
Agenda

1. Plattform
2. Kommunikation
3. Cloud
4. Firmware-Update
5. Datenschutz und Security
Art der Internetverbindung

• Bandbreite, Latenz, Unterbrüche
• Wieviele Stunden pro Tag verfügbar?
• NAT, Firewall

• Teilen sich viele Geräte eine langsame Leitung?

Priorisierung, Koordination, gestaffelter Zugriff?

z.B. bei Firmware-Update
Direkter Zugriff aufs Gerät

Arten von Clients:
- Desktop
- Mobile App
- Web-UI

Web-UI:
Erzeugt ev. viel Aufwand im Gerät (z.B. Übersetzungen, Zahlenformate).
Direkter Zugriff aufs Gerät

Viele Geräte sind nicht direkt aus dem Internet erreichbar.
Indirekter Zugriff aufs Gerät (1)

Optional:
Umwandlung von Low-Level (z.B. MQTT, CoAP)
In High-Level (z.B. REST).
Indirekter Zugriff aufs Gerät (2)

Umfangreichere Datenverarbeitung und -speicherung
Kommunikation zwischen Geräten

Auch hierzu kann eine Netzwerkverbindung verwendet werden.
Kommunikation: Weitere Aspekte

Gleichzeitiger Zugriff von mehreren Clients

- Race Conditions
- Daten zwischen allen Clients synchronisieren
- Individuelle Sitzungen
- Auditing (wer hat was gemacht)

Umgang mit Verbindungsunterbrüchen

- Vermeidung zu vieler aufgestauter Befehle
- Daten nachfordern / erneut senden
Agenda

1. Plattform
2. Kommunikation
3. Cloud
4. Firmware-Update
5. Datenschutz und Security
Cloud: Skalierbarkeit bei der Datenverarbeitung

Ziel:
Spezialisierte Behandlung und Speicherung der Daten anhand ihrer Charakteristik.

Nicht für alle Daten braucht es atomare Transaktionen und jederzeitige Konsistenz.
# Cloud: Skalierbarkeit bei der Datenverarbeitung

<table>
<thead>
<tr>
<th>Welche Daten</th>
<th>Charakteristik</th>
<th>Speicherung in</th>
</tr>
</thead>
</table>
| Periodische Messwerte | Kontinuierlicher Datenstrom vom Gerät  
Zugriff darauf ausschnittweise, zusammengefasst in Intervallen | Zeitreihen-Datenbank (z.B. TimescaleDB)       |
| Konfiguration     | Synchronisation der Konfiguration zum Gerät («Device Twin»)                  | egal (z.B. irgendeine SQL-Datenbank)         |
| Befehle           | Fire-and-Forget                                                              | -                                            |
| Gerätestatus      | Nur letzter Status wichtig                                                   | Flüchtiger Cache (z.B. Redis)               |
| Ereignisse        | Logging von Ereignissen                                                      | Zeitlich limitierter Cache (z.B. Redis)     |
Agenda

1. Plattform
2. Kommunikation
3. Cloud
4. Firmware-Update
5. Datenschutz und Security
Firmware-Update: Partitionierung des Flashspeichers

Kernel
Device-Tree
Root-Dateisystem

Firmware-Update im vollständig laufenden System:
Benötigt **zwei Kopien**

Bootloader
Bootloader Variablen
Software-Kopie 1
Software-Kopie 2
Daten
Firmware-Update: Einfluss auf viele Ebenen

- **Cloud**
  - Updates bereitstellen und steuern
  - Automatisch vs. manuell

- **Updater-Applikation**
  - Download und Anwenden des Updates

- **Bootloader**
  - Entscheidung, welche Kopie gebootet werden soll

- **Flash-Partitionierung**
  - Platz für zwei Software-Kopien
Firmware-Update: Fertige Komponenten

- **SWUpdate** von DENX Engineering: https://sbabic.github.io/swupdate/
- **RAUC**: https://www.rauc.io/
- **Mender**: https://mender.io/

Die meisten Komponenten stellen nur eine Teillösung für den Firmware-Update dar.

Zudem unterscheidet sich die Menge der unterstützten Bootloader.
Agenda

1. Plattform
2. Kommunikation
3. Cloud
4. Firmware-Update
5. Datenschutz und Security
Datenschutz

Technische Massnahmen:
• Pseudonymisierung
• Anonymisierung
• Daten verlassen Gerät nicht
• Keine unnötige Daten speichern

Hier lässt sich schon vieles machen.

Organisatorische Massnahmen:
• Aufklärung und Einverständnis des Benutzers
• Löschen / Sperren / Herausgeben nach Aufforderung
• Daten schützen
• usw.
Verbindungen lassen sich nur streckenweise schützen (keine End-zu-End-Verschlüsselung).
## Security: Initialisierung von Schlüsseln

<table>
<thead>
<tr>
<th>Variante</th>
<th>Wie</th>
<th>Anforderung</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symmetrisch</td>
<td>Extern generierter Schlüssel auf Gerät und Server verteilen</td>
<td>In abhörsicherer Umgebung verteilt</td>
</tr>
<tr>
<td>Asymmetrisch</td>
<td>Extern generierter Schlüssel und Zertifikat auf Gerät (privat) und Server (öffentlich) verteilen</td>
<td>In abhörsicherer Umgebung verteilt</td>
</tr>
<tr>
<td>Asymmetrisch mit Fingerprint</td>
<td>Gerät generiert Schlüssel, Server merkt sich Fingerprint</td>
<td>Öffentlicher Schlüssel erstmalig über vertrauenswürdigen Kanal</td>
</tr>
<tr>
<td>Asymmetrisch mit Zertifikat</td>
<td>Gerät generiert Schlüssel und lässt sich von Server daraus ein Zertifikat erstellen</td>
<td>Öffentlicher Schlüssel erstmalig über vertrauenswürdigen Kanal</td>
</tr>
</tbody>
</table>
Cloud: Mehr oder weniger Security?

**Pro:**
- Verwendung geprüfter/bewährter Komponenten und Protokolle
- Gut bewachte Rechenzentren
- Professionelle Wartung
- Schnelle Reaktion bei Bekanntwerden von Lücken

**Kontra:**
- Aufgeblähte Standard-Software bietet mehr Angriffsfläche als massgeschneiderte (proprietäre) Software
- Standard-Software ist ein lohnendes Ziel für opportunistische, großflächige Attacken
- Security und Datenschutz „nur“ vertraglich zugesichert, keine eigene Kontrolle darüber
  - Basiert letztlich nur auf Reputation
«Gemeinsam treiben wir weltweit Innovationen voran.»

- Willi Flühmann
- willi.fluehmann@noser.com