

# Herausforderungen für den IP-Schutz in eingebetteten Systemen

Jamshid Shokrollahi

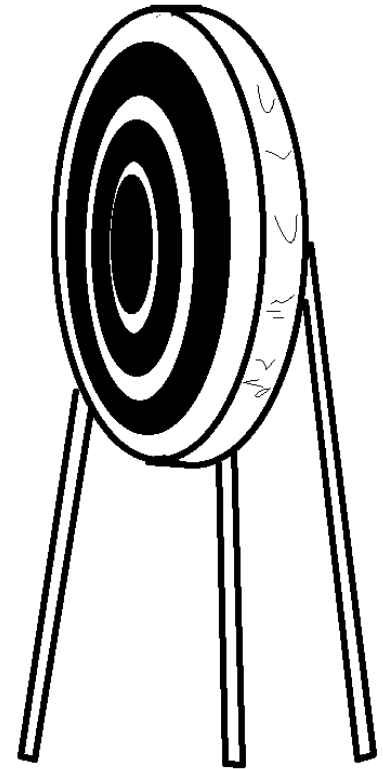
BITKOM Forum Embedded Security  
Nürnberg  
15. October 2009



**BOSCH**

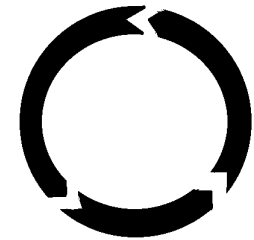
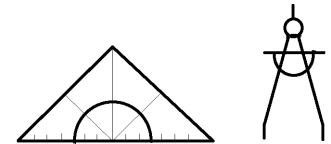
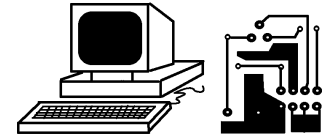
## Ziel

- Geheimhaltung von (Teilen der) Software als wertvolles Ressource



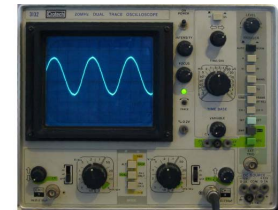
## IP-Schutz Herausforderungen

- Sichere Umsetzung der Mechanismen
- Während des Entwurfs
  - Richtige Auswahl der Mechanismen aus umfangreichen Katalogen von Lösungen mit unterschiedlichen Kosten-Nutzen Kriterien
- **Prozess**
  - **Beurteilung der Entscheidungen**



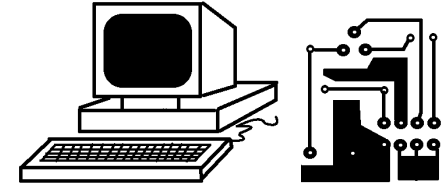
## Angreifer

- Einfach (Layman):  
Kennt das System und die Schwachstellen des Betriebssystems. Teilweise kann er seine Software auf das System laden
- Mittel:  
Kann mit Oszilloskopen die Signale auf der Platine lesen
- Stark:  
Hat Zugang zu aufwändigen Geräten wie FIB und kann geheime Information aus dem Chip lesen

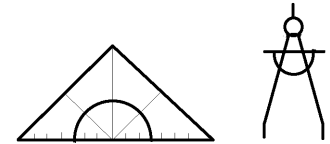


## Methoden für die Umsetzung

- Software-basiert
  - Disassembly verhindern
  - Debug erkennen
  - Tamper resistant software
  
- Hardware-basiert
  - Multichip Methoden
  - Platinenentwurf
  - Schutz innerhalb des Chips
  
- Betriebssystem
  - MPU und MMU
  - Virtualisierung



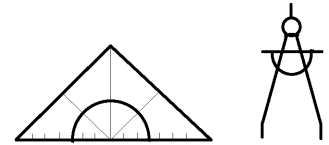
## Die Evolution der Methoden für den Schutz gegen Software-Piraterie



- Best, 1980:  
Kryptographische Mikroprozessoren gegen Software-Piraterie
- Kent 1980:  
Die Lieferung der Software in verschlüsselter Form
- White and Liam, 1990:  
Abyss sicherer Koprozessor: *partitioned computations*

Verschlüsselung der Software

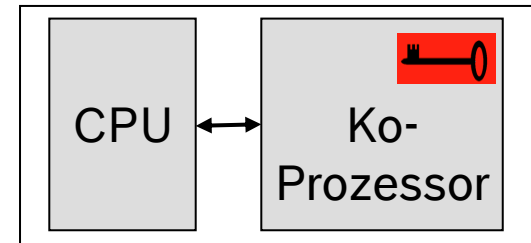
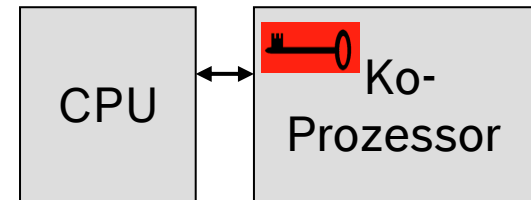
## Verschlüsselung der Software



- Schutz der verschlüsselten Software
- **Schutz der kryptographischen Schlüssel**
- Schutz der entschlüsselten Software während des Betriebs
- Kann mit Schutzmechanismen des Betriebssystems kombiniert werden

## Schutz der Schlüssel

- Im internen Speicher
- In einem externen kryptographischen Koprozessor
- In einem kryptographischen Koprozessor innerhalb desselben Chips



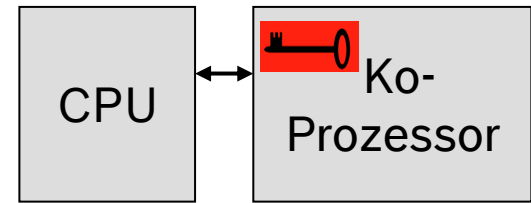
## Schlüssel im internen Speicher



- Falls ein externer Code geladen wird, können die Schlüssel ausgelesen werden
- Debug ist ein Risiko
- Die Schwachstellen in der ursprünglichen Software können ausgenutzt werden
  
- Das Risiko für die Schwachstellen in der Software kann mit Hilfe von CPUs mit Memory Protection Units und Betriebssystem reduziert werden:
  - Erhöhte Kosten
  - Reduzierte Performanz

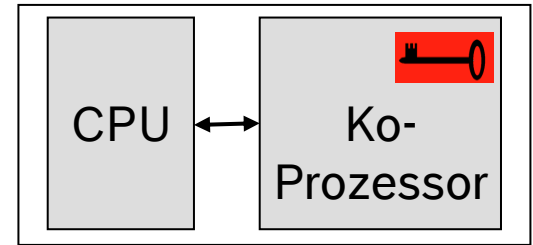
## Externer Koprozessor

(z.B. Trusted Platform Module)



- TPM speichert den Schlüssel und gibt ihn an die CPU nur weiter, wenn ein bestimmter (vertrauenswürdiger) Software-Stack auf der CPU läuft
- Kombination mit den Schutzmechanismen für Multichip-Systeme
- Große Software-Stacks haben meistens mehr Schwachstellen, die ausgenutzt werden können
- Die Methode kann mit Virtualisierung kombiniert werden. Security-Teil wird auf einem Mikrokern und in einem getrennten Compartment laufen.
  - **Erhöhte Kosten**
  - **Reduzierte Performanz**

## Interner Koprozessor



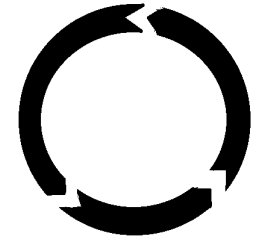
- Der kryptographische Koprozessor wird auf demselben Chip eingebaut wie die CPU
- Hat die gleichen Aufgaben wie der interne Koprozessor
- Angriff auf die Verbindung zwischen CPU und Koprozessor braucht aufwändige Geräte
- **Der Entwurf eines neuen Chips ist teuer**

## Szenarien

Interner Speicher	<ul style="list-style-type: none"><li>→ Schwachstellen gegen einfachen Angreifer</li><li>→ Debug als Schwachstelle</li></ul>	<ul style="list-style-type: none"><li>→ Günstig</li><li>→ Low End Lösungen</li></ul>
Externer Koprozessor	<ul style="list-style-type: none"><li>→ Schützt gegen einfachen Angreifer</li><li>→ Kombiniert mit Multichip Schutzmechanismen</li></ul>	<ul style="list-style-type: none"><li>→ Günstig für kleine Stückzahlen</li><li>→ Mittelwertige Lösungen</li></ul>
Interner Koprozessor	<ul style="list-style-type: none"><li>→ Schützt gegen mittleren Angreifer</li></ul>	<ul style="list-style-type: none"><li>→ Günstig für große Stückzahlen</li><li>→ Teure Lösungen</li></ul>

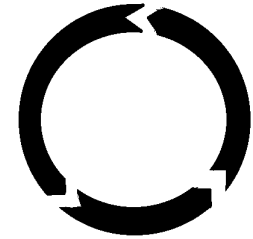
Falls CPU geeignet und die Performanz nicht wichtig ist: Kombination mit Schutzmechanismen des Betriebssystems empfohlen

## Embedded Security als Prozess



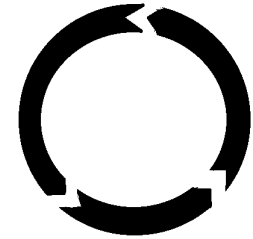
- Die Integration in die Organisation:  
Anpassen an die vorhandenen Prozesse
- Die Analyse der Kosten und des Schadens
- Die Auswertung der Rückmeldungen und die Analyse des Erfolgs eingesetzter Mechanismen

## IT-Grundschutz



- Entwickelt vom BSI
- Stellt wertvolle Richtlinien zur Verfügung, wie Security in einer Organisation eingesetzt werden kann
- Beinhaltet Kataloge für verschiedene Systeme
- **Kein dedizierter Vorschlag für eingebettete Systeme**

## Security Metriken



- Die Darstellung der Eigenschaften des Prozesses als Zahlen
- Werden benutzt, um zu beobachten, ob die investierten Kosten die Mechanismen rechtfertigen
- Die Darstellung des Erfolgs als Zahlen
  - Im Bereich Netzwerksicherheit kann z.B. die Anzahl der gescheiterten Angriffe eine Metrik sein
  - Was kann in eingebetteten Systemen und insbesondere für den IP-Schutz als ähnliche Metrik benutzt werden?

## Zusammenfassung

- Die Umsetzung der IP-Schutz Mechanismen in eingebetteten Systemen stellt Herausforderungen an viele Personen dar: An diejenigen die zu schützende Systeme entwerfen, die sie implementieren und die Managern der betroffenen Organisationen
- Viele Ergebnisse sind für die ersten beiden Gruppen vorhanden, wogegen für die dritte Gruppe nur wenig Informationen existieren
- Zukünftige Technologien wie „Internet of Things“ stellen neue Herausforderungen dar, sie können aber auch die Möglichkeit geben, die Lösungen aus Netzwerksicherheit in eingebetteten Systemen umzusetzen.

## Bibliography

- Phil Giordano and Wassim Bassalee, Locking down intellectual property in embedded systems, embedded.com, 2007  
<http://www.embedded.com/design/embeddeddsp/201801324>
- Sean Smith, Magic Boxes and Boots: Security in Hardware, Computer, vol. 37, no. 10, pp 106 - 109, Oct. 2004
- Norbert Pohlmann und Helmut Reimer, Trusted Computing, Ein Weg zu neuen IT-Sicherheitsarchitekturen, 2008, vieweg Verlag
- Trusted Computing Group, TPM Main Specification Level 2 Version 1.2, Revision 103,  
[http://www.trustedcomputinggroup.org/resources/tpm\\_main\\_specification](http://www.trustedcomputinggroup.org/resources/tpm_main_specification)