Karlsruher Institut für Technologie

Sicherheit in Kognitiven Automobilen

Martin Thoma | 23. Juli 2015

Chrysler Uconnect-Hack

Einleitung

- Kognition: Wahrnehmung der Umwelt
  - Sensoren: Reifenrotation und -druck, Unfall-Detektion, LIDAR, ...
  - Aktoren: Motor, Bremsen, Licht
- VIEL Elektronik: 30 und mehr ECUs in modernen Autos

Einleitung

- Kognition: Wahrnehmung der Umwelt
  - Sensoren: Reifenrotation und -druck, Unfall-Detektion, LIDAR, ...
  - Aktoren: Motor, Bremsen, Licht
- VIEL Elektronik: 30 und mehr ECUs in modernen Autos

Einleitung

- Kognition: Wahrnehmung der Umwelt
  - Sensoren: Reifenrotation und -druck, Unfall-Detektion, LIDAR, ...
  - Aktoren: Motor, Bremsen, Licht
- VIEL Elektronik: 30 und mehr ECUs in modernen Autos

Einleitung

- Kognition: Wahrnehmung der Umwelt
  - Sensoren: Reifenrotation und -druck, Unfall-Detektion, LIDAR, ...
  - Aktoren: Motor, Bremsen, Licht
- VIEL Elektronik: 30 und mehr ECUs in modernen Autos

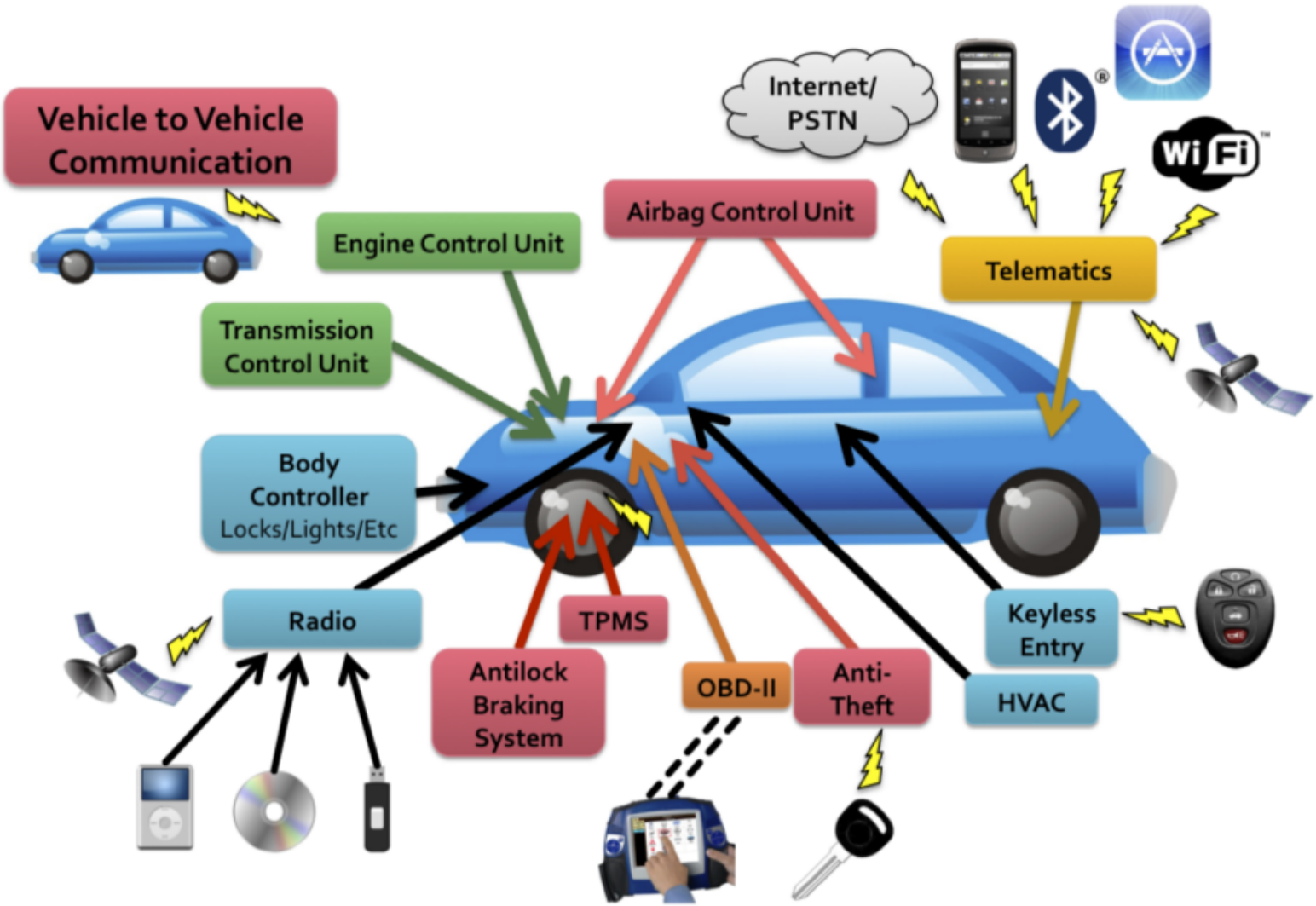
Standards und Verordnungen

- CAN-Bus
- OBD (On-Board-Diagnostics)
- TPMS (Tire preassure measurement system)
- eCall (Elektronischer Notruf)

Angriffsziele

- *Mord / Körperverletzung*: Beschleunigen, Bremsen, Lenken
- *Diebstahl*: Auto öffnen, Position des Autos, Metadaten
- *Ablenkung*: Lichter, Radio
- *Trollen*: Klimaanlage, Dashboard

Angriffsvektoren



Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

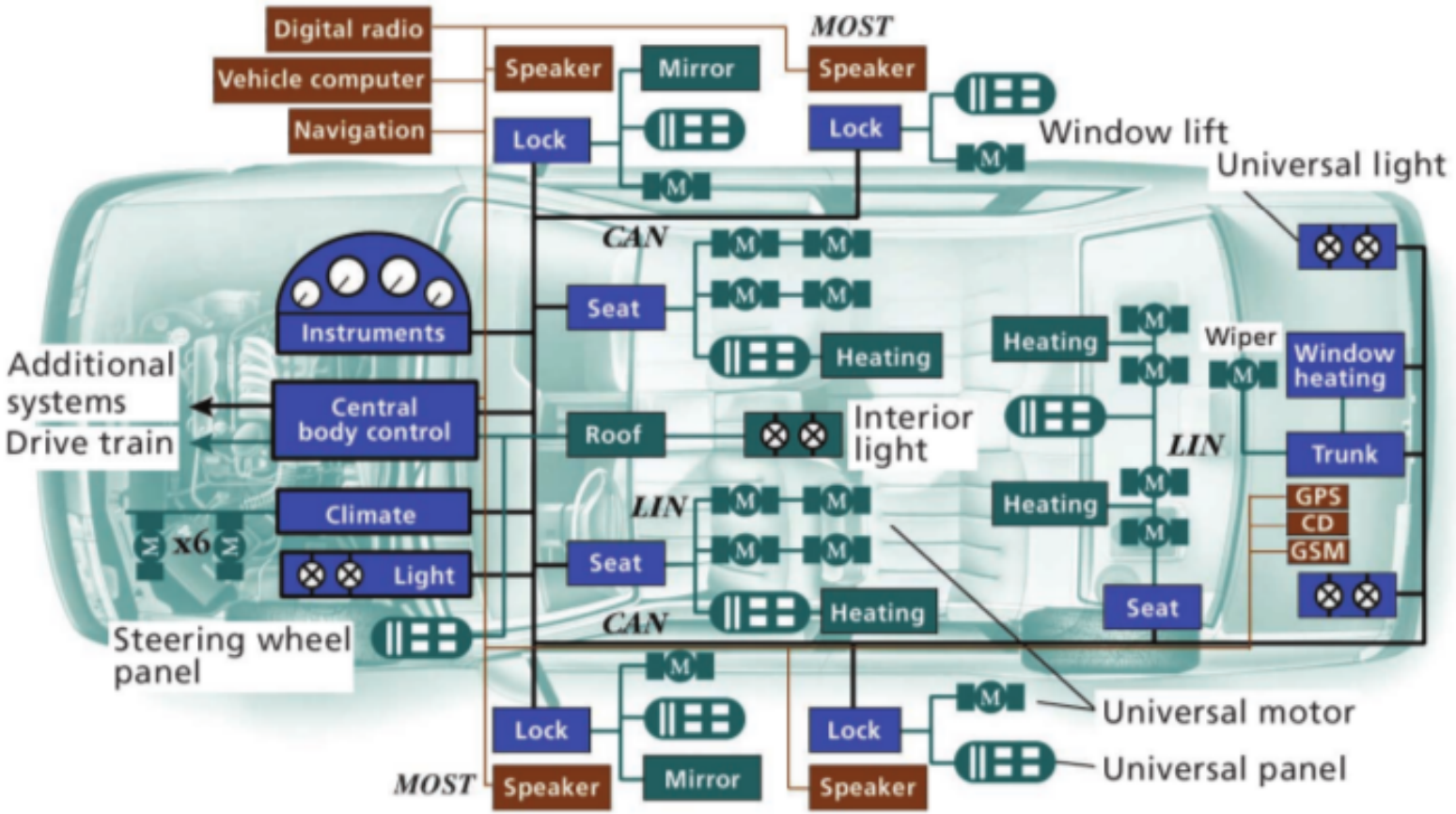
Verteidigung

- Auto-Spezifisch
  - Weniger Elektronik? Getrennte Bus-Systeme?
  - Obfuscation: Debugging-Symbole entfernen um Reverse-Engineering zu erschweren
  - Plausibilität von CAN-Nachrichten prüfen
  - Blackbox
- Allgemein
  - Authentifizierung vor Software-Updates
  - Andere Sprachen (Java, Rust)
  - Besserer Code
    - Code Reviews
    - Static Code Analysis
  - Speizielles (Stack Cookies, `strncpy()` anstatt `strcpy()`)

Thanks for Your Attention!

We also found that the entire attack can be implemented in a completely blind fashion— without any capacity to listen to the car's responses. Demonstrating this, we encoded an audio file with the modulated post-authentication exploit payload and loaded that file onto an iPod. By manually dialing our car on an office phone and then playing this „song“ into the phone's microphone, we are able to achieve the same results and compromise the car.

Thanks for Your Attention!



CAN Controller area network
GPS Global Positioning System
GSM Global System for Mobile Communications
LIN Local interconnect network
MOST Media-oriented systems transport

Quellen

Wichtigste Paper

- Checkoway et al.: Comprehensive Experimental Analyses of Automotive Attack Surfaces
- Koscher et al: Experimental Security Analysis of a Modern Automobile

Bildquellen:

- Folie 2: wired.com/2015/07/hackers-remotely-kill-jeep-highway
- Folie 6: Paper „Comprehensive Experimental Analyses of Automotive Attack Surfaces“
- Folie 8: dailymail.co.uk/sciencetech/article-2553026

Videos and Talks

- A. Davis: Black Hat 2015 - Broadcasting Your Attack: Security Testing DAB Radio in Cars, 2015.
- C. Miller, C. Valasek: Black Hat 2015 - Remote Exploitation of an Unaltered Passenger Vehicle, 2015.
- C. Miller, C. Valasek: Hackers Remotely Kill a Jeep on the Highway—With Me in It, 2015. 5:05 Minuten.
- C. Miller, C. Valasek: Black Hat 2014 - Embedded: A Survey of Remote Automotive Attack Surfaces, 2014. 57:36 Minuten.
- C. Miller, C. Valasek: Defcon 22 - A Survey of Remote Automotive Attack Surfaces, 2014. 50:24 Minuten.
- C. Miller, C. Valasek: Defcon 21 - Adventures in Automotive Networks and Control Units, 2013. 48:10 Minuten.