Karlsruher Institut für Technologie

Seminar Kognitive Automobile

Sicherheit in Kognitiven Automobilien

Seminararbeit
von

Martin Thoma

Fakultät für Informatik
Institut für Anthropomatik
und
FZI Forschungszentrum Informatik

Betreuender Mitarbeiter: Dipl.–Inform. Ralf Kohlhaas

Sommersemester 2015

Kurzfassung

Moderne Automobile verfügen über eine Vielzahl von Assistenz- und Fahrsicherheitssystemen. Diese Systeme haben Schnittstellen, welche das Ziel von Angriffen sein können. In dieser Seminararbeit wird der aktuelle Stand der IT-Sicherheit kognitiver Automobilie untersucht. Dabei wird auf mögliche Angriffe sowie Möglichkeiten zum Schutz eingegangen.

Inhaltsverzeichnis

1. Einleitung

Kognitive Automobile sind, im Gegensatz zu klassichen Automobilen, in der Lage ihre Umwelt und sich selbst wahrzunehmen und dem Fahrer zu assistieren oder auch teil- bzw. vollautonom zu fahren. Diese Systeme benötigen Zugriff auf Sensoren und Aktoren, um ihre Aufgabe zu erfüllen. So benötigt ein Auto mit Antiblockiersystem beispielsweise die Drehzahl an jedem Reifen und die Möglichkeit die Bremsen zu beeinflussen; für Einparkhilfen werden Sensoren benötigt, welche die Distanz zu Hindernissen wahrnehmen sowie Aktoren, die das Auto lenken und beschleunigen können. Weitere dieser Systeme sind Spurhalteassistenz, Spurwechselassistenz und Fernlichtassistenz.

Als immer mehr elektronische Systeme in Autos verbaut wurden, die teilweise sich überschneidende Aufgaben erledigt haben, wurde der CAN-Bus entwickelt[15]. Über ihn kommunizieren elektronische Steuergeräte, sog. *ECUs* (engl. *electronic control units*). Diese werden beispielsweise für ABS und ESP eingesetzt.

Der folgende Abschnitt geht auf Standards wie den CAN-Bus und Verordnungen, die in der Europäischen Union gültig sind, ein. In Kapitel 3 werden Angriffsziele und Grundlagen zu den Angriffen erklärt, sodass in Kapitel 4 mögliche Verteidigungsmaßnahmen erläutert werden können.

2. Standards und Verordnungen

Für den Automobilbereich existieren viele Standards und Verordnungen. In diesem Abschnitt wird eine Auswahl vorgestellt, die Fahrzeuge der Klassen M_1 und N_1 betrifft. Das sind Fahrzeuge zur Personenbeförderung „mit mindestens vier Rädern und höchstens acht Sitzplätzen außer dem Fahrersitz“ sowie „für die Güterbeförderung ausgelegte und gebaute Kraftfahrzeuge mit einer zulässigen Gesamtmasse von 3,5 Tonnen“[8].

In der EU wurde mit [9] die OBD-Schnittstelle verpflichtend für Fahrzeuge der Klasse M_1 und N_1 mit Fremdzündungsmotor ab 1. Januar 2004. Die EU-Direktive führt weiter die in der ISO DIS 15031-6 Norm aufgeführten Fehlercodes als Minimalstandard ein. Diese müssen „für genormte Diagnosegeräte [. . .] uneingeschränkt zugänglich sein“. Außerdem muss die Schnittstelle im Auto so verbaut werden, dass sie „für das Servicepersonal leicht zugänglich [. . .] ist“.

Der Software-Zugang wird mit J2534 [14] standardisiert. Dieser Standard stellt sicher, dass unabhängig vom OBD-Reader Diagnosen über das Auto erstellt und die ECUs umprogrammiert bzw. mit Aktualisierungen versorgt werden können.

Um die Daten bereitzustellen, werden verschiedene elektronische Komponenten über den CAN-Bus vernetzt. Dieser ist in ISO 11898 genormt.

Weiterhin wurde in der EU mit [10] beschlossen, dass ab 1. November 2012 alle PKWs für Neuzulassungen ein System zur Reifendrucküberwachung (engl. *tire pressure monitoring system*, kurz *TPMS*) besitzen müssen. Ab 1. November 2014 müssen alle Neuwagen ein solches System besitzen. Da sich die Räder schnell drehen ist eine kabelgebundene Übertragung der Druckmesswerte nicht durchführbar. Daher sendet jeder Reifen kabellos ein Signal, welches von einem oder mehreren Sensoren im Auto aufgenommen wird.

Mit [11] wird für Fahrzeuge, die ab dem 31. März 2018 gebaut werden das eCall-System, ein elektronisches Notrufsystem, verpflichtend. Dabei müssen dem eCall-System „präzise[-] und verlässliche[-] Positionsdaten“ zur Verfügung stehen, welche über das globales Satellitennavigationssystem Galileo und dem Erweiterungssystem EGNOS geschehen soll. eCall soll über öffentliche Mobilfunknetze eine „Tonverbindung zwischen den Fahrzeuginsassen und einer eCall-Notrufabfragestelle“ herstellen können. Außerdem muss ein Mindestdatensatz übermittelt werden, welcher in DIN EN 15722:2011 geregelt ist. Diese Funktionen müssen im Fall eines schweren Unfalls automatisch durchgeführt werden können.

3. Angriffe

Eine Reihe von elektronischen Systemen wurde zum Diebstahlschutz entwickelt [21, 22, 13]. Allerdings passen sich auch Diebe an die modernen Gegebenheiten, insbesondere Funkschlüssel [17], an. Außerdem gehen diese Systeme von einem klassischen Angreifer aus, der sich ausschließlich auf der Hardware-Ebene bewegt.

Im Folgenden werden zunächst Möglichkeiten von netzwerk-internen Angreifer, d.h. Angreifern welche physischen Zugang zum CAN-Bus haben, aufgelistet. Es folgt eine Beschreibung wie Angreifer ohne direkten physischen Zugriff auf das Auto sich mit dem CAN-Bus verbinden können.

3.1. CAN-interne Angriffe

Koscher et al. haben in [16] zwei nicht näher spezifizierte Autos der selben Marke und des selben Modells untersucht. Sie waren in der Lage über den CAN-Bus etliche Funktionen des Autos, unabhängig vom Fahrer, zu manipulieren. Das beinhaltet das deaktivieren und aktivieren der Bremsen, stoppen des Motors, steuern der Klimaanlage, Heizung, Lichter, Manipulation der Anzeigen im Kombiinstrument sowie das Öffnen und Schließen der Schlösser. Durch moderne Systeme wie eCall kann der Angreifer sich sogar einen Kommunikationskanal zu dem Auto aufbauen. Dies setzt allerdings voraus, dass der Angreifer sich bereits im auto-internen Netzwerk befindet.

3.2. CAN-externe Angriffe

In [6] wurde an einer Mittelklasselimosine mit Standardkomponenten gezeigt, dass der Zugang zum auto-internen Netzwerk über eine Vielzahl an Komponenten erfolgen kann. So haben Checkoway et al. CD-Spieler, Bluetooth und den OBD-Zugang als mögliche Angriffsvektoren identifiziert.

Bei dem Angriff über den Media Player haben Checkoway et al. die Tatsache genutzt, dass dieser am CAN-Bus angeschlossen ist und die Software des Mediaplayers über eine CD mit einem bestimmten Namen und Dateiformat aktualisiert werden kann. Außerdem wurde ein Fehler beim abspielen der Audio-Dateien genutzt um einen Buffer Overflow zu erzeugen. Es wurde gezeigt, dass dieser genutzt werden kann um die Software des Media-Players zu aktualisieren. Dafür muss nur eine modifizierte Audio-Datei, welche immer noch abspielbar ist, auf der CD sein.

Der von Checkoway et al. durchgeführte Angriff via Bluetooth benötigt ein mit dem Media Player verbundenes Gerät. Dieses nutzt dann unüberprüfte `strcpy`-Befehle bei der Bluetooth-Konfiguration aus um beliebigen Code auf der Telematik-Einheit des Autos ausführen zu können. Daher ist das Smartphone des Autobesitzers ein Angriffsvektor.

Die Bluetooth-Verbindung kann jedoch auch ohne ein verbundenes Gerät für Angriffe genutzt werden. Dazu muss der Angreifer genügend Zeit in der Nähe des Autos verbringen um die Bluetooth-MAC-Adresse zu erfahren. Damit kann er ein Anfrage zum Verbindungsaufbau starten. Diese müsste der Benutzer normalerweise mit der Eingabe einer PIN bestätigen. Checkoway et al. haben für ein Auto gezeigt, dass die Benutzerinteraktion nicht nötig it und die PIN via Brute-Force, also das Ausprobieren aller Möglichkeiten, innerhalb von 10 Stunden gefunden werden kann. Allerdings kann dieser Angriff parallel ausgeführt werden. Es ist also beispielsweise möglich diesen Angriff für alle Autos in einem Parkhaus durchzuführen.

Die standardisierte und von Automechanikern zu Diagnosezwecken genutzte OBD-Schnittstelle stellt einen weiteren Angriffspunkt dar. Für die verschiedenen Marken gibt es Diagnosewerkzeuge, wie z.B. NGS für Ford, Consult II für Nissan und der Diagnostic Tester von Toyota. Diese dedizierten Diagnosegeräte werden allerdings über PCs mit aktualisierungen versorgt. Modernere Diagnosewerkzeuge sind nicht mehr bei der Diagnose vom PC getrennt, sondern werden direkt, über ein Kabel, W-LAN oder Bluetooth, mit einem PC verbunden. Daher stellt die Diagnose- und Aktualisierungstätigkeit von Automechanikern einen weiteren Angriffsvektor dar. Wenn der Mechaniker ein Diagnosegerät benutzt, welches ein W-LAN aufbaut, so können Angreifer sich mit diesem verbinden und selbst Aktualisierungen durchführen. Außerdem wurde von Checkoway et al. gezeigt, dass auch das Diagnosegerät selbst so manipuliert werden kann, dass es automatisch die gewünschten Angriffe ausführt.

Wie in Kapitel 2 beschrieben wird eCall-System ab 2018 in Europa verpflichtend eingeführt. Dieses nutzt das Mobilfunknetz zur Kommunikation. Checkoway et al. haben gezeigt, dass Telematik-Einheiten von außerhalb des Autos angewählt werden und die Software auf diese Weise aus beliebigen Entfernungen aktualisiert werden kann. Dazu wurden zahlreiche Schwachstellen der Telematik-Einheit von Airbiquitys Modem aqLink genutzt. Dieses Modem wird unter anderem von BMW und Ford eingesetzt [1, 2].

Ein Angriff auf die Privatsphäre ist mit TPMS möglich. In [20] wurde gezeigt, dass TPMS-Signale zur Identifikation von Autos genutzt werden können. Die Identifikation eines vorbeifahrenden Autos ist aus bis zu 40 m Entfernung möglich.

Genauso stellt das Mikrofon, welches wegen eCall ab 2018 in jedem Auto sein muss, eine Möglichkeit zum Angriff auf die Privatsphäre dar.

3.3. Buffer-Overflow Angriffe

Dieser Abschnitt erklärt anhand eines einfachen Beispiels wie Buffer Overflow Angriffe durchgeführt werden. Der gezeigte Assembler-Code wurde mit der GCC auf einem 64-Bit Linux-Rechner erstellt. Grundlagen über Assembler-Code sind in [3] zu finden.

Buffer-Overflow Angriffe nutzen die Tatsache aus, dass bestimmte Befehle wie Beispielsweise `gets` Strings in eine Puffer schreiben, ohne die Größe des Puffers zu beachten. `gets` erhält als Parameter einen Zeiger auf die Startadresse des Puffers. Wenn der Benutzer eine längere Eingabe macht als der Puffer erlaubt, so wird in nachfolgende Speicherbereiche geschrieben. Dies kann an folgendem Beispiel aus [4] beobachtet werden:

simple.c

```c
#include <stdio.h>

int main(void) {
    char buff[10];
    int pass = 0;

    printf("Enter password: ");
    gets(buff);

    if (strcmp(buff, "correct")) {
        printf("Wrong Password\n");
    } else {
        printf("Correct Password\n");
        pass = 1;
    }

    if (pass) {
        printf ("Password protected code. pass=%i\n", pass);
    }

    return 0;
}
```

Kompiliert man dieses Programm mit `gcc -O0 -fno-stack-protector -g simple.c -o simple`, so kann man mit der Eingabe von 16 Zeichen die Variable `pass` überschreiben.

Allerdings ist es nicht nur möglich interne Variablen zu überschreiben, sondern sogar beliebigen Code auszuführen. Dies wird in [19] mit einem sehr ähnlichem Beispiel gezeigt und im Detail erklärt. Dabei wird nicht beliebiger Text in den Puffer geschrieben, sondern sogenannter *Shellcode*. Unter Shellcode versteht man Assemblerbefehle, welche in Opcodes umgewandelt wurden.

4. Verteidigungsmaßnahmen

Der CAN-Bus ist eine große Schwachstelle der IT-Sicherheit in Autos. Über ihn müssen viele ECUs kommunizieren und einige, wie das Autoradio, werden nicht als Sicherheitskritsch wahrgenommen.

Daher ist es wichtig die Nachrichten, welche über den CAN-Bus empfangen werden, zu filtern. Die Informationen müssen auf Plausibilität geprüft werden.

Alle von Checkoway et al. beschriebenen Angriffe basieren zum einen auf Reverse-Engineering, also der Rekonstruktion der Software-Systeme und Protokolle, zum anderen auf Fehlern in der Software. Das Reverse-Engineering wurde in einigen Fällen laut Checkoway et al. stark vereinfacht, da Debugging-Symbole in der Software war. Diese können und sollten einfach entfernt werden.

Außerdem sollten laut Checkoway et al. die Diagnosegeräte Authentifizierung und Verschlüsselung wie beispielsweise OpenSSL nutzen.

Gegen Buffer-Overflow-Angriffe können zum einen Sprachen wie Java oder Rust verwendet werden, welche die Einhaltung der Bereichsgrenzen automatisch überprüfen. Des weiteren kann anstelle der C-Funktion `strcpy()` die Funktion `strncpy()` verwendet werden, welche die Anzahl der zu schreibenden Zeichen begrenzt [7]. Ein weiteres Konzept zum Schutz vor Buffer-Overflow-Angriffen sind Stack Cookies [5].

Code Reviews können auch solche Sicherheitslücken aufdecken [12]. Code Reviews können teilweise automatisch mit Werkzeugen zur statischen Code Analyse durchgeführt werden [18].

A. Literaturverzeichnis

[1] Airbiquity signs telematics deal with bmw, Oktober 2006.

[2] Airbiquity link highlights ford's telematics strategy, 2008.

[3] D. Albert. Understanding c by learning assembly, September 2012. Available at `https://www.recurse.com/blog/7-understanding-c-by-learning-assembly`.

[4] H. Arora. Buffer overflow attack explained with a c program example, Juni 2013. Available at `http://www.thegeekstuff.com/2013/06/buffer-overflow/`.

[5] B. Bray. Compiler security checks in depth. MSDN, Februar 2002.

[6] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno. Comprehensive experimental analyses of automotive attack surfaces. In *Proceedings of the 20th USENIX Conference on Security*, SEC'11, pages 6–6, Berkeley, CA, USA, 2011. USENIX Association.

[7] C. Eckert. *IT-Sicherheit*. Oldenbourd Wissenschaftsverlag GmbH, 2012.

[8] Europäischer Rat. Richtlinie des rates 70/156/ewg, Februar 1970.

[9] European Parliament, Council of the European Union. Richtlinie 98/69/ec des europäischen parlaments und des rates, Oktober 1998. `http://eur-lex.europa.eu/legal-content/en/ALL/?uri=CELEX:31998L0069`.

[10] European Parliament, Council of the European Union. Verordnung (eg) nr. 661/2009 des europäischen parlaments und des rates, Juli 2009.

[11] European Parliament, Council of the European Union. Verordnung (eu) 2015/758 des europäischen parlaments und des rates, April 2015.

[12] M. Howard. A process for performing security code reviews. *Security Privacy, IEEE*, 4(4):74–79, July 2006.

[13] S. Hwang. Wireless car security system, July 15 1997. US Patent 5,648,754.

[14] S. International. Recommended practice for pass-thru vehicle programming, Dezember 2004.

[15] U. Kiencke, S. Dais, and M. Litschel. Automotive serial controller area network. Technical report, Robert Bosch GmbH, Februar 1986.

[16] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental security analysis of a modern automobile. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, SP '10, pages 447–462, Washington, DC, USA, 2010. IEEE Computer Society.

[17] D. Lee. Keyless cars 'increasingly targeted by thieves using computers', Oktober 2014.

[18] G. McGraw. Automated code review tools for security. *Computer*, 41(12):108–111, Dec 2008.

[19] Mixter. Writing buffer overflow exploits - a tutorial for beginners. Available at `http://www.eecis.udel.edu/~bmiller/cis459/2007s/readings/buff-overflow.html`.

[20] I. Rouf, R. Miller, H. Mustafa, T. Taylor, S. Oh, W. Xu, M. Gruteser, W. Trappe, and I. Seskar. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *Proceedings of the 19th USENIX Conference on Security*, USENIX Security'10, pages 21–21, Berkeley, CA, USA, 2010. USENIX Association.

[21] H. Song, S. Zhu, and G. Cao. Svats: A sensor-network-based vehicle anti-theft system. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages –, April 2008.

[22] V. Turner. Automotive vehicle anti-theft and anti-vandalism and anti-carjacking system, Dec. 14 1999. US Patent 6,002,326.