Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projektbericht

Jonas Schäufler

Ausgewählte sicherheitsrelevante Use-Cases einer zonen-basierten autmotive Netzwerk Architektur

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Jonas Schäufler

Ausgewählte sicherheitsrelevante Use-Cases einer zonen-basierten autmotive Netzwerk Architektur

Projektbericht eingereicht im Rahmen der Prüfungsleistung des Grundprojekts

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Franz Korf

Eingereicht am: 22. März 2019

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Einleitung

Diverse Forschungsprojekte befassten sich ausgiebig mit der Entwicklung einer Sicherheitsarchitektur für *autmotive* Systeme. Das EVITA Projekt (2008-2011) entwickelte eine Architektur [1], prototypisch Implementiert und getestet, welche als Basis für weitere Forschung diente. Insbesondere die formale Herleitung der Anforderungen und die zur Modellierung und Verifikation entwickelten Methodologien[2][3][2] und Werkzeuge[4] lieferten großen Mehrwert. Für *safety-critical* Systeme wie Automobile ist es nötig das Design zu verifizieren. Deshalb war es wichtig die verwendeten Methodologien und Werkzeuge zu erweitern, um die neuen, durch Sicherheitsanforderungen hinzugekommenen, Attribute im System, wie zum Beispiel *Vertraulichkeit* oder *Integrität*, zu unterstützen. So dass *Safety*- und *Security*- Aspekte und deren Zusammenhänge mit den selben Werkzeugen modelliert und formal verifiziert werden können.[1]

Auch in der Industrie können sich Ergebnisse des EVITA Projektes zu Nutzen gemacht werden. Die Gerätetreiber für die Kommunikation mit den *Hardware Security Modulen* (HSM) sind AUTOSAR konform und eine Software-Implementierung dessen Funktionen ist bereits validiert worden, so dass unabhängige Weiterentwicklungen in der Industrie Kompatibel sind und eine Standardisierung möglich ist. Einer der treibenden Faktoren für diese Entwicklungen ist die zunehmende Vernetzung des Automobils mit externen Netzen und die steigende Anzahl der internen Steuergeräte, Sensoren und Aktuatoren.
Die steigende Komplexität des Systems treibt die Weiterentwicklung des internen Automobilnetzwerks voran. In heutigen Automobilen wird jede Funktionalität auf einer eigenen ECU integriert. Um die höheren Anforderungen bewältigen zu können und die Anzahl der ECUs zu verringern sollen Funktionen in Zukunft immer zentralisierter integriert werden.

[1] Um im folgenden eine Verwechslung von *Safety* und *Security* durch die deutsche Übersetzung *Sicherheit* zu vermeiden ist Anzumerken, dass in diesem Dokument das Wort *Sicherheit* ausschließlich im Kontext der Informationssicherheit verwendet wird.

2 Referenzarchitektur

In diesem werden die Grundlagen der, in der Referenzarchitektur verwendeten, Protokolle und Komponenten als auch Begrifflichkeiten erläutert.

2.1 Protokolle

Dieser Abschnitt erläutert die in der Architektur verwendeten Protokolle Ethernet und CAN (Controller Area Network). Eine betrachtung der Protokolle, unabhängig der Anwendungsebene, ist wichtig um Angriffsvektoren durch die Sicherungsschicht in einem automotive Kontext zu identifizieren. Angriffe auf die Sicherungsschicht sind nicht teil dieser Arbeit, jedoch ist es wichtig zu erläutern mit welchen Möglichkeiten, und auf welche Art und Weise, Nachrichten im Netzwerk ausgetauscht werden können.

2.1.1 Ethernet

Mechanismen des Ethernet Protokolls, die der automatischen Konfiguration des Netzwerkes dienen (Flooding, STP), als auch im Zusammenhang mit Ethernet verwendete Management-Protokolle für die Transportschicht, wie ARP und DHCP, liefern Oberfläche für Angriffe. Hierzu gehören unter anderem *Denial of Service* durch *Resource Exhaustion*, *Man-in-the-Middle* und damit verbundene Folge-Angriffe wie *Session-Hijacking* auf darüberliegenden Schichten, eine unauthorisiertes hinzufügen von Hosts in das Ethernet Netzwerk, als auch das Stören und Mitschneiden der Kommunikation im Netzwerk. [5]

Für die Bitübertragungsschicht existieren bereits Standards, wie 100BASE-T1 (BroadR-Reach), um Ethernet für die Verwendung im automotive Bereich anwendbar zu machen. Mit dieser Technologie können bis zu 100 Megabit pro Sekunde im Gegenbetrieb über einzelnes *Twisted-Pair* Kabel übertragen werden.[6] Nicht nur die ehöhte Datenrate ist für automotive Applikationen eine Vorraussetzung, auch das geringere Gewicht, und Kosteneffizienz, der Kabel ist für die Industrie von Vorteil.[7] Auch auf logischer Ebene wird das Protokoll adaptiert, jedoch geht es dabei Hauptsächlich darum Echtzeitanforderungen einhalten zu können und die Energieeffizienz zu optimieren, auf den Sicherheitsaspekt wird bei Adaptionen kaum eingegangen.

Time-Triggered Ethernet und AV-Bridging sind Technologien, die in diesem Zusammenhang zu nennen wären.[8][9] Auch VLAN Technologie kommt in automotive System zum Einsatz. Dies erhöht einerseits die Sicherheit durch minimierung der Broadcast Domäne, jedoch bedarf es hierfür auch einen erhöhten Konfigurationsaufwand und Angriffe wie *Switch-Spoofing* und *Double-Tagging* sind möglich, wenn Switches nicht korrekt konfiguriert sind.[10]

Für diese Arbeit wird davon ausgegangen, dass die Konfiguration der Ethernet-Komponenten ein vollständiges Routing ermöglicht, und die Sicherheitsrisiken im Zusammenhang mit der Eigenkonfiguration des Netzwerkes durch dessen initiale Konfiguration negiert werden. Zum Beispiel sind bei Router, Switches und Hosts alle nötigen MAC-Addressen in den Tabellen vorhanden, so dass *Flooding* für *Host Discovery* auf allen Geräten deaktiviert werden kann, und eine Nachricht von einer unbekannten MAC-Addresse als Indiz für einen Angriff gewertet werden muss. Die Eigenkonfiguration des Ethernet-Netzwerks ist zwar auch für ein automotive Netzwerk ein großer Vorteil, jedoch sind die hierfür verwendeten Mechanismen nur in bestimmten Fällen, wie der initialen Konfiguration oder beim austauschen eines Bauteils, nötig. Aus einer Sicherheitsperspektive wäre es besser diese Mechanismen im normalen Betrieb zu deaktivieren oder für diesen Anwendungsfall zu adaptieren, um Angriffe die diese Mechanismen ausnutzen, wie zum Beispiel MAC-Flooding bei Switches, zu verhindern. Wie genau ein automotive Netzwerk mit Ethernet Komponenten hinsichtlich dessen Sicherheit konfiguriert werden muss, ist nicht Teil dieser Arbeit.

2.1.2 CAN

Im Gegensatz zu Ethernet ist CAN (Controller Area Network) speziell für die Anwendung in industriellen Systemen entwickelt. Ein grundlegender sicherheitsrelvanter Aspekt in welchem sich die Protokolle unterscheiden ist, dass CAN ein addressloses Protokoll (*Message-Based*) ist das nicht geroutet wird. Alle Knoten eines CAN-Netzwerks hängen am gleichen Seriellen Bus. Ein CAN-Frame enthält keine Zieladdresse sondern lediglich eine ID, den Communication Object Identifier (COB-ID), welche den Typ der Nachricht kennzeichnet. COB-IDs können zwar, je nach Anwendungsfall, mit einer Sender-Identifikation kodiert werden, jedoch ist die dekodierung der Nachricht wiederrum teil der Anwendungsebene. Übertragene Frames werden an alle Knoten im Netzwerk gesendet, und verworfen wenn die Anwendung diesen Nachrichtentyp nicht nutzt oder erwartet. Diese Eigenschaft macht Angriffe auf Anwendungen in einem CAN-Netzwerk deutlich leichter, da Nachrichten leicht in das Netzwerk eingespeist und mitgeschnitten werden können.

Ein weiterer, dem Protokoll inheränter, sicherheitskritischer Aspekt ist das prioritätsbasierte Scheduling der Nachrichtenübertragung. Welcher Knoten an der Reihe ist, eine Nachricht

zu senden, wird durch die zu sendende COB-ID bestimmt. Sendet ein Knoten einen Frame, aber zur selben Zeit ein Frame mit einer niedrigeren COB-ID auf den Bus gelegt wird, erkennt der Sender eine Nachricht mit höherer Priorität, abgeleitet durch den bitweisen Vergleich der IDs, und stoppt daraufhin die weitere Übertragung seines Frames. Dieser Mechanismus des Präemtiven Scheduling macht CAN einerseits Echtzeitfähig, bietet Angreifern aber auch eine einfache Möglichkeit das Netzwerk komplett lahmzulegen, dadurch dass der Bus mit Frames von höchster Priorität geflutet werden kann.[11]

Der Aspekt der Sicherheit ist beim Entwurf des CAN Protokolls nicht addressiert worden. Da auch automotive Netzwerke bis vor kurzem komplett geschlossene Systeme waren, und ein Angreifer für einen Angriff Zugriff auf den Bus haben muss, waren die ergriffenen Maßnamen hauptsächlich die Abschirmung des Buses und die geheimhaltung der Kommunikationsmatrix. Um Knoten im CAN-Netzwerk vollständig zu manipulieren, muss also lediglich, meist durch ausnutzen einer Schwachstelle in einer Komponente des Netzwerks, Zugriff auf den Bus erlangt werden, und die Kommunikationsmatrix aus mitgeschnittenen Übertragungen *reverse-engineered* werden.[12] Das Reverse-Engineering der CAN Nachrichten ist mittlerweile ein gut Dokumentiertes vorgehen, und das *Know-How* für jederman zugänglich. [13] Automatisierte Verfahren, welche *Machine-Learning* Techniken verwenden, sind Bestand aktueller Forschungen. [14] Es existiert auch eine Erweiterung des CAN Protokolls, das nicht *Event-Triggered* sondern *Time-Triggered* ist (TTCAN), um besser mit Echtzeitanforderungen umzugehen. Auf diese Abwandlung wird jedoch hier nicht weiter eingegangen, da es in Anbetracht der Sicherheit die selben Eigenschaften besitzt.

2.1.3 LIN, MOST und Flexray

Weitere im automotive Bereich verwendeten Protokolle sind LIN (Local Interconnect Network), Most (Media Oriented Systems Tranpsport) und Flexray. Eine nähere Betrachtung dieser Protokolle wird für diese Arbeit außen vor gelassen, und in der Architektur durch CAN und Ethernet ersetzt, da sie ähnliche Charakteristiken besitzen. CAN wird hier räpresentativ für ein Nachrichtenbasiertes Broadcast-Protokoll eines Feldbusses verwendet, und Ethernet für ein Addressen-basiertes, geroutetes, Unicast-Protokoll.

2.1.4 Netzwerk- und Transportschicht

Für CAN existieren zahlreiche *Higher Level Protocols* für verschiedene Anwendungsfälle. Beispiele für mehrzweck Protokolle, die in der Automobilindustrie verwendet werden und auf CAN aufsetzen, wären zum Beispiel ISO-TP/TP2.0 und CANopen. ISO-TP ist ursprünnglich

entwickelt worden um verschiedene existierende Diagnoseprotokolle über den CAN-Bus verwendendbar zu machen. Es erweitert das Konzept mit Addressierung einzelner Nodes (Unicast), Verbindungen (Stateful), Flusskontrolle und Segmentierung. CANopen soll für die Interoperabilität selber Gerätetypen unterschiedlicher Hersteller sorgen. Hierfür werden für Knoten eines CAN-Netzwerks *Communication Profiles*, *Device Profiles* und das *Object Dictionary*, eine standartisierte Geräte-Spezifikation definiert. Die standartisierte Repräsentation des Gerätetypes ermöglicht es die Soft- und Hardwarekomponenten zu entwickeln die mit Gerätetypen verschiedener Hersteller kompatibel ist. So kann das Protokoll einen Zugriff auf Zustände der Geräte, deren Synchronisierung, und verschiedene Arten des Datentransfers, unabhängig des Herstellers und der genauen Bauweise, zur Verfügung stellen. [15] Es existieren Zahlreiche weitere Protokolle die auf CAN aufsetzen. Die AUTSOAR Spezifikation der CAN Transportsschicht basiert auf ISO-TP (ISO-15765) mit diversen Einschränkungen und Erweiterungen, bleibt jedoch mit den geeigneten Einstellungen vollständig Kompatibel. [16] Ein naheliegender Kandidat für ein Netzwerkprotokoll auf Ethernet ist natürlich das Internet Protokoll (IP), und eine Spezifikation eines TCP/IP-Stacks ist auch in AUTOSAR enthalten. Da die *socket*-Schnittstelle aber nicht mit dem Konzept der AUTOSAR PDUs (Protocol Data Unit) ohne weiters vereinbar ist, wird auch ein *Socket Adaptor Module* Vorrausgesetzt, dass diese Schnittstellen miteinander verbindet.[17][18]

Für den Zweck dieser Arbeit wird kein bestimmtes Netzwerk-, Transport-, oder Anwendungsprotokoll spezifiziert. Konzepte die, wie zum Beispiel Verbindungen und zusätzliche Addressierung, von Protokollen, überhalb der Vermittlungsschicht angesiedelt, eingeführt werden, können durch direkten Zugriff auf das Interface umgangen werden, sofern diese Protokolle nicht auch in Hardware implementiert, beziehungsweise die Mechanismen nicht kryptographisch abgesichert sind. Außerdem soll das Ergebnis dieser Arbeit auch die Wahl geeigneter Protokolle für die jeweiligen Anwendungsfälle, durch die Identifizierung von Risiken, unterstützen.

2.1.5 AUTOSARs *Security Modules*

Dieser Abschnitt widmet sich den AUTOSAR Komponenten die für die Absicherung der Kommunikation zuständig sind. Dies sind zwar keine Protokolle, jedoch eng mit dem *Communication Stack* (ComStack) verwoben, und sollen der Vollständigkeit halber kurz erläutert werden, um zu zeigen wie die nötigen Sicherheitsmechanismen in einem automotive System integriert werden.

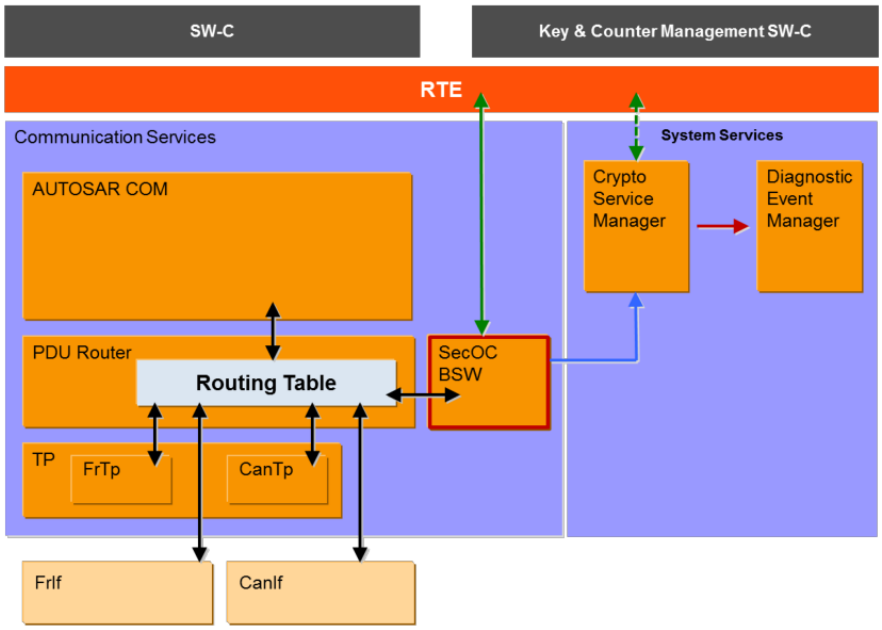


Abbildung 2.1: Security Modules im ComStack [19]

Die zentrale Komponente stellt das SecOC (Secure Onboard Communication) dar. Sichere *Protocol Data Units* (PDU) werden vom PDU Router (PDUR) an das SecOC Module weitergeleitet. Dessen Aufgabe ist es, die Authenzitiät der Nachrichten zu gewährleisten. Die Spezifikation des SecOC wurde unter der Annahme entwickelt, dass hauptsächlich symmetrische Verfahren mit *Message Authentication Codes* (MACs) verwendet werden, die Spezifikation ist aber so allgemein gehalten das auch asymmetrische Verfahren auf das Interface abbildbar sind, wobei man dann von Signaturen als Authenzitiätsnachweis spricht. In der Layer orientierten Architektur von AUTOSAR werden Mechanismen auf verschiedenen Ebenen implementiert, um alle Sicherheitsanforderungen abzudecken. Hier wird unterschieden zwischen Authentifizierung, Verschlüsselung und *Intrusion Protection*. Das SecOS ist zuständig für die Authentifizierung der PODs, während die Verschlüsselung teil der Anwendungsebene ist. Hierfür verwendet das SecOS Interfaces des CSM: MAC-generate, MAC-verify, Signature-generate, Signature-verify, und Key Management Funktionen (Csm_KeyElementSet, Csm_KeySetValid). Die Interfaces, durch einen Port in das Runtime Environment (RTE) eingebunden, des CSM zur ver- und entschlüsselung werden von der SWC (Software Component) auf Anwendungsebene verwendet. Integrität und Authenzitiät der Nachrichten werden durch die verwendung der MAC (bzw. Signatur) gewährleistet. Weiterhin fügt das SecOS einer PDU einen Zählerstand hinzu, welcher

die Sicherheitsanforderung der *Freshness* einer Nachricht abdeckt (siehe nächster Abschnitt). [20][21][22]

2.1.6 Sicherheitsanforderungen eines IVN

Im Rahmen von EVITA wurden folgende Sicherheitsanforderungen für ein IVN identifizert. Diese Anforderungen werden im Kapitel 3 den erläuterten Anwendungsfällen zugeordnet.

Tabelle 2.1: Security requirements of vehicle networks [23][24][25]

	Security requirements	Description
Data origin authenticity	The data source is authentic and reliable	
Integrity	The data is not modified when transferring	
Access control	Authorization before the access to information	
Freshness	Time information of related message	
Non-repudiation	The actions of entity is undeniable	
Privacy/anonymity	The information of entity is confidential	
Confidentiality	Only authorized entities can get the information	
Availability	The services provided are operational	

2.1.7 CAN über Ethernet

Es existieren verschiedene Möglichkeiten CAN Frames, oder dessen Applikationslayer, über Ethernet zu übertragen. Ein standartisiertes Protokoll hierfür ist EtherCAT, das es CANopen *Devices* (siehe Abschnitt 2.1.4) ermöglicht, über eine Ethernet Sicherungsschicht zu kommunizieren. In diesem Fall wird Ethernet gemäß EtherCAT als Feldbus verwendet und CAN ersetzt. Eine andere Möglichkeit ist, die Frames mit einem Hardware-Converter direkt zu übersetzten. Auch Converter für Ethernet mit TCP/IP auf CAN oder CANopen sind auf dem Markt verfügbar. Welche Technologie für die Übertragung der CAN-Frames durch das Ethernet-Backbone verwendet wird, ist für diese Arbeit nicht genau spezifiziert, Jedoch werden für die *Zone-Controller* (Siehe 2.2.2) folgende Funktionalitäten vorrausgesetzt:

Tunneling CAN Frames werden nicht direkt in Ethernet Frames übersetzt sondern getunnelt. Ein CAN FD Frame besitzt eine MTU von 64 Oktetts, mit einer maximalen Framegröße von 184 Byte. Die MTU eines Ethernet Frames mit 1500 Oktetts ist also ausreichend um 8 CAN FD Frames zu transportieren.

Aggregation Da mehrere CAN Frames in einem Ethernet Frame übertragen werden können. Besteht die Möglichkeit, dass in den Switches aggregiert wird.

Logische Zuordnung Zonen (siehe 2.2.1) können mehrere Feldbus-Linien enthalten. Diese CAN-Subnetze können Linien in anderen Zonen zugeordnet werden, und wirken innerhalb des Subnetzes transparent als der selbe Bus. Dafür ist es nötig das CAN-Frames innerhalb eines Ethernet-Frames auf irgendeine Weise annotiert werden, so dass der Controller diese an den richtigen Bus weiterleitet. Dies kann auf verschiedene Art und Weisen geschehen, zum Beispiel codiert in der COB-ID, oder mit einem Tunneling-Protokoll, das CAN-Frames auf Ethernet um diese Routing-Daten erweitert. Eine weitere Möglichkeit wäre es CAN-Interfaces eine Ethernet-MAC zu geben, jedoch kann in diesem Fall nicht aggregiert werden. Wie genau die Zuordnung erfolgt ist nicht näher spezifiziert.

Bidirektionale Kommunikation Frames von CAN-Bussen die aneinander zugeordnet sind, werden bidirektional weitergeleitet und die Verbindung ist vollständig transparent für die Komponenten des logischen Busses.

2.2 Begriffserklärungen

In diesem Abschnitt werden verwendete Begriffe im Zusammenhang mit automotive Architekturen erläutert. Hierzu gehören Begrifflichkeiten die verschiedene Arten von E/E-Architekturen beschreiben, Typen von Komponenten derer, und Begriffe die für den Zweck dieser Arbeit definiert worden sind.

2.2.1 Architektur

Neue Komponenten und Services in Automobilen, sowie die Öffnung des automotive Netzwerks mit der Vernetzung in Infrakstruktursysteme sind die Treiber für steigende Anforderungen an das fahrzeuginterne Netzwerk (IVN). Gleichzeitig will man, durch reduktion der physikalischen Verbindungen, an Kosten und Gewicht einsparen. Folglich strebt man an fahrzeuginterne Netzwerke weniger modular zu gestalten und die Funktionalitäten so weit wie möglich zu zentralisieren, wie in Abbildung 2.2 dargestellt. Ein IVN ist heutzutage ein sehr verteiltes System, wobei jeder Funktion einer, oder sogar mehrerer, ECUs zugeordnet sind. Diese werden oft von verschiedenen Herstellern geliefert, und es wird auch hier wiederrum oft Software mehrerer Subunternehmer auf ihnen integriert. Der erste Schritt ist es diese Funktionen auf immer weniger ECUs zu integrieren, diese aber dafür mit mehr Resourcen auszustatten. Siehe Abbildung 2.2: *Today - Functional integration.*

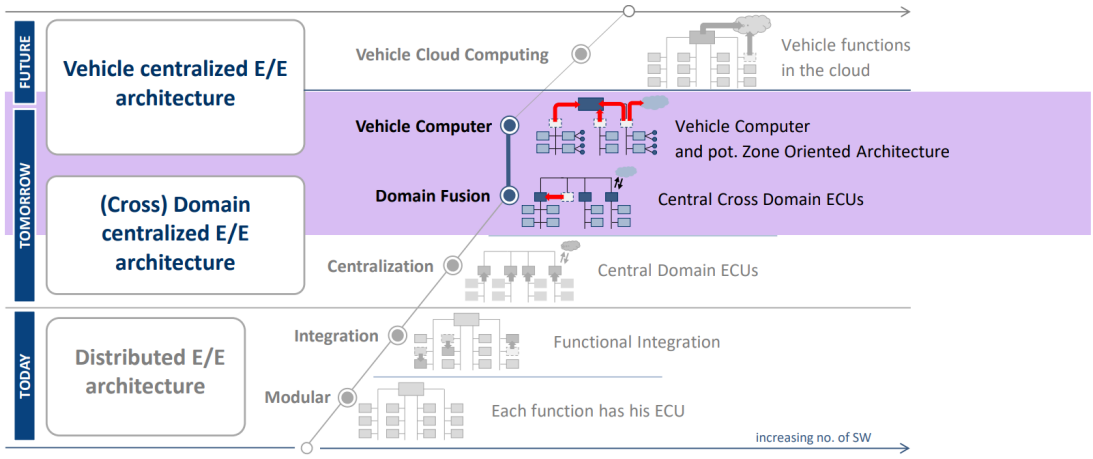


Abbildung 2.2: Autmotive E/E-Architektur Roadmap [26]

Eine sich wandelnte Netzwerkarchitektur erfordert auch eine Anpassung der Software des Systems. So bewegt man sich auf Softwareebene hin zu einer Serviceorientierten-Architektur (SOA). Davon verspricht man sich, die im Wandel der Digitalisierung weiterentwickelten, Infrakstruktursysteme und Infrakstruktur-Dienste besser mit dem Automobil in Einklang bringen zu können. Auch was Wartung, und Dinge wie Software-Updates, angeht soll eine Serviceorientierten-Architektur Verbesserung mit sich bringen. Wenn heutzutage Software eines Bauteiles aktualsiert werden soll, ist dies, durch die Verteilung der Komponenten, kein einfaches Unterfangen, da man auch *over-the-air* keinen direkten Zugriff auf die meisten Bauteile hat, sondern Umwege über andere Komponenten des Netzwerks gegangen werden muss, damit das Update am Zielsystem eingespielt werden kann. [26][27][28][29][30]

Domain Komponenten eines Automobils werden in verschiedene Domänen (im folgenden Domain oder Domains) eingeteilt. Die genaue zuordnung unterscheidet sich teilweise von Hersteller zu Hersteller es lassen sich aber folgende übergreifende Kategorien bilden:

Body and Convenience Unter diese Kategorie fallen Komfort-Komponenten und Komponenten die mit der Außenhülle der Karosserie in Verbindung stehen, zum Beispiel *Door-Control*, *Head-Up-Display*, *Seat-Control*, *Trunk-Control*, *Heating-System*, *Car Exterior Lighting*.

Chassis and Safety Hierzu zählen *Safety*-relevante Komponenten, welche im *Vehicle-Frame* der Karosserie eingebaut sind. Zum Beispiel: Antiblockiersystem (ABS), Aktive Radaufhängung (im folgenden *Active Suspension*, elektrische Handbremse (EPB), Airbags, Servolenkung (EPS), Fahrdynamikregelung (ESC) und Immobilizer.

Infotainment Enternaintment und Statusanzeigen. Beispielsweise Audioanlage, *Head-Unit* und andere im Automobil verbaute Displays und Empfänger für Radio und DVB.

Powertrain Komponenten für Motor und Getriebe. Kraftstoffpumpe, Einspritzpumpe, *Engine-Control*, Wechselstromgenerator und *Glow-Plug-Control* (Glühkerze).

ADAS (Advanced Driver Assistance Systems) Sensoren für Fahrzeugassistenzsysteme wie RADAR, LIDAR, Kameras und auch gerätespezifische Algorithmen für die Generation von Informationen aus Rohdaten (*perception*).

Telematics and Networking Netzwerk Infrakstruktur wie interne Gateways, V2X Module, und *Firmware Over-The-Air* (FOTA) Module.

Services Dienste für Flottenmanagement, *Tolling*, *Car-Sharing* und Notrufe.

Nach der Integration der Funktionen auf ein einzelnes Steuergerät, in Abschnitt 2.2.1 beschrieben, wäre der nächste Schritt, hin zu einer zentralisierten Architektur, und die Funktionen einer Domain alle auf ein einzelnes Steuergerät zu integrieren, der *Central-Domain ECU* (in Abbildung 2.2: *Tomorrow - Centralisation*).

Zone Die, durch die zentralisierung, sinkende Anzahl der ECUs in einer Domäne und neue Technologien wie autmotive Ethernet, ermöglichen eine andere Persepektive auf das Netzwerk, auf logischer Ebene. [29][30]

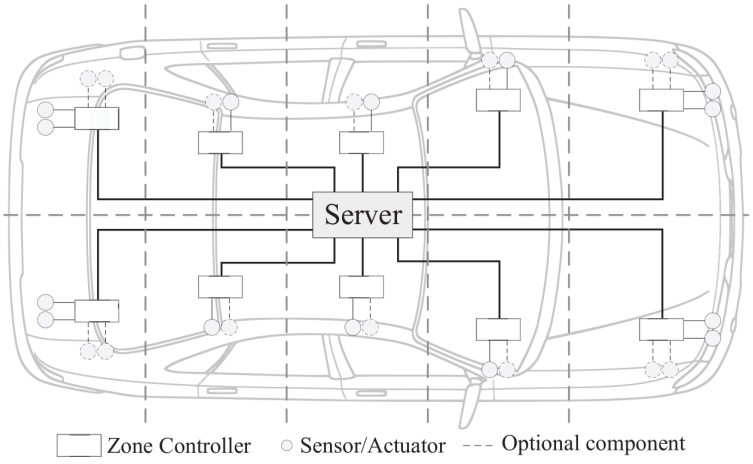


Abbildung 2.3: Autmotive E/E-Architektur als Zonen-Architektur [29]

Autmotive Netzwerkarchitekturen können so in Zonen untereilt, wie in Abbildung 2.3 dargestellt. Jede Zone enthält einen *Zone-Controller*, ein Layer-2 Gateway, welches an ein Backbone angeschlossen ist und die Kommunikation, zwischen den Zonen und verschiedenen Bussen einer Zone, ermöglicht.

2.2.2 Komponententypen

Controller Ein *Controller* beschreibt in diesem Dokument eine Art von Steuergerät welches ausschließlich dazu dient die Kommunikation anderer Netzwerkentitäten zu ermöglichen. Er bietet Funktionen wie *Switching*, *Forwarding*, *Encapsulation* und das Übersetzen von verschiedenen Protkollen (Gateway). Wenn nicht genauer spezifiziert bezieht sich *Controller* auf einen Zonen-Controller.

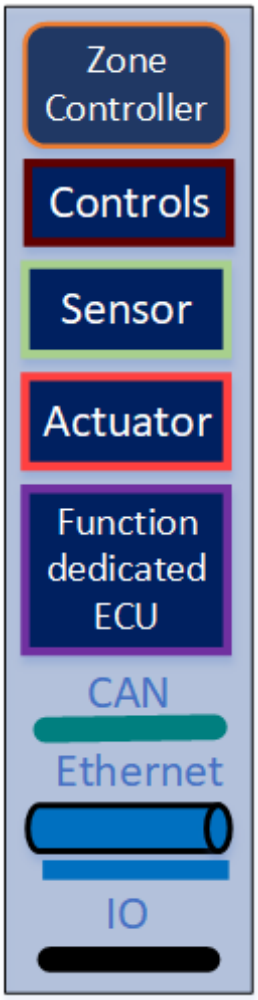
ECU Eine *ECU* beschreibt, wenn nicht weiter spezifiziert, ein Steuergerät auf welchem die Logik einer automotive Funktion implementiert ist. Zum Beispiel ABS, EPS, ESC, etc.

Sensor/Actuator Unter Sensoren und Aktuatoren werden Bauteile zusammengefasst welche keine Funktionen implementieren, sondern entweder Information aufnehmen und Daten produzieren oder die Änderung eines physikalischen Zustandes herbeiführen, wenn das Gerät angesteuert wird.

2.2.3 Kommunikation

Inter/Intra-Zone Inter-Zonen Kommunikation ist der Nachrichtenaustausch von Komponenten, welche sich in verschiedenen Zonen der Netzwerk-Architektur befinden (physikalisch örtliche Trennung). Intra-Zonen Kommunikation beschreibt den Nachrichtenaustausch von Komponenten in der selben Zone.

Inter/Intra-Bus Inter-Bus Kommunikation ist der Nachrichtenaustausch von Komponenten welche sich an verschiedenen Feldbussen befinden. Intra-Bus beschreibt die Kommunikation von Kompo-



Legende

nenten am selben Bus. Da in diesem Dokument eine Zonen-Architektur behandelt wird, gibt es keine physikalisch zonen-übergreifenden Busse. Es kann jedoch durchaus eine logische Zuordnung zwischen Bussen in verschiedenen Zonen geben.

I/O Sensoren und Aktuatoren können auch per serieller Schnittstelle direkt an ein Steuergerät angeschlossen werden. Diese Art der Anbindung wird hier unter *I/O* zusammengefasst.

2.3 verwendete Komponenten

In diesem Abschnitt werden die abgeleiteten Nachrichten der Komponenten, die in der Referenzarchitektur verwendet werden, aufgelistet. Das Reverse-Engineering des CAN-Busses, verschiedener Hersteller und Produkte, ist ein andauernder Prozess in der *Car-Hacking Community*, dessen Ergebnisse in Internetforen und Wikis veröffentlicht werden. Diese Quellen wurden, zusätzlich zu den referenzierten wissenschaftlichen Arbeiten, für die Definition der Nachrichten herangezogen. Diverse Aktoren können von verschiedenen Steuergeräten getrieben werden. Nachrichten eines Typs werden jedoch nur ein mal in der Tabelle aufgelistet, für mehrere mögliche Empfänger oder Sender wird '-' als Platzhalter verwendet. Auf welche der Nachricht agiert wird, ist abhängig von dessen Prioritäten. Da Priorität nur in bestimmten Fällen eine Rolle spielt wird nicht für jede Nachricht eine Priorität vergeben sondern nur in der Beschreibung der Komponente eine Anmerkung gemacht.

2.3.1 Wheel Speed Sensor (WSS)

Misst die Winkelgeschwindigkeit eines Rades.

Quelle	Ziel	Nachricht	Beschreibung	ID
WSS	-	Wheel Speed FL	Winkelgeschwindigkeit des linken Vorderades	WS1
WSS	-	Wheel Speed FR	Winkelgeschwindigkeit des rechten Vorderrades	WS2
WSS	-	Wheel Speed RL	Winkelgeschwindigkeit des linken Hinterrades	WS3
WSS	-	Wheel Speed RR	Winkelgeschwindigkeit des rechten Hinterrades	WS4

2.3.2 Steering Angle Sensor (SAS)

Misst den Winkel des Lenkrades.

Quelle	Ziel	Nachricht	Beschreibung	ID
SAS	-	Steering Wheel Angle	Lenkradposition	SWA1

2.3.3 Gyroscopic Sensor (GS)

Misst die Drehgeschwindigkeit um eine geometrische Achse.

Quelle	Ziel	Nachricht	Beschreibung	ID
GS	-	Vertical Rotation Speed	Drehgeschwindigkeit um zentrale vertikale Achse (Gierachse)	VRA

2.3.4 Brake

Übt den empfangenen Druck-Wert auf die Bremsscheibe eines Rades aus.

Quelle	Ziel	Nachricht	Beschreibung	ID
-	Brake FR	Brake Pressure FR	Bremsdruck auf das rechte Vorderrad	BP1
-	Brake FL	Brake Pressure FL	Bremsdruck auf das linke Vorderrad	BP2
-	Brake RR	Brake Pressure RR	Bremsdruck auf das rechte Hinterrad	BP3
-	Brake RL	Brake Pressure RL	Bremsdruck auf das linke Hinterrad	BP4

2.3.5 Brake Pedal

Misst den ausgeübten Druck auf das Bremspedal

Quelle	Ziel	Nachricht	Beschreibung	ID
Brake Pedal	-	Brake Pedal Pressure	Druck auf das Bremspedal	BPP1

2.3.6 ABS Module

Misst das ABS eine, im vergleich zu den anderen Rädern, zu niedrige Drehzahl, wird der Bremsdruck auf dieses Rad verringert.

Vom ABS ausgehende Nachrichten BP1-4 sind von höherer Priorität als die des EBB und ESC.

Quelle	Ziel	Nachricht	Beschreibung	ID
WSS	ABS	Wheel Speed FL	Winkelgeschwindigkeit des linken Vorderades	WS1
WSS	ABS	Wheel Speed FR	Winkelgeschwindigkeit des rechten Vorderrades	WS2
WSS	ABS	Wheel Speed RL	Winkelgeschwindigkeit des linken Hinterrades	WS3
WSS	ABS	Wheel Speed RR	Winkelgeschwindigkeit des rechten Hinterrades	WS4
-	ABS	Brake Pressure FR	Bremsdruck auf das rechte Vorderrad	BP1
-	ABS	Brake Pressure FL	Bremsdruck auf das linke Vorderrad	BP2
-	ABS	Brake Pressure RR	Bremsdruck auf das rechte Hinterrad	BP3
-	ABS	Brake Pressure RL	Bremsdruck auf das linke Hinterrad	BP4
ABS	Brake FR	Brake Pressure FR	Bremsdruck auf das rechte Vorderrad	BP1
ABS	Brake FL	Brake Pressure FL	Bremsdruck auf das linke Vorderrad	BP2
ABS	Brake RR	Brake Pressure RR	Bremsdruck auf das rechte Hinterrad	BP3
ABS	Brake RL	Brake Pressure RL	Bremsdruck auf das linke Hinterrad	BP4

2.3.7 EBB Module

Berechnet, ausgehend vom auf das Bremspedal ausgeübten Druck, den Druck auf die Bremsscheiben.

Quelle	Ziel	Nachricht	Beschreibung	ID
Brake Pedal	EBB	Brake Pedal Pressure	Druck auf das Bremspedal	BPP1
EBB	Brake FR	Brake Pressure FR	Bremsdruck auf das rechte Vorderrad	BP1
EBB	Brake FL	Brake Pressure FL	Bremsdruck auf das linke Vorderrad	BP2
EBB	Brake RR	Brake Pressure RR	Bremsdruck auf das rechte Hinterrad	BP3
EBB	Brake RL	Brake Pressure RL	Bremsdruck auf das linke Hinterrad	BP4

2.3.8 ESC Module

Weicht die Gewünschte Fahrtrichtung merklich von der Eigenbewegung ab (Über-/Untersteuern), bremst das ESC individuelle Räder ab um einem Kontrollverlust entegenzuwirken.

Vom ESC ausgehende Nachrichten BP1-4 sind von höherer Priorität als die des EBB und ESC.

Quelle	Ziel	Nachricht	Beschreibung	ID
WSS	ESC	Wheel Speed FL	Winkelgeschwindigkeit des linken Vorderades	WS1
WSS	ESC	Wheel Speed FR	Winkelgeschwindigkeit des rechten Vorderrades	WS2
WSS	ESC	Wheel Speed RL	Winkelgeschwindigkeit des linken Hinterrades	WS3
WSS	ESC	Wheel Speed RR	Winkelgeschwindigkeit des rechten Hinterrades	WS4
-	ESC	Brake Pressure FR	Bremsdruck auf das rechte Vorderrad	BP1
-	ESC	Brake Pressure FL	Bremsdruck auf das linke Vorderrad	BP2
-	ESC	Brake Pressure RR	Bremsdruck auf das rechte Hinterrad	BP3
-	ESC	Brake Pressure RL	Bremsdruck auf das linke Hinterrad	BP4
GS	ESC	Vertical Rotation Speed	Drehgeschwindigkeit um zentrale vertikale Achse (Gierachse)	VRA
SAS	ESC	Steering Wheel Angle	Lenkradposition	SWA1
ESC	Brake FR	Brake Pressure FR	Bremsdruck auf das rechte Vorderrad	BP1
ESC	Brake FL	Brake Pressure FL	Bremsdruck auf das linke Vorderrad	BP2
ESC	Brake RR	Brake Pressure RR	Bremsdruck auf das rechte Hinterrad	BP3
ESC	Brake RL	Brake Pressure RL	Bremsdruck auf das linke Hinterrad	BP4

2.4 Kategorisierung der Kommunikation

Um ein hohes Maß an Abdeckung sicherzustellen werden, für die Konzeption der Referanzarchitektur und dieß Herleitung der Anwendungsfälle, verschiedene Kategorien der Kommunikation unterschieden. Typen der Kommunikationspartner, verwendete Protokolle, und die Art der Anbindung an das Netzwerk sind die kategorisierten Merkmale.

2.4.1 Matrix

Die folgende Tabelle enthält eine Zuordnung der Komponeten und die möglichen Kommunikationstypen, wie sie in Abschnitt 2.2.2 definiert wurden.

Tabelle 2.2: Kommunikationsmatrix

	Controller	ECU	Sensor/Actuator
Controller	• Inter-Zone	• Inter-Zone • Intra-Zone	• Inter-Zone • Intra-Zone
ECU		• Inter-Zone • Intra-Zone • Inter-Bus • Intra-Bus	• Inter-Zone • Intra-Zone • Inter-Bus • Intra-Bus
Sensor/Actuator			-

Die übersicht der Kommunikation in Tabelle 2.2 wird nun durch den vollständigen Kommunikationspfad und die verwendeten Protokolle erweitert.

2.4.2 Pfade

1. Inter-Zone Controller to Controller
   1.1. Controller $\xrightarrow{\text{Ethernet}}$ Controller
2. Inter-Zone ECU to Controller
   2.1. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller
   2.2. ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller
3. Intra-Zone ECU to Controller
   3.1. ECU $\xrightarrow{\text{CAN}}$ Controller
   3.2. ECU $\xrightarrow{\text{Ethernet}}$ Controller
4. Inter-Zone Sensor/Actuator to Controller
   4.1. Sensor/Actuator $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller
   4.2. Sensor/Actuator $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller
   4.3. Sensor/Actuator $\xrightarrow{\text{IO}}$ ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller
   4.4. Sensor/Actuator $\xrightarrow{\text{CAN}}$ ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller
5. Intra-Zone Sensor/Actuator to Controller
   5.1. Sensor/Actuator $\xrightarrow{\text{CAN}}$ Controller
   5.2. Sensor/Actuator $\xrightarrow{\text{IO}}$ Controller
6. Inter-Zone ECU to ECU
   6.1. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
   6.2. ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
   6.3. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ ECU

6.4. ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ ECU

7. Intra-Zone Inter-Bus ECU to ECU
   7.1. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
   7.2. ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
   7.3. ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ ECU
   7.4. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ ECU
8. Intra-Zone Intra-Bus ECU to ECU
   8.1. ECU $\xrightarrow{\text{CAN}}$ ECU
   8.2. ECU $\xrightarrow{\text{Ethernet}}$ ECU
9. Inter-Zone Sensor/Actuator to ECU
   9.1. Sensor/Actuator $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
   9.2. Sensor/Actuator $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ ECU
   9.3. Sensor/Actuator $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
   9.4. Sensor/Actuator $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ ECU
10. Intra-Zone Inter-Bus Sensor/Actuator to ECU
    10.1. Sensor/Actuator $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
    10.2. Sensor/Actuator $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{CAN}}$ ECU
    10.3. Sensor/Actuator $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ ECU
11. Intra-Zone Intra-Bus Sensor/Actuator to ECU
    11.1. Sensor/Actuator $\xrightarrow{\text{CAN}}$ ECU
12. Inter-Zone ECU to Sensor/Actuator
    12.1. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ Sensor/Actuator
    12.2. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Sensor/Actuator
    12.3. ECU $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ Sensor/Actuator
    12.4. ECU $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Sensor/Actuator
    12.5. ECU $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow{\text{CAN}}$ Sensor/Actuator
13. Intra-Zone Inter-Bus ECU to Sensor/Actuator
    13.1. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{CAN}}$ Sensor/Actuator
    13.2. ECU $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{CAN}}$ Sensor/Actuator
    13.3. ECU $\xrightarrow{\text{CAN}}$ Controller $\xrightarrow{\text{Ethernet}}$ Sensor/Actuator
    13.4. ECU $\xrightarrow{\text{IO}}$ Controller $\xrightarrow{\text{Ethernet}}$ Sensor/Actuator
14. Intra-Zone Intra-Bus ECU to Sensor/Actuator
    14.1. ECU $\xrightarrow{\text{CAN}}$ Sensor/Actuator

Controller sind für die Anwendungsschicht transparent und sind nur zuständig für die Kommunikation der funktionalen Komponenten. Deshalb werden Pfade die von einem Controller ausgehen, oder an einem Controller enden außenvor gelassen. Pfade, die relevant sind, sind in der oberen Auflistungen also Pfade der Kategorien 6-14. Alle Pfade der Kategorien 1-5, sind Teilpfade dieser.

2.5 Grundgerüst der Referenzarchitektur

Die Referezarchitektur wird in die Sechzehn, in Abbildung 2.5 zu sehenden, Zonen unterteilt. Jede Zone enthält einen *Zone-Controller*, wie er in Abschnitt 2.2.1 beschrieben ist. Im folgenden Kapitel 3 werden Anwendungsfälle, der in diesem Kapitel besprochenen, Komponenten vorgestellt. Diese Komponenten werden dann iterativ dem hier dargestellten Schema hinzugefügt.

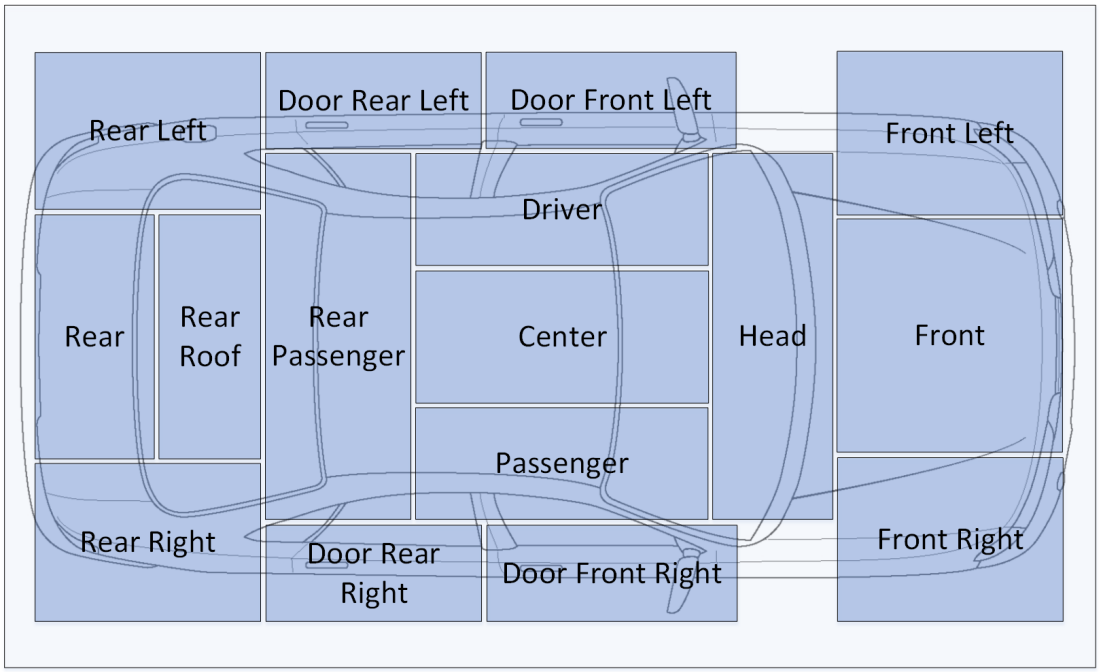


Abbildung 2.5: Zonen der Referenzarchitektur

3 Use Cases

3.1 Bremssystem

In diesem Szenario wird die Kommunikation der Komponenten des Bremssystems für verschiedene Szenarien untersucht. Hierfür sind beteiligte Komponenten in das Schema der Referenzarchitektur hinzugefügt worden. 3.3.

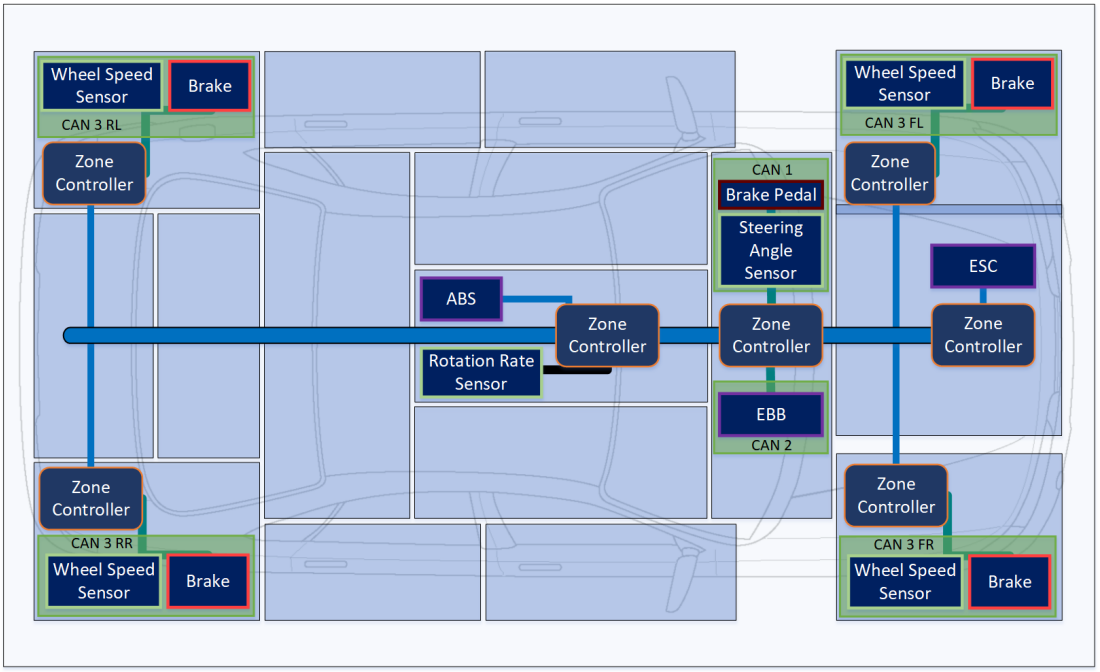


Abbildung 3.1: Komponenten des Bremssystems

3.1.1 Normales Bremsen

Nachrichten des EBB werden stets, unabhängig von ESC und ABS, gesendet. Diese werden durch Nachrichten von höher Priorität 'überschrieben'.

- Brake Pedal $\xrightarrow[BPP1]{\text{CAN1}}$ Controller $\xrightarrow[BPP1]{\text{CAN2}}$ EBB (Pfad 10.1)

- EBB $\xrightarrow[BP1]{\text{CAN2}}$ Controller $\xrightarrow[BP1]{\text{Ethernet}}$ Controller $\xrightarrow[BP1]{\text{CAN3 FR}}$ Brake FR (Pfad 12.1)
- EBB $\xrightarrow[BP2]{\text{CAN2}}$ Controller $\xrightarrow[BP2]{\text{Ethernet}}$ Controller $\xrightarrow[BP2]{\text{CAN3 FL}}$ Brake FL (Pfad 12.1)
- EBB $\xrightarrow[BP3]{\text{CAN2}}$ Controller $\xrightarrow[BP3]{\text{Ethernet}}$ Controller $\xrightarrow[BP3]{\text{CAN3 RR}}$ Brake RR (Pfad 12.1)
- EBB $\xrightarrow[BP4]{\text{CAN2}}$ Controller $\xrightarrow[BP4]{\text{Ethernet}}$ Controller $\xrightarrow[BP4]{\text{CAN3 RL}}$ Brake RL (Pfad 12.1)

- **Authentication** Die Authentizität der kommunizierenden Steuergeräte muss gewährleistet sein, um sicherzustellen, dass keine und unauthorisierten Geräte einen Bremsvorgang ausführen.

- **Integrity** Die Integrität der Nachrichten muss gewährleistet sein, um sicherzustellen, dass auf dem Kommunikationspfad keine bösartige manipulation der Bremskraftwerte erfolgt ist, die das Fahrverhalten beeinflussen könnte.

- **Freshness** Die Aktualität der Nachrichten muss gewährleistet sein, um ein Replay zu verhindern, welches durch aktivierung der Bremsen das Fahrverhalten beeinflussen könnte.

3.1.2 ESC

Auf den folgendenen drei Pfaden werden kontinuierlich Nachrichten an das ESC Modul versendet.

- Steering Angle Sensor $\xrightarrow[SWA1]{\text{CAN1}}$ Controller $\xrightarrow[SWA1]{\text{Ethernet}}$ Controller $\xrightarrow[SWA1]{\text{Ethernet}}$ ESC (Pfad 9.2)
- Wheel Speed Sensor $\xrightarrow[WS1-4]{\text{CAN3}}$ Controller $\xrightarrow[WS1-4]{\text{Ethernet}}$ Controller $\xrightarrow[WS1-4]{\text{Ethernet}}$ ESC (Pfad 9.2)
- Gyroscopic Sensor $\xrightarrow[VRA]{\text{IO}}$ Controller $\xrightarrow[VRA]{\text{Ethernet}}$ Controller $\xrightarrow[VRA]{\text{Ethernet}}$ ESC (Pfad 9.4)

Erkennt das ESC ein Unter- oder Übersteuern wird der Bremsdruck auf bestimmte Räder erhöht.

- EBB $\xrightarrow[BP1-4]{\text{CAN2}}$ Controller $\xrightarrow[BP1-4]{\text{Ethernet}}$ Controller $\xrightarrow[BP1-4]{\text{Ethernet}}$ ESC (Pfad 6.3)
- ESC $\xrightarrow[BP1]{\text{Ethernet}}$ Controller $\xrightarrow[BP1]{\text{Ethernet}}$ Controller $\xrightarrow[BP1]{\text{CAN3 FR}}$ Brake FR (Pfad 12.5)
- ESC $\xrightarrow[BP2]{\text{Ethernet}}$ Controller $\xrightarrow[BP2]{\text{Ethernet}}$ Controller $\xrightarrow[BP2]{\text{CAN3 FL}}$ Brake FL (Pfad 12.5)
- ESC $\xrightarrow[BP3]{\text{Ethernet}}$ Controller $\xrightarrow[BP3]{\text{Ethernet}}$ Controller $\xrightarrow[BP3]{\text{CAN3 RR}}$ Brake RR (Pfad 12.5)
- ESC $\xrightarrow[BP4]{\text{Ethernet}}$ Controller $\xrightarrow[BP4]{\text{Ethernet}}$ Controller $\xrightarrow[BP4]{\text{CAN3 RL}}$ Brake RL (Pfad 12.5)

- **Authentication**
- **Integrity**
- **Freshness**

3.1.3 ABS

Auf den folgendenen drei Pfaden werden kontinuierlich Nachrichten an das ABS Modul versendet.

- Wheel Speed Sensor $\xrightarrow[WS1-4]{\text{CAN3}}$ Controller $\xrightarrow[WS1-4]{\text{Ethernet}}$ Controller $\xrightarrow[WS1-4]{\text{Ethernet}}$ ABS (Pfad 9.2)
- EBB $\xrightarrow[BP1-4]{\text{CAN2}}$ Controller $\xrightarrow[BP1-4]{\text{Ethernet}}$ Controller $\xrightarrow[BP1-4]{\text{Ethernet}}$ ABS (Pfad 6.3)

Erkennt das ABS eine zu geringe Drehzahl wird der Bremsdruck auf bestimmte Räder verringert.

- EBB $\xrightarrow[BP1-4]{\text{CAN2}}$ Controller $\xrightarrow[BP1-4]{\text{Ethernet}}$ Controller $\xrightarrow[BP1-4]{\text{Ethernet}}$ ABS (Pfad 6.3)
- ESC $\xrightarrow[BP1]{\text{Ethernet}}$ Controller $\xrightarrow[BP1]{\text{Ethernet}}$ Controller $\xrightarrow[BP1]{\text{CAN3 FR}}$ Brake FR (Pfad 12.5)
- ESC $\xrightarrow[BP2]{\text{Ethernet}}$ Controller $\xrightarrow[BP2]{\text{Ethernet}}$ Controller $\xrightarrow[BP2]{\text{CAN3 FL}}$ Brake FL (Pfad 12.5)
- ESC $\xrightarrow[BP3]{\text{Ethernet}}$ Controller $\xrightarrow[BP3]{\text{Ethernet}}$ Controller $\xrightarrow[BP3]{\text{CAN3 RR}}$ Brake RR (Pfad 12.5)
- ESC $\xrightarrow[BP4]{\text{Ethernet}}$ Controller $\xrightarrow[BP4]{\text{Ethernet}}$ Controller $\xrightarrow[BP4]{\text{CAN3 RL}}$ Brake RL (Pfad 12.5)
- **Authentication**
- **Integrity**
- **Freshness**

3.2 Body Control

3.2.1 NFC unlock car

- Smartphone $\xrightarrow[\text{UL}]{\text{NFC}}$ NFC Reciever $\xrightarrow{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{Ethernet}}$ NFC Module
- NFC Module $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{CAN3 FR}}$ Body Domain Master

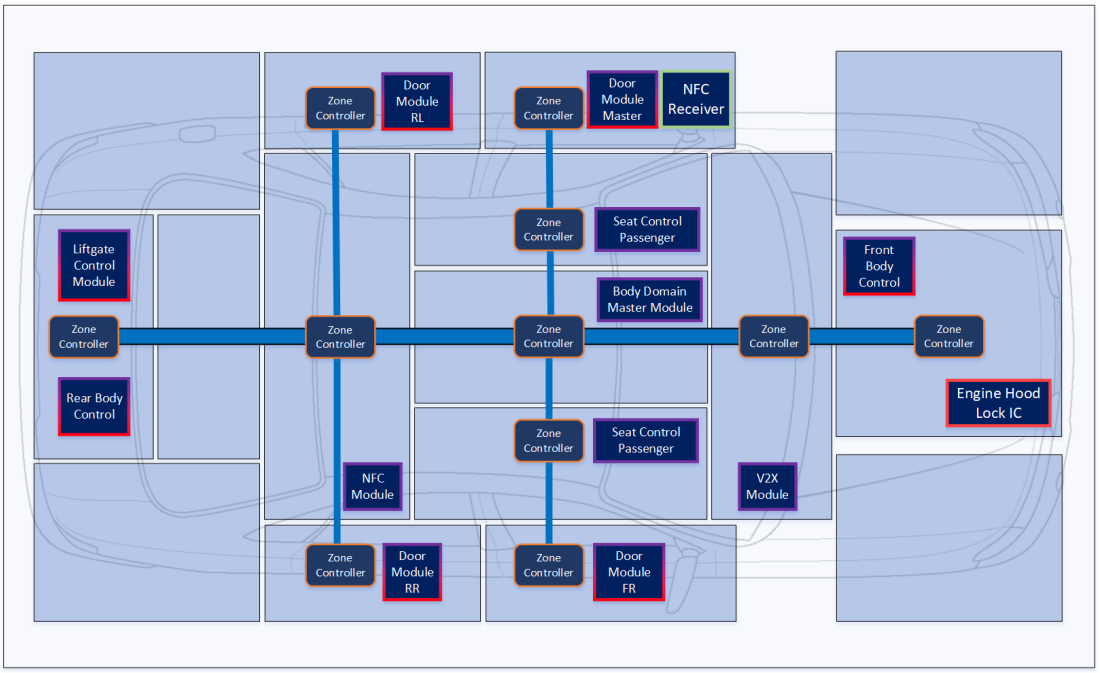


Abbildung 3.2: Komponenten der für Body Domain Anwendungsfälle

- Body Domain Master $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{CAN3 FL}}$ Door Module Master
- Body Domain Master $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{CAN3 FL}}$ Door Module FR
- Body Domain Master $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{CAN3 FL}}$ Door Module RR
- Body Domain Master $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{UL}]{\text{CAN3 FL}}$ Door Module RL
- **Authentication**
- **Integrity**
- **Freshness**

3.2.2 Radio Unlock trunk

- Car Key Radio Transmitter $\xrightarrow[\text{ULT}]{\text{Radio}}$ V2X Module $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Body Domain Master
- Body Domain Master $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{CAN3 FL}}$ Rear Body Control

- Rear Body Control $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{CAN3 FL}}$ Liftgate Control Module
- **Authentication**
- **Integrity**
- **Freshness**

3.2.3 Unlock engine hood

- Center Controls $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Body Domain Master
- Body Domain Master $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{CAN3 FL}}$ Front Body Control
- Front Body Control $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{Ethernet}}$ Controller $\xrightarrow[\text{ULT}]{\text{CAN3 FL}}$ Engine Hood Lock IC
- **Authentication**
- **Integrity**
- **Freshness**

3.3 Fahrerassistenzsysteme

3.3.1 Lane Keep Assist

- LIDAR $\xrightarrow[LRAW]{\text{Ethernet}}$ Controller $\xrightarrow[LRAW]{\text{Ethernet}}$ Controller $\xrightarrow[LRAW]{\text{Ethernet}}$ Perception
- RADAR $\xrightarrow[RRAW]{\text{Ethernet}}$ Controller $\xrightarrow[RRAW]{\text{Ethernet}}$ Controller $\xrightarrow[RRAW]{\text{Ethernet}}$ Perception
- Camera $\xrightarrow[CRAW]{\text{Ethernet}}$ Controller $\xrightarrow[CRAW]{\text{Ethernet}}$ Controller $\xrightarrow[CRAW]{\text{Ethernet}}$ Perception
- Perception $\xrightarrow[LP]{\text{Ethernet}}$ Controller $\xrightarrow[LP]{\text{Ethernet}}$ Controller $\xrightarrow[LP]{\text{CAN3 FL}}$ Lane Keep Assist
- Ego Motion $\xrightarrow[VS]{\text{Ethernet}}$ Controller $\xrightarrow[VS]{\text{Ethernet}}$ Controller $\xrightarrow[VS]{\text{CAN3 FL}}$ Lane Keep Assist
- Lane Keep Assist $\xrightarrow[WPD]{\text{Ethernet}}$ Controller $\xrightarrow[WPD]{\text{Ethernet}}$ Controller $\xrightarrow[WPD]{\text{CAN3 FL}}$ Steering Wheel
- Lane Keep Assist $\xrightarrow[WT]{\text{Ethernet}}$ Controller $\xrightarrow[WT]{\text{Ethernet}}$ Controller $\xrightarrow[WT]{\text{CAN3 FL}}$ EPS

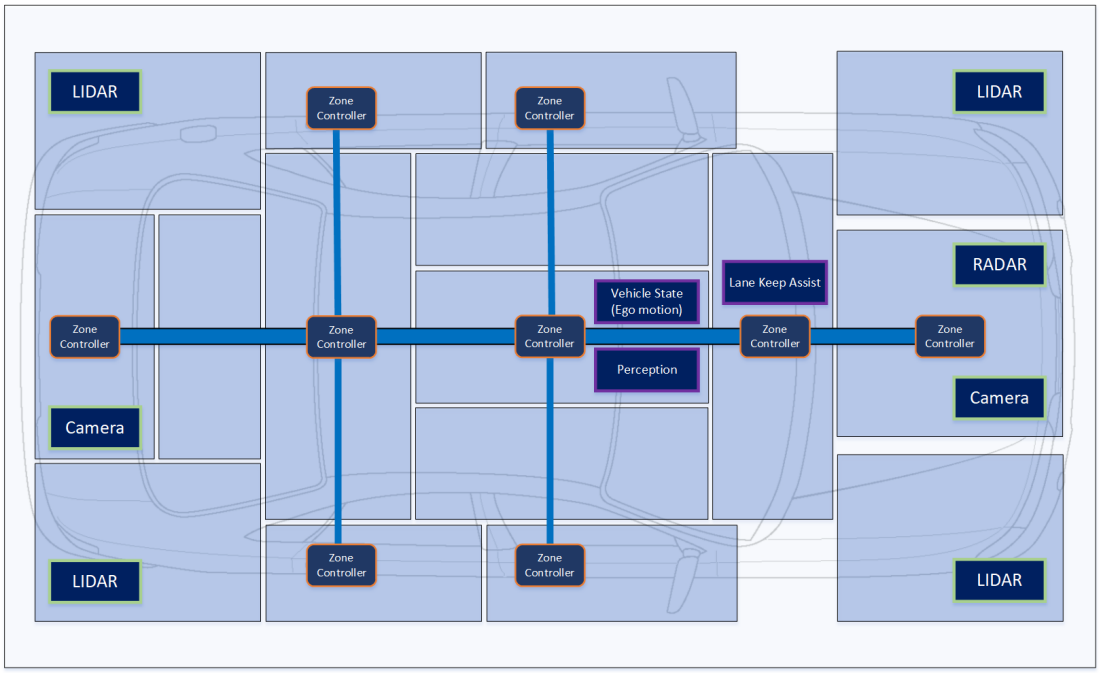


Abbildung 3.3: Komponenten des Bremssystems

Literatur

[1] B. Weyl, M. Wolf, F. Zweers, T. Gendrullis, M. S. Idrees, Y. Roudier, H. Schweppe, H. Platzdasch, R. E. Khayari, O. Henniger, D. Scheuermann, A. Fuchs, L. Apvrille, G. Pedroza, H. Seudié, J. Shokrollahi und A. Keil, "Secure on-board architecture specification", EVITA project, EVITA Deliverable D3.2, Aug. 2011.

[2] G. Pedroza, M. S. Idrees, L. Apvrille und Y. Roudier, "A formal methodology applied to secure over-the-air automotive applications", in *2011 IEEE Vehicular Technology Conference (VTC Fall)*, Sep. 2011, S. 1–5. DOI: 10.1109/VETECF.2011.6093061.

[3] A. Fuchs, S. Gürgens und C. Rudolph, "Formal notions of trust and confidentiality-enabling reasoning about system security", *Journal of Information Processing*, Jg. 19, S. 274–291, 2011. DOI: 10.2197/ipsjjip.19.274.

[4] M. S. Idrees, Y. Roudier und L. Apvrille, "A framework towards the efficient identification and modeling of security requirements", in *SAR-SSI 2010, 5th Conference on Network Architectures and Information Systems Security, May 18-21, 2010, Menton, France*, M, Mai 2010.

[5] T. Kiravuo, M. Sarela und J. Manner, "A Survey of Ethernet LAN Security", *IEEE Communications Surveys Tutorials*, Jg. 15, Nr. 3, S. 1477–1491, März 2013, ISSN: 1553-877X. DOI: 10.1109/SURV.2012.121112.00190.

[6] *IEEE 802.3bw-2015 - IEEE Standard for Ethernet amendment 1: Physical Layer Specifications and management parameters for 100 mb/s operation over a single balanced twisted pair cable (100BASE-T1)*, März 2016.

[7] D. Porter, "100base-t1 ethernet: The evolution of automotive networking", Texas Instruments, Techn. Ber., Apr. 2018.

[8] P. Hank, S. Müller, O. Vermesan und J. Van Den Keybus, "Automotive ethernet: In-vehicle networking and smart mobility", in *Proceedings of the Conference on Design, Automation and Test in Europe*, Ser. DATE '13, Grenoble, France: EDA Consortium, 2013, S. 1735–1739, ISBN: 978-1-4503-2153-2.

[9] I. Cooperation, "Automotive ethernet: An overview", ixia, Techn. Ber., Mai 2014.

[10] H. Fan, Y. Dong, M. Yu und L. Tung, "Security threats against the communication networks for traffic control systems", in *2013 IEEE International Conference on Systems, Man, and Cybernetics*, Okt. 2013, S. 4783–4788. DOI: 10.1109/SMC.2013.814.

[11] D. Wampler, H. Fu und Y. Zhu, "Security threats and countermeasures for intra-vehicle networks", in *2009 Fifth International Conference on Information Assurance and Security*, Bd. 2, Aug. 2009, S. 153–157. DOI: 10.1109/IAS.2009.350.

[12] R. Currie, "Hacking the can bus: Basic manipulation of a modern automobile through can bus reverse engineering", SANS Institute, Mai 2017.

[13] C. Smith, *The Car Hacker's Handbook: A Guide for the Penetration Tester*, 1st. San Francisco, CA, USA: No Starch Press, 2016, ISBN: 1593277032, 9781593277031.

[14] T. Huybrechts, Y. Vanommeslaeghe, D. Blontrock, G. Van Barel und P. Hellinckx, "Automatic Reverse Engineering of CAN Bus Data Using Machine Learning Techniques", Jan. 2018, S. 751–761, ISBN: 978-3-319-69834-2. DOI: 10.1007/978-3-319-69835-9_71.

[15] T. Strang und M. Roeckl, *Vehicle Networks: CAN-based Higher Layer Protocols*, https://www.sti-innsbruck.at/sites/default/files/courses/fileadmin/documents/vn-ws0809/03-vn-CAN-HLP.pdf, 2009.

[16] AUTOSAR, "Specification of CAN Transport Layer", Techn. Ber., Dez. 2017.

[17] ——, "Specification of TCP/IP Stack", Techn. Ber., Dez. 2017.

[18] ——, "Requirements on Ethernet Support in AUTOSAR", Techn. Ber., Dez. 2017.

[19] P. Werner, A. Happel, R. Fritz und S. Keul, *AUTOSAR Security Modules*, https://assets.vector.com/cms/content/know-how/automotive-cyber-security/AUTOSAR_Security_Modules_Lecture_ESCAR_2015.pdf, 2015.

[20] AUTOSAR, "Requirements on Secure Onboard Communication", Techn. Ber., Okt. 2018.

[21] ——, "Specification of Crypto Service Manager", Techn. Ber., Okt. 2018.

[22] ——, "Specification of Secure Onboard Communication", Techn. Ber., Dez. 2017.

[23] Q. Hu und F. Luo, "Review of secure communication approaches for in-vehicle network", *International Journal of Automotive Technology*, Jg. 19, Nr. 5, S. 879–894, Okt. 2018, ISSN: 1976-3832. DOI: 10.1007/s12239-018-0085-1.

[24] A. Ruddle, D. Ward, B. Weyl, S. Idrees, Y. Roudier, M. Friedewald, T. Leimbach, A. Fuchs, S. Gürgens, O. Henniger, R. Rieke, M. Ritscher, H. Broberg, L. Apvrille, R. Pacalet und G. Pedroza, "Security requirements for automotive on-board networks based on dark-side scenarios", EVITA project, EVITA Deliverable D2.3, Dez. 2009.

[25] O. Henniger, L. Apvrille, A. Fuchs, Y. Roudier, A. Ruddle und B. Weyl, "Security requirements for automotive on-board networks", in *2009 9th International Conference on Intelligent Transport Systems Telecommunications, (ITST)*, Okt. 2009, S. 641–646. DOI: 10.1109/ITST.2009.5399279.

[26] Robert Bosch GmbH. (8. März 2017). E/E-Architecture in a connected world, ASAM, Adresse: https://www.asam.net/index.php?eID=dumpFile&t=f&f=798&token=148b5052945a466cacfe8f31c44eb22509d5aad1 (besucht am 27. 09. 2018).

[27] M. Wille und O. Krieger, *Ethernet & Adaptive AUTOSAR key elements of the new Volkswagen E/E architecture.*

[28] D. Reinhardt und M. Kucera, "Domain Controlled Architecture - A New Approach for Large Scale Software Integrated Automotive Systems", in *PECCS*, 2013.

[29] S. Brunner, J. Roder, M. Kucera und T. Waas, "Automotive E/E-architecture enhancements by usage of ethernet TSN", in *2017 13th Workshop on Intelligent Solutions in Embedded Systems (WISES)*, Juni 2017, S. 9–13. DOI: 10.1109/WISES.2017.7986925.

[30] M. Ziehensack und M. Kunz, "Smart ethernet switch architecture", 2017 IEEE Standards Association (IEEE-SA) Ethernet & IP @ Automotive Technology Day, Techn. Ber., 2017.