A Security Testing Framework for the Automotive SOME/IP Protocol

Christoph Schmid

Master Thesis
VS-2020-02M

Begutachtung
Prof. Dr. rer. nat. Frank Kargl (Universität Ulm)
Prof. Dr. Ing. Franz Hauck (Universität Ulm)

Betreuung
Dr. rer. nat. Bastian Könings (SCHUTZWERK GmbH)
Dr. rer. nat. Henning Kopp (SCHUTZWERK GmbH)
M. Sc. Matthias Matousek (Universität Ulm)

Institut für Verteilte Systeme
Fakultät Ingenieurwissenschaften, Informatik und Psychologie
Universität Ulm

7. Dezember 2020

© 2020 Christoph Schmid

Fassung vom: 7. Dezember 2020

Dieses Werk ist unter einer Creative Commons Lizenz vom Typ Namensnennung zugänglich.

Um eine Kopie dieser Lizenz einzusehen, konsultieren Sie https://creativecommons.org/licenses/by/4.0/deed.de oder wenden Sie sich brieflich an Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Titelbild [21]: pixabay. Gray Scale Photo of Gears. 2016 URL https://www.pexels.com/photo/gray-scale-photo-of-gears-159298/
License: CCo 1.0, https://creativecommons.org/publicdomain/zero/1.0/

Hiermit versichere ich, dass ich die vorliegende Arbeit mit dem Titel

A Security Testing Framework
for the Automotive SOME/IP Protocol

bis auf die offizielle Betreuung, selbstständig und ohne fremde Hilfe angefertigt habe und die benutzten Quellen und Hilfsmittel vollständig angegeben sind. Aus fremden Quellen direkt oder indirekt übernommene Gedanken sind jeweils unter Angabe der Quelle als solche kenntlich gemacht.
Ich erkläre außerdem, dass die vorliegende Arbeit entsprechend den Grundsätzen guten wissenschaftlichen Arbeitens gemäß der „Satzung der Universität Ulm zur Sicherung guter wissenschaftlicher Praxis“ erstellt wurde.

Ulm, den 7. Dezember 2020

Christoph Schmid

ZUSAMMENFASSUNG

SOME/IP ist ein Netzwerkprotokoll, das dazu genutzt wird, um Informationen zwischen Steuergeräten, insbesondere in Fahrzeugen, auszutauschen. Für die Sicherheit der Verkehrsteilnehmer ist es sehr wichtig, dass diese Steuergeräte nicht von außen manipuliert werden können. Um dies zu erreichen, sind Sicherheitsanalysen und Penetrationstests während der Produktentwicklung wichtige Maßnahmen. Um dies zu erreichen, wurde ein Framework entwickelt, mit dem die Implementierung von SOME/IP auf Steuergeräten auf deren Sicherheit überprüft werden kann. Dazu wurden die möglichen Angriffsvektoren auf deren Brisanz untersucht. Hierzu wurden verschiedene Angriffsvektoren erarbeitet und klassifiziert, die bei der Entwicklung berücksichtigt wurden. Das Framework selbst wurde modular aufgebaut, wobei jeder Angriff einem Modul entspricht. Da im Laufe der Arbeit festgestellt wurde, dass herstellerspezifische Anpassungen an SOME/IP genutzt werden, wurde ein Modul entwickelt, das diese Funktionalität bereitstellt. Mit den implementierten Angriffsmodulen lassen sich Informationen über das SOME/IP-Netzwerk auslesen und Events abonnieren und deabonnieren. Das aktive Angriffsmodul wurde in einem Netzwerk mit einem Steuergerät an einer Restbussimulation getestet, während für das Information Gathering Modul eine qualitative Studie mit Sicherheitstestern durchgeführt wurde. Ergebnis dieser Studie war, dass das Framework bereits unterstützend für Sicherheitsanalysen genutzt werden kann. Mit dem entwickelten Framework konnten mehrere Angriffe praktisch durchgeführt werden. Durch diese Arbeit ist es nun möglich, Steuergeräte sicher zu entwickeln und so für mehr Sicherheit im Verkehr zu sorgen. Durch den modularen Aufbau können neu gefundene Angriffsvektoren und auch zukünftig entwickelte, auf SOME/IP basierende Protokolle getestet werden.

DANKSAGUNG

Ich bedanke mich bei Allen, die mich während meines Studiums, insbesondere beim Anfertigen dieser Arbeit, unterstützt haben. Im Besonderen bedanke ich mich bei meinen Betreuern Dr. Bastian Könings und Dr. Henning Kopp von der SCHUTZWERK GmbH und Prof. Dr. Frank Kargl und Matthias Matousek aus dem Institut für Verteilte Systeme der Universität Ulm für ihre Unterstützung, Hilfe und Geduld. Des Weiteren bedanke ich mich bei der SCHUTZWERK GmbH, die diese Masterarbeit erst möglich gemacht hat, sowie allen ihren Mitarbeitern , die mich durch hilfreiche Tipps und der Teilnahme an der Studie unterstützt haben. Zum Schluss bedanke ich mich bei meiner Familie und im Besonderen bei meiner Ehefrau durch deren Beistand und Halt ich den Abschluss dieses Studiums erreichen konnte. Entschuldigen möchte ich mich bei meiner Tochter Ronja: Papa ist jetzt wieder mehr daheim.

INHALTSVERZEICHNIS

Zusammenfassung **v**

Inhaltsverzeichnis **ix**

1 Einleitung **1**
- 1.1 Forschungsgebiet . . . 1
- 1.2 Bereich der Arbeit . . . 2
- 1.3 Verwandte Forschung . . . 2
- 1.4 Ziel der Arbeit . . . 3
- 1.5 Aufbau der Arbeit . . . 4

2 Technische Grundlagen **5**
- 2.1 Das ISO-OSI Netzwerkschichtenmodell . . . 5
- 2.2 Historische Entwicklung der internen Vernetzung von Automobilen . . . 7
- 2.3 Aufbau von Controller Area Network (CAN) in Automobilen . 7
- 2.4 Anforderungen an Netzwerksysteme in Automobilen . . . 7
- 2.5 Autonome Fahrzeuge . . . 8
- 2.6 Testverfahren . . . 9
- 2.7 Informationen über zu testende Komponenten . . . 10
- 2.8 Verfügbarkeit von Hard- und Software . . . 11
- 2.9 Verfügbare Netzwerkbibliotheken . . . 11

3 SOME/IP **13**
- 3.1 Anwendungszweck als Middleware . . . 13
- 3.2 Services . . . 13
  - 3.2.1 Events . . . 14
  - 3.2.2 RPC-Aufruf von Methoden . . . 14
  - 3.2.3 Setter- und Getter-Methoden von Feldern . . . 15
- 3.3 Gateways . . . 15
- 3.4 SOME/IP Frames . . . 16
  - 3.4.1 Message ID . . . 16
  - 3.4.2 Länge . . . 17
  - 3.4.3 Request ID . . . 17
  - 3.4.4 Weitere Datenfelder . . . 17
- 3.5 Service Discovery . . . 18
- 3.6 Sicherheitsfunktionen für SOME/IP . . . 19
- 3.7 Zusammenfassung . . . 19

4 Sicherheitsanalyse **21**
- 4.1 Methodik . . . 21
  - 4.1.1 Ziele der Angriffe . . . 21
  - 4.1.2 Risikoanalyse mit der HEAVENS-Methode . . . 22
  - 4.1.3 Feedbackkanal . . . 25
- 4.2 Angriffsvektoren . . . 26

4.2.1 SOME/IP Header 27
4.2.2 Events 28
4.2.3 Datenfelder 34
4.2.4 Remote Procedure Call (RPC) 36
4.2.5 Service Discovery 41
4.2.6 Kombinierte Angriffe 45
4.3 Zusammenfassung der Angriffsvektoren 47
4.4 Zu implementierende Module 47

5 Security Testing Framework für SOME/IP 51
5.1 Anforderungen an das Framework 51
5.1.1 Anforderungen Modularität 51
5.1.2 Anforderungen in die Module 52
5.1.3 Anforderungen an die Netzwerkbibliothek 52
5.2 Implementierung 54
5.2.1 Aufbau der Implementierung 55
5.2.2 Analyse der Netzwerkpakete mit scapy 55
5.2.3 In-Memory Datenbank mit sqlite 56
5.2.4 Modularität mit Yapsy 57
5.2.5 GUI-Implementierung mit PyQt5 57
5.3 Implementierte Module 57
5.3.1 Information Gathering Modul 58
5.3.2 Subscribe Modul 58
5.3.3 Paket Modul für von der Spezifikation abweichende SOME/IP Pakete 59
5.4 Zusammenfassung 59

6 Evaluation 65
6.1 Evaluation des Information Gathering Moduls 65
6.1.1 Benotung der Werkzeuge durch die Probanden 66
6.1.2 Abweichung der Benotung in Abhängigkeit der Erfahrung 67
6.1.3 Zeitaufwand der Bearbeitung der Übungsaufgaben . . 69
6.2 Evaluation aktiver Angriffsmodule 71
6.3 Evaluationsergebnisse 71

7 Fazit 73
7.1 Zusammenfassung 73
7.2 Ausblick 75

A Appendix 77

Literatur 85

1 EINLEITUNG

Das Auto ist aus dem heutigen Leben in Deutschland als Fortbewegungs- und Transportmittel nicht mehr wegzudenken. Immer mehr Menschen besitzen ein Auto und nutzen dies täglich um zum Beispiel zur Arbeit zu gelangen. Um die Fortbewegung für die Fahrer angenehmer gestalten zu können, werden zu den Fahrzeugen immer mehr Funktionen hinzugefügt. Dabei handelt es sich nicht nur um unterstützende Automatisierungen, die dem Fahrer Aufgaben beim Bewegen des Autos erleichtern sollen, wie das Einparken oder das Wechseln einer Spur auf einer Autobahn, sondern auch um multimediale Funktionen, wie das Hören von Musik über das Internet.

Um zu verhindern, dass solche Funktionen angegriffen werden können, müssen diese zuvor getestet werden. Da diese Fortbewegungsmittel heute eine solch große Komplexität erreicht haben, müssen die Sicherheitsaspekte die überprüft werden, in viele Teilaufgaben zerlegt und getrennt voneinander betrachtet werden. So kann sichergestellt werden, dass das Komplettsystem, das aus sicheren Einzelteilen erstellt wurde, ebenfalls keine Angriffsziele bietet. Diese Arbeit beschäftigt sich daher mit dem Erstellen eines Frameworks, das solche Tests in einem Teilbereich der Informationsverarbeitung in einem Fahrzeug, dem Netzwerkprotokoll SOME/IP ermöglicht.

Um Angriffe auf Fahrzeuge ausüben zu können, müssen die Angreifer heute nicht mehr direkten Zugriff auf das Auto haben. Da die Fahrzeuge über kabellose Systeme wie Bluetooth und Mobilfunk mit anderen Fahrzeugen, Smartphones und Servern verbunden sind, können über diese Zugänge Angreifer eindringen. Ist einem Angreifer dies gelungen, muss ein weiteres Vordringen ins Fahrzeugnetzwerk durch die Absicherung der einzelnen Steuergeräte vor Fremdzugriff verhindert werden.

1.1 Forschungsgebiet

Die Automobilindustrie erlebt derzeit einen Wendepunkt. Immer mehr eingebaute Sensoren wie Kameras, Radar oder Laser, übertragen immer mehr Informationen in das Auto, die für Assistenzsysteme genutzt werden können. Diese Informationen werden auch dazu genutzt, um über lange Strecken autonom fahren zu können. Um dies zu erreichen, müssen diese einzelnen Systeme in den Fahrzeugen sehr stark vernetzt sein, um möglichst viele Informationen der Sensoren auswerten und nutzen zu können. Doch während die Zuverlässigkeit und Funktionalität dieser Systeme bereits auf einem sehr hohen Niveau ist, hinkt die Sicherheit der Entwicklung hinterher [20]. Es muss klargestellt werden, dass Angreifer nicht in der Lage sind, autonom gesteuerte Systeme wie Lenkung, Beschleunigung oder Bremsen aus der Ferne zu manipulieren. Dies würde nicht nur das Material, sondern auch das Leben der Insassen und anderer Verkehrsteilnehmer gefährden. Aus diesem Grund fördert das Bundesministerium für Bildung und Forschung ein Projekt, das die Entwicklung dieser Sicherheitsmechanismen vorantreiben soll. Dieses Projekt heißt SecForCARs und wird von

mehreren Unternehmen und Bildungseinrichtungen [7] unterstützt. So sind die IT-Sicherheitsfirma SCHUTZWERK GmbH und die Universität Ulm, unter deren Betreuung diese Arbeit entstandt, Teil dieses Projektes.

Um ein entsprechendes Sicherheitsniveau zu erreichen, müssen die eingesetzten Systeme individuell auf ihre Sicherheit überprüft werden. Dazu müssen geeignete Testmethoden geschaffen und diese getestet werden. Zu den getesteten Systemen gehören sowohl die Sensoren, die von Angreifern nicht überlistet werden dürfen, als auch die Steuergeräte, die die gesammelten Informationen auswerten und verarbeiten sollen. Systeme mit Schwachstellen müssen dann verbessert oder sogar neu entwickelt werden. Darüber hinaus müssen auch die verwendeten Kommunikationsprotokolle auf die vorhandenen Sicherheitsmerkmale überprüft werden. Hierfür werden Testprotokolle und Testsysteme benötigt, um für zukünftige Entwicklungen standardisierte Tests zu haben.

1.2 Bereich der Arbeit

Die Vernetzung der Fahrzeuge begann 1915, als das erste Fahrzeug mit elektrischem Licht ausgestattet wurde. Seitdem werden immer mehr Systeme in ein Fahrzeug eingebaut [26]. In den 1980er Jahren entwickelte die Firma Bosch den CAN-Bus, der es ermöglicht, die komplexen Kommunikationsanforderungen im Fahrzeug mit einem einzigen Bussystem zu erfüllen [25]. Inzwischen sind jedoch durch eine Vielzahl neuer Sensoren wie Kameras und Radar die Anforderungen an die Übertragungsraten so stark gestiegen, dass die Übertragungsrate des CAN-Busses nicht mehr ausreicht. Deshalb wird im Fahrzeug zunehmend Ethernet eingesetzt, das die geforderten Übertragungsraten ermöglicht. Deshalb wurden in den letzten Jahren neue Kommunikationsprotokolle für die Komponenten eines Fahrzeugs entwickelt [30]. Ein vielversprechender und häufig genutzter Kandidat ist das skalierbare, serviceorientierte Middleware over IP (SOME/IP). Dieses basiert auf dem bewährten Internet Protokoll (IP) in den Versionen IPv4 oder IPv6 und kann je nach Übertragungsmedium bis zu 1000MBit pro Sekunde übertragen. Für langsamere Übertragungsgeschwindigkeiten, wie zum Beispiel 100 Mbit/s, wird jedoch nur eine 2-Draht-Verkabelung, u.a. auf Basis von IEEE 100BASE-T1, verwendet. Da diese Verkabelung eine Peer-to-Peer-Verbindung darstellt, ist zur Realisierung einer Sternverkabelung [5] ein Switch erforderlich.

1.3 Verwandte Forschung

Das SOME/IP-Protokoll basiert auf dem häufig verwendeten Internet-Protokoll (IP) in den Versionen IPv4 und IPv6. SOME/IP bietet nur wenige Funktionen, um den Zielen der IT-Sicherheit gerecht zu werden. Daher gibt es verschiedene Erweiterungen des Protokolls, um die erforderlichen Funktionen bereitzustellen. Diese zusätzlichen Methoden wurden von Jochen Kreissl in seiner Masterarbeit [18] untersucht und verglichen. Insbesondere Datagram Transport Layer Security (DTLS) für Unicast und TimedEfficient Stream Loss-tolerant Authentication (TESLA) für Multicast-Schutz werden in der oben genannten Arbeit eingesetzt. Diese bieten den Schutz von Authentifizierung, Geheimhaltung und Frische für den jeweiligen Zweck. In der folgenden Masterarbeit soll ein Konzept

für ein SOME/IP Security Testing Framework entwickelt werden. Auch für die anderen in Fahrzeugen genutzten Kommunikationsmedian, wie den oft genutzten CAN-Bus wurden einige Arbeiten zus IT-Sicherheit durchgeführt. so erstellte Phillip Schmid in seiner Bachelorarbeit das Fuzzing- und Testingingwerkzeug CANalyzator [25].

1.4 Ziel der Arbeit

Ziel dieser Ausarbeitung ist es, mögliche Sicherheitslücken des Netzwerkprotokolls SOME/IP zu identifizieren. Für diese soll ein Framework erstellt werden, das es ermöglicht zu überprüfen, ob bestimmte Steuergeräte für diese Sicherheitslücken empfänglich sind. Da in Zukunft neue Angriffsvektoren entdeckt werden können, soll dieses Framework modular aufgebaut sein. So soll ein Angriffsmodul einen oder mehrere der Angriffsvektoren abdecken. Um evaluieren zu können, ob ein solcher Angriffsvektor genutzt werden kann, sollen diese Module automatisch erkennen können, ob ein Angriff erfolgreich war oder nicht. Dies ermöglicht in Zukunft ein automatisiertes Testen dieser Schwachstellen.

Nachdem die Sicherheitsaspekte von SOME/IP durch eine erste Arbeit [18] überprüft wurden, müssen konkrete Angriffe auf das Protokoll ausgearbeitet werden. Dazu soll eine modulare und erweiterbare Bibliothek erstellt werden, die das SOME/IP-Protokoll nutzen kann. Diese kann auch als Grundlage für spätere Ansätze und Implementierungen dienen. Dabei werden nicht nur Testfälle abgedeckt, die das Protokoll selbst betreffen. Mit den Modulen können auch Tests auf Basis von SOME/IP entwickelt werden, wie zum Beispiel Fuzzing auf PDU-Ebene. Auch sollte es möglich sein, die in der Praxis durchgeführten Tests möglichst automatisch auszuwerten und damit ein Steuergerät bewerten zu können.

Dazu wurde eine Anzahl von Angriffsvektoren identifiziert (siehe Kapitel 4.2). Um diese Aufzählung möglichst vollständig zu generieren, wurde SOME/IP in unterschiedliche Bereiche aufgeteilt und diese dann nacheinander analysiert. Die Ziele dieser Angriffsvektoren wurden dann mit der HEAVENS-Methode klassifiziert. Zusätzlich wurde für jede Schwachstelle mit der STRIDE-Methode überprüft, wie groß die Wahrscheinlichkeit eingeschätzt werden kann, dass diese ausgenutzt werden. Dazu wird in dieser Methode der Aufwand und der zu erwartende Schaden beim Ausnutzen einer Schwachstelle eingeschätzt. Daraus kann dann für jeden Angriffsvektor eine Kennziffer generiert werden, die die Gefahr angibt, die davon ausgeht. Um die Tests durchführen zu können, wurde das Framework implementiert, das die Grundlage für solche Angriffe bietet. Die einzelnen Angriffe werden dann als Module erstellt, von denen bereits zwei implementiert wurden. Diese beiden Module sind in der Lage, einen Teil der identifizierten Angriffsvektoren zu überprüfen.

Forschungsfragen

Die folgenden Forschungsfragen sollen in dieser Arbeit beantwortet werden.

- Was sind die Anforderungen an ein Sicherheitstest-Framework von SOME/IP? (Definition der Testfälle, Protokollierung der Testfallergebnisse, etc.)
- Ist es möglich, bestehende SOME/IP Schwachstellen automatisch zu erkennen?
- Wie kann das Framework über Module erweitert werden, um die Kompatibilität zu verschiedenen OEM-Protokollschichten (zum Beispiel auf PDU-Ebene oder SecOC) auf Basis von SOME/IP zu gewährleisten?

Die Antworten auf diese Fragen, die während der Arbeit erarbeitet werden, werden in Kapitel 7.1 nochmals zusammengefasst.

1.5 Aufbau der Arbeit

Nach der Einleitung gliedert sich die Arbeit in fünf weitere Kapitel ein. Zuerst werden in Kapitel 2 technische Grundlagen erläutert, die benötigt werden, um den technischen Teil der Arbeit zu verstehen. Danach folgt ein Kapitel, in dem die Funktionsweise von SOME/IP und der Service Discovery erläutert wird. Im vierten Kapitel wird die Sicherheit von SOME/IP untersucht. Dazu werden die identifizierten Angriffsvektoren aufgelistet und analysiert. Kapitel 5 widmet sich dann dem implementierten Framework und dessen Funktionsweise. In Kapitel 6 werden die implementierten Module auf ihre Funktion evaluiert und im letzten Kapitel werden die Erkenntnisse der Arbeit nochmals zusammengefasst.

2 TECHNISCHE GRUNDLAGEN

Im folgenden Abschnitt der Arbeit werden die Grundlagen erläutert, die zum Verständnis der Ausarbeitung notwendig sind. Hier wird zuerst ein Blick auf die historische Entwicklung der elektronischen Bauteile in Fahrzeugen und deren Vernetzung geworfen, bevor die Grundlagen des zur Zeit meist genutzten Bussystems in Automobilen (der CAN-Bus) diskutiert wird. Danach wird erläutert, welche Anforderungen an moderne Netzwerkssysteme in Automobilen gestellt werden und inwiefern sich autonom fahrende Fahrzeuge von den herkömmlichen Modellen der letzten Jahrzehnten unterscheiden. Außerdem wird kurz erläutert, welche unterschiedliche Methoden es gibt, IT-Systeme auf ihre Sicherheit zu überprüfen. Zusätzlich werden die technischen Grundlagen genannt, die für das Erstellen eines Testingframeworks gegeben sind.

2.1 Das ISO-OSI Netzwerkschichtenmodell

Für ein besseres Verständnis der Netzwerkfunktionen und eine vereinfachte Fehlersuche bei Problemen in Netzwerken wurden die einzelnen genutzten Protokolle in Schichten eingeteilt. Es existieren verschiedene Interpretationen dieser Netzwerkschichten, wobei das ISO-OSI Schichtenmodell mit seinen sieben Schichten ein oft genutztes ist (siehe Bild 2.1). Die Informationen über den Aufbau dieses Systems stammen aus dem Buch Computernetzwerke: Der Top-Down-Ansatz (Pearson Studium - IT) von James F. Kurose und Keith W. Ross [16]. Es wird bei der ersten, der niedrigsten Schicht begonnen, welche die Übertragung der einzelnen Bits über ein geeignetes Medium beschreibt. Für die darüberliegende zweite Schicht des Schichtenmodell ist es dann nicht weiter relevant, wie die unterste Schicht umgesetzt wurde, solange alle Anforderungen der darüberliegenden Schicht eingehalten werden. Für diese Arbeit ist die Funktionsweise der unteren vier Schichten relevant, weshalb auf diese genauer eingegangen wird.

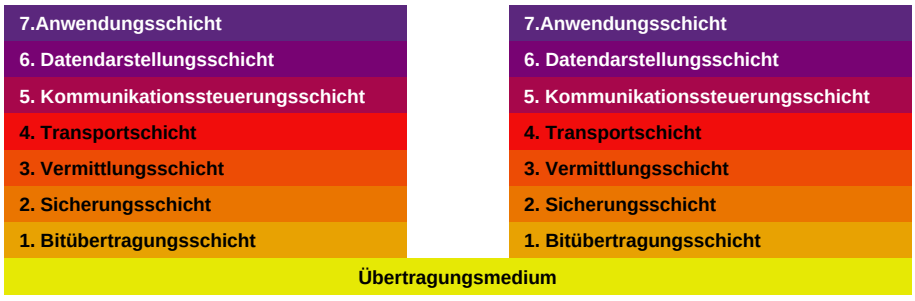

Abbildung 2.1: Die 7 Schichten des ISO-OSI Netzwerkschichtenmodells

- **1. Bitübertragungsschicht**
  In der Bitübertragungsschicht wird festgelegt, über welche Methode die einzelnen Bits übertragen werden. Oft genutzte Medien sind hierbei das 8-adrige Ethernetkabel nach IEEE 802.3 [14]. Alternativ werden häufig

kabellose Übertragungsmedien, wie Wireless LAN genutzt. Beim Automotive Ethernet wird unter anderem auf den Standard IEEE 100BASE-T1 [13] gesetzt, der es ermöglicht über 2 Drähte 100 MBit zu übertragen.

- **2. Sicherungsschicht**
  Bei der Sicherungsschicht werden die Datenströme in Pakete unterteilt und diese mit Prüfsummen versehen, um sicherzustellen, dass die Übertragung fehlerlos erfolgt. Für die Adressierung des Empfängers innerhalb eines Netzwerkes werden MAC-Adressen genutzt.

- **3. Vermittlungsschicht**
  Die Vermittlung der Pakete zwischen verschiedenen Netzwerken wird in der dritten Schicht koordiniert. Hierfür wird standardmäßig das Internet Protokoll in der Version 4 (IPv4) oder in der Version 6 (IPv6) genutzt. IPv4 nutzt dabei 32 Bit lange Adressen, während IPv6 128 Bit lange Adressen nutzt. Um festzustellen, ob sich eine andere IP-Adresse im selben Netz befindet, wie die eigene Netzwerkschnittstelle, wird eine Netzwerkmaske genutzt. Diese gibt an, wie groß ein gewähltes Netzwerk ist.

- **4. Transportschicht**
  In der Transportschicht wird der Transport von Nutzdaten geregelt. Weit verbreitete Protokolle in dieser Schicht sind das User Datagramm Protocol (UDP) und das Transmission Control Protocol (TCP). TCP ist ein verbindungsorientiertes Protokoll. Dabei wird vor dem Versenden von Daten eine Verbindung zwischen den Kommunikationspartnern initiiert. Nach dem Übertragen eines Datenpaketes teilt der Empfänger der Daten mit, dass diese erfolgreich übertragen wurden. Im Fehlerfall wird eine entsprechende Antwort an den Versender gesendet, damit dieser das fehlerhafte Paket erneut senden kann. So müssen sich die höher liegenden Schichten im ISO-OSI Modell nicht mehr darum kümmern, ob alle Pakete fehlerlos angekommen sind. UDP bietet keine solche Sicherheit. Allerdings ist UDP dadurch deutlich flexibler im Einsatz und einfacher zu implementieren. Sowohl TCP als auch UDP unterstützen Ports, die das direkte Adressieren von Services ermöglichen. Dabei werden 32 Bit-Nummern genutzt, die beim Anlegen eines Services festgelegt werden. Es werden für bestimmte Services immer wieder die gleichen Ports genutzt. So nutzt zum Beispiel das auf UDP basierende Domain Name System (DNS) den UDP Port 53. Der Absender eines DNS-Anfrage Paketes wählt dann einen freien Absenderport, an den der Server seine Antwort versenden kann.

Welches Protokoll oder welche Technik in der jeweiligen Schicht eingesetzt wird, muss von Fall zu Fall an die Anforderungen angepasst werden. Dadurch wird ein Netzwerk sehr vielseitig und flexibel. Für SOME/IP wird zumeist ein Automotive Ethernet, wie IEEE 100BASE-T1 [13] genutzt. Darauf baut dann die Sicherungsschicht und die Vermittlungsschicht auf. Für die Vermittlungsschicht wird das Internet Protokoll genutzt. Dies kann in den Versionen 4 und 6 Verwendung finden. Für die Transportschicht kann sowohl UDP als auch TCP genutzt werden, wobei hier häufig auf UDP gesetzt wird. Dieser Standard ist einfacher zu implementieren, was eine günstigere Hardware ermöglicht.

2.2 Historische Entwicklung der internen Vernetzung von Automobilen

Im Jahre 1915 wurde das erste Automobil mit einem elektronischen Bauteil, einem Scheinwerfer ausgeliefert [26]. Seit dem wurden immer mehr elektronische Komponenten in Fahrzeuge integriert. Allerdings hatte auch im Jahr 1972 ein Käfer von Volkswagen erst eine Komplexität, dass der Schaltplan des Fahrzeuges auf einer einzelnen DIN A4 Seite Platz findet [26]. Diese Entwicklung wurde allerdings sehr stark angekurbelt, als in den 1970er Jahren die ersten elektronischen Kontrolleinheiten (engl. Electric Control Unit(ECU)) verbaut wurden[8]. Mit dieser Entwicklung wurde die Komplexität der Automobile schlagartig gesteigert, da die ECUs, wie Antiblockiersystem (ABS) auf eine größere Anzahl von Informationen zurückgreifen müssen, die unter anderem von Sensoren erhoben werden. Diese Informationen wurden zu Beginn noch mit Punkt-zu-Punkt Verbindungen übertragen, was aber ein großer Aufwand an die Verkabelung bedeutet [33]. Daher wurde ein System gesucht, das die Verkabelung in einem Fahrzeug auf einen vereinheitlichten Standard bringt. Bussysteme bieten hier den Vorteil, dass alle Informationen über eine gemeinsame Kabelverbindung versandt und empfangen werden können. Dadurch wird eine einfachere Skalierbarkeit, eine Kostenminderung, eine erleichterte Realisierbarkeit und ein verringertes Fahrzeuggewicht erreicht. Im Jahre 1986 begann die Firma Robert Bosch GmbH mit der Arbeit an dem Controller Area Network (CAN). Dieses Bussystem entwickelte sich schnell zum meistgenutzten Bussystem in Automobilen [9].

2.3 Aufbau von Controller Area Network (CAN) in Automobilen

Das seit 1986 entwickelte Controller Area Network (CAN) ist bis heute ein oft genutzter Standard für die Vernetzung der einzelnen Komponenten in einem Fahrzeug. Beim so genannten CAN-Bus werden zwei Datendrähte und ein Nulleiter genutzt, um Informationen auszutauschen. Dabei verwenden alle Netzwerkteilnehmer die selben Drähte und werden somit über ein Bussystem miteinander verbunden. Die maximale Datenübertragungsrate des CAN-Busses ist bei 1 MB/s limitiert. In modernen Fahrzeugen werden immer mehr Steuergeräte verwendet. Außerdem steigt die Datenmenge, die die jeweiligen Steuergeräte verarbeiten, an. So werden für das autonome oder teilautonome Fahren immer mehr Sensoren verbaut, die hochauflösend die Umgebung wahrnehmen. Durch die Kombination der erhöhten Datenmenge und der erhöhten Anzahl an Steuergeräten, reicht die Übertragungsrate des CAN-Busses nicht länger aus, um die Kommunikation innerhalb eines Fahrzeuges zu gewährleisten. Daher sind die Hersteller in den letzten Jahren dazu übergegangen, mehrere spezialisierte CAN-Busse in ein Fahrzeug zu integrieren oder auf alternative Verkabelung für einzelne Verbindungen umzusteigen [9].

2.4 Anforderungen an Netzwerksysteme in Automobilen

Da Fahrzeuge in sehr großen Stückzahlen gefertigt werden und aus einer großen Anzahl einzelner Komponenten bestehen, wird bei der Konstruktion eines Auto-

mobils stark auf die Kosten eines einzelnen Bauteils geachtet. Daher muss die genutzte Technik der Datenkommunikation zwischen den Komponenten eines Fahrzeuges einfach und kostengünstig realisierbar sein. Um das Gewicht eines Fahrzeuges gering zu halten, werden außerdem möglichst leichte Komponenten verbaut. Daher ist es von großem Vorteil, wenn die Verkabelung des Fahrzeugnetzwerkes einfach realisiert werden kann. Außerdem muss die Kommunikation schnell, ohne große Verzögerungen und zuverlässig funktionieren, da die Sicherheitssysteme im Fahrzeug auf Informationen arbeiten, die im internen Fahrzeugnetzwerk ausgetauscht werden. Der CAN-Bus hat eben diese Anforderungen an die Vernetzung der Fahrzeugkomponenten erfüllt. Allerdings kommen in modernen Fahrzeugen immer weitere Anforderungen hinzu, wie zum Beispiel autonomes Fahren. So wird eine immer höhere Übertragungsrate im Netzwerk benötigt. Diese Anforderung kann der CAN-Bus nicht erfüllen. Daher wurden in den letzten Jahren immer wieder neue Alternativsysteme entwickelt, die sowohl die alten, als auch die neuen Anforderungen der Automobilindustrie an ein Netzwerk in einem Fahrzeug darstellen.

2.5 Autonome Fahrzeuge

Autonom oder selbstfahrende Fahrzeuge sind Fahrzeuge, die ohne das Zutun eines Menschen in der Lage sind, sich gezielt fortzubewegen. Dies kann geschehen um Personen oder Waren zu transportieren oder um Arbeit zu verrichten. Dabei setzen diese Fahrzeuge auf Informationen, die durch Sensoren wie Kameras, GPS, Abstandssensoren und anderem gewonnen werden. Das bedeutet, ein Fahrzeug, das ohne einen Fahrer in der Lage ist zu fahren aber ferngesteuert werden muss, ist kein autonom fahrendes Fahrzeug.

Die Entwicklung autonom fahrender Fahrzeuge wird weltweit stark vorangetrieben. Dabei sind vor allem IT- und Automobilfirmen maßgeblich an dieser Entwicklung beteiligt. Um den Fortschritt dieses Prozesses einteilen zu können, wurde das autonome Fahren in fünf Stufen eingeteilt [15], die vom assistierten Fahren bis zum autonomen Fahren reichen. Diese Stufen werden im Folgenden kurz erläutert.

- **1. Stufe** Assistiertes Fahren
  Der Fahrer ist für das Fahren und die Kontrolle des Verkehrs verantwortlich und wird durch Assistenzsysteme, wie einem Abstandstempomaten oder Spurhalteassistenten unterstützt.

- **2. Stufe** Teilautomatisiertes Fahren
  Beim teilautomatisierten Fahren werden verschiedene Assistenzsysteme miteinander kombiniert, damit das Fahrzeug selbstständig einige Fahrmanöver ausführen kann. So kann die Kombination aus einen Abstandstempomaten und Spurhalteassistenten genutzt werden, um Spurwechsel oder Überholmanöver zu unterstützen oder gar ganz zu übernehmen.

- **3. Stufe** Hochautomatisiertes Fahren
  In dieser Stufe der Automatisierung können bestimmte Fahraufgaben vollständig ohne das Zutun eines Fahrers übernommen werden. Allerdings nur über einen begrenzten Zeitraum, bis der Fahrer wieder die Kontrolle

über das Fahrzeug übernehmen muss. So kann das Fahrzeug zum Beispiel kurzzeitig das Fahren auf der Autobahn übernehmen. Während dieses Zeitraums kann der Fahrer seine Aufmerksamkeit kurzzeitig vom Verkehr lösen.

- **4. Stufe** Vollautomatisiertes Fahren
  Der Fahrer gibt für bestimmte Fahraufgaben die komplette Kontrolle an das Fahrzeug ab und wird während dieses Zeitraums zum Passagier. Das Fahrzeug ist in der Lage bestimmte Situationen, wie das Fahren in einem Parkhaus, komplett ohne Insassen zu absolvieren.

- **5. Stufe** Autonomes Fahren
  Es gibt nur noch Passagiere ohne spezielle Fahraufgabe. Das Fahrzeug ist komplett in der Lage autonom und ohne das Zutun von Menschen zu fahren.

2.6 Testverfahren

Um Computersysteme wie Fahrzeugnetzwerke auf ihre IT-Sicherheit zu testen werden verschiedene Verfahren genutzt, die die Genauigkeit und die Realitätsnähe solcher Tests abdecken sollen [17]. So ist ein entscheidender Bestandteil eines Angriffes auf ein System, die Information, die über dieses System zur Verfügung steht. Alle Systeme lassen sich nur effektiv angreifen, wenn genügend Informationen darüber vorhanden sind. Dabei darf allerdings nicht davon ausgegangen werden, dass Angreifer keinen Zugriff auf geheim gehaltene Informationen erlangen können. Durch frühere Angriffe oder durch Reverse-Engineering, also das Untersuchen des fertigen Produkts, ist es Angreifern oft möglich, an geheime Informationen zu gelangen. Beim White-, Black- oder Greyboxtesting werden Angriffe von Testern ausgeführt, denen nur bestimmte Informationen zur Verfügung stehen. Im Folgenden werden diese Testverfahren kurz erläutert.

- **Whiteboxtesting**
  Beim Whiteboxtesting stehen den Testern, bzw. Angreifern alle internen, geheimen Informationen über das anzugreifende System zur Verfügung. Dabei können die Angreifer auf den Quellcode von Software, auf Schaltpläne von Hardware und auf Spezifikationen geheimer Protokolle zugreifen und diese untersuchen und angreifen. Allerdings ist ein solcher Angriff nicht sehr realistisch, da Angreifer nur selten einen solch tiefgehenden Einblick in Systeminformationen erlangen. So kann durch reverse-engineering zwar der Ablauf eines Programmes ermittelt werden, allerdings ist der reverse-engineerte Quellcode meist deutlich weniger lesbar, da Variablennamen und Kommentare der Entwickler fehlen. Außerdem kann die große Menge an Informationen zum Nachteil für die Tester werden, da es passieren kann, dass wichtige Details in der Informationsflut untergehen.

- **Blackboxtesting**
  Das Blackboxtesting ist das genaue Gegenteil vom Whiteboxtesting. Hier erhalten die Tester keinerlei Informationen über das anzugreifende System. Dies ist das realistischste Szenario eines Angriffes auf ein System. Die

Angreifer müssen sich alle Informationen selbst organisieren. Dies kann durch zusätzliche Angriffe, wie Netzwerkscans oder reverse-engineering erfolgen. Allerdings benötigt die Informationsgewinnung viel Zeit und Ressourcen. Zusätzlich kann es passieren, dass die Angreifer eine bestimmte Schwachstelle übersehen, die von böswilligen Angreifern gefunden und später ausgenutzt werden kann.

- **Greyboxtesting**
  Das Greyboxtesting ist ein Verfahren, bei dem ein ausgewählter Teil an Informationen an die Tester übergeben wird. Diese Informationen werden vom Betreiber oder Entwickler des zu testenden Systems so ausgewählt, dass die Angreifer in der Lage sind, alle relevanten Bestandteile dieses Systems zu analysieren, ohne durch zu viele unwichtige Informationen abgelenkt zu werden. Allerdings wird für diesen Test eine Person oder Gruppe von Personen benötigt, die diese Vorauswahl der Informationen sinnvoll und vollständig erledigen können. Werden mehrere Informationen benötigt, können die Angreifer diese, wie beim Blackboxtesting selbst durch Analysen erlangen oder die fehlenden Informationen vom Auftraggeber nachgereicht bekommen.

Für ein möglichst umfassendes Testergebnis empfiehlt es sich, mehrere Tests mit verschiedenen Testern durchzuführen. Dabei können die unterschiedlichen Tests aus verschiedenen Testverfahren stammen. So kann durch das Blackboxtesting in Erfahrung gebracht werden, wie einfach der Informationsgewinn für einen Angreifer ist, während ein Grey- oder Whiteboxtest genutzt werden kann, um die Sicherheit des Systems bei ausreichend Informationen des Angreifers zu testen.

2.7 Informationen über zu testende Komponenten

Wird ein Fahrzeug oder eine Komponente eines Fahrzeuges getestet, muss der Tester zuerst in Erfahrung bringen, welche Funktion einzelne Bestandteile des Fahrzeuges oder der zu testenden Komponente haben. Diese Komponenten arbeiten in einem Fahrzeug zumeist mit Identifikationsnummern, die einem Außenstehenden keinerlei Informationen über die Funktionsweise verraten. Dies hat für den Fahrzeughersteller gleich mehrere Vorteile. Zum Einen kann ein Konkurrent oder Angreifer nicht einfach ergründen, welche Funktion einzelne Komponenten oder versendete Pakete haben. Zum Anderen können diese Identifikationsnummern aber auch deutlich kompakter gehalten werden, als sprechende Benennungen. Für die Hersteller und Tester steht hier das von AUTOSAR spezifizierte ARXML [4] zur Verfügung, in dem spezifiziert ist, wie die Informationen der einzelnen Komponenten und Funktionen in einer XML Datei abgespeichert werden müssen. Ist ein Tester oder Angreifer im Besitz einer solchen ARXML Datei, ist dieser in der Lage den kompletten Funktionsablauf einer Netzwerkkommunikation zu rekonstruieren. Ohne diese Datei müssen die Informationen aufwendig reverse-engineered werden.

Damit die Steuergeräte selbst wissen, welche IDs von den jeweiligen Services genutzt werden, werden diese Informationen in sogenannten Manifest-Dateien

innerhalb eines Steuergerätes abgelegt. Darin wird festgelegt, welche IDs die eigenen Services besitzen und welche fremden IDs für dieses Steuergerät von Bedeutung sind und welche Informationen diese enthalten.

2.8 Verfügbarkeit von Hard- und Software

Bei der Entwicklung von Fahrzeugen wird Soft- und Hardware mit großem Funktionsumfang benötigt, die großen Aufwand bei der Entwicklung benötigen und dann nur in vergleichsweise kleinen Stückzahlen vertrieben werden. Daher sind die Kosten für die Anschaffung solcher Werkzeuge verglichen mit anderer Software recht hoch. Auch gibt es nur wenige zuverlässige OpenSource-Lösungen, die weiterentwickelt, gepflegt und dokumentiert werden. Dies stellt sich als Hindernis heraus, wenn ein Projekt in diesem Arbeitsumfeld bearbeitet werden soll. Ein weiteres Hindernis besteht darin, dass die Automobilbranche ein hart umkämpftes Feld ist, bei der die Hersteller und Entwickler sehr darauf bedacht sind, keine internen Informationen weiterzugeben. Daher ist es selbst dann schwer, an Hard- und Software zu gelangen, die getestet werden kann, wenn Fahrzeughersteller und Zulieferer im selben Projekt arbeiten.

2.9 Verfügbare Netzwerkbibliotheken

Als Grundlage für die Netzwerkanalyse wird eine Python-Bibliothek genutzt, die in der Lage ist SOMEIP-Pakete zu identifizieren, analysieren, erzeugen und zu versenden. Es wurden die Bibliotheken scapy, pyshark und pypacker untersucht und mit Wireshark [2], einem Netzwerkanalysetool, verglichen. Für die Wahl einer geeigneten Bibliothek wurden Anforderungen formuliert und deren Erfüllung analysiert. Da scapy die meisten Anforderungen erfüllt und Lösungen für die aufgetretenen Probleme umsetzbar sind, wurde diese Bibliothek für die Implementierung des Frameworks genutzt.

Ein großer Vorteil von Python ist die große Verfügbarkeit von Bibliotheken, die für die Implementation der eigenen Software genutzt werden kann. So werden verschiedene Bibliotheken angeboten, die das Arbeiten mit dem Netzwerk unterstützen.

scapy
Scapy [3, 24] ist eine seit 2003 entwickelte Python-Bibliothek für das Arbeiten mit Netzwerkpaketen. Die Bibliothek unterstützt das Empfangen, Verändern, Erstellen und Versenden von Netzwerkpaketen und beherrscht eine große Anzahl von Protokollen, die automatisiert erkannt werden. Außerdem lässt sich die Auswahl der unterstützten Protokolle durch eigene Paket-Klassen erweitern.

pyshark
Pyshark [12] ist ein Wrapper für tshark, die Kommandozeilenversion von Wireshark. Dieser Wrapper ermöglicht es, tshark in eine Python-Implementation zu integrieren. Bei der Analyse von tshark wurde allerdings festgestellt, dass die ab Version 3.1.1rc0 [31] von Wireshark integrierte Unterstützung von SOMEIP-SD noch fehlerhaft ist. Da die Pakete in

der Kommandozeilenversion fehlerhaft dargestellt werden, kann pyshark die Informationen nicht korrekt auslesen.

pypacker
Pypacker [28] ist eine von Michael Stahn entwickelte Bibliothek, die einen ähnlichen Funktionsumfang wie scapy besitzt. Da pypacker aber wenig verbreitet ist, wird diese Bibliothek von keiner Community unterstützt, welche die Dokumentation und Pflege der Bibliothek übernimmt. Daher kann nicht vorausgesagt werden, wie lange pypacker unterstützt wird.

Wireshark
Wireshark [2] ist ein von einer großen Community entwickeltes Werkzeug für die Analyse von Netzwerken, das seit der Version 3.1.1rc0 [31] SOME/IP unterstützt. Durch den großen Funktionsumfang und den vielen unterstützten Protokollen hat sich das quelloffene Werkzeug zu einem Standard für sehr viele Anwendungsfälle in der Analyse von Kommunikationen entwickelt. Im gegensatz zu den oben genannten Werkzeugen kann Wireshark allerdings nicht als Python Bibliothek genutzt werden. Es wird im weiteren Verlauf der Arbeit aber als Referenz für den Vergleich der Bibliotheken genutzt.

3 SOME/IP

Das von AUTOSAR spezifizierte Netzwerkprotokoll SOME/IP [5] soll die Anforderungen der internen Fahrzeugvernetzung auf das Automotive Ethernet adaptieren. Dazu wird eine weitere Schicht oberhalb von UDP/IP beziehungsweise TCP/IP hinzugefügt. Diese Schicht erlaubt es den einzelnen Komponenten miteinander zu kommunizieren und Daten auszutauschen. Dabei muss sich die Software, die auf diesen Komponenten betrieben wird, nicht mehr darum kümmern, welche Ressource auf welchem Steuergerät verfügbar ist. Dieses Konzept nennt AUTOSAR serviceorientierte Middleware. Im folgenden Kapitel wird diese Funktionalität und die Funktionsweise von SOME/IP näher erläutert. Dieses Wissen ist für das Verständnis der Angriffsvektoren und die Implementierung wichtig.

3.1 Anwendungszweck als Middleware

Um die Software, die auf den einzelnen Steuergeräten in einem Fahrzeug läuft, einfacher halten zu können, übernimmt SOME/IP die Vermittlung der Daten zwischen diesen Komponenten. Benötigt eine Software Informationen, stellt diese direkt eine Anfrage an die SOME/IP Implementation des eigenen Steuergerätes. Werden die Informationen lokal gehalten, gibt es diese direkt an die anfragende Funktion zurück. Sind die Informationen lokal nicht bekannt, dem SOME/IP Stack des Steuergerätes ist aber bekannt, wo diese Information liegt, stellt es eine entsprechende Anfrage an das entfernte Steuergerät, andernfalls stellt SOME/IP eine Funktionalität bereit, die das Auffinden entfernter Ressourcen erlaubt. Durch dieses Verfahren ist es nicht weiter notwendig, dass der Entwickler der eigentlichen Software auf dem Steuergerät überprüfen muss, wo bestimmte Ressourcen erreichbar sind. Selbst wenn sich der Standort der Ressourcen verändert, kann SOME/IP als Middleware dies erkennen und entsprechend reagieren. Dieser Ablauf wird in Abbildung 3.1 nochmals dargestellt. Dabei ist ein exemplarischer Aufbau eines fahrzeuginternen Netzwerkes dargestellt, wobei die grünen Pfeile die Kommunikation der Steuergeräte untereinander darstellen.

3.2 Services

Um verschiedene Arten von Ressourcen bereitstellen zu können, werden von SOME/IP mehrere Services angeboten. So kann auf einer beliebigen Komponente ein Event oder eine Eventgruppe abonniert werden. Wird ein Event abonniert, versendet der Anbieter des Events die betreffenden Daten, wenn diese sich ändern oder in einem vorgeschriebenen Intervall. Eventgruppen fassen mehrere Events zusammen (siehe Abschnitt 3.2.1). Zusätzlich wird über SOME/IP die Ausführung von Remote Procedure Call (RPC)- Methoden angeboten. Dabei ist ein Teilnehmer des Netzwerks in der Lage auf einer entfernten Netzwerkkomponente eine Methode auszuführen (siehe Abschnitt 3.2.2). Außerdem ist es

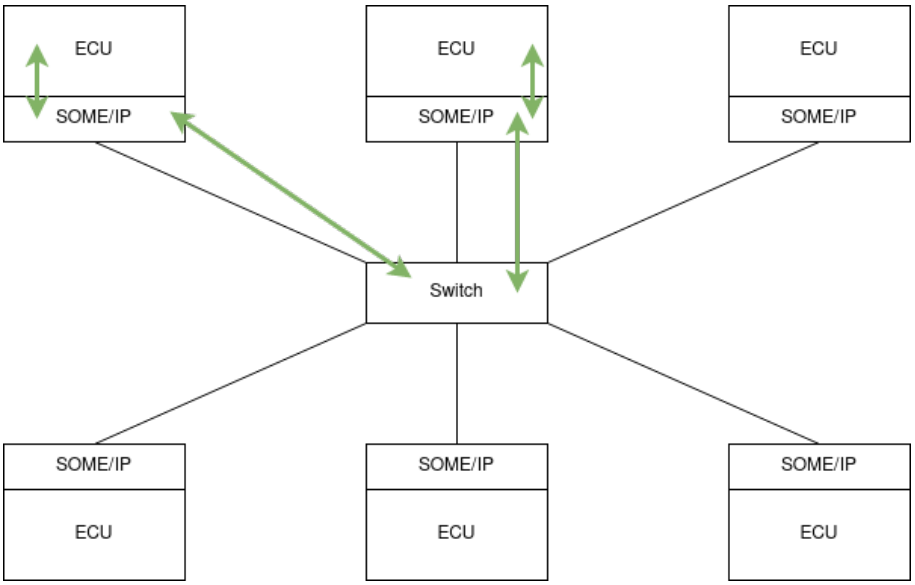

Abbildung 3.1: SOME/IP als Middleware in der Fahrzeugvernetzung

mit Setter und Getter Methoden möglich, Datenfelder direkt auszulesen oder zu ändern (siehe Abschnitt 3.2.3).

3.2.1 Events

Events sind Datenfelder, über deren Zustand automatisch informiert werden soll. Dies kann zum Beispiel von der Head Unit genutzt werden, um die Geschwindigkeit des Fahrzeuges, die Drehzahl des Motors, die Öltemperatur und alle anderen Informationen zu erhalten, die dem Fahrer angezeigt werden sollen. Dazu muss das Steuergerät, das die entsprechenden Werte hält Events anbieten, die ein anderes Steuergerät abonnieren kann. Wird ein solches Event abonniert, kann festgelegt werden, auf welche weise das abonnierende Steuergerät über Änderungen des Wertes informiert werden soll. Hierfür stehen drei Varianten zur Auswahl: *Zyklische Updates*, die nach immer dem gleichen Zeitabstand den aktellen Wert übermitteln, *änderungsbasierte Updates*, die den Wert aktualisieren, wenn Änderungen eintreten und *Epsilon-Updates*. *Epsilon-Updates* entsprechen den änderungsbasierten Updates. Allerdings werden hier erst Update-Pakete versendet, wenn ein bestimmter Schwellwert überschritten wurde.

Events, die zumeist gemeinsam abonniert werden, können in Eventgruppen zusammengefasst werden. Diese Eventgruppen können dann auf einmal abonniert oder deabonniert werden.

3.2.2 RPC-Aufruf von Methoden

SOME/IP bietet die Möglichkeit, auf entfernten Geräten Methodenaufrufe durchzuführen. Dazu muss ein entsprechendes Paket versendet werden, das die Methoden ID und die Übergabeparameter enthält. Dabei muss der Typ und die Anzahl

der Übergabeparameter denen entsprechen, die auch der Methode auf dem ausführenden Gerät entsprechen. Das Antwortpaket enthält dann den Return-Code des Aufrufs. Der Return Code Null bedeutet hierbei, dass die Methode erfolgreich ausgeführt wurde.

Es gibt verschiedene Arten, eine Methode auszuführen. Fire & Forget Methoden erwarten nur einen Methodenaufruf und geben keinen Rückgabewert in einem Antwortpaket zurück. Bei diesen Methoden wird der aufrufende Kommunikationspartner nicht über den Erfolg oder Misserfolg des Methodenaufrufs informiert. Request & Response Methoden erhalten nach dem RPC-Aufruf ein Antwortpaket mit dem Ergebnis des Aufrufs zurück.

3.2.3 Setter- und Getter-Methoden von Feldern

Setter und Getter-Methoden sind Methodenaufrufe, die ein Datenfeld direkt auslesen oder verändern. Dabei wird bei jeder Anfrage ein Antwortpaket versendet, das den Inhalt des Datenfeldes zurückgibt. So kann sichergestellt werden, dass bei einem Set-Befehl der Wert korrekt eingetragen wurde.

3.3 Gateways

Viele Bauteile in modernen Fahrzeugen basieren immer noch auf dem CAN-Bus. Da die Umstellung auf ein modernes Kommunikationsmittel nicht ohne Weiteres umgesetzt werden kann, werden im Fahrzeug weiterhin CAN-Busse verbaut. Damit moderne SOME/IP Steuergeräte mit diesen Bauteilen kommunizieren können, können Gateways genutzt werden, die an beiden Systemen angeschlossen werden. Sollen die Informationen eines CAN-Sensors an ein SOME/IP Steuergerät übertragen werden, kann dieses Steuergerät die entsprechenden Events am Gateway abonnieren (siehe Abbildung 3.2). Das Gateway wiederrum empfängt die Sensordaten am CAN-Bus und leitet die Informationen mittels SOME/IP an den Empfänger weiter. Solche Gateways können auch so spezifiziert werden, dass auch Kommunikationen über andere Wege aufgebaut werden können. So können die SOME/IP-fähigen Steuergeräte über Mobilfunk oder andere externe Kommunikationskanäle mit anderen Fahrzeugen (Car to Car) oder Servern kommunizieren. Diese Verbindungsmöglichkeiten können auch wieder für einen Angriff auf das Fahrzeug genutzt werden. Da sich diese Arbeit aber ausschließlich auf SOME/IP fokussiert, werden diese Möglichkeiten nicht weiter diskutiert.

Blackboxen, die in den Fahrzeugen installiert werden, abonnieren hierbei alle Events und sind so in der Lage, den kompletten Zustandsverlauf eines Fahrzeuges aufzuzeichnen und für spätere Analysen, zum Beispiel im Falle eines Unfalles, bereitzustellen.

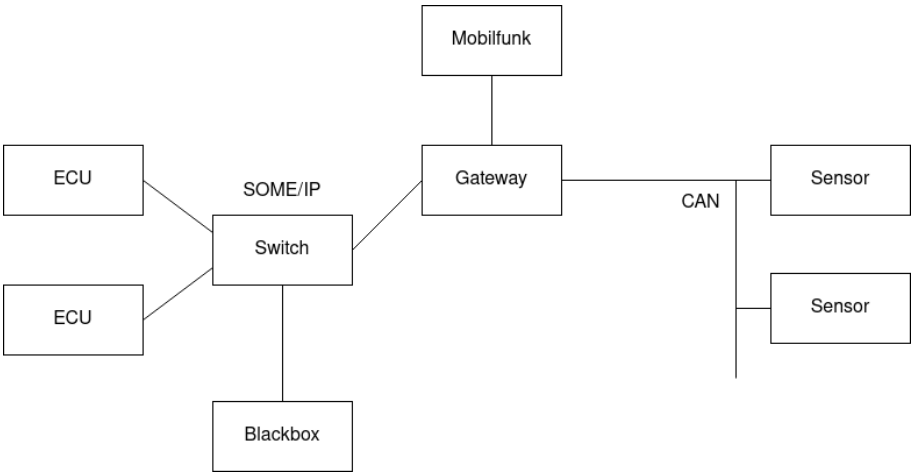

Abbildung 3.2: Interne Fahrzeugkommunikation über ein Gateway

3.4 SOME/IP Frames

SOME/IP baut auf TCP/IP oder auf UDP/IP auf. Da TCP und UDP der Transportschicht im ISO-OSI-Schichtenmodell, also der vierten Schicht angehören, bildet SOME/IP die fünfte Schicht. Jedem SOME/IP Paket wird ein Header vorangestellt, der 128 Bit lang ist. Im Header werden die Message ID, die Länge, die Request ID und weitere Informationen übertragen. Diese werden im folgenden Abschnitt genauer diskutiert und in Abbildung 3.3 dargestellt.

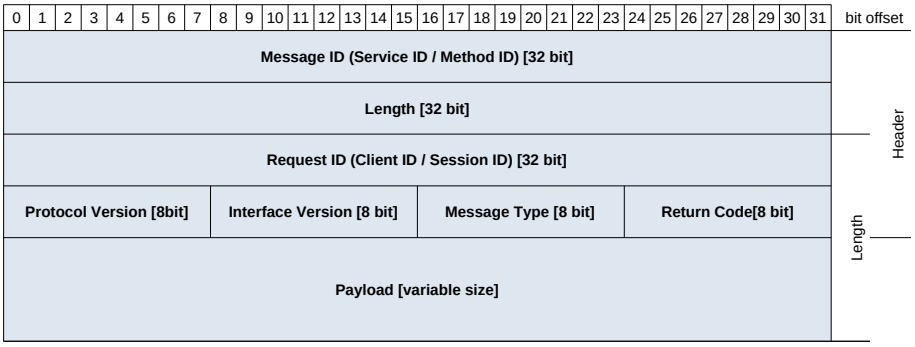

Abbildung 3.3: Der 128 Bit Header von SOME/IP

3.4.1 Message ID

Die 32 Bit lange Message ID gibt den Empfänger eines Paketes an. Diese ID besteht aus drei Teilen. Die ersten 16 Bit bilden die Service ID, die einen Service auf diesem Gerät und diesem UDP- oder TCP-Port eindeutig kennzeichnet. Dies bedeutet, dass es in einem Netzwerk oder auf einem Steuergerät mehrere Services mit der selben ID geben kann. Um diese eindeutig adressieren zu können, werden die niedrigeren Schichten des Netzwerks genutzt. So können zwar mehrere Instanzen von SOME/IP auf einem Gerät betrieben werden, diese müssen aber auf unterschiedlichen TCP- oder UDP-Ports erreichbar sein. Innerhalb einer Instanz muss die Service ID einzigartig sein. Da SOME/IP zumeist auf mehreren

Geräten betrieben wird, muss auch die IP-Adresse dieser Geräte mit einbezogen werden, um einen Service sicher eineindeutig identifizieren zu können. Das Bit das der Service ID nachgestellt ist, also das 17. Bit im Header von SOME/IP, gibt an, ob es sich bei der nachfolgenden ID um eine Methoden oder eine Event ID handelt. Ist dieses Bit eine Null, wird mit diesem SOME/IP Paket eine Methode, sonst ein Event adressiert. Bei den restlichen 15 Bits der Message ID handelt es sich dann um die Methoden oder Event ID, die eine konkrete Methode oder ein Event adressieren.

3.4.2 Länge

Das Längenfeld von SOME/IP ist 32 Bit lang und gibt die Länge des SOME/IP Paketes in Byte an. Die Länge wird ab dem ersten Bit nach dem Längenfeld gemessen. Da der Header ab dieser Stelle noch mindestens 8 Byte Lang ist, muss die angegebene Länge des Paketes mindestens diesem Wert entsprechen.

Die Länge der Nachrichten ist auf 1400 Byte beschränkt. Werden mehrere SOME/IP Nachrichten von einem Sender an einen Empfänger verschickt, können diese in einem einzelnen TCP oder UDP Paket verschickt werden. Der Empfänger erkennt dies daran, wenn die Länge der niedrigeren Schicht zu groß ist für das nächste SOME/IP Paket. In diesem Fall kann er den Rest des Payloads als weiteres SOME/IP Paket interpretieren.

3.4.3 Request ID

Die Request ID ist in zwei gleich lange Teile zerlegt. Der erste Teil ist die Client ID. Diese entspricht der Service ID des Absenders und kann für die Antwort-Pakete des Server genutzt werden. Um die Antwort Pakete eindeutig zuordnen zu können, wird jeder Anfrage eine fortlaufende Nummer zugeordnet. Dies ist die Session ID, die beim ersten Paket mit Null beginnt. Sind die 16 Bit der Session ID voll ausgenutzt, wird wieder bei Null begonnen.

3.4.4 Weitere Datenfelder

Die letzten 32 Bit des SOME/IP Headers werden für vier unterschiedliche Datenfelder genutzt. Dabei handelt es sich um die Protokoll Version, die zum aktuellen Zeitpunk nur einer Eins entsprechen kann, da SOME/IP bislang nur in der ersten Version spezifiziert wurde. Daraufhin folgt die Interface Version, die vom Hersteller selbst angepasst werden kann und die angibt, in welcher Interface Version die Implementation der Steuergeräte arbeitet. Die nächsten 8 Bit des Headers geben an, welcher Typ die folgende Nachricht ist. Die möglichen Werte dieses Message Type Feldes werden in Tabelle 3.1 erläutert. Die letzten 8 Bit des SOME/IP Headers ergeben den Return Code, den Antwortpakete auf Methodenaufrufe enthalten. Dieser Return Code ist allerdings bei allen Paketen enthalten und wird auf Null gesetzt, wenn kein Return Code übertragen werden soll.

Nummer	Wert	Beschreibung
0x00	REQUEST	Eine Anfrage, die eine Antwort erwartet
0x01	REQUEST_NO_RETURN	Eine Anfrage, die keine Antwort erwartet
0x02	NOTIFICATION	Benachrichtigung, die keine Antwort erwartet
0x80	RESPONSE	Antwortpaket auf eine vorhergehende Anfrage
0x81	ERROR	Antwortpaket, das einen Fehler enthält
0x20	TP_REQUEST	Wie 0x00 für ein Transport Protokoll Paket (mit Überlänge)
0x21	TP_REQUEST_NO_RETURN	Wie 0x01 für ein Transport Protokoll Paket (mit Überlänge)
0x22	TP_NOTIFICATION	Wie 0x02 für ein Transport Protokoll Paket (mit Überlänge)
0x23	TP_RESPONSE	Wie 0x80 für ein Transport Protokoll Paket (mit Überlänge)
0x24	TP_ERROR	Wie 0x81 für ein Transport Protokoll Paket (mit Überlänge)

Tabelle 3.1: Mögliche Werte des Nachrichten Typs von SOME/IP

3.5 Service Discovery

Die Service Discovery wird von SOME/IP genutzt, um Services anzubieten und zu suchen. Dazu gibt es zwei unterschiedliche Arten von Paketen: Die Offer-Service und die Find-Service Pakete. Wird ein Fahrzeug gestartet, werden in den meisten Fällen alle Komponenten gleichzeitig mit Strom versorgt. Die Steuergeräte starten automatisch beim Starten des Fahrzeuges und warten dann eine vorgeschriebene Zeit. Diese Wartezeit soll langsameren Steuergeräten die Möglichkeit geben auch in einen betriebsbereiten Modus zu wechseln. Nach dieser Wartezeit beginnen die Steuergeräte an zuvor festgelegte IP-Multicast-Adressen die eigenen Services per Offer-Service-Pakete bekannt zu geben. Dies hat den Zweck, dass kurze Zeit nach dem Starten des Fahrzeuges alle Kommunikationspartner aller Sterngeräte bekannt sind und das Gesamtsystem betriebsbereit ist. Ändert sich der Zustand eines Steuergerätes dahin gehend, dass es zu einem späteren Zeitpunkt ein neues Event abonnieren muss, kann es dieses Event durch ein Find-Service Paket ausfindig machen. Dazu sendet es mittels Multicast-Paket an alle anderen Steuergerät das entsprechende Paket mit der ID des zu findenden Services. Die ECU die diesen Service anbietet antwortet mit einem Offer-Service Paket. Die Offer-Service Pakete enthalten eine Time-To-Live (TTL). Also einen Zeitstempel, der Angibt, wie lange der Service von dem absendenden Steuergerät angeboten wird. Ein dediziertes Stop-Offer-Service Paket gibt es nicht, kann aber durch ein Offer-Service Paket mit der TTL Null umgesetzt werden.

3.6 Sicherheitsfunktionen für SOME/IP

SOME/IP enthält keine eigenen Sicherheitsfunktionen. Allerdings können entsprechende Funktionen durch zusätzliche Netzwerkschichten hinzugefügt werden. Wird TCP als Transportschicht gewählt, kann zum Beispiel Transport Layer Security (TLS) [22] genutzt werden, um eine Verschlüsselung aller Daten zu erreichen. Allerdings muss dann eine zusätzliche Verwaltungseinheit eingerichtet werden, die die einzelnen Schlüssel der Steuergeräte kontrolliert. Auch für UDP kann eine zusätzliche Verschlüsselungsebene im Netzwerkstack eingesetzt werden. Hier kann dann zum Beispiel die UDP-Version von TLS, Datagram Transport Layer Security (DTLS) [23] eingesetzt werden. Allerdings sind diese Verschlüsselungsmethoden nur in der Lage, Unicast Pakete zu verschlüsseln, da Symmetrische Schlüssel immer zwischen zwei Netzwerkteilnehmern eingesetzt werden. Für die häufig bei SOME/IP eingesetzten Multicast-Pakete muss daher auf eine andere Verschlüsselungstechnologie gewechselt werden. Hier bietet sich zum Beispiel TESLA [1] an, das diese Funktionalität ermöglicht.

Sollen die Informationen, die im SOME/IP Netzwerk ausgetauscht werden nicht verschlüsselt, aber die Integrität der Daten geschützt werden, kann unter anderem SecOC [6] eingesetzt werden. Dies ist ein Protokoll, das an alle Nachrichten kurze Binärblöcke anhängt, die aus dem Inhalt der Nachricht und einem geheimen Schlüssel generiert werden. Dadurch wird sichergestellt, dass nur der Besitzer dieses geheimen Schlüssels der Versender dieser Nachricht gewesen sein kann.

Diese Sicherheitsmechanismen werden in dieser Arbeit nicht weiter beleuchtet, da sie nicht Bestandteil von SOME/IP sind. Sie können aber in der späteren Entwicklung des Frameworks als Modul in dessen Funktionsumfang integriert werden.

3.7 Zusammenfassung

In diesem Kapitel der Arbeit wurde die Funktionsweise von SOME/IP erläutert. Das Protokoll wird als Middleware bezeichnet. Dies bedeutet, dass die Steuergeräte die SOME/IP nutzen, nicht wissen müssen, wer einen gewissen Service anbietet. Die SOME/IP-Service Discovery kümmert sich hierbei darum, dass den Steuergeräten die benötigten Services bekannt gemacht werden. Wird der benötigte Service von einem anderen Steuergerät angeboten, steuert SOME/IP die Informationsweiterleitung. Für diese Kommunikation stehen mehrere Arten von Paketen bereit.

4 SICHERHEITSANALYSE

Da SOME/IP auf die oft genutzten Protokolle der unteren Schichten im Netzwerkschichtenmodel zurückgreift (siehe Abschnitt 2.1), sind auch Angriffe auf diese Netzwerkschichten durchführbar. In diesem Kapitel wird diskutiert, welche Angriffe auf SOME/IP durchführbar sind. Hierfür wird davon ausgegangen, dass Techniken, wie das ARP-Spoofing [32] genutzt werden können, um zum Beispiel den Datenverkehr zwischen zwei Netzwerkteilnehmern auf die angreifende Maschine umzuleiten. Auch Angriffe auf die im Netzwerk genutzte Hardware sind möglich. Auf diese Angriffe wird in dieser Arbeit nicht weiter eingegangen. Es ist unter Umständen möglich, die SOME/IP Konfiguration der Steuergeräte anzugreifen, indem die genutzte Manifestdatei auf den Netzwerkteilnehmern manipuliert oder entfernt wird. Allerdings wird hierbei das Gerät selbst angegriffen und nicht das SOME/IP-Protokoll, weswegen diese Angriffsvektoren in der folgenden Auflistung nicht weiter beachtet werden.

4.1 Methodik

Um eine möglichst flächendeckende Sicherheitsanalyse eines Systems durchführen zu können, muss ein methodisches und analytisches Vorgehen gewählt werden. Dadurch ist es auch möglich, die einzelnen Schwachstellen besser einschätzen und miteinander vergleichen zu können. Dadurch kann eine Priorisierung der Schwachstellen erfolgen um die Reihenfolge und Priorität der Fehlerbehebung festgelegt werden. Hierfür werden für jeden Angriff die Ziele dieses Angriffes nach der STRIDE Methode [27] bestimmt. Für die Risikoeinschätzung wird die HEAVENS-Methode [19] genutzt und für jeden Angriff werden dessen Voraussetzung und ein Ansatz für dessen Umsetzung diskutiert.

4.1.1 Ziele der Angriffe

Wenn Angriffe klassifiziert werden sollen, kann dies nach unterschiedlichen Kriterien geschehen. Eine Methode besteht darin festzustellen, welche Ziele ein Angreifer durch sein Handeln erreichen möchte. Hierfür kann das STRIDE-Verfahren [27] genutzt werden. Hierbei steht jeder Buchstabe des Akronyms STRIDE für ein mögliches Angriffsziel. Im Folgenden werden diese Ziele aufgelistet und erläutert, welchen Nutzen der Angreifer durch dieses Ziel erreichen kann. Später können dann die einzelnen Angriffe dahingehend kategorisiert werden.

S Spoofing - Dies kann in diesem Szenario am besten mit Vortäuschen übersetzt werden. Hierbei gibt ein Angreifer vor, ein legitimierter Teilnehmer oder anderer Nutzer eines Systems zu sein. Dies kann er wiederrum nutzen, um falsche Informationen zu verteilen oder durch eine falsche Legitimation an Informationen zu gelangen.

T Tampering - Dies beschreibt die Manipulation von Daten. Hierbei verändert der Angreifer Informationen, die von einem Teilnehmer an einen Anderen versandt werden oder die auf einem Datenmedium gespeichert sind. Ohne entsprechenden Schutz kann der Empfänger der Informationen diese Manipulation nicht erkennen.

R Repudiation - Beschreibt die Abstreitbarkeit des Angreifers. Dies ist dann der Fall, wenn der Angreifer in der Lage ist, ein Paket zu versenden, ohne sich zuvor zu authentifizieren oder anderweitig von den anderen Teilnehmern eines Systems identifiziert werden kann.

I Information Disclosure - Beschreibt das Erlangen von Informationen, die eigentlich verborgen bleiben sollten. Diese können von einem Angreifer eventuell genutzt werden, um weitere Angriffe durchzuführen oder sind bereits das endgültige Ziel eines Angriffs.

D Denial of Service - Oftmals ist es wichtig für einen Dienstleister, dass ein bestimmter Service verfügbar gemacht wird. Ein Denial of Service (DoS) Angriff zielt darauf ab, diesen Service nicht länger verfügbar halten zu können. Dies geschieht in vielen Fällen durch Überlastung des Anbieters des Services. Aber auch andere Angriffsszenarien sind denkbar. So kann ein Angreifer viele gehackte Geräte gleichzeitig nutzen, um die Überlastung eines Systems zu erreichen. Dieser Angriff wird dann als Distributed Denial of Service (DDoS) bezeichnet.

E Elevation of Privilege - Die Nutzer eines Systems sind zumeist in ihren Möglichkeiten eingeschränkt, um bestimmte Funktionen oder Informationen nur ausgewählten Teilnehmern verfügbar zu machen. Ziel eines Angriffs kann sein, diese Rechte auszuweiten und so an Informationen zu gelangen oder Dienste in Anspruch zu nehmen, die eigentlich außerhalb der Berechtigung des Angreifers liegen.

4.1.2 Risikoanalyse mit der HEAVENS-Methode

Um das Risiko eines Angriffes Einschätzen zu können, muss eingeschätzt werden, wie einfach der Angriff durchzuführen ist und wie groß der zu erwartende Schaden ist. Hierfür bietet sich die HEAVENS-Methode [19] an. Diese nutzt für die Einschätzung die Threat Level und Impact Level Kenngrößen, um den Aufwand und den Schaden eines Angriffes Einschätzen zu können. Aus diesen Kennzahlen wird dann eine Risikobewertung ermittelt.

4.1.2.1 Bestimmung des Threat Levels

Für die Bestimmung des Threat Levels, also des Aufwands, der für einen Angriff erfüllt werden muss, werden in der HEAVENS-Methode Punkte in vier Rubriken vergeben (siehe Tabelle 4.1). Diese Rubriken sind das benötigte Fachwissen, die Verfügbarkeit von Informationen, die benötigten Werkzeuge und der Zugriff, der benötigt wird, um einen entsprechenden Angriff durchführen zu können. Je größer die Schwirigkeiten die in der jeweiligen Rubrik gelöst werden müssen, desto weniger Punkte werden in dieser Rubrik vergeben. Für jede Rubrik sind null bis drei Punkte möglich. Im Bereich Fachwissen werden zum Beispiel drei

Punkte vergeben, wenn ein Amateur mit wenig Kenntnissen über die Materie in der Lage sein kann diesen Angriff auszuführen. Wird allerdings das Wissen von mehreren Experten benötigt, wird in dieser Rubrik kein Punkt vergeben. Um das Gesamtergebnis zu erhalten muss die Summe aller für die Rubriken vergebenen Punkte gebildet werden.

Ein Beispiel: Wird für einen Angriff fortgeschrittenes Fachwissen, öffentlich zugängliche Informationen, spezialisierte Werkzeuge und ein physischer Zugriff benötigt, ergibt dies ein Threat Level von sieben.

Da alle Angriffe auf SOME/IP einen einen physischen Zugriff auf das Automotive Ethernet haben und die gleichen Voraussetzungen an Information, Werkzeug und Fachwissen haben, wird bei der Bestimmung des Threat Levels zumeist die gleiche Punktzahl erreicht. Diese wird gebildet aus einem fortgeschrittenen Fachwissen (zwei Punkte), internen Informationen (ein Punkt), einem spezialisierten Werkzeug (zwei Punkte) und einem einfachen physischen Zugriff (ein Punkt). Dies ergibt ein Threat Level von sechs Punkten, welches für die meisten Angriffe genutzt wird. Falls keine internen Informationen vonnöten sind, wie es bei den Enumerations-Angriffen (siehe 4.2.2.7, 4.2.3.3 und 4.2.4.6) der Fall ist, werden nur öffentlich zugängliche Informationen benötigt. Dadurch steigt der Threat Level auf acht Punkte. Dieser Unterschied fällt aber, wie in der Tabelle 4.3 zu sehen ist, bei der Risikoeinschätzung nicht ins Gewicht.

Komponente	**Einstufung**	**Wert**
Fachwissen	Amateur	3
	Fortgeschritten	**2**
	Experte	1
	Expertenteam	0
Verfügbarkeit von Informationen über das System	Öffentlich	**3**
	Eingeschränkt	2
	Interna	**1**
	Geheim	0
Werkzeug	Standard	3
	Spezialisiert	**2**
	Maßgeschneidert	1
	Mehrere Maßgeschneiderte	0
Zugriff	Unlimitierter Fernzugriff	3
	Zeitweiser Fernzugriff	2
	Einfacher physischer Zugriff	**1**
	Physischer Zugriff zwingend notwendig	0

Tabelle 4.1: Bestimmung des Threat Levels in der HEAVENS-Methode

4.1.2.2 Bestimmung des Impact Levels

Das Impact Level der HEAVENS-Methode beschreibt den Schaden, der durch einen Angriff zu erwarten ist. Dazu wird der Schaden in vier unterschiedlichen Rubriken beurteilt (siehe Tabelle 4.2). Diese sind die Betriebssicherheit, also welchen Schaden Personen durch einen Angriff erleiden können, der Datenschutz,

der finanzielle Schaden und die Funktionseinschränkung der Services, die vom Fahrzeug bereitgestellt werden. Für jede dieser Rubriken können null bis vier Punkte vergeben werden. Null Punkte in jeder Rubrik besagen, dass kein Schaden zugefügt wird. Vier Punkte in der Rubrik Betriebssicherheit besagen zum Beispiel, dass das Leben mehrerer Personen bedroht wird. Um den erwarteten Schaden für die Risikoeinschätzung zu erhalten, wird das Maximum der vier Rubriken verwendet.

Beispiel: Wird durch einen Angriff die Betriebssicherheit des Fahrzeuges so beeinträchtigt, dass dies für den Fahrer lebensgefährlich ist (drei Punkte), keine Informationen gewonnen werden können (null Punkte), ein mittlerer finanzieller Schaden (zwei Punkte) und eine mittlere Funktionseinschränkung (zwei Punkte) des Fahrzeuges zu erwarten sind, ergibt dies durch das Maximum bei der Betriebssicherheit einen Impact Level von drei.

Auch bei der Bestimmung des Impact Levels werden nur wenige Fälle wirklich genutzt. So werden bei den Enumerations-Angriffen (siehe Abschnitt 4.2.2.7, 4.2.3.3 und 4.2.4.6) nur anonyme Daten erhoben, während sonst kein großer weiterer Schaden erzeugt wird. Bei den anderen Angriffen, die es auf Informationen abgesehen haben, besteht immer die Möglichkeit, dass die Bewegungsdaten des Fahrzeuges und somit des Fahrers ermittelt werden können und dieser dadurch deanonymisiert werden kann. Dies ergibt dann ein Impact Level von drei. Bei den Angriffen, bei denen eine Datenmanipulation oder ein Denial of Service erreicht werden kann, muss davon ausgegangen werden, dass dies zu einem Unfall führen kann. Da in diesem Fall auch mehrere Fahrzeuge betroffen sein können, ergibt dies einen Impact von vier.

Impact Level	**Betriebssicherheit**	**Datenschutz**	**Finanziell**	**Funktionalität**
0	Keine	Keine	Keine	Keine
1	Leichte bis mittlere Verletzungen	**Anonyme Daten**	Gering	Gering
2	Schwere oder mehrere mittlere Verletzungen	Identifikation des Fahrers oder des Fahrzeugs	Mittel	Mittel
3	Lebensgefährlich	**Bewegungsdaten von Fahrer und Fahrzeug**	Groß	Schwer
4	**Lebensgefährlich für mehrere Personen**	Bewegungsdaten mehrerer Fahrzeuge	Groß, mehrere Fahrzeuge	Schwer, mehrere Fahrzeuge

Tabelle 4.2: Bestimmung des Impact Levels in der HEAVENS-Methode

4.1.2.3 Risikoeinstufung

Für die endgültige Risikoeinstufung werden nun die Summe der Threat-Level-Einschätzung und der maximale Impact Level genutzt und in der folgenden Tabelle das resultierende Risiko ausgelesen. Hierbei spielt der Aufwand des Angriffs eine Rolle, da ein Angriff unwahrscheinlicher wird, wenn dieser nur sehr

Impact \ Threat	0-2	3-5	6-8	9-11	11+
0	QM	QM	QM	QM	QM
1	QM	Gering	**Gering**	Gering	Mittel
2	QM	Gering	Mittel	Mittel	Hoch
3	QM	Gering	**Mittel**	Hoch	Hoch
4	Gering	Mittel	**Hoch**	Hoch	Kritisch

Tabelle 4.3: Bestimmung des Risikos mithilfe des Threat und des Impact Levels in der HEAVENS-Methode

schwer umzusetzen ist. Ist der Angriff dagegen sehr leicht umzusetzen, steigt die Wahrscheinlichkeit, dass dieser auch geschieht. Ebenso spielt der zu erwartende Schaden eine große Rolle, da zum einen auch die Wahrscheinlichkeit steigt, dass ein Angriff durchgeführt wird, wenn dieser einen großen Nutzen für den Angreifer zu Folge hat. Ebenso wird es wichtiger für den Hersteller einen Angriff zu verhindern, wenn dieser mit schwerwiegenden Folgen für den Fahrer, den Besitzer oder den Hersteller des Fahrzeuges verbunden ist. In diesem Fall muss der Angriff möglichst durch zusätzliche Sicherheitsmaßnahmen verhindert werden. In der Tabelle 4.3 werden die Risiken folgendermaßen eingeteilt: QM (Qualitätsmanagement) – Der Angriff ist unwahrscheinlich und die Absicherung hilft nur, die grundsätzliche Qualität eines Produktes zu steigern, Mittel bis Hoch – beschreibt den Grad des Risikos, Kritisch – Es ist extrem wichtig diese Sicherheitslücke zu schließen.

4.1.3 Feedbackkanal

Um feststellen zu können, ob ein Angriff erfolgreich war, muss überprüft werden können, ob sich der Zustand des angegriffenen Steuergerätes verändert hat. Ob und wie dies möglich ist, lässt sihc grob in drei unterschiedliche Kategorieen eingliedern. Es gibt Angriffe, die mittels SOME/IP-Paketen überprüft werden können, Angriffe deren Erfolg mit einem Diagnoseprotokoll für Steuergeräte überprüft werden kann und Angriffe, deren Evaluation nicht mit diesen Mitteln erfolgen kann. Im folgenden werden diese Klassifikationen diskutiert.

4.1.3.1 Evaluation mittels SOME/IP

Einige SOME/IP-Pakete erhalten von dem Empfänger eines Paketes eine Antwort, ob das Paket verarbeitet wurde oder nicht. bassieren Angriffe auf solchen Packeten, kann direkt an der Antwort des Steuergerätes erkannt werden, ob ein Angriff erfolgreich war oder nicht. Schwieriger wird dies, wenn der Angriff einen gefälschten Absender enthält, also wenn das Paket mit der IP-Adresse eines dritten Steuergerätes versendet wurde. In diesem Fall erhält dieses Steuergerät die Antwort des Steuergerätes, dem der Angreifer das initiale Paket versendet het. In diesem Fall muss zur Evaluation sichergestellt werden, dass der Angreifer dieses Antwortpaket ebenfalls empfangen kann. Dazu müssen Angriffe auf tiefere Netzwerkebenen geführt werden, um den Netzwerkstrom auf dem Angreifer

umzuleiten (zum Beispiel Man-in-the-Middle mittels ARP-Spoofing). Besteht keine Möglichkeit für solch einen Angriff, kann die Evaluation über diesen Weg nicht erfolgen und es müssen andere Mittel genutzt werden. Hier besteht auch die Möglichkeit, durch zusätzliche SOME/IP-Pakete den internen Zustand eines Steuergerätes zu überprüfen. Werden Getter-Methoden vom Steuergerät für Variablen angeboten, die durch einen Angriff verändert werden, kann der Angreifer durch das Abfragen dieser überprüfen, ob und wie sich der Zustand dieser Variablen verändert hat.

4.1.3.2 Evaluation über ein Diagnoseprotokoll

Fahrzeughersteller nutzen für das Entwickeln und Testen von Steuergeräten ein Diagnoseprotokoll, dass den internen Zustand eines solchen Gerätes auslesen kann. Dies erfolgt über die Automotive-Ethernet-Schnittstelle oder überandere Kommunikationsmöglichkeiten, wie dem CAN-Bus. Ein solches Diagnoseprotokoll (Universal Measurement and Calibration Protocol (XCP)) wird in vielen Fällen vom Hersteller deaktiviert, bevor das Fahrzeug den Endkunden erreicht und kann zumeist nicht ohne weiteres wieder aktiviert werden. In manchen Fällen wurde das deaktivieren des Diagnoseprotokolls durch den Hersteller allerdings unterlassen oder der Hersteller gibt ein Steuergerät mit aktivem Diagnoseprotokoll für eine Sicherheitsanalyse an eine externe Firma. Soll ein solches Steuergerät angegriffen werden, kann der Angreifer die Informationen nutzen, die dadurch erhoben werden können. So kann überprüft werden, ob das Sterugerät nach dem Empfangen eines Netzwerkpaketes den internen Zustand ändert und so der erfolg eines Angriffes evaluiert werden.

4.1.3.3 Andere Angriffe

Steht kein Diagnoseprotokoll zur Verfügung und kann auch SOME/IP nicht genutzt werden, um die Evaluierung eines Angriffes durchzuführen, kann diese Evaluierung schwirig werden. Ist es möglich, ein Steuergerät über andere Anriffsvektoren zu Kontollieren, kann der interne Zustand dieses Gerätes überprüft werden. Allerdings ist es in diesem Fall unter umständen nicht mehr vonnöten, dieses Steuergerät über SOME/IP von außen anzugreifen.

4.2 Angriffsvektoren

Um die angeführten Angriffsvektoren vergleichbar zu machen, wird für jeden Angriff ermittelt, welcher Kategorie oder welchen Kategorien 4.1.1 dieser zugeordnet werden kann und welches Risiko dieser Angriff besitzt. Also wie aufwändig er durchzuführen ist und welcher Schaden durch den Angriff zu erwarten ist. Zusätzlich wird ein Ansatz gesucht, wie dieser Angriff praktisch umzusetzen wäre. Zum Schluss wird erörtert, wie der Feedback-Kanal des Angriffs aussehen könnte, also wie überprüft werden kann, ob eine bestimmte Netzwerkkomponente für den gewählten Angriff anfällig ist.

Um eine möglichst vollständige Auflistung der Angriffsvektoren zu erhalten, werden die einzelnen Funktionen von SOME/IP nacheinander untersucht. Dabei wird beim eigentlichen SOME/IP begonnen und zuerst die Parameter des Headers von SOME/IP inspiziert. Danach wird der Fokus auf die Events, Fields,

Eventgruppen und auf RPC gelegt. Zum Schluss wird die Sicherheit der Service Discovery von SOME/IP diskutiert.

4.2.1 SOME/IP Header

Der SOME/IP Header wird mit jedem SOME/IP Paket versandt und enthält Informationen, die für die Verwendung des Protokolls vonnöten sind. Werden diese Parameter manipuliert, können die Reaktionen des Kommunikationspartners verändert werden.

4.2.1.1 Downgrading-Angriff auf die Protokoll-Version

Durch einen Downgrade der genutzten Protokoll-Version können eventuell hinzugefügte Sicherheitsmechanismen neuerer Versionen deaktiviert werden. Da zum aktuellen Zeitpunkt aber noch die erste Version von SOME/IP genutzt wird, ist dieser Angriff momentan noch nicht durchzuführen. Da in Zukunft allerdings Sicherheitsfeatures zu SOME/IP hinzugefügt werden können, kann der Downgradingangriff zu einem Sicherheitsrisiko werden.

Ziel des Angriffes ist eine Elevation of Privilege, da eventuell hinzukommende Sicherheitsmechanismen dadurch umgangen werden sollen, dass eine Protokollversion ohne diese Mechanismen genutzt wird. Für die Risikoanalyse wird von einem Threat Level von sechs ausgegangen (siehe Abschnitt 4.1.2.1). Da dieser Angriff keinen direkten Schaden anrichtet, wird der Impact Level mit eins bewertet. Dies resultiert in einem geringen Risiko.

Um über SOME/IP mit einem Service kommunizieren zu können, muss eine Verbindung zum Netzwerk bestehen. Zusätzlich wird ein Werkzeug benötigt, das in der Lage ist SOME/IP Pakete zu erstellen und zu versenden und die Antwortpakete entgegen zu nehmen. Der Feedback-Kanal dieses Angriffs erfolgt direkt durch Antwort-Pakete des Services, da die genutzte Version des Protokolls in jedem Paket mit angegeben ist (siehe Tabelle 4.4).

	Risikoeinschätzung
Angriffsziele	Elevation of Privilege
Risikoeinstufung	TL: 6, IL: 1, Risiko: Gering

	Technische Umsetzung
Voraussetzungen	Netzwerkzugriff, Werkzeug zum Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers

Tabelle 4.4: Risikoeinschätzung Downgrading-Angriff auf die Protokoll-Version

4.2.1.2 Fuzzing der Parameter im SOME/IP Header

Im Header der SOME/IP Pakete werden mehrere Parameter gesetzt. So sind die Längeninformationen des Paketes in den Bits 32 bis 63 des Headers angegeben. Wie das empfangende Steuergerät mit den Headerinformationen umgeht, hängt von dessen Implementierung ab. Die Parameter, die im Header übertragen werden, können durch Fuzzing getestet werden. Dadurch kann unvorhergesehenes Verhalten des Steuergerätes gefunden werden.
Das Ziel eines solchen Fuzzings ist in erster Linie das Finden von unerwartetem Verhalten des Steuergerätes. Für die Risikoanalyse wird von einem Threat Level von acht (siehe Abschnitt 4.1.2.1) ausgegangen. Da nicht vorhergesagt werden kann, wie ein Steuergerät reagiert, muss davon ausgegangen werden, dass ein Unfall das Resultat des unvorhergesehenen Verhaltens ist. Daher wird von einem Impact Level von vier ausgegangen, was in einem hohen Risiko resultiert.

Um den Angriff durchführen zu können, wird Zugang zum Netzwerk und ein Werkzeug benötigt, das in der Lage ist SOME/IP Pakete zu generieren und zu versenden. Die Feststellung, ob ein Angriff erfolgreich war gestaltet sich hier allerdings schwer, da nicht immer Antwortpakete vom angegriffenen Kommunikationspartner zu erwarten sind. Daher muss über andere Wege (zum Beispiel das Auslesen der Datenfelder mittels Get-Paketen oder das Abonnieren von Events) der aktuelle Zustand des Opfers ermittelt werden (siehe Tabelle 4.5).

	Risikoeinschätzung
Angriffsziele	Denial of Service
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang und Werkzeug zum Erzeugen und Versenden von SOME/IP Paketen
Art	Aktiv
Feedback Kanal	Zusätzliche Abfragen nötig

Tabelle 4.5: Risikoeinschätzung Fuzzing des SOME/IP Headers

4.2.2 Events

Events können von Services abonniert werden, wenn zyklisch Informationen über Änderungen an Datenfelder weitergegeben werden sollen. Die Events werden vom Service Dienstleister per Offer-Service-Nachrichten aus dem Service Discovery Protokoll von SOME/IP den anderen Netzwerkteilnehmern als Verfügbar angegeben.

4.2.2.1 Auslesen der Event-Daten aus fremden Paketen

Um die Informationen auslesen zu können, die durch Event-Nachrichten weitergegeben werden, können diese Pakete abgefangen und gelesen werden, da SOME/IP standardmäßig keine Verschlüsselung unterstützt.

Das Ziel eines solchen Angriffes ist es, an Informationen zu gelangen, die sonst unzugänglich wären. Für das Threat Level werden sechs Punkte vergeben (siehe Abschnitt 4.1.2.1). Da durch diesen Angriff Informationen bezogen werden können, die es unter Umständen ermöglichen, den Fahrer zu deanonymisieren, wird der Impact Level auf drei festgelegt (siehe 4.1.2.2), was ein mittleres Risiko ergibt.

Für den Angriff wird eine Anbindung an das SOME/IP Netzwerk und ein Tool benötigt, um Netzwerkverkehr mithören zu können. Zusätzlich muss der Netzwerkverkehr so umgeleitet werden, oder die Anbindung an das Netzwerk so realisiert werden, dass die benötigten Pakete mitgelesen werden können. Ist dies erfolgt, handelt es sich beim eigentlichen Angriff auf SOME/IP um einen passiven Angriff (siehe Tabelle 4.6).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 6, IL: 3, Risiko: Mittel
	Technische Umsetzung
Voraussetzungen	Netzwerkanbindung, Werkzeug zum Empfangen und Auswerten von SOME/IP Paketen, Umleitung fremder Pakete
Art	Passiver Angriff
Feedback Kanal	Direkt

Tabelle 4.6: Risikoeinschätzung Auslesen der Event-Daten aus fremden Paketen

4.2.2.2 Unautorisiertes Abonnieren von Events

Eine weitere Methode, um an die Informationen eines Events zu kommen besteht darin, Events von anderen Services zu abonnieren. Dies ist in SOME/IP ohne Authentifizierung möglich. Allerdings können die Hersteller hier zusätzliche Hindernisse einbauen, indem zum Beispiel nur Netzwerkteilnehmer, die in einer Whitelist aufgelistet sind Zugriff auf ein bestimmtes Event erlangen.

Das Ziel dieses Angriffes ist das Erlangen von Informationen über das Fahrzeug oder den Fahrer. Für die Risikoeinschätzung wird von einem Threat Level von acht Punkten ausgegangen (siehe Abschnitt 4.1.2.1). Da die erlangten Informationen einem Angreifer unter Umständen erlauben, das Fahrzeug oder den Fahrer durch Bewegungsdaten oder Ähnlichem zu identifizieren, wird von einem Impact Level von drei ausgegangen, was ein mittleres Risiko zur Folge hat.

Für die Bestätigung, ob ein Angriff erfolgreich war, also das Abonnement akzeptiert wurde, muss der Angreifer auf ein Eventpaket warten. Daher empfiehlt es sich, ein periodisches Updateverhalten des Events zu planen, da dann innerhalb einer bestimmten Zeit nach dem erfolgten Abonnement das erste Eventpaket

versendet wird. Zur Umsetzung dieses aktiven Angriffes wird eine Netzwerkanbindung und ein Werkzeug zur Durchführung benötigt. Dieses Werkzeug muss in der Lage sein, SOME/IP Pakete zu generieren, zu versenden und zu empfangen (siehe Tabelle 4.7).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 6, IL: 3, Risiko: Mittel
	Technische Umsetzung
Voraussetzungen	Netzwerkanbindung, Werkzeug für das Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Event-Pakete des Services

Tabelle 4.7: Risikoeinschätzung Unautorisiertes Abonnieren von Events

4.2.2.3 Versenden unautorisierter Event-Pakete

Da in SOME/IP keine Funktion existiert, um sicherzustellen, von welchem Absender ein Paket versendet wurde, ist ein Angreifer unter Umständen in der Lage, Pakete zu versenden, die einen gefälschten Absender besitzen. So kann ein Angreifer Event-Pakete versenden, die falsche Informationen enthalten um ein Steuergerät zu täuschen.

Ziel dieses Angriffes ist das Verbreiten von Fehlinformationen, also das Spoofing. Für die Risikoeinschätzung wird ein Threat Level von sechs (siehe Abschnitt 4.1.2.1) und ein Impact Level von vier gesetzt. Der Impact Level kommt dadurch zustande, dass ein Angreifer, der in der Lage ist ein Steuergerät zu täuschen, eventuell unvorhergesehenes Verhalten hervorrufen kann und dadurch einen Unfall mit mehreren beteiligten Personen provozieren kann. Dies ergibt ein hohes Risiko für dieses Angriffsszenario.

Für die Durchführung des Angriffes wird ein Zugang zum SOME/IP Netzwerk und eine Software als Werkzeug benötigt. Diese Software muss in der Lage sein, SOME/IP Pakete zu generieren und zu versenden. Um festzustellen zu können, ob ein Angriff erfolgreich war oder nicht, muss überprüft werden, ob das Steuergerät entsprechend den veränderten Daten reagiert. Dies ist nicht ohne weiteres möglich. Daher müssen weitere Pakete versendet werden, um den Zustand des Steuergerätes festzustellen (siehe Tabelle 4.8).

4.2.2.4 Replay fremder Events

Eine Methode, bestimmte Absicherungen zu umgehen besteht darin, zuvor aufgezeichnete Pakete erneut zu versenden. Dazu müssen allerdings zuerst die Pakete anderer Netzwerkteilnehmer abgefangen werden können. Dies kann

	Risikoeinschätzung
Angriffsziele	Spoofing
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkanbindung, Werkzeug zum Generieren und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Durch weitere Informationsgewinnung

Tabelle 4.8: Risikoeinschätzung Versenden unautorisierter Event-Pakete

durch Angriffe auf tiefere Netzwerkschichten im ISO-OSI Schichtenmodel erreicht werden. Wird ein Paket eines anderen Absenders empfangen, kann dies zu einem späteren Zeitpunkt erneut versandt werden. Das Vorgehen kann auch dann erfolgreich sein, wenn der Inhalt des Paketes durch zusätzliche Mechanismen verschlüsselt oder anderweitig kodiert ist und vom Angreifer nicht gelesen werden kann.

Ziel des Angriffes ist, im Namen eines anderen Netzwerkteilnehmers ein Paket zu versenden, ohne dass die eigene Identität bekannt gemacht wird. Für die Risikoanalyse wird ein Threat Level von sechs festgelegt (siehe Abschnitt 4.1.2.1). Da durch diesen Angriff die automatischen Sicherheitssysteme eines Fahrzeugs deaktiviert und eventuell Aktionen wie ein Lenk- oder Bremsvorgang ausgelöst werden kann, was bei schneller Fahrt einen Unfall erzeugen kann, muss der Impact Level bei der Betriebssicherheit auf vier, also auf Lebensgefahr für mehrere Personen festgelegt werden. Dadurch wird ein Gesamtrisiko von hoch ermittelt.

Für den Feedback Kanal dieses Angriffs müssen anderweitig Informationen gewonnen werden. So muss zum Beispiel der Zustand des Angriffsziels vor und nach dem Angriff ausgelesen werden, um herausfinden zu können, ob der Angriff erfolgreich war (siehe Tabelle 4.9).

	Risikoeinschätzung
Angriffsziele	Repudiation, Spoofing
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkanbindung, Werkzeug zum Empfangen und Versenden von SOME/IP Paketen, Umleitung fremder Pakete
Art	Aktiver Angriff
Feedback Kanal	Durch weitere Informationsgewinnung

Tabelle 4.9: Risikoeinschätzung Replay fremder Events

4.2.2.5 Modifikation fremder Event-Pakete

Werden fremde Pakete mitgelesen und sollen weiterverschickt werden, kann dies auch nach der Manipulation der Pakete erfolgen. Dieser Angriff wird von zusätzlichen Sicherheitsmechanismen, wie Message Authentication Codes (Mac) oder Message Integration Codes (MIC) zwar erkannt, allerdings werden diese Funktionen von SOME/IP standardmäßig nicht unterstützt und müssen zusätzlich eingeführt werden. Für diesen Angriff gelten die gleichen Voraussetzungen, die auch für die Umsetzung des Replay-Angriffs auf SOME/IP Event Ebene gelten (siehe Abschnitt 4.2.2.4).

Auch die Ziele und Risikoanalyse decken sich mit dem in 4.2.2.4 beschriebenen Angriff. So werden als Ziele die Repudiation und das Spoofing gesetzt. Das Threat Level wird auf sechs festgelegt (siehe Abschnitt 4.1.2.1), während das Impact Level wegen des Risikos eines Unfalles auf vier festgelegt wird. So wird für diesen Angriff ein hohes Risiko festgelegt.

Für den Angriff muss ein Zugang zum SOME/IP Netzwerk bestehen, die Pakete müssen auf den Angreifer umgeleitet werden und ein Werkzeug vorhanden sein, das das Mitlesen, Verändern und Versenden von SOME/IP Paketen beherrscht. Für den Feedback Kanal wird eine zusätzliche Analyse des angegriffenen Steuergeräts benötigt, um feststellen zu können, ob sich der Zustand des Steuergerätes entsprechend verändert hat (siehe Tabelle 4.10).

	Risikoeinschätzung
Angriffsziele	Repudiation, Spoofing
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch
	Technische Umsetzung
Voraussetzungen	Netzwerkanbindung, Werkzeug zum Empfangen, Verändern und Versenden von SOME/IP Paketen, Umleitung fremder Pakete
Art	Aktiver Angriff
Feedback Kanal	Durch weitere Informationsgewinnung

Tabelle 4.10: Risikoeinschätzung Modifikation fremder Event-Pakete

4.2.2.6 Denial of Service Angriff durch übermäßiges Abonnieren von Events

Bietet eine ECU ein Event an, das abonniert werden kann, muss es für jedes Abo abspeichern, wer bei einem Update ein entsprechendes Infopaket zugeschickt bekommt und in welchem Zyklus diese Pakete versandt werden sollen. Besteht keine Absicherung durch Whitelists oder Ähnlichem, können unter Umständen so viele Event Abonnements an diesen Service versendet werden, dass durch Überlauf des Speichers die Funktionen des Steuergeräts deaktiviert werden. Dazu muss der Angreifer mit einer oder mehreren IP-Adressen und vielen unterschiedlichen Client-ID's Abonnements an den Service versenden.

Für die Risikoanalyse wird von einem Threat Level von sechs ausgegangen (siehe Abschnitt 4.1.2.1). Das Impact Level wird mit vier festgelegt. Zwar kann davon ausgegangen werden, dass Fahrzeuge dementsprechend abgesichert werden, dass bei einem Ausfall eines Steuergerätes kein gefährliches Verhalten des Fahrzeuges ausgeführt wird, es kann aber nicht vorhergesagt werden, wie ein Autofahrer auf den Ausfall einzelner Systeme reagiert. Ist ein Fahrer zum Beispiel unaufmerksam, da er sich auf ein Assistenzsystem wie den Spurhalteassistent verlässt und dieser fällt bei hoher Geschwindigkeit aus, kann ein Unfall nicht ausgeschlossen werden bei dem mehrere Fahrzeuge und Personen involviert sind. Daher muss ein hohes Risiko für diesen Angriff festgelegt werden.

Nutzt der Angreifer für den Angriff auch IP-Adressen, auf deren Antwortpakete er selbst zugreifen kann, kann so der Feedback Kanal realisiert werden. Der Angreifer muss nur überprüfen, ob vom Steuergerät noch neue Abonnements angenommen und die alten Abos ebenfalls noch mit Informationen versorgt werden. Dazu müssen einige der Abonnements derart gestaltet werden, dass die Eventpakete wirklich an den Angreifer verschickt werden. Ist dies der Fall, muss mit der Zeit unweigerlich der Speicher des Steuergeräts ausgelastet werden. Vorausgesetzt wird hier, dass der Angreifer sich im Netzwerk des Automobils befindet und in der Lage ist, SOME/IP Pakete zu generieren und zu versenden (siehe Tabelle 4.11).

	Risikoeinschätzung
Angriffsziele	Denial of Service
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Durch Antwortpakete des Servers

Tabelle 4.11: Risikoeinschätzung Denial of Service Angriff durch übermäßiges Abonnieren von Events

4.2.2.7 Enumeration von Events, für die keine Offer-Service-Nachrichten versendet werden

Standardmäßig werden alle Events, die von einem Steuergerät angeboten werden durch die Service Discovery von SOME/IP verfügbar gemacht. Sind Events verfügbar, für die keine Offer-Nachrichten existieren, können diese von besonderem Interesse für Angreifer sein. Daher besteht die Möglichkeit, dass alle möglichen Event-ID's von Angreifern getestet werden und untersucht wird, für welche ID's der Servicedienstleister Antwortpakete sendet. Dieses Vorgehen ist vergleichbar mit dem Portscannen, das im Netzwerk-Bereich auf UDP oder TCP Ebene durchgeführt wird.

Ziel des Angriffes ist es, zusätzliche Informationen über den Aufbau und die Architektur des Fahrzeuges zu erlangen. Für die Risikoanalyse wird ein Threat

Level Wert von acht festgelegt, da bei diesem Angriff keine Informationen des Herstellers benötigt werden (siehe Abschnitt 4.1.2.1). Für den Impact Level wird ein Punkt vergeben, da die gewonnen Informationen nicht genutzt werden können, um den Fahrer zu identifizieren und kein anderweitiger Schaden angerichtet wird. Dies ergibt ein geringes Risiko.

Da der Service direkt mit dem Versenden von Event Informationen beginnt, wird ein existierendes Event abonniert, kann der Angreifer direkt festellen, ob ein Event existiert oder nicht. Um den Angriff zu starten wird nur ein physikalischer Zugang zum Netzwerk und ein entsprechendes Werkzeug für das versenden der SOME/IP Pakete benötigt (siehe Tabelle 4.12).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 8, IL: 1, Risiko: Gering
	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers

Tabelle 4.12: Risikoeinschätzung Enumeration von Events, für die keine Offer-Service-Nachrichten versendet werden

4.2.3 Datenfelder

Datenfelder werden in SOME/IP vergleichsweise genutzt, wie Variablen innerhalb eines Objektes bei einer objektorientierten Programmiersprache. Hier werden Getter genutzt, um diese Variable auszulesen und Setter, um den Inhalt der Variable zu verändern. Da in SOME/IP keine Sicherheitsmechanismen integriert sind, muss eine Zugriffskontrolle lesend oder schreibend durch zusätzliche Schichten hinzugefügt werden. Andernfalls ist es Angreifern möglich die verfügbaren Werte auszulesen oder zu manipulieren.

4.2.3.1 Unautorisiertes Auslesen von Datenfeldern

Werden keine zusätzlichen Sicherheitsmechanismen genutzt, ist es einem Angreifer problemlos möglich, eine Getter Nachricht an einen Service zu versenden, um den Inhalt eines Datenfeldes auszulesen.

Ziel eines solchen Angriffes ist das Erlangen von Informationen. Für die Risikoanalyse wird ein Threat Level von sechs festgelegt (siehe Abschnitt 4.1.2.1). Da durch das Erlangen dieser Informationen die Möglichkeit besteht, die Bewegungsdaten des Fahrzeuges auszulesen, und der Fahrer identifiziert werden kann, wird ein Impact Level von vier genutzt. Dies resultiert in einem mittleren Risiko.

Hierfür benötigt der Angreifer nur Zugriff auf das Netzwerk und ein Werkzeug, das in der Lage ist SOME/IP Pakete zu erzeugen, zu versenden und das Antwortpaket zu empfangen, das den Inhalt des Datenfeldes enthält. Ob der Angriff funktioniert hat, erfährt der Angreifer direkt, wenn er das Antwortpaket vom Server erhält (siehe Tabelle 4.13).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 6, IL: 3, Risiko: Mittel

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen, Versenden und Empfangen von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers

Tabelle 4.13: Risikoeinschätzung Unautorisiertes Auslesen von Datenfeldern

4.2.3.2 Unautorisiertes Ändern von Datenfeldern

Ohne zusätzlichen Sicherheitsmechanismus ist es auch möglich, Datenfelder mit der entsprechenden Setter Methode zu verändern.

Ziel des Angriffes ist, Fehlinformationen in einem Steuergerät zu verbreiten. Die hohe Risikoeinstufung wird durch einen Threat Level von sechs (siehe Abschnitt 4.1.2.1) und einen Impact Level von vier erzeugt. Der Impact Level rührt von der Tatsache, dass durch Manipulation der Variablen in einem Steuergerät das Fahrzeug zu einem Unfall gebracht werden kann. Dies ergibt ein hohes Risiko.

Für einen erfolgreichen Angriff wird nur eine Anbindung an das interne Netzwerk und ein Werkzeug, das in der Lage ist SOME/IP Pakete zu versenden, benötigt. Da die Setter Methoden von SOME/IP ein Antwortpaket erzeugen, die das Setzen einer Variable bestätigen, wird ein erfolgreicher Angriff direkt bestätigt. Zusätzlich kann nach dem Ändern der Variable die Getter Methode genutzt werden, um das Resultat auszulesen (siehe Tabelle 4.14).

4.2.3.3 Enumeration von Datenfeldern, für die keine Offer-Service-Nachrichten versandt werden

Um zu erfahren, welche Datenfelder zur Verfügung stehen, wird die Service Discovery von SOME/IP genutzt. Allerdings kann der Hersteller diese Informationen auch in einer Manifest Datei fest in die Clients einspeichern und keine Service Discovery Pakete versenden. Um solche versteckten Datenfelder zu finden, müssen die möglichen ID's getestet werden, indem getField Pakete an die

	Risikoeinschätzung
Angriffsziele	Spoofing
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Generieren und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers, Nutzen der Getter Methode

Tabelle 4.14: Risikoeinschätzung Unautorisiertes Ändern von Datenfeldern

entsprechenden Steuergeräte mit den möglichen ID's versendet werden.

Ziel dieses Angriffs ist das Auslesen zusätzlicher Informationen. Diese Informationen können genutzt werden, um einen weiteren Angriff zu starten. Für die Risikoanalyse wird daher von einem Threat Level von acht ausgegangen (siehe Abschnitt 4.1.2.1). Dieser resultiert daraus, dass keine internen Informationen des Herstellers benötigt werden. Der Impact Level wird auf eins festgelegt, da es sich bei den gefundenen Informationen um anonyme Daten handelt. Dies resultiert in einem geringen Risiko.

Voraussetzung für diesen Angriff ist eine Verbindung an das SOME/IP Netzwerk und ein Werkzeug zum generieren der Getter Pakete. Da der Server direkt auf eine Getter Nachricht antwortet, kann dies dem Angreifer als Feedback für einen geglückten Angriff, also ein gefundenes Datenfeld, dienen (siehe Tabelle 4.15).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 8, IL: 1, Risiko: Gering

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers

Tabelle 4.15: Risikoeinschätzung Enumeration von Datenfeldern, für die keine Offer-Service-Nachrichten versandt werden

4.2.4 Remote Procedure Call (RPC)

Bei einem Remote Procedure Call (RPC) wird eine Methode direkt auf einem entfernten Steuergerät aufgerufen und als Antwort ein Paket mit dem Rückgabewert verschickt. Enthält die Antwort den Rückgabewert Null, so wurde die Funktion fehlerlos ausgeführt. Wird diese Funktion, wie standardmäßig bei SOME/IP

ohne zusätzliche Authentifizierung ausgeführt, kann dieser Mechanismus für Angriffe genutzt werden.

4.2.4.1 Mitlesen von RPC-Methoden-Aufrufen und deren Antworten

Werden Angriffe auf niedrigeren Netzwerkschichten genutzt, um den Netzwerkverkehr auf die Schnittstelle des Angreifers weiterzuleiten, ist dieser in der Lage die empfangenen Pakete aufzuzeichnen und auszuwerten. Dies ist auch bei den Methodenaufrufen und den Antwortpaketen des Servers möglich.

Ziel des Angriffes ist das Auslesen von Informationen. Für die Risikoanalyse werden für das Threat Level sechs Punkte vergeben (siehe Abschnitt 4.1.2.1). Dies ergibt mit einem Impact Level von drei ein mittleres Risiko. Dieses Impact Level ergibt sich durch die eventuelle Möglichkeit des Angreifers, den Fahrer und das Fahrzeug zu orten und somit zu identifizieren.

Für den Angriff wird ein Zugang zum Netzwerk und ein Netzwerksniffer benötigt. Um die Pakete interpretieren zu können, benötigt der Sniffer eine Integration von SOME/IP. Um die Pakete mit einem Sniffer aufzeichnen zu können, müssen die Pakete der anderen Netzwerkteilnehmer empfangen werden können. Dazu wird ein Angriff auf eine niedrigere Netzwerkschicht benötigt, um dies zu erreichen (siehe Tabelle 4.16).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 6, IL: 3, Risiko: Mittel
	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Angriff auf niedrigere Netzwerkschicht, Werkzeug zum Aufzeichnen von Netzwerkpaketen
Art	Passiver Angriff
Feedback Kanal	Durch das auslesen der Pakete

Tabelle 4.16: Risikoeinschätzung Mitlesen von RPC-Methoden-Aufrufen und deren Antworten

4.2.4.2 Unautorisierter Aufruf von RPC-Methoden

Werden von einem Service RPC-Methoden zur Verfügung gestellt, können diese von anderen Netzwerkteilnehmern aufgerufen werden, sofern kein zusätzlicher Sicherheitsmechanismus dies verhindert. In diesem Falle ist ein Angreifer in der Lage unautorisiert Methoden aufzurufen und die Ergebnisse auszulesen.

Ziel eines solchen Angriffes ist das Ausführen von Methoden, die eigentlich nur von berechtigten Netzwerkteilnehmern ausgeführt werden sollten. Für die Risikoanalyse wird von einem Threat Level von sechs ausgegangen (siehe Abschnitt 4.1.2.1). Für das Impact Level werden vier Punkte vergeben, da durch

das Ausführen von Methoden der Zustand von Steuergeräten verändert werden kann und dadurch Unfälle provoziert werden können. Dies ergibt im Gesamten ein hohes Risiko.

Voraussetzung für diesen Angriff ist ein Zugriff auf das SOME/IP Netzwerk des Fachzeuges. Außerdem wird ein Werkzeug benötigt, das in der Lage ist SOME/IP Pakete zu versenden. Da bei einem Aufruf einer RPC-Methode, wie auch beim Aufruf einer C-Funktion ein Rückgabewert in Form einer Integer Zahl erfolgt, kann ein Angreifer direkt durch diese Antwort beurteilen, ob ein Angriff erfolgreich war oder nicht (siehe Tabelle 4.17).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zur Generierung von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers

Tabelle 4.17: Risikoeinschätzung Mitlesen von RPC-Methoden-Aufrufen und deren Antworten

4.2.4.3 Manipulation fremder RPC-Methoden-Aufrufe und deren Antwortpakete

Wird ein Angriff auf niedrigerer Netzwerkebene durchgeführt, kann aufgrund der fehlenden Signatur von SOME/IP eine Manipulation von RPC-Methoden-Aufrufen und den Antwortpaketen des Servers erfolgen, wenn die Netzwerkkommunikation durch einen Angreifer abgefangen und weitergeleitet wird.

Ziel eines solchen Angriffs ist das Manipulieren, also das Spoofing von Informationen, ohne dass zurückverfolgt werden kann, wer der eigentliche Absender des Paketes ist. Dafür wird ein vorhandenes Paket genutzt, das verändert wird. Für die Risikoanalyse wird beim Threat Level eine Punktzahl von sechs Punkten erreicht (siehe Abschnitt 4.1.2.1). Da durch die Manipulation der Pakete Fehlverhalten der Steuergeräte hervorgerufen werden kann, was Unfälle provozieren kann, wird der Impact Level auf vier, also auf lebensgefährlich für mehrere Personen festgelegt.

Um diesen Angriff durchführen zu können, muss Zugriff auf das SOME/IP Netzwerk bestehen und ein Angriff auf niedrigerer Netzwerkebene durchgeführt werden, um die Netzwerkpakete zum angreifenden Netzwerkteilnehmer umleiten zu können. Dann wird ein Werkzeug benötigt, das Netzwerkpakete empfangen und manipulieren kann. Um zu erkennen ob ein Angriff erfolgreich gewesen ist, muss der Angreifer das Antwortpaket des Servers interpretieren (siehe Tabelle 4.18).

	Risikoeinschätzung
Angriffsziele	Spoofing und Repudiation
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Angriff auf niedrigerer Netzwerkebene, Werkzeug zur Manipulation von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Analyse des Antwortpaketes

Tabelle 4.18: Risikoeinschätzung Manipulation fremder RPC-Methoden-Aufrufe und deren Antwortpakete

4.2.4.4 Denial of Service Angriff durch übermäßiges Aufrufen von RPC-Methoden

Wird der Zugriff auf die RPC-Methoden nicht durch Sicherheitsmechanismen geregelt, kann ein Angreifer Methodenaufrufe starten, die das angegriffene Steuergerät ausführt. Verbraucht das Ausführen dieser Methode genügend Ressourcen oder hat der Angreifer genügend Netzwerkkapazitäten, kann der Angreifer das Steuergerät mit der Ausführung der Methode vollständig auslasten und es für andere Aufgaben unerreichbar machen. Dazu muss der Angreifer die Methode in genügender Häufigkeit aufrufen.

Ziel dieses Angriffes ist ein Denial of Service, also die Verfügbarkeit eines Dienstes im Negativen zu beeinflussen. Für die Risikoanalyse wird von einem Threat Level von sechs ausgegangen (siehe Abschnitt 4.1.2.1). Der Impact Level wird mit vier festgelegt, da nicht direkt in die Steuerung des Fahrzeuges eingegriffen wird, viele Fahrer aber so durch ausfallende Systeme, wie Spurhalteassistenz, abgelenkt werden, dass Unfälle möglich werden. Wichtige Sicherheitseinrichtungen von Fahrzeugen, wie die Bremse, darf aber durch Sicherheitsbestimmungen, die die Fahrsicherheit betreffen nicht durch ausfallende Steuergeräte beeinflusst werden. Mit diesen Sicherheitsüberlegungen wird ein hohes Risiko ermittelt.

Für die Durchführung dieses Angriffes wird ein Zugang zum Netzwerk und ein Werkzeug benötigt, das SOME/IP Pakete erzeugen und versenden kann. Da für jede aufgerufene Methode ein Antwortpaket generiert wird, kann der Angreifer sehr gut nachvollziehen, ob der entsprechende Methodenaufruf ausgeführt wurde oder nicht (siehe Tabelle 4.19).

4.2.4.5 Fuzzing der Parameter von RPC-Methoden-Aufrufen

RPC-Methoden unterstützen, wie auch Funktionen in der Programmiersprache C, die Übergabe von Parametern beim Aufrufen einer Methode. Hierbei kann vom Angreifer getestet werden, wie das getestete Steuergerät auf verschiedene Parameter reagiert. Bei diesem Fuzzing genannten Angriff werden die verschiedenen Parameter generiert und an die RPC-Methode übertragen. Ein Orakel wird genutzt, das voraussagen kann, wie die Antwort des Steuergeräts aussehen

	Risikoeinschätzung
Angriffsziele	Denial of Service
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Analyse des Antwortpaketes

Tabelle 4.19: Risikoeinschätzung Denial of Service Angriff durch übermäßiges Aufrufen von RPC-Methoden

müsste um das Ergebnis zu evaluieren.

Das Ziel dieses Angriffes ist es, ein angegriffenes Steuergerät in einen Zustand zu bringen, der vom Programmierer nicht vorausgesagt wurde, indem bislang ungetestete Eingabeparameter getestet werden. Innerhalb der STRIDE Methode lässt sich dieses Ziel am besten in der Kategorie Elevation of Privilege einordnen. Für die Risikoanalyse wird von einem Threat Level von sechs (siehe Abschnitt 4.1.2.1) und einem Impact Level von vier (siehe 4.1.2.2) ausgegangen. Daher ergibt die Analyse ein hohes Risiko.

Für dieses Angriffsszenario wird ein Zugang zum Netzwerk und ein Werkzeug benötigt, das SOME/IP Pakete erzeugen kann. Außerdem muss ein Orakel in Form einer zweiten Implementierung oder einer Abschätzung vorliegen, um die Ergebnisse evaluieren zu können (siehe Tabelle 4.20).

	Risikoeinschätzung
Angriffsziele	Elevation of Privilege
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum erstellen und versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Orakel zur Auswertung des Antwortpaketes

Tabelle 4.20: Risikoeinschätzung Fuzzing der Parameter von RPC-Methoden-Aufrufen

4.2.4.6 Enumeration von RPC-Methoden, für die keine Offer-Service-Nachrichten versandt werden

Da nicht alle verfügbaren RPC-Methoden durch Service Discovery Pakete angegeben werden müssen, kann ein Durchprobieren aller möglicher RPC-Methoden ID's neue verfügbare RPC-Methoden zutage fördern. Hierbei muss allerdings beachtet werden, wie ein Steuergerät mit einem Methodenaufruf für eine RPC-

Methode umgeht, für die die angegebenen Parameter nicht mit den von der Methode erwarteten übereinstimmt.

Ziel eines solchen Angriff ist es, bisher unbekannte RPC-Methoden ID's zu finden. Für die Risikoeinschätzung wird hier von einem Threat Level von acht (siehe Abschnitt 4.1.2.1) und einem Impact Level von eins (siehe 4.1.2.2) ausgegangen. Die ergibt ein geringes Risiko.

Um diesen Angriff durchführen zu können, benötigt der Angreifer Zugang zum Netzwerk und ein Werkzeug zum Erzeugen von SOME/IP Paketen. Wird ein RPC-Methoden-Aufruf akzeptiert, erhält der Angreifer ein entsprechendes Antwortpaket (siehe Tabelle 4.21).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 8, IL: 1, Gering

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum erstellen und versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers

Tabelle 4.21: Risikoeinschätzung Enumeration von RPC-Methoden, für die keine Offer-Service-Nachrichten versandt werden

4.2.5 Service Discovery

Die Service Discovery von SOME/IP wird dazu genutzt, um anderen Teilnehmern des SOME/IP Netzwerkes die eigenen Services als verfügbar bekannt zu machen und um Services anderer Steuergeräte zu finden.

4.2.5.1 Versenden von unautorisierten Find-Service-Nachrichten

Durch das Versenden von Find-Service Nachrichten können Informationen über die von anderen Steuergeräten angebotenen Services wie Events, RPC-Methoden oder Datenfeldern eingeholt werden. Wird innerhalb der SOME/IP-SD Pakete kein zusätzlicher Sicherheitsmechanismus eingerichtet, kann dies jeder Teilnehmer des SOME/IP Netzwerkes bewerkstelligen.

Ziel dieses Angriffes ist das Erlangen zusätzlicher Informationen. Da die Informationen selbst nicht genutzt werden können um den Fahrer zu lokalisieren, wird von einem Impact Level von eins ausgegangen. Dies kommt dadurch Zustande, dass nur anonyme Daten erfasst werden können. Dies ergibt mit einem Threat Level von sechs (siehe Abschnitt 4.1.2.1) ein geringes Risiko.

Um den Angriff durchführen zu können, muss Zugang zum SOME/IP Netzwerk bestehen und ein Werkzeug vorhanden sein, das das Erstellen und Versenden von SOME/IP Paketen erlaubt. Der Angriff ist dann erfolgreich, wenn die anderen Teilnehmer des SOME/IP Netzwerkes auf die versendenten Find-Service Pakete mit Offer-Service Paketen antworten (siehe Tabelle 4.22).

	Risikoeinschätzung
Angriffsziele	Information Disclosure
Risikoeinstufung	TL: 6, IL: 1, Risiko: Gering

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete des Servers

Tabelle 4.22: Risikoeinschätzung Versenden von unautorisierten Find-Service-Nachrichten

4.2.5.2 Versenden von falschen Offer-Service-Nachrichten

Benötigt ein Steuergerät einen Service, kann er die SOME/IP-SD Nachrichten nutzen, um herauszufinden, welcher anderen Netzwerkteilnehmer diesen anbietet. Dies kann genutzt werden, indem vom Angreifer falsche Offer-Service Nachrichten versandt werden und das Opfer dadurch dazu gebracht wird den falschen Service des Angreifers zu abonnieren. Dadurch ist der Angreifer in der Lage Fehlinformationen in das Netzwerk einzubringen.

Ziel eines solchen Angriffs ist, dass der Angreifer als ein reguläres Steuergerät im SOME/IP Netzwerk anerkannt wird und dessen Services abonniert werden. Für die Risikoanalyse muss hier von einem Impact Level von vier ausgegangen werden, da das angegriffene Steuergerät je nach Implementierung der Software den eigentlichen zu nutzenden Service nicht weiter verwendet und keine oder fehlerhafte Informationen erhält. Dadurch kann wieder ein Unfall Provoziert werden. Mit dem Threat Level von sechs (siehe Abschnitt 4.1.2.1) wird ein hohes Risiko erreicht.

Für den Angriff wird ein Zugang zum SOME/IP Netzwerk und ein Werkzeug zum Erzeugen und Versenden von SOME/IP Paketen benötigt. Versendet ein Angreifer Offer-Nachrichten, erkennt er, ob dies erfolgreich war, wenn andere Steuergeräte den angebotenen Dienst abonnieren (siehe Tabelle 4.23).

4.2.5.3 Versenden von falschen Stop-Offer-Service-Nachrichten

Wird ein Service nicht länger von einem Steuergerät angeboten, kann es dies durch das Versenden einer Stop-Offer-Service-Nachricht mitteilen. Ein Angreifer kann diesen Pakettyp nutzen, indem er im Namen eines anderen diese

	Risikoeinschätzung
Angriffsziele	Spoofing
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete der Clients

Tabelle 4.23: Risikoeinschätzung Versenden von falschen Offer-Service-Nachrichten

Nachrichten versendet. Je nach Implementierung hat dieser Angriff keine Auswirkung, da der Client Find-Service-Nachrichten nutzen kann, um den Service erneut zu finden und zu abonnieren. Der Angreifer kann aber auch wiederholt Stop-Offer-Service-Nachrichten versenden, um den Service dauerhaft zu blockieren.

Angriffsziel ist ein Denial of Service (DoS), da der Service für die anderen Netzwerkteilnehmer als nicht mehr verfügbar markiert wird. Für die Risikoanalyse wird von einem Threat Level von sechs ausgegangen (siehe Abschnitt 4.1.2.1). Da durch einen DoS-Angriff Services nicht mehr erreichbar sein können, kann der Fahrer abgelenkt werden und ein Unfall provoziert werden. Daher wird ein Impact Level von 4 festgelegt. Das ergibt ein hohes Risiko durch diesen Angriff.

Für den Angriff wird ein Netzwerkwerkzeug benötigt, das in der Lage ist SOME/IP Pakete erstellen und verschicken zu können. Um festzustellen, ob ein Angriff erfolgreich war, muss das Netzwerk belauscht werden, da der Angriff keine direkt erkennbare Auswirkung hat (siehe Tabelle 4.24).

	Risikoeinschätzung
Angriffsziele	Denial of Service
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch

	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Zusätzliche Netzwerkanalyse

Tabelle 4.24: Risikoeinschätzung Versenden von falschen Stop-Offer-Service-Nachrichten

4.2.5.4 Denial of Service über übermäßiges Versenden von Offer-Nachrichten

Werden Offer-Service Nachrichten empfangen, muss ein Steuergerät entscheiden, ob es sich diesen Service merkt, damit die Adresse bekannt ist, sollte der Service später benötigt werden. Ist die Software auf dem Steuergerät so programmiert, dass sie alle Services für späteren Bedarf speichert, kann sie eventuell durch Versenden sehr vieler offer-Nachrichten dazu gebracht werden, den gesamten Speicher voll zu schreiben.

Ziel des Angriffes ist ein Denial of Service. Für die Risiko Analyse wird von einem Threat Level von acht ausgegangen (siehe Abschnitt 4.1.2.1). Der Impact Level wird mit vier festgelegt, da nicht direkt in die Steuerung des Fahrzeuges eingegriffen wird, viele Fahrer aber so durch ausfallende Systeme, wie Spurhalteassistenz, abgelenkt werden, dass Unfälle möglich werden. Dies ergibt ein hohes Risiko.

Für den Angriff wird ein Zugang zum Netzwerk benötigt. Außerdem müssen SOME/IP Pakete generiert und versendet werden. Der Feedback Kanal stellt ein Problem dar, da nicht einfach festgestellt werden kann, ob die Service Pakete von nicht benötigten Services aufgezeichnet werden oder nicht (siehe Tabelle 4.25).

	Risikoeinschätzung
Angriffsziele	Denial of Service
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch
	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Erst bei Ausfall des Services

Tabelle 4.25: Risikoanalyse Denial of Service über übermäßiges Versenden von offer Nachrichten

4.2.5.5 Abonnentenstatus eines anderen Services ändern

Ohne zusätzliche Sicherheitsmechanismen wird nicht sichergestellt, welcher Netzwerkteilnehmer ein Service Discovery Paket versendet hat. Daher kann ein Angreifer im Namen eines anderen Netzwerkteilnehmers einen Service abonnieren oder das Abonnement beenden.

Ziel dieses Angriffes ist das Ändern des Zustands eines Abonnements fremder Steuergeräte. Für die Risikoanalyse des Angriffs wird davon ausgegangen, dass es einem Angreifer möglich ist, einen Service eines dritten Steuergerätes zu deabonnieren, dass dieser die benötigten Informationen nicht länger erhält und deshalb fehlerhafte Entscheidungen trifft. Dies führt zu einem nicht vorhersagbaren Verhalten, was Unfälle provozieren kann. Darum wird ein Impact Level von vier (Lebensgefahr mehrerer Personen) festgelegt. Dies ergibt mit einem

Threat Level von acht (siehe Abschnitt 4.1.2.1) ein hohes Risiko.

Für den Angriff wird ein Zugang zum SOME/IP Netzwerk benötigt. Zusätzlich muss ein Werkzeug vorhanden sein, das in der Lage ist SOME/IP Pakete zu erzeugen und zu versenden. Um feststellen zu können, ob der Angriff erfolgreich war, muss der Angreifer erkennen können, ob der Server die Zustandsänderung des Abonnements akzeptiert hat. Dazu muss der Angreifer allerdings in der Lage sein, Netzwerkpakete anderer Teilnehmer zu empfangen. Dies bedarf eines weiteren Angriffs auf niedrigerer Netzwerkebene (siehe Tabelle 4.26).

	Risikoeinschätzung
Angriffsziele	Spoofing
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch
	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Weiterer Angriff auf niedrigerer Netzwerkebene nötig

Tabelle 4.26: Risikoanalyse Abonnentenstatus eines anderen Services ändern

4.2.6 Kombinierte Angriffe

Für einige der zuvor diskutierten Angriffe werden verschiedene Angriffe kombiniert, um ein vorgesehenes Ziel zu erreichen. Dabei werden Angriffe auf niedrigere Netzwerkschichten genutzt, um den Datenverkehr auf die Adresse des Angreifers umzuleiten. Es können allerdings auch verschiedene Angriffe auf SOME/IP kombiniert werden, um neue Angriffsszenarien zu generieren. Diese Art von Angriff wird im Folgenden beschreiben.

4.2.6.1 Senden von Stop-Offer-Service-Nachrichten zur Unterstützung von Offer-Service-Nachrichten

Der Angriff „Versenden von falschen Offer-Service-Nachrichten“, der in Abschnitt 4.2.5.2 diskutiert wird, hat das Ziel einem anderen Steuergerät einen gefälschten Service anzubieten. Falls die Implementation des Steuergerätes vorsieht, keine neueren Offer-Nachrichten zu registrieren, kann eine Stop-Offer-Nachricht genutzt werden, um das Steuergerät auf das Empfangen der Offer-Nachricht vorzubereiten.

Ziel dieses Angriffes ist es, das Opfer zu täuschen und den eigenen Service anzubieten. Für die Risikoanalyse wird von einem Thread Level von sechs (siehe 4.1.2.1) und einem Impact Level von vier ausgegangen. Der Impact Level kommt dadurch zustande, dass dem Opfer Fehlinformationen zugespielt werden und so Sicherheitsmechanismen umgangen werden können.

Für den Angriff wird ein Zugang zum SOME/IP Netzwerk und ein Werkzeug zum Erzeugen und Versenden von SOME/IP Paketen benötigt. Versendet ein Angreifer Offer-Nachrichten, erkennt er, ob dies erfolgreich war, wenn andere Steuergeräte den angebotenen Dienst abonnieren (siehe Tabelle 4.27).

	Risikoeinschätzung
Angriffsziele	Spoofing
Risikoeinstufung	TL: 6, IL: 4, Risiko: Hoch
	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Antwortpakete der Clients

Tabelle 4.27: Risikoeinschätzung Senden von Stop-Offer-Service-Nachrichten zur Unterstützung von Offer-Service-Nachrichten

4.2.6.2 Man-in-the-Middle mit SOME/IP

Soll ein Man-in-the-Middle Angriff nur auf der Netzwerkschicht von SOME/IP erfolgen, können die Angriffe „Unautorisiertes Abonnieren von Events“ (siehe 4.2.2.2) und „Versenden unautorisierter Event-Pakete“ (siehe 4.2.2.3) kombiniert werden. Dazu wird dem anzugreifenden Steuergerät eine Offer-Nachricht mit dem entsprechenden Event verschicken. Ist die Software des Steuergerätes so implementiert, dass immer die aktuellste Offer-Nachricht akzeptiert wird, wird das Steuergerät das eigene Event abonnieren. Um die Informationen, die dieses Event versendet zu erhalten, kann nun das Event abonniert werden, das vom Opfer ursprünglich abonniert wurde. Nun können die erhaltenen Eventdaten weitergeleitet und bei Bedarf manipuliert werden.

Durch die Kombination der Angriffe und die damit verbundene Kontrolle über den Datenfluss wird eine große Anzahl von Schutzzielen verletzt. So gibt sich der Angreifer dem Opfer gegenüber als anderes Steuergerät aus, kann die Eventinformationen aus den empfangenen Eventdaten auslesen und bei Bedarf kann das Opfer so mit Fehlinformationen versorgt werden, dass es seinen Dienst nicht weiter anbieten kann. Für die Risikoanalyse wird von einem Threat Level von acht (siehe Abschnitt 4.1.2.1) und einem Impact Level von vier ausgegangen. Der Impact Level kommt durch die Möglichkeit des Angreifers, Unfälle durch Fehlfunktionen der Steuergeräte zu erzeugen, zustande. Dies ergibt ein hohes Risiko für dieses Angriffsszenario.

Für diesen Angriff wird ein Zugang zum Netzwerk vorausgesetzt. Außerdem wird ein Werkzeug benötigt, das Netzwerkpakete aufzeichnen, auswerten, manipulieren, erstellen und verschicken kann. Ob der Angriff erfolgreich war, wird direkt ersichtlich, wenn das Opfer den eigenen Service abonniert. Allerdings kann dabei nicht festgestellt werden, ob das originale Event deabonniert wurde. Um dies herausfinden zu können, wird ein Angriff auf tiefere Netzwerkebenen

benötigt, um fremde Netzwerkpakete zu empfangen (siehe Tabelle 4.28).

	Risikoeinschätzung
Angriffsziele	Spoofing, Information Disclosure und Denial of Service
Risikoeinstufung	TL: 6, IL: 4; Risiko: Hoch
	Technische Umsetzung
Voraussetzungen	Netzwerkzugang, Werkzeug zum Empfangen, Erstellen und Versenden von SOME/IP Paketen
Art	Aktiver Angriff
Feedback Kanal	Direkt

Tabelle 4.28: Risikoanalyse Man in the Middle mit SOME/IP

4.3 Zusammenfassung der Angriffsvektoren

In den Tabellen 4.34, 4.33, 4.32, 4.31, 4.30 und 4.29 werden die identifizierten Angriffsvektoren nochmal zusammgefasst dargestellt.

4.4 Zu implementierende Module

Anhand der erstellten Angriffsvektoren kann nun ermittelt werden, welche Angriffsmodule entwickelt werden müssen und welche Vektoren diese abdecken.

Angriff	**Ziel**	**Risiko**	**Feedback**
Downgrading-Angriff auf die Protokoll-Version	E	Gering	✓
Fuzzing der Parameter im SOME/IP Header	S & I	Hoch	?

Tabelle 4.29: Zusammenfassung Angriffsvektoren SOME/IP Header

Angriff	**Ziel**	**Risiko**	**Feedback**
Auslesen der Event-Daten aus fremden Paketen	I	Mittel	✓
Unautorisiertes Abonnieren von Events	I	Mittel	✓
Versenden unautorisierter Event-Pakete	S	Hoch	?
Replay fremder Events	S	Hoch	?
Modifikation fremder Event-Pakete	S	Hoch	?
Denial of Service Angriff durch übermäßiges Abonnieren von Events	D	Hoch	✓
Enumeration von Events, für die keine Offer-Service-Nachrichten versendet werden	I	Gering	✓

Tabelle 4.30: Zusammenfassung Angriffsvektoren Events

Angriff	Ziel	Risiko	Feedback
Unautorisiertes Auslesen von Datenfeldern	I	Mittel	✓
Unautorisiertes Ändern von Datenfeldern	S	Hoch	✓
Enumeration von Datenfelder, für die keine Offer-Service-Nachrichten versandt werden	I	Gering	✓

Tabelle 4.31: Zusammenfassung Angriffsvektoren Datenfelder

Angriff	Ziel	Risiko	Feedback
Mitlesen von RPC-Methoden und Aufrufen derer Antworten	I	Mittel	✓
Unautorisierter Aufruf von RPC-Methoden	S	Hoch	✓
Manipulation fremder RPC-Methoden-Aufrufe und deren Antwortpaketen	S & T	Hoch	?
Denial of Service Angriff durch übermäßiges Aufrufen von RPC-Methoden	D	Hoch	?
Fuzzing der Parameter von RPC-Methoden-Aufrufen	S & I	Mittel	✓
Enumeration von RPC-Methoden, für die keine Offer-Service-Nachrichten versandt werden	I	Gering	✓

Tabelle 4.32: Zusammenfassung Angriffsvektoren RPC-Methoden

Angriff	Ziel	Risiko	Feedback
Versenden von unautorisierten find-Service-Nachrichten	I	Gering	✓
Versenden von falschen Offer-Service-Nachrichten	S	Hoch	✓
Versenden von falschen Stop-Offer-Service-Nachrichten	D	Hoch	?
Denial of Service über übermäßiges Versenden von Offer-Nachrichten	D	Hoch	?
Abonnentenstatus eines anderen Services ändern	S	Hoch	?

Tabelle 4.33: Zusammenfassung Angriffsvektoren Service-Discovery

Angriff	Ziel	Risiko	Feedback
Senden von Stop-Offer-Service-Nachrichten zur Unterstützung von Offer-Service-Nachrichten	S	Hoch	✓
Man-in-the-Middle mit SOME/IP	S & I & D	Hoch	✓

Tabelle 4.34: Zusammenfassung Angriffsvektoren Kombinierte Angriffe

Aus zeitlichen Gründen können während der Durchführung dieser Arbeit nicht genügend Module implementiert werden, um alle ermittelten Angriffsvektoren abzudecken. Es wurden zwei Module implementiert. Dabei handelt es sich um das Information Gathering und das Subscribe Modul.

Das Information Gathering Modul wurde als erstes zu entwickelndes Modul gewählt, das das Sammeln von Informationen bei einem Angriff oder Sicherheitstest immer zuerst durchgeführt wird und die Informationen, die erlangt werden von großer Bedeutung für das weitere Vorgehen und den Erfolg eines Angriffes sind.

Als weiteres zu implementierendes Modul wurde das Subscribe Modul gewählt, da dadruch gleich mehrere Angriffe realisiert werden können und es ein Teil des Man-in-the-Middle Angriffes bildet.

5 SECURITY TESTING FRAMEWORK FÜR SOME/IP

Im folgenden Abschnitt wird das Vorgehen bei der Implementierung des Frameworks erläutert. Dabei werden Entscheidungen und Problemlösungen bei der Umsetzung erläutert und auf die Evaluierung des Frameworks eingegangen. Dazu wurden zuerst die Anforderungen analysiert, die das Framework an die Netzwerkbibliothek stellt, die als Basis für die Kommunikation und die Paketanalyse genutzt wird. Anschließend wurden die Anforderungen untersucht, die an das Framework selbst gestellt werden. Mit diesen Informationen wurde dann ein Plan für die Implementierung des Frameworks erstellt und die Implementierung selbst durchgeführt.

5.1 Anforderungen an das Framework

Da das Framework innerhalb einer Abschlussarbeit nicht den kompletten Funktionsumfang erreichen kann, der für einen ausführlichen Test der SOME/IP Implementierung eines Fahrzeugbauteils benötigt wird, soll es modular aufgebaut werden. Dies ermöglicht auch zukünftig ein einfache Erweiterung des Frameworks, wenn neue Erkenntnisse über SOME/IP-Angriffe gewonnen werden. In der Grundfunktionalität soll hierbei das Empfangen und Versenden von SOME/IP Paketen implementiert werden, das für die verschiedenen Angriffe benötigt wird. Außerdem soll die Funktionalität bestehen, als Quelle für die SOME/IP Paketanalyse sowohl live Daten aus einem Netzwerk zu empfangen, als auch diese Daten aus einem Netzwerkdump zu beziehen. Der Zustand den das Programm bei der Benutzung hat, also sowohl die schon eingelesenen Daten, als auch die gesetzten Einstellungen und die geladenen Module sollen als Projekt abgespeichert und zu einem späteren Zeitpunkt wieder geladen werden können.

5.1.1 Anforderungen Modularität

Um ein möglichst modulares Framework zu erstellen, werden alle nicht internen Funktionen als Module realisiert. Dazu muss ein Standard erstellt werden, nach dem diese Module erstellt werden. Es müssen unterschiedliche Typen von Modulen realisiert werden, damit Angriffe und zusätzliche Funktionen als Module ermöglicht werden. Diese werden in Funktionsmodule und Angriffsmodule eingeteilt. Angriffsmodule bieten hierbei eine grafische Benutzeroberfläche, um die einzelnen Angriffe steuern zu können. Diese Module müssen auch die Anforderungen erfüllen, die an das gesamte Framework gestellt werden. So müssen die getroffenen Einstellungen speicherbar und ladbar gehalten werden, dass diese Einstellungen im Projekt mit abgespeichert werden können. Dadurch ist es auch möglich Angriffe, die aus verschiedenen Angriffsmodulen erstellt werden, wie ein Stop-Offer-Angriff, aus mehreren Modulen zu erstellen und diesen Angriff als Projekt abzuspeichern. Die Angriffsmodule müssen auch, wenn möglich, eine Auswertung des Erfolgs eines Angriffes bieten. Funktionsmodule können Funktionen, wie abweichende Spezifikationen durch Hersteller in das Framework integrieren. Die Module müssen durch die Oberfläche des Frameworks

dynamisch nachgeladen werden können. Um eine einfache Erweiterbarkeit des Frameworks ermöglichen zu können, müssen die Module einfach implementierbar und in das Framework integrierbar sein.

5.1.2 Anforderungen in die Module

Im folgenden Abschnitt werden die Anforderungen definiert, die an die beiden während der Arbeit implementierten Module gestellt werden.

5.1.2.1 Information Gathering Modul

Beim Information Gathering Modul handelt es sich um ein Modul, das die gesammelten Informationen möglichst übersichtlich anzeigen soll und aktiv neue Informationen aus einem bestehenden Netzwerk auslesen können soll. Durch dieses Modul soll es ermöglicht werden, den Aufbau und die Funktionalität eines Systems zu extrahieren. Dazu werden die Informationen in Tabellen angezeigt, die in der Detailstufe angepasst werden können, um die angezeigten Informationen an die Anforderungen eines Tests anpassen zu können. Diese Informationen sollen sich nach Steuergeräten, Services bis hin zu einzelnen Paketen auswerten lassen.

5.1.2.2 Man-in-the-Middle Angriff

Der Man-in-the-Middle Angriff besteht aus den Einzelangriffen Abonnieren, Deabonnieren, Stop-Offer und Offer. Diese werden in einzelne Module aufgeteilt, die im verlauf der Arbeit teilweise implementiert werden sollen. Dabei handelt es sich um die Angriffe Stop-Offer, Offer, Deabonnieren eines fremden Abonnements und Abonnieren eines Events. Durch die Kombination dieser Angriffe ist es möglich, den Datenverkehr in einem SOME/IP Netzwerk so zu verändern, dass ein oder mehrere Clients die eigenen Daten erhalten. Dazu wird ein eigener Service mit der selben Service-ID gestartet und die Daten aus dem originalen Service bezogen. Um die gesendeten Informationen gezielt manipulieren zu können, werden die konkreten Events vom eigentlichen Steuergerät abonniert und an den Client weitergeleitet. Um zu verifizieren, ob die Einzelangriffe erfolgreich verlaufen sind, soll jedes einzeln genutzte Angriffsmodul ein Ergebnis anzeigen.

5.1.3 Anforderungen an die Netzwerkbibliothek

In der Tabelle 5.1 sind die Anforderungen eines Frameworks an die Netzwerkbibliothek aufgelistet. Im Folgenden werden diese Anforderungen nochmals erläutert. Wie in Abschnitt 2.9 bereits diskutiert, besitzt Python eine große Auswahl nutzbarer Bibliotheken, die nun im folgenden auf die Anforderungen an die Bibliothek untersucht werden.

- **Empfangen und Interpretieren von Paketen**
  Alle genannten Werkzeuge sind in der Lage Pakete über eine Netzwerkschnittstelle zu empfangen und Netzwerkdumps einzulesen.

Anforderungen	Wireshark	pyshark	scapy	pypacker
Empfangen und Interpretieren von Paketen	✓	✓	✓	✓
SOME/IP	✓	✓	✓	✓
SOME/IP-SD	✓	(✓)*	✓	X
Erweiterbar und Anpassbar	✓	✓	✓	✓
Zukunftssicher	✓	✓	✓	X
Manipulation und Erstellen von Paketen	X	X	✓	✓
Versenden von Paketen	X	X	✓	✓
In Python integriert	X	✓	✓	✓
Qualitative Dokumentation	✓	✓	✓	X
Echtzeitfähig	✓	✓	X	✓

* fehlerhafte Implementierung

Tabelle 5.1: Anforderungen an Netzwerkwerkbibliothek

- **SOME/IP**
  Alle genannten Werkzeuge unterstützen SOME/IP.

- **SOME/IP Service Discovery (SOME/IP-SD)**
  In der Implementierung von pypacker fehlt zum Zeitpunkt der Recherche die Erweiterung Service Discovery für SOME/IP. Tshark, das von Pyshark verwendet wird, unterstützt die Service Discovery. Allerdings werden die Informationen fehlerhaft dargestellt. Daher kann pyshark diese Informationen nicht korrekt interpretieren. Die anderen Bibliotheken erkennen SOME/IP-SD Pakete.

- **Erweiterbar und Anpassbar**
  Die genannten Bibliotheken sind opensource verfügbar und in den Programmiersprachen Python und im Falle von tshark in C++ geschrieben. Daher lassen sich diese an die Anforderungen des Frameworks anpassen.

- **Zukunftssicher**
  Die Zukunftssicherheit der Bibliotheken wurde an der Anzahl der Entwickler und der Größe der Community ermittelt, die die jeweilige Implementierung unterstützt und nutzt. Für pypacker gibt es außerhalb von gitlab keinen Support über Foren oder Mailinglisten. Dies ist bei den anderen Bibliotheken gegeben.

- **Manipulation und Erstellen von Paketen**
  Für viele Angriffe wird die Funktion, Pakete zu manipulieren, zu erstellen und zu versenden benötigt. Tshark unterstützt diese Funktionalität nicht. Folglich hat pyshark diese Funktion nicht. Die anderen Bibliotheken hingegen schon.

- **Versenden von Paketen**
  Pyshark ist ein reines Werkzeug zum Empfangen und Analysieren von Netzwerkverkehr und ist nicht in der Lage, Netzwerkpakete zu versenden.

- **In Python integriert**
  Die genannten Werkzeuge lassen sich alle problemlos mittels pip installieren und ohne weiteres Zutun in das zu implementierende Framework integrieren.

- **Qualitative Dokumentation**
  Pyshark und scapy besitzen jeweils eine ausführliche Projektseite und sind gut dokumentiert. Für scapy umfasst das Handbuch zum Zeitpunkt der Arbeit 181 Seiten. Im Falle von Pypacker umfasst die Dokumentation nur wenige hundert Zeilen(siehe [28]).

- **Echtzeitfähig**
  Um die Echtzeitfähigkeit der Bibliotheken zu überprüfen, wurde die Aufzeichnung des Netzwerkverkehrs eines Steuergerätes mit Restbussimulation genutzt. In diesem Netzwerkdump wurden in 30 Sekunden circa 27000 Pakete versendet. Es wurde für jede Bibliothek ein Script geschrieben, das auf einer Netzwerkschnittstelle Daten empfangen und diese analysieren kann. Das erneute Versenden des Netzwerkverkehrs und das gleichzeitige Aufzeichnen mit den einzelnen Bibliotheken zeigte, dass scapy nur 80% der Pakete aufzeichnete. Die anderen Bibliotheken zeichneten den Datenverkehr verlustlos auf. Um zu verifizieren, dass das Sniffingmodul von scapy den Datenverlust verursachte, wurden die Daten mit einer anderen Bibliothek komplett aufgezeichnet und die Rohpakete an die Analysefunktion von scapy weitergeleitet. Die Daten wurden fehlerlos empfangen und analysiert. Allerdings benötigte die Analyse durch scapy mit ungefähr 60 Sekunden etwa doppelt so lange wie das Versenden und Empfangen der Daten. Daher wurde die Analyse der Pakete auf mehrere Prozesse aufgeteilt, wodurch ein echtzeitfähiges Sniffing- und Analysescript erstellt wurde, das auf der Netzwerkbibliothek scapy basiert.

Nach der Analyse der Anforderungen an die zu benutzende Netzwerkbibliothek in Tabelle 5.1 wurde scapy gewählt. Scapy erfüllt, bis auf die Echtzeitfähigkeit, alle Anforderungen. Um die Echtzeitfähigkeit zu erhalten, wurde überprüft, wie pypacker das Mitschneiden von Paketen ausführt. Diese Lösung wurde dann auf scapy adaptiert, um die Echtzeitfähigkeit zu ermöglichen. Dazu wurde die Bibliothek socket genutzt, die das Empfangen und Versenden von Daten ermöglicht. Die so aufgezeichneten Daten werden dann in eine Warteschlange weitergeleitet und von den Analyseprozessen weiterverarbeitet.

5.2 Implementierung

Wie in Kapitel 5.1.3 diskutiert, wurde bei der Implementierung auf die Sprache python3 gesetzt, da diese eine große Auswahl an zusätzlichen Bibliotheken bietet. Dazu wurde scapy genutzt, um die Netzwerkpakete zu analysieren und anzuzeigen. Um die Daten im Programm zu speichern und abrufen zu können, wurde eine sqlite-Datenbank in den Arbeitsspeicher geladen. Um das Framework modular gestalten zu können wurde die Python Bibliothek yapsy genutzt. Diese ermöglicht es, Module aus einer Ordnerstruktur während der Laufzeit nachzuladen. Im Folgenden werden die einzelnen Bestandteile der Implementierung und die Umsetzung dieser genauer erläutert.

5.2.1 Aufbau der Implementierung

Für die Durchführung der Aufgaben wurde das Framework in mehrere Bestandteile zerlegt (siehe Abbildung 5.1), die in eigenen Prozessen und Threads nebenläufig arbeiten. So steuert der Hauptprozess der Implementierung die Benutzeroberfläche, damit diese auch reagiert, wenn im Hintergrund andere Aufgaben erfüllt werden. Von diesem werden die anderen Prozesse gestartet. Ein Prozess wird genutzt um die Daten im Hintergrund in die Datenbank zu speichern. Außerdem werden beim Einlesen von Paketen aus einem Netzwerkdump oder über ein Netzwerkinterface Prozesse gestartet, die dieses Einlesen übernehmen. Zusätzlich werden Prozesse gestartet, die die Analyse dieser eingelesenen Pakete übernehmen. Dabei wird automatisch erkannt, welche Protokolle innerhalb der Pakete genutzt werden und die entsprechenden Informationen aus diesen Protokollen ausgelesen. Die erkannten Daten werden dann über multiprozessfähige Warteschlangen an die Datenbank übertragen. Werden über die Benutzeroberfläche Informationen abgefragt, erfolgt dies ebenfalls über Warteschlangen. In Richtung des Datenbankprozesses werden die Abfragen versendet. Nach der Verarbeitung dieser sendet der Prozess das Ergebnis an den anfragenden Prozess zurück. In den nächsten Abschnitten wird die Funktionsweise dieser einzelnen Elemente detaillierter erläutert.

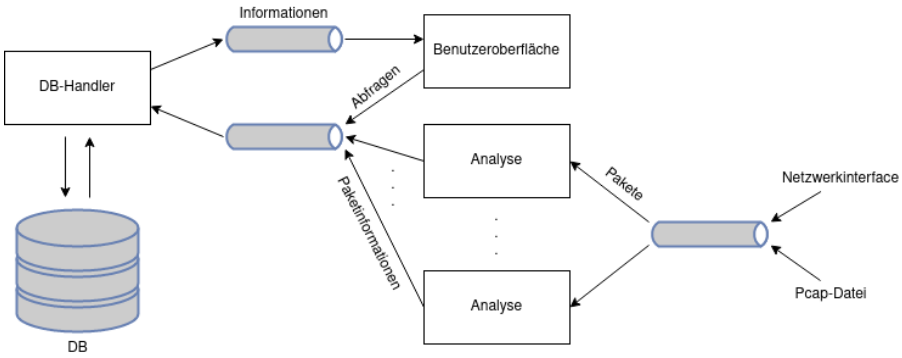

Abbildung 5.1: Programmablaufplan des Frameworks

5.2.2 Analyse der Netzwerkpakete mit scapy

Scapy ist eine Netzwerkbibliothek für Python, die unter anderem das Aufzeichnen, Erstellen, Modifizieren und Versenden von Netzwerkpaketen unterstützt. Von scapy wird eine sehr große Anzahl von Protokollen direkt unterstützt. Es lassen sich aber auch neue Protokollversionen zusätzlich implementieren und in scapy integrieren. Dazu muss die Python Klasse, die das entsprechende Paket-Objekt in Python implementiert von der Klasse *Packet* von scapy erben. Dann kann das erstellte Paket direkt in Scapy genutzt und mit in die Stapelstruktur der Netzwerkpakete aufgenommen werden. Mit der Funktion *bind_layer* von scapy kann bei der automatischen Erkennung von Netzwerkprotokollen mitgeteilt werden, um welches Protokoll es sich handelt. Bei verschiedenen Protokollen, wie IP, wird die nachfolgende Protokollschicht durch einen Bereich im IP-Header angegeben. Bei UDP oder TCP wird dies durch festgelegte Zielports angegeben.

Zusätzliche Protokolle, die scapy nicht selbst erkennt, werden im *contrib* Ordner von scapy mit ausgeliefert und können während der Laufzeit direkt in scapy nachgeladen werden, was ab der Version 2.4.2 von scapy auch bei SOME/IP und bei der Service Discovery von SOME/IP der Fall ist. Diese können mit dem Befehl *load_contrib("automotive.someip")* und *load_contrib("automotive.someip_sd")* nachgeladen werden. Dann werden diese Protokolle direkt von scapy erkannt.

Da die Geschwindigkeit von scapy bei dem Erkennen und Analysieren von Paketen im Vergleich zu anderen Implementationen, wie Wireshark oder pypacker sehr langsam ist, wurden für diesen Zweck mehrere Prozesse gleichzeitig gestartet, die das Analysieren übernehmen. Dazu existiert ein Prozess, der die Pakete, die an einer Netzwerkschnittstelle aufgezeichnet werden oder die Daten aus einem Netzwerk-Dump ausliest und unbearbeitet in eine Warteschlange speichert. Damit während der Verarbeitung der Daten die Reihenfolge der Pakete nicht verletzt wird, protokolliert dieser Prozess die Reihenfolge direkt bei dem Einlesen des Paketes. Die Analyseprozesse, die parallel laufen, erhalten die Pakete aus der Warteschlange und verarbeiten diese weiter. Das Resultat der Analyse wird dann in eine zentrale Datenbank gespeichert, die von einem zusätzlichen Prozess verwaltet wird.

Beim Aufzeichnen der Pakete mit *scapy* wurde festgestellt, dass diese Bibliothek bei einer erhöhten Anzahl von Netzwerkverkehr Pakete verliert, die nicht im System erkannt werden. Bei der Analyse dieses Problems, wurde mit dem Werkzeug *bittwist* ein zuvor aufgezeichneten Netzwerk-Dump erneut versendet und mit den verschiedenen Werkzeugen erneut mitgelesen. Dabei verlor *scapy* circa 20 Prozent der Pakete, während *Wireshark* und *pypacker* die korrekte Anzahl der Pakete aufzeichneten. Um dieses Problem zu lösen, wurde untersucht, wie *pypacker* die Pakete aufzeichnet und dessen Lösung in die entsprechende Funktion des Frameworks integriert.

5.2.3 In-Memory Datenbank mit sqlite

Für die zentrale Datenhaltung des Frameworks wurde eine In-Memory Datenbank mit *sqlite3* [11] erstellt. Da *sqlite3* nicht threadsafe ist, das heißt nicht von mehreren Threads oder Prozessen gleichzeitig darauf zugegriffen werden kann, wird diese Datenbank von einem Prozess verwaltet. Dieser erhält über Warteschlangen die Datenbankzugriffe der anderen Prozesse, führt diese aus und gibt eventuelle Ergebnisse der Anfragen über andere Warteschlangen zurück. Dieses Verfahren ist nicht sonderlich effizient, erlaubt aber die Benutzung einer einfachen Datenbank, ohne dass zusätzliche Software auf dem ausführenden Gerät installiert werden muss.

In der Datenbank selbst existieren verschiedene Tabellen, die die Informationen abspeichern, die bei der Analyse der Pakete aufgezeichnet wurden. Im Folgenden werden die einzelnen Tabellen und deren Funktion erläutert, die im Speicher gehalten werden.

Die Datenbankstruktur erlaubt es nun, die benötigten Informationen über die Netzwerkkommunikation in einem SOME/IP Netzwerk auszulesen. Dabei ist unter anderem interessant, welche Steuergeräte wie viele Services und Methoden beziehungsweise Events anbieten und welche Geräte mit welchen anderen kommunizieren. Dies kann nun durch entsprechende Datenbankabfragen aus der Informationssammlung extrahiert werden. Um diese Informationen darzustellen, werden im weiteren Verlauf Angriffsmodule erstellt, die auf spezielle Angriffsziele angepasst werden können.

5.2.4 Modularität mit Yapsy

Um die Erweiterbarkeit des Frameworks zu ermöglichen, wurde mit *Yapsy* eine *Python* Bibliothek genutzt, die *Python* Dateien dynamisch einlesen und ausführen kann. Dazu werden sowohl eine oder mehrere Dateien benötigt, die die Implementierung und eine Datei, die die Beschreibung des Moduls enthalten. Es wurden zwei Typen von Modulen erstellt, die es ermöglichen neue Funktionen beziehungsweise OEM-Eigene Protokollvariationen von SOME/IP zum Funktionsumfang des Frameworks hinzuzufügen. Diese Modultypen können im laufenden Betrieb des Frameworks durch unterschiedliche Ladeformulare nachgeladen werden. Dabei werden bei den Angriffsmodulen zusätzliche Anpassungen an der graphischen Oberfläche des Frameworks vorgenommen, um die Einstellungen und Rückmeldungen eines Angriffes ermöglichen zu können.

5.2.5 GUI-Implementierung mit PyQt5

Für die Implementierung des Frameworks wurde auf eine graphische Oberfläche gesetzt. Dies ermöglicht eine übersichtliche Darstellung der empfangenen Pakete und erkannten Geräte, Services, Methoden und Events von SOME/IP in Tabellen. Es existieren mehrere Bibliotheken, die es *python3* ermöglichen, eine Oberfläche zu generieren. *QT5* [10] ist allerdings eine weit verbreitete Oberfläche, da sie in sehr ähnlicher Form für Programmiersprachen, wie *C++*, *Go*, *Rust* und vielen anderen umgesetzt wurde. Außerdem existiert für *QT5* eine große Community und eine aktuelle und ausführliche Dokumentation.

5.3 Implementierte Module

Im Verlauf der Arbeit wurden die ersten Angriffsmodule implementiert. Diese wurden so gewählt, dass die Analyse eines SOME/IP Netzwerkes durchgeführt und ein aktiver Angriff getestet werden kann. Daher wurde für die Analyse ein Information Gathering Modul geschrieben, welches eine große Anzahl von Informationen über die SOME/IP fähigen Geräte im Netzwerk und den Verbindungen untereinander erlaubt. Außerdem wurde ein erstes Modul implementiert, das für den Man-in-the-Middle Angriff auf SOME/IP-Ebene benötigt wird, das Subscribe Module. Da die zur Verfügung stehende Netzwerkaufzeichnung SOME/IP Datenverkehr enthielt, die vom Standard abweicht, musste ein *Packet Modul* erstellt werden, das diese Änderungen der Spezifikation erkennt. Welche Funktionen diese Module erfüllen und wie diese implementiert wurden, wird in den nächsten Abschnitten erläutert.

5.3.1 Information Gathering Modul

Das Information Gathering Modul nutzt die Informationen aus der Datenbank, um dem Nutzer einen möglichst übersichtlichen Einblick in ein SOME/IP-Netzwerk zu geben. Dazu werden die Informationen in unterschiedlichen Tabellen angezeigt. Im Folgenden werden diese Tabellen und deren Funktionsweise kurz erläutert.

Packets
In der Packets-Ansicht werden alle Pakete angezeigt, die vom Framework bekannte Protokolle beinhalten. Also alle Pakete, die SOME/IP oder ein über die Module nachgeladenes Protokoll enthält. Diese Ansicht ist in Abbildung 5.2 dargestellt. Hierbei werden in der oberen Tabelle die Pakete mit einigen Informationen angezeigt. Nach einem Klick auf eines dieser Pakete wird dieses in der Darstellung von scapy im unteren Textfeld angezeigt.

Tables
Diese Ansicht gibt einen Überblick über die im Netzwerk gefundenen Geräte, deren Services, Methoden und Events. Es werden in der in Abbildung 5.3 zu sehenden oberen Tabelle die Geräte angezeigt. Dabei entspricht jede gefundene IP-Adresse einem Gerät. Wird eines dieser Geräte angeklickt, werden in der zweiten Tabelle die Services angezeigt, die dieses Gerät besitzt. Nach einem Klick auf einen Service enthält die dritte Tabelle Informationen über die angebotenen Methoden und Events. In diesen Tabellen können jeweils durch einen Rechtsklick mehr Informationen über ein Gerät, einen Service, eine Methode oder ein Event angefordert werden. Dann wird ein neuer Reiter geöffnet, der die Details-Ansicht beinhaltet.

Details
In der Details-Ansicht, die in Abbildung 5.4 zu sehen ist, werden zusätzliche Informationen über das gewählte Gerät angegeben. Dabei wird in der ersten Tabelle eine Übersicht über die Verbindungen angezeigt. Dabei wird jeweils die Anzahl und die Größe der Pakete zwischen zwei Kommunikationspartners summiert. Wird eine Kommunikationsrichtung gewählt, zeigt die zweite Tabelle alle Pakete dieser Kommunikation, während in der Textansicht die Pakete im Detail begutachtet werden können.

5.3.2 Subscribe Modul

Das Subscribe Modul bietet die Möglichkeit für einen aktiven Angriff. Hierbei kann ein beliebiges SOME/IP Subscribe Paket der Service Discovery versendet werden. Es kann der Absender, der Empfänger und alle Eigenschaften des Absender und Empfänger-Services frei gewählt werden. Nach dem Betätigen des „Run"-Knopfes unter den Ausgabefeldern, wie in Abbildung 5.5 zu sehen ist, versendet das Modul das Paket. Dann wird auch die Anzeige im Angriffsmodul aktiviert. Hier werden die Pakete herausgefiltert, die zwischen den beiden angegriffenen Modulen versandt werden. Hierbei kann das Paket automatisch markiert werden, wenn das Abonnieren akzeptiert wurde. Wird im Paket die

TTL auf Null gesetzt, handelt es sich hierbei um das Deabonnieren eines Services, was ebenfalls durch dieses Modul erfolgen kann.

5.3.3 Paket Modul für von der Spezifikation abweichende SOME/IP Pakete

Da die Firma SCHUTZWERK GmbH bereits Steuergeräte auf Sicherheitslücken untersucht, stand während der Ausarbeitung ein Netzwerkdump zur Verfügung, der SOME/IP-Pakete enthält. Da das in diesem Netzwerk genutzt SOME/IP allerdings von den offiziellen Spezifikationen von AUTOSAR abweicht, musste ein *Packet Modul* entwickelt werden, das dem Framework diese Abweichungen bekannt macht. Dazu wurde die öffentlich zugängliche SOME/IP-Implementation von Scapy genutzt und diese an die OEM-Spezifikationen des Fahrzeugherstellers angepasst. Das modulare Nachladen dieser Pakettypen ermöglicht es auch für andere herstellerspezifische Abänderungen von SOME/IP entsprechende Module zu entwerfen.

5.4 Zusammenfassung

In diesem Kapitel wurde der Aufbau des Frameworks erläutert. Dieses besteht aus verschiedenen Bestandteilen: Dem Einlese- und Analyseprozess, der die Pakete mit scapy untersucht, der Datenbank, die die gewonnenen Informationen abspeichert, dem Modulsystem und der Benutzeroberfläche. Werden Daten über eine Netzwerkschnittstelle oder aus einem Netzwerkdump eingelesen, analysiert das Framework diese und speichert die Informationen in der Datenbank ab. Diese Informationen können dann von den Angriffsmodulen genutzt werden. So gibt das Information Gathering Modul eine übersichtliche Einsicht in die Funktionsweise eines SOME/IP-Netzwerkes. Das Subscribe-Tool ist in der Lage festzustellen, ob ein Angriff erfolgreich verlaufen ist.

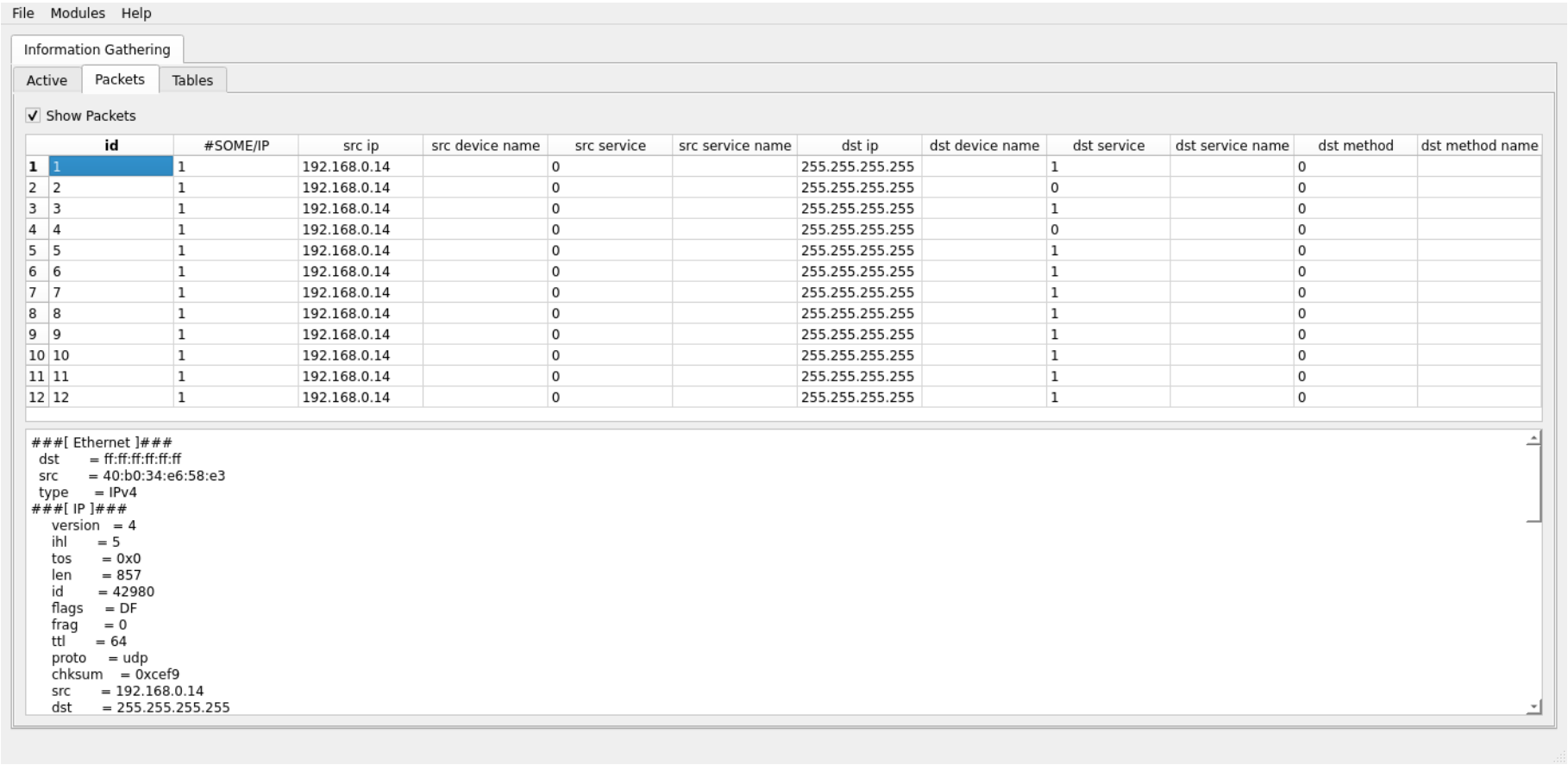

Abbildung 5.2: Packets Ansicht im Information Gathering Modul des Frameworks. Es werden alle Pakete und deren Detailansicht angezeigt.

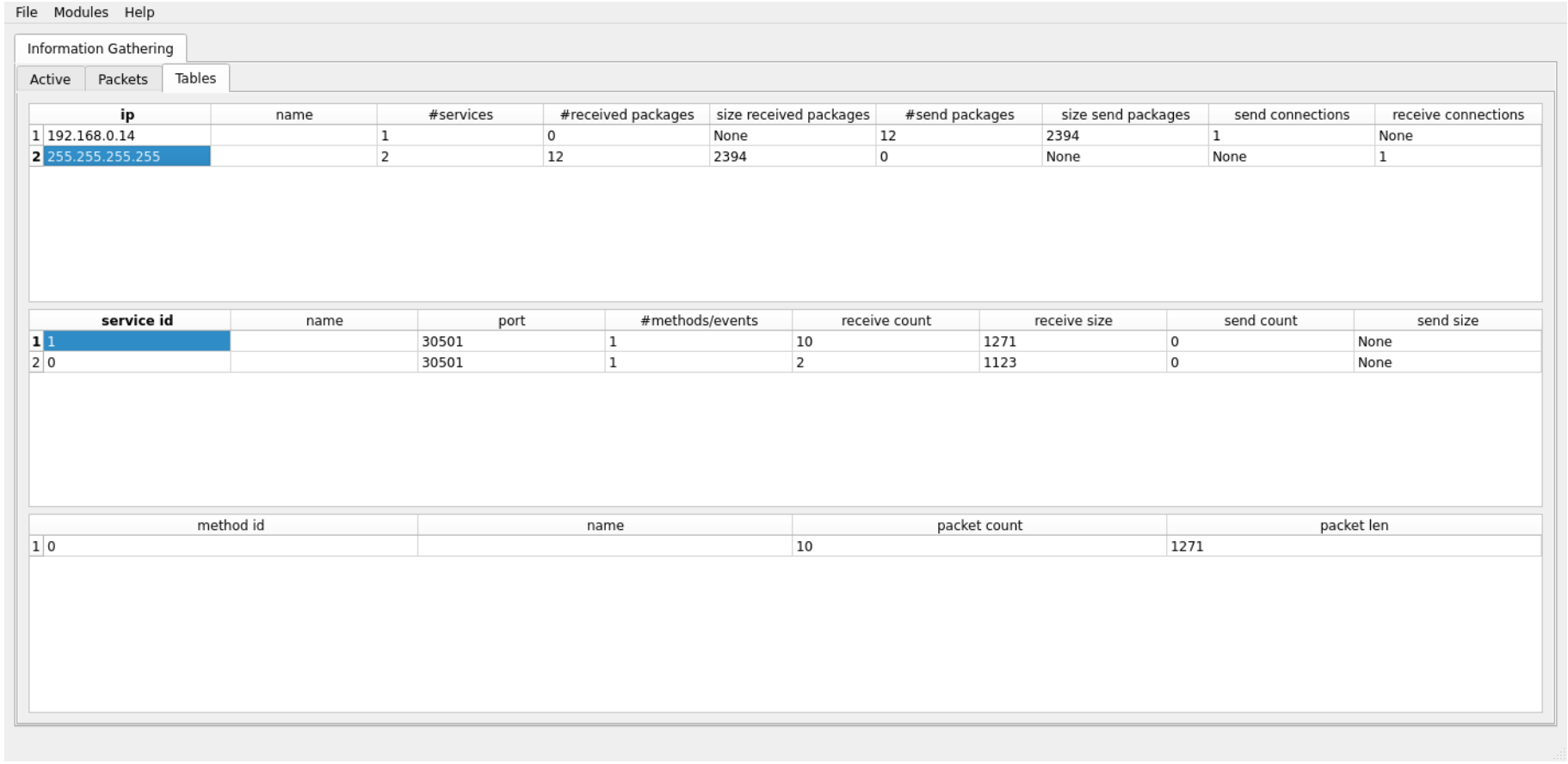

Abbildung 5.3: Table Ansicht im Information Gathering Modul des Frameworks. Es werden die Geräte und deren Services, Methoden und Events angezeigt.

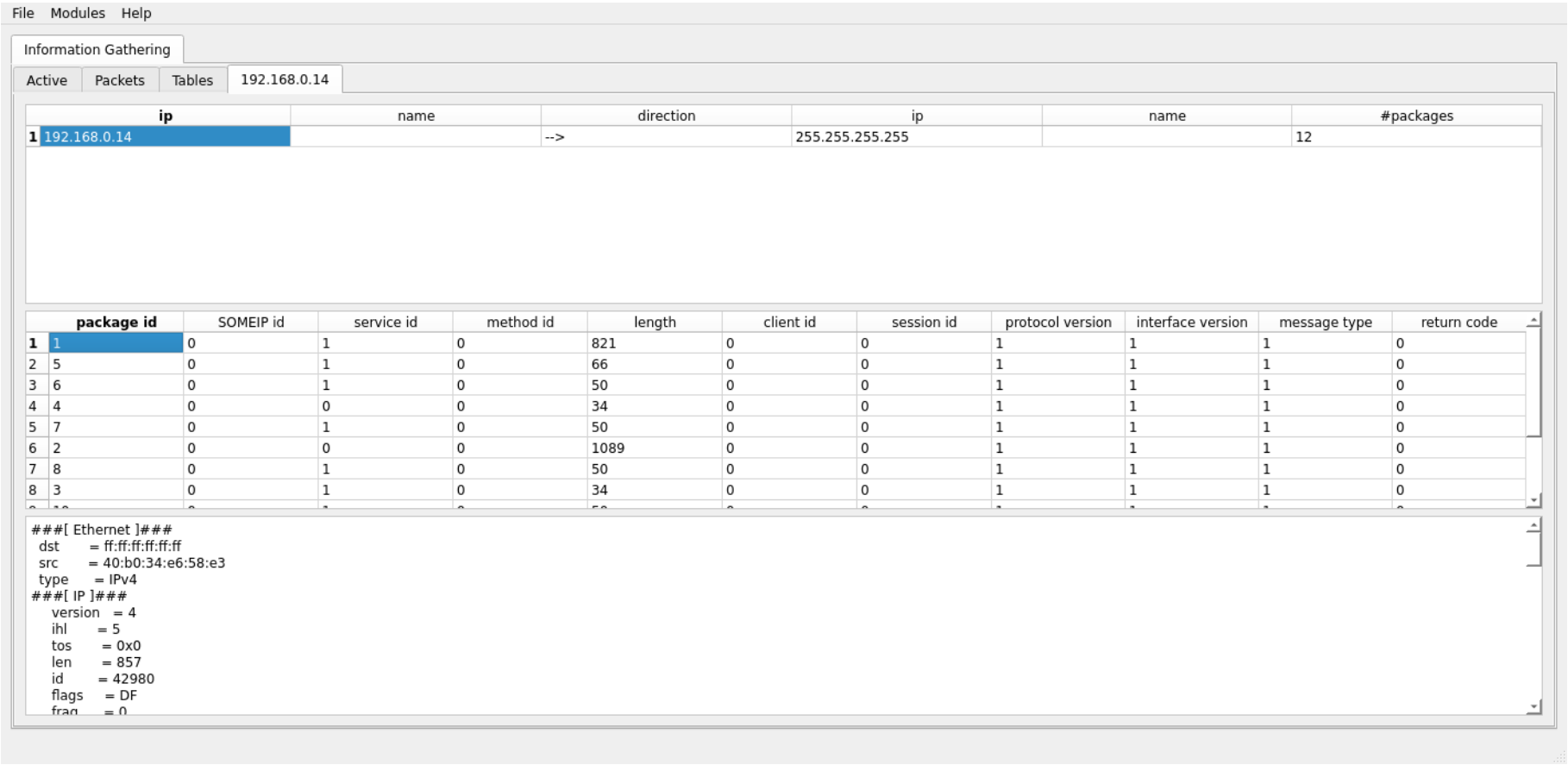

Abbildung 5.4: Detail Ansicht eines Gerätes im Information Gathering Modul des Frameworks. Es werden die Kommunikationen und deren Pakete angezeigt.

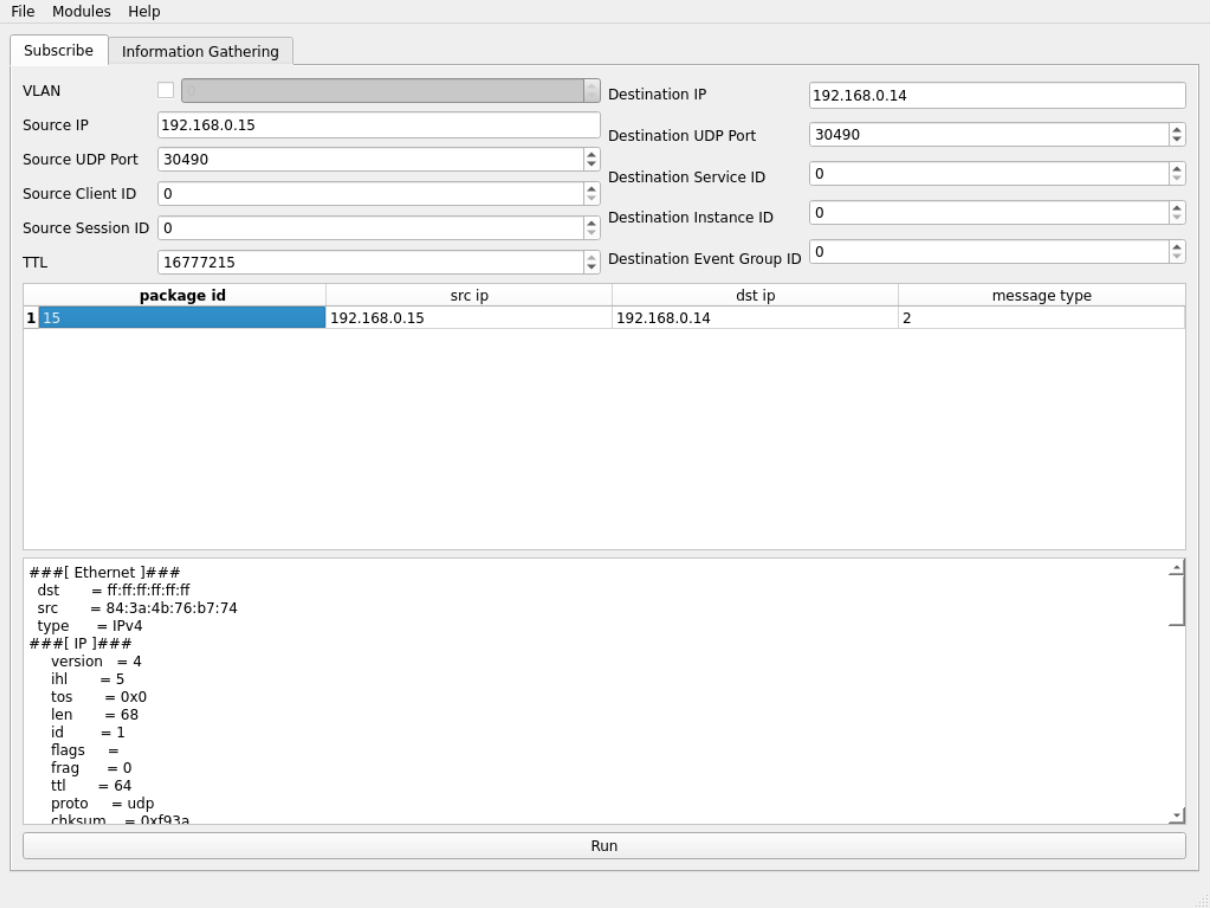

Abbildung 5.5: Das Subscribe Modul des Frameworks. Hierbei wird ein Paket versendet, das ein Event abonniert oder deabonniert.

6 EVALUATION

Um die Funktion der Angriffsmodule zu verifizieren, wurden diese an einem SOME/IP-fähigen Steuergerät getestet. Für das *Subscribe Modul* wurde hierfür ein Angriff durchgeführt und überprüft, wie das angegriffene System darauf reagierte. Da das *Information Gathering Modul* Informationen für den Nutzer darstellt, wurde dies von mehreren Mitarbeitern der *SCHUTZWERK GmbH* in einer Nutzerstudie getestet. Die Aufgaben wurden zum Vergleich auch mit *Wireshark* gelöst. Die Durchführung und die Ergebnisse dieser Evaluation werden in diesem Kapitel besprochen.

6.1 Evaluation des Information Gathering Moduls

Um zu überprüfen, ob das *Information Gathering Modul* einen Mehrwert beim Untersuchen von SOME/IP-Datenverkehr erfüllt, testeten fünf Probanden aus der SCHUTZWERK GmbH das Framework. Dazu haben diese insgesamt 16 Aufgaben, sowohl mit dem implementierten Modul, als auch mit dem verbreiteten Netzwerkanalysewerkzeug *Wireshark*, durchgeführt. Diese Übungsaufgaben wurden mit zwei Netzwerkdumps genutzt. Der erste Netzwerkdump enthielt eine Aufzeichnung des Datenverkehrs einer Restbussimulation mit einem Steuergerät mit über 26.000 Netzwerkpaketen. Diese Pakete enthielten über 134.000 SOME/IP Pakete. Der zweite Netzwerkdump ist eine Beispieldatei, die von Michael Stahn mit dem *pypacker* [29] generiert wurde und nur 13 IP-Pakete mit jeweils einem SOME/IP Paket enthielten. Es wurden beide Netzwerkdumps genutzt, da das realistische Dump kein standardmäßiges SOME/IP nutzt. Das zweite Netzwerkdump war jedoch zu klein, um aufwändigere Analysen durchzuführen.

Die Aufgaben, die von den Probanden gelöst wurden mussten, waren in drei Gruppen unterteilt. So mussten von dem großen Netzwerkdump allgemeine Informationen über die Pakete und die Geräte im Netzwerk ausgelesen werden. In der zweiten Aufgabengruppe wurden Informationen über ein spezifisches Netzwerkgerät abgefragt. Für die dritte Aufgabengruppe wurde der zweite, kleine Dump genutzt und Informationen aus diesen Paketen ausgelesen. Die Aufgaben wurden jeweils abwechselnd mit den beiden Analysewerkzeugen bearbeitet. Nach jeder Teilaufgabe wurden Noten für das jeweilige Werkzeug vergeben. Dabei sollte bewertet werden, wie gut die Information aus dem Dump ausgelesen werden konnten. Zusätzlich wurde die Zeit gemessen, die für das Durchführen der Aufgaben benötigt wurde. Die Aufgaben selbst können im Anhang A eingesehen werden.

Um feststellen zu können, wie sich ein erfahrener Wireshark-Nutzer im Vergleich zu einem unerfahrenen die Umstellung auf ein neues Werkzeug erarbeitet, wurden die Probanden gebeten sich selbst zuvor einzuschätzen. Dazu wurde ermittelt, wie viel Erfahrung die Probanden allgemein mit Angriffen auf IT-Systemen, mit Fahrzeugtechnik, mit SOME/IP und mit der Bedienung von Wireshark haben.

Während der Übungen wurde ein PC genutzt, auf dem die Software mit den benötigten Versionen, also Wireshark in der Version 3.2.1 und scapy in der Version 2.4.3 für das implementierte Framework, bereits vorkonfiguriert wurde. Während der Arbeit wurden die Probanden darum gebeten, laut ihr Vorgehen zu beschreiben, damit das Vorgehen nachvollzogen werden konnte. Im Folgenden werden die Resultate dieser Studie beschrieben und diskutiert.

6.1.1 Benotung der Werkzeuge durch die Probanden

In Abbildung 6.1 werden die Bewertungen der Teilaufgaben dargestellt. Dabei sind die blauen Balken jeweils die Bewertungen für das in der Arbeit implementierte Framework und die roten die Ergebnisse des Vergleichstools Wireshark. Die Noten wurden dabei wie Schulnoten, aber von eins bis fünf, vergeben, wobei die eins das bestmögliche und die fünf das schlechtestmögliche Ergebnis darstellt. Die Balken geben jeweils den kompletten Bereich, also das Maximum und das Minimum an, in dem die Noten vergeben wurden. So wurden für das Framework für die Aufgaben nur Einsen und Zweien vergeben. Die schwarze Markierung innerhalb des Balkens gibt den berechneten Durchschnitt der vergebenen Noten für diese Aufgabe an. Die Durchschnittswerte sind nochmals in der Tabelle 6.1 aufgeführt. Auffällig bei der Auswertung ist, dass die Benotung für Wireshark in den Teilaufgaben 1.5 und 2.1 von der Note eins bis zur Note fünf reichen.

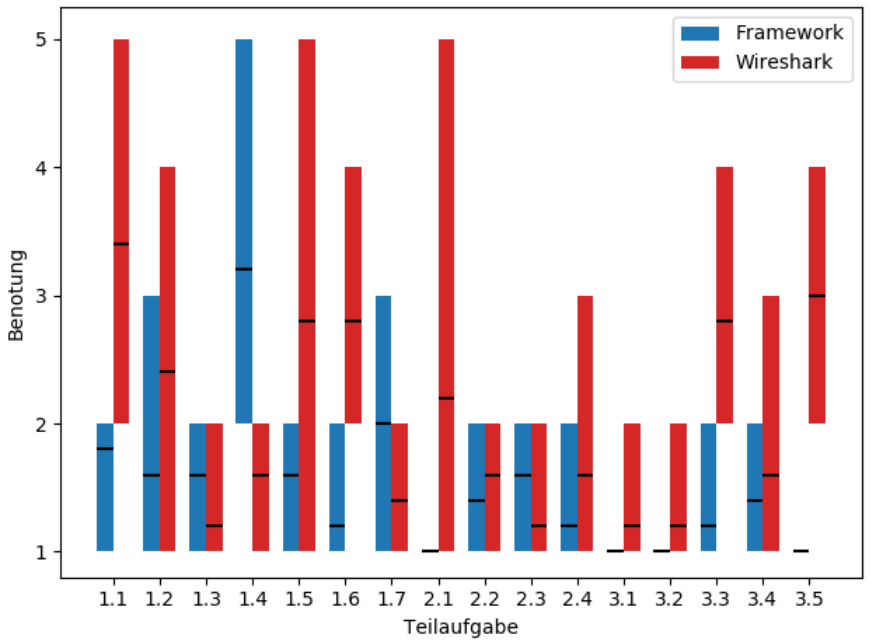

Abbildung 6.1: Durchschnittliche Benotung der Übungsaufgaben durch die Tester

In Abbildung 6.2 ist die durchschnittliche Bewertung aller Teilaufgaben aller teilnehmenden Probanden dargestellt. Das *Information Gathering* Modul wurde im Durchschnitt mit der Note 1,5 und Wireshark mit der Note 2,0 bewertet.

Aufgabe	1.1	1.2	1.3	1.4	1.5	1.6	1.7	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5
Framework	1.8	1.6	1.6	3.2	1.6	1.2	2.0	1.0	1.4	1.6	1.2	1.0	1.0	1.2	1.4	1.0
Wireshark	3.4	2.4	1.2	1.6	2.8	2.8	1.4	2.2	1.6	1.2	1.6	1.2	1.2	2.8	1.6	3.0

Tabelle 6.1: Durchschnittliche Bewertung der Teilaufgaben

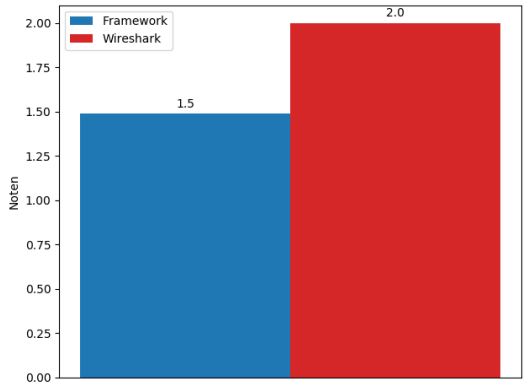

Abbildung 6.2: Durchschnittliche Benotung pro Aufgabe

Diskussion der Ergebnisse

Im Durchschnitt wurde das *Information Gathering* Modul um eine halbe Note besser bewertet, als Wireshark, was bedeutet, dass das Werkzeug schon jetzt eine Alternative zum etablierten Wireshark darstellt, um bestimmte Informationen aus SOME/IP-Datenverkehr auszulesen. Allerdings wurde Wireshark bei manchen Aufgaben besser bewertet. Dies zeigt, dass das Werkzeug an den betreffenden Teilgebieten nochmals überarbeitet werden muss, um diese Aufgaben besser erfüllen zu können.

Auffällig ist, dass Wireshark in den Teilaufgaben 1.5 und 2.1 sowohl mit der Note eins, als auch mit der Note fünf bewertet wurde. Eine Erklärung für diese große Spannweite der Bewertungen ist die, dass die Aufgaben hier zu ungenau erläutert wurden. In der Aufgabe 1.5 wurde nach der Anzahl der Pakete im Netzwerkdump gefragt. Ein Teil der Tester gab als Ergebnis die 26.000 IP-Pakete an, die von beiden Werkzeugen angezeigt werden können. Diese Tester bewerteten Wireshark bei dieser Aufgabe gut. Suchten die Tester allerdings nach der Anzahl der versandten SOME/IP-Pakete, wurde Wireshark schlecht bewertet, da diese Information nicht ohne Weiteres durch Wireshark erhoben werden kann. Das *Information Gathering* Modul zeigt die Information, dass 134.000 SOME/IP Pakete versandt wurden, an.

6.1.2 Abweichung der Benotung in Abhängigkeit der Erfahrung

Um feststellen zu können, wie sich die Erfahrung der Probanden in den Bereichen Automotive, SOME/IP und mit Wireshark auf die Bewertung der genutzten Werkzeuge auswirkt, wurden die Teilnehmer gebeten, sich selbst einzuschätzen, bevor die Aufgaben gelöst wurden. In den Abbildungen 6.3, 6.4 und 6.5 sind die Bewertungen der Werkzeuge getrennt nach der Selbsteinschätzung dargestellt. Bei der Bewertung mussten sich die Teilnehmer selbst Punkte geben. Wobei ein

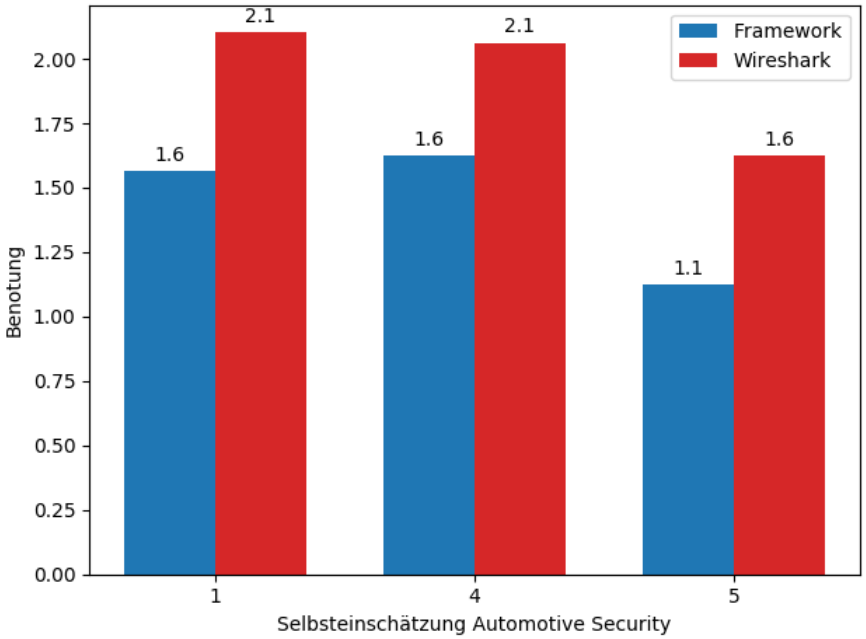

Abbildung 6.3: Benotung in Abhängigkeit der Erfahrung in Automotive Security

Punkt bedeutet, der Proband besitzt keine Erfahrung in diesem Bereich und fünf Punkte bedeuten sehr viel Erfahrung. Hat sich keiner der Teilnehmer mit einem Punkt bewertet, wird dies nicht in der Abbildung angezeigt.

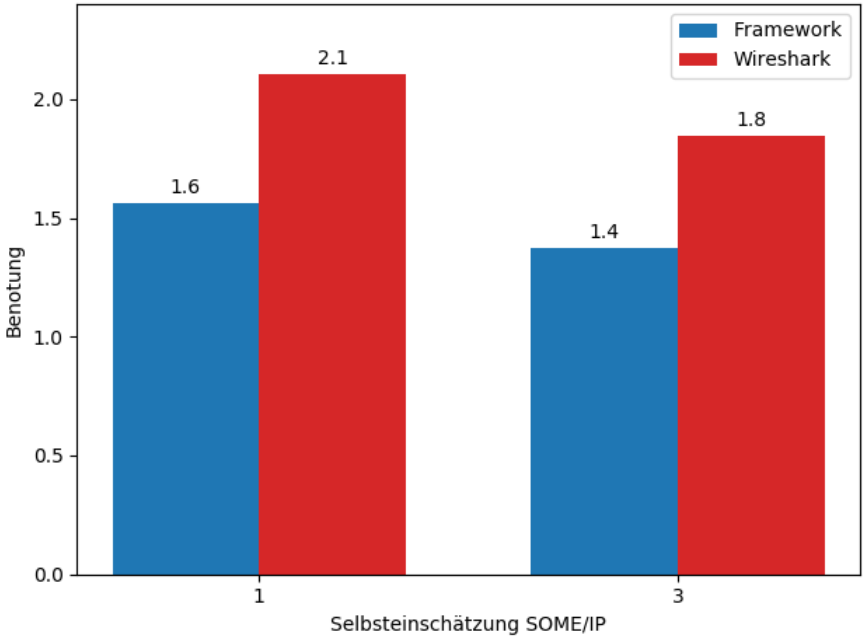

Abbildung 6.4: Benotung in Abhängigkeit der Erfahrung mit SOME/IP

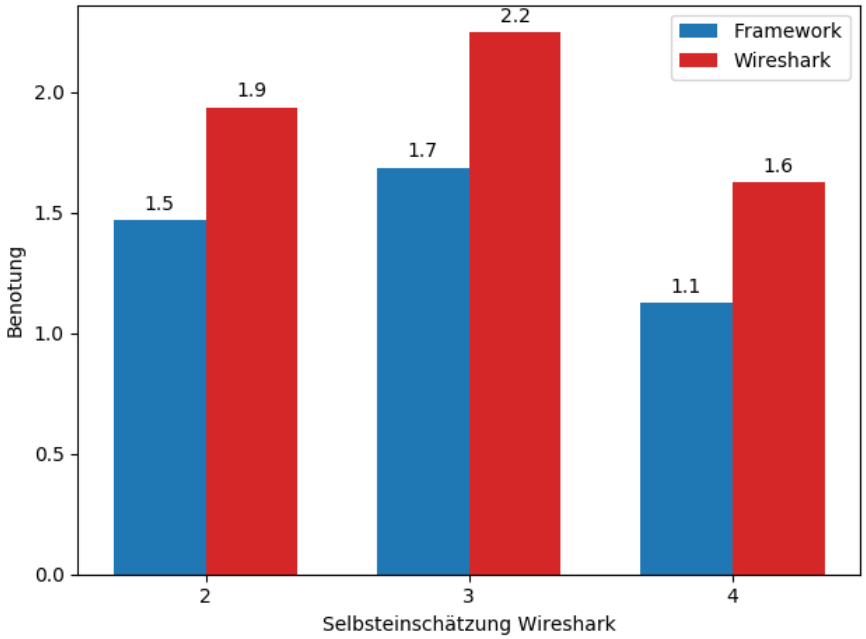

Abbildung 6.5: Benotung in Abhängigkeit der Erfahrung mit Wireshark

Diskussion der Ergebnisse

Da nur fünf Probanden an der Studie teilgenommen haben, kann aus den hier erhobenen Daten kein Rückschluss darauf geschlossen werden, ob sich die Selbsteinschätzung der Erfahrung aus den unterschiedlichen Bereichen auf die Bewertung auswirkt. Vor allem, da für SOME/IP nur zwei Gruppen von Punkten vergeben wurden. Auch ist das Resultat bei der Einschätzung zur Erfahrung in Wireshark nicht aussagekräftig, da die Bewertung beider genutzter Werkzeuge beim Vorwissen von drei Punkten am schlechtesten ausfällt. Allerdings ist hervorzuheben, dass das *Information Gathering* Modul in allen Teilbereichen besser bewertet wurde, als Wireshark.

6.1.3 Zeitaufwand der Bearbeitung der Übungsaufgaben

In den Abbildungen 6.6 und 6.7 ist der Zeitaufwand zum Lösen der einzelnen Teilaufgaben in Minuten dargestellt. In der Darstellung jeder Aufgabe sind wie in Abbildung 6.1 sowohl die Minimal- als auch die Maximalwerte und der Durchschnitt zu sehen. Tabelle 6.2 zeigt diese Werte nochmals als Zahlen. Im Durchschnitt benötigten die Tester mit dem *Information Gathering* Modul eine Minute und sieben Sekunden für die Bearbeitung der Aufgaben. Mit Wireshark benötigten sie eine Minute und 19 Sekunden.

Aufgabe	**1.1**	**1.2**	**1.3**	**1.4**	**1.5**	**1.6**	**1.7**	**2.1**	**2.2**	**2.3**	**2.4**	**3.1**	**3.2**	**3.3**	**3.4**	**3.5**
Framework	1.7	1.2	2.15	2.5	1.9	0.9	1.7	0.7	1.1	1.1	0.6	0.5	0.3	0.5	0.4	0.5
Wireshark	3.7	1.2	0.7	0.9	2.8	3.8	1.2	0.9	1.2	0.8	0.9	0.5	0.4	1.1	0.4	0.5

Tabelle 6.2: Durchschnittlicher Zeitaufwand für die Teilaufgaben

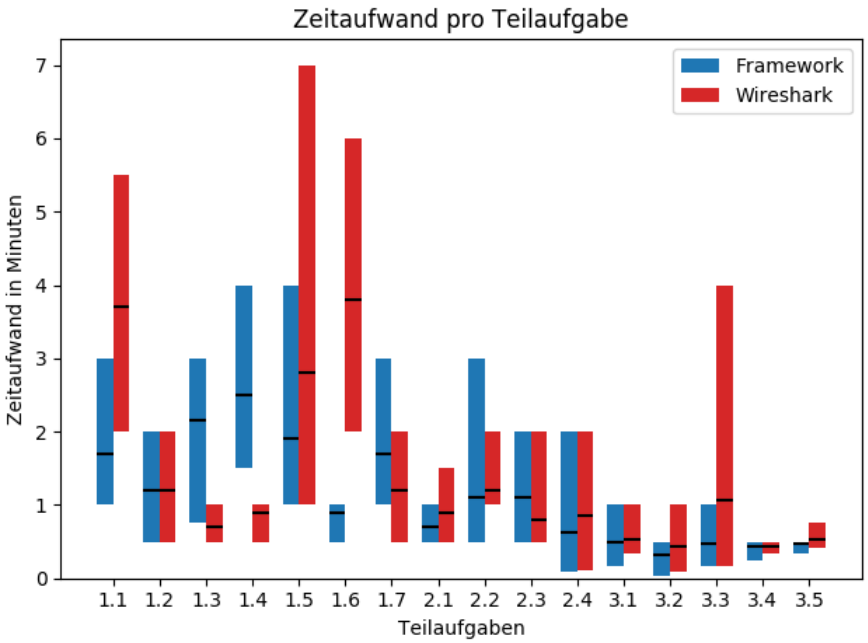

Abbildung 6.6: Durchschnittlicher Zeitaufwand für die Übungsaufgaben durch die Tester

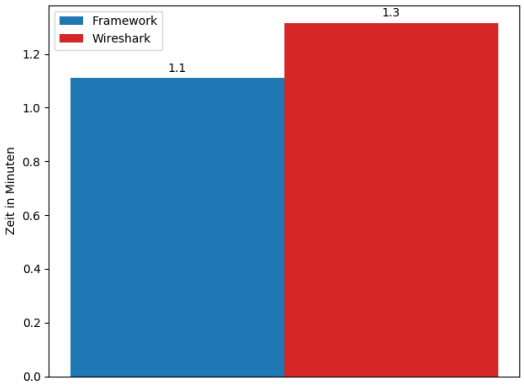

Abbildung 6.7: Durchschnittlicher Zeitaufwand pro Aufgabe

Diskussion der Ergebnisse

Die Auswertung des Zeitaufwands pro Teilaufgabe kann ebenso als Qualitätskriterium zur Auswertung der Werkzeuge genutzt werden, da hier die Effizienz dieser abgelesen werden kann, ohne diese subjektiv einschätzen zu müssen. Kann eine Information durch ein Werkzeug schneller ausgelesen werden, kann dieses Werkzeug als effizienter für diese Aufgabe bewertet werden. Da die Teilaufgaben im Durchschnitt um 12 Sekunden beziehungsweise um circa 15 Prozent schneller mit dem Information Gathering Modul gelöst werden konnten, gibt es für dieses Modul in der frühen Entwicklungsphase bereits eine Tendenz, dass es effizienter für diese Analysen genutzt werden kann. Ein gutes Beispiel für die bessere Effizienz bildet hierbei die Aufgabe 1.6, in der die Lösung nach Durchschnittlich circa einem Viertel der Zeit mit dem von uns implementierten

Framework gefunden werden konnte. Die Aufgabe 1.4 konnte allerdings mit Wireshark deutlich schneller gelöst werden.

6.2 Evaluation aktiver Angriffsmodule

Um die Funktionsweise des *Subscribe Moduls* zu evaluieren, wurde ein Netzwerk, bestehend aus einem SOME/IP-fähigen Steuergerät und einer Restbussimulation mit einer *Vector Box* aufgebaut. Um die Angriffe durchführen zu können, wurde der Datenverkehr komplett über einen PC übertragen, der sowohl in der Lage war, die Netzwerkpakete aufzuzeichnen, als auch eigene Pakete ins Netzwerk zu versenden. Der Versuchsaufbau ist in Abbildung 6.8 schematisch dargestellt. Es wurden Medienkonverter genutzt, um den Netzwerksstandard 100BASE-T1 des ECU am angreifenden PC nutzen zu können. Die VectorBox, die den Restbus simulierte, wurde durch einen PC mit CANoe konfiguriert. Durch zusätzliche USB-Ethernet-Schnittstellen konnten genügend Netzwerkanschlüsse an dem angreifenden PC geschaffen werden. Diese wurden durch eine Bridge miteinander verbunden. Durch diese Bridge-Schnittstelle war das Framework in der Lage, die Pakete, die in beiden Richtungen versendet wurden aufzuzeichnen und selbst Pakete in das Netzwerk zu versenden.

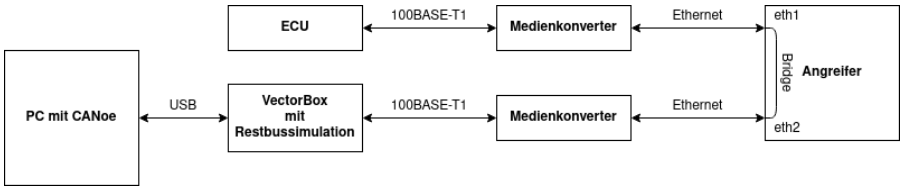

Abbildung 6.8: Schematische Darstellung des Versuchsaufbaus für die Evaluation des Subscribe Moduls

Durch die Analyse des Netzwerks konnten die versendeten SOME/IP-SD Pakete aufgezeichnet werden und ein Service für den Angriff ausgewählt werden. Dieser Service konnte dann durch ein Abonnementpaket mit der *Time To Live* von Null im Namen des simulierten ECUs beendet werden. Dieser reagiert direkt mit einem Find-Paket und abonniert den Service erneut. Dieses Verhalten zeigt, dass der Angriff erfolgreich verlaufen ist. Folglich kann das Tool in realistischen Szenarien genutzt werden.

6.3 Evaluationsergebnisse

Die Evaluation zeigt, dass das *Information Gathering* Modul schon als Alternative zu Wireshark zum Auslesen von SOME/IP-Informationen genutzt werden kann. Allerdings ist Wireshark ein sehr komplexes Werkzeug, das einige hundert Funktionen besitzt und noch nicht alle Informationen von dem selbst implementierten Modul so dargestellt, dass sie einfach und übersichtlich abgelesen werden können. Um dies zu verbessern, müssen die Rückmeldungen der Tester und Anwender genutzt werden. Allerdings beherrscht das stark spezialisierte

Werkzeug, das in dieser Arbeit geschaffen wurde, Funktionen die Wireshark nicht oder nur mit großem Wissen des Anwenders besitzt.

7 FAZIT

Im folgenden Kapitel werden die Erkenntnisse, die durch diese Arbeit errungen wurden, nochmals zusammengefasst. Außerdem wird nochmal auf die Forschungsfragen eingegangen, die im ersten Kapitel gestellt wurden. Als Abschluss der Arbeit werden Ausblicke auf zukünftig noch zu leistende Arbeiten gegeben. So können noch weitere Angriffsmodule erstellt werden, um weitere Angriffsvektoren abzudecken.

7.1 Zusammenfassung

Ziel dieser Arbeit war die Entwicklung eines Frameworks für die Sicherheitsanalyse von SOME/IP-basierten Steuergeräten und der automatisierten Auswertung der Ergebnisse. Hierfür wurden im Kapitel 2 die Grundlagen erörtert, die zum Verständnis der Zusammenhänge benötigt werden. So baut SOME/IP auf den gleichen Netzwerkprotokollen TCP/IP beziehungsweise UDP/IP auf, auf denen auch die Netzwerke in Firmen oder Privathaushalten basieren. SOME/IP besitzt die Möglichkeit über Gateways mit nicht SOME/IP-fähigen Geräten zu kommunizieren. Dies ermöglicht Angriffe auf Fahrzeuge über diese Wege. Daher wurden diese Schnittstellen und die Funktionsweise des fahrzeuginternen Kommunikationsprotokolls CAN erläutert.

In Kapitel 3 wird der Aufbau und die Funktionsweise von SOME/IP diskutiert. Das Protokoll ermöglicht die Kommunikation und den Austausch von Information und Befehlen über Services, die über die Service-ID, die IP-Adresse und den Port des Services eindeutig zugeordnet werden können. Über diese Services können Variablen ausgelesen, Befehle gesendet oder Events abonniert werden. Durch das Abonnieren eines Events kann ein Service regelmäßige Updates über bestimmte Informationsänderungen eines anderen Services beantragen. Um herauszufinden, welches andere Steuergerät einen bestimmten Service anbietet und um ein Event zu abonnieren, kann die Service Discovery genutzt werden. Dies ist ein weiteres Protokoll, das auf SOME/IP aufbaut.

Nach diesen Grundlagen befasst sich Kapitel 4 mit den Angriffsvektoren. Dabei wurden die Verfahren der HEAVENS und der STRIDE Methode genutzt, um die Angriffe in Kategorien einzusortieren und den zu erwartenden Aufwand und den zu erwartenden Schaden einzuschätzen. In der STRIDE Methode wird untersucht, welche Ziele die Angriffe haben. In der HEAVENS Methode wird für verschiedene Bereiche untersucht, welche Voraussetzungen erfüllt sein müssen, dass ein Angriff gelingen kann. So wird überprüft, welches Fachwissen, welche Werkzeuge, welche Informationen und welcher Zugriff auf ein System gegeben sein muss, um einen Angriff darauf durchzuführen. Außerdem wird bestimmt, welchen Einfluss der einzelne Angriff auf ein System hat. So muss eingeschätzt werden, wie groß der zu erwartende Schaden auf die Systeme werden kann und ob Personenschäden zu erwarten sind. Aus diesen Kennzahlen wird dann ein Gefahrenpotenzial ermittelt, das von den Angriffen ausgeht. Diese Verfahren

wurden dann auf verschiedenen Bereichen von SOME/IP angewandt. Die komplette Liste der ermittelten Angriffe und die Ergebnisse der Analyse kann in der Zusammenfassung des 4. Kapitels entnommen werden.

Auf Basis dieser Sicherheitsanalyse wurde das in Kapitel 5 beschriebene Framework entwickelt. Dabei wurde das Design des Frameworks so erstellt, dass durch einzelne Module die in Kapitel 4 diskutierten Angriffe implementiert werden können und die Basis des Frameworks die benötigten Grundfunktionen bereitstellt. Die Implementation besteht aus mehreren Bestandteilen. So wurde die Python Bibliothek scapy genutzt, um die eingelesenen Netzwerkpakete auf SOME/IP zu untersuchen und die erhobenen Informationen werden in eine Datenbank gespeichert. Außerdem bietet das Framework eine Benutzeroberfläche auf Basis von Qt5. Dies ermöglicht eine einfache Bedienung und eine übersichtliche Darstellung der Informationen. Die Angriffsmodule, von denen zwei im Rahmen der Arbeit implementiert wurden, können ebenfalls über die Benutzeroberfläche nachgeladen werden. Für die Evaluation, ob ein Angriff erfolgreich verlaufen ist, können die benötigten Informationen aus der Datenbank extrahiert werden. Konkret wurden die Module Information Gathering und Subscribe implementiert. Das Information Gathering Modul ermöglicht es einem Angreifer die enthaltenen Informationen aus einer Netzwerkaufzeichnung zu analysieren und die Informationen für weitere Angriffe zu extrahieren. Das Subscribe Modul ermöglicht Angriffe auf die Abonnementfunktionen von SOME/IP. Dabei kann festgelegt werden, welcher Service angegriffen wird. Außerdem kann der Absender des Service Discovery Paketes und die Dauer eines Abonnements frei gewählt werden. Dadurch kann auch ein Abonnement im Namen eines anderen Services gestartet oder beendet werden.

Um zu verifizieren, ob das Framework einen mehrwert gegenüber etablierten Systemen bietet, wurden die Angriffsmodule evaluiert. Dafür wurden die beiden Module getrennt voneinander untersucht. Da der Mehrwert des Information Gathering Moduls nicht ohne weiteres durch einen Angriff getestet werden kann, wurde dieses Modul von mehreren Probanden getestet und mit dem etablierten Netzwerkanalysetool Wireshark verglichen. Die Ergebnisse dieser Studie werden im Kapitel 6 diskutiert. Ergebnis dieser Diskussion ist, dass das Information Gathering Modul genutzt werden kann, um die benötigten Informationen aus dem Datenverkehr auszulesen. Dabei können diese Informationen schneller ausgelesen werden als mit dem nicht auf SOME/IP spezialisierten Wireshark. Für die Evaluation des Subscribe Moduls wurde ein SOME/IP-fähiges Steuergerät angegriffen. Damit für den Test dieses Modul ein aktiver Versuchsaufbau verfügbar war, wurde eine ECU mit Restbussimulation in betrieb genommen. Es wurde im Namen des Steuergerätes ein Service deabonniert, der im Restbus simuliert wurde. Der Angriff war erfolgreich und das Steuergerät abonnierte das deabonnierte Event direkt wieder. Dieser Sachverhalt konnte direkt im Angriffsmodul abgelesen werden.

Beantwortung der Forschungsfragen

Die in Kapitel 1 gestellten Forschungsfragen wurden im Verlauf der Arbeit bereits beantwortet. Diese Antworten sollen an dieser Stelle nochmals zusammengefasst werden.

Was sind die Anforderungen an ein Sicherheitstest-Framework von SOME/IP?
Die Anforderungen des Frameworks wurden in Kapitel 5.1 diskutiert. Diese sind, dass das Framework modular erweiterbar sein muss, um noch nicht implementierte und zukünftig entdeckte Angriffe einfach zum Funktionsumfang des Angriffs- und Testingframeworks hinzufügen zu können. Außerdem müssen der Erfolg beziehungsweise die Auswirkungen eines Angriffes, soweit möglich, automatisiert erkannt werden. In Kapitel 4.1.3 wird erläutert, über welche Informationskanäle diese Information erhoben werden kann. Außerdem wurde bei der Analyse der möglichen Angriffe untersucht, bei welchem dieser Feedbackkanäle diese Information direkt aus dem SOME/IP-Datenverkehr entnommen werden kann.

Ist es möglich, bestehende SOME/IP Schwachstellen automatisch zu erkennen?
Dieser Frage wurde im Kapitel 4 nachgegangen. Dabei ist es wichtig, welche Informationen dem Angreifenden über das angegriffene System zur Verfügung stehen. Es können nicht alle notwendigen Informationen über SOME/IP erhalten werden und es muss sichergestellt werden, dass die Informationen, die benötigt werden, zum Angreifer gelangen. Dabei ist es hinderlich, dass in modernen Netzwerken, wie es in Fahrzeugen die SOME/IP nutzen, eingesetzt wird, Informationen direkt zwischen den Kommunikationspartnern ausgetauscht werden. Damit ein Angreifer an diese Informationen gelangen kann, müssen weitere Angriffe erfolgen oder ein entsprechendes Testszenario aufgebaut werden. Liegen die benötigten Informationen hierfür vor, kann die Auswertung eines Angriffes automatisiert realisiert werden.

Wie kann das Framework über Module erweitert werden, um die Kompatibilität zu verschiedenen OEM-Protokollschichten auf Basis von SOME/IP zu gewährleisten?
Im Verlauf der Arbeit wurde ein Framework implementiert, bei dem die einzelnen Angriffe als Module nachgeladen und unabhängig voneinander entwickelt werden können. Außerdem können spezielle Abwandlungen am SOME/IP-Protokoll, die Fahrzeughersteller unter Umständen vornehmen können, durch Paketmodule hinzugefügt werden. In Kapitel 5 werden die Anforderungen diskutiert, die für ein solch modulares Angriffswerkzeug erfüllt sein müssen.

7.2 Ausblick

Während der Planung und Ausarbeitungen wurden einige Ideen entwickelt, die den Funktionsumfang des entwickelten Frameworks erweitern und weitere

Sicherheitsanalysen auf SOME/IP ermöglichen werden. Aus Zeitgründen und dem Mangel an verfügbarer Hardware konnten allerdings nicht alle dieser Ideen umgesetzt werden. In der folgenden Aufzählung werden diese Ideen aber für weiterführende Projekte erläutert.

Nutzung zusätzlicher Informationen für Angriffe
Informationen, die über Steuergeräte zur Verfügung stehen, können genutzt werden, um zusätzliche Angriffe zu realisieren, Angriffe automatisch zu generieren oder die angezeigten Informationen übersichtlicher gestalten zu können. So werden von den Fahrzeugherstellern ARXML-Dateien genutzt, die Informationen über die Steuergeräte und Services enthalten. Diese Dateien können automatisiert ausgelesen und die Informationen genutzt werden, um die Angriffe zu planen. Zusätzliche Module können diese Funktionalität zum Framework hinzufügen und die Informationen in der Datenbank abspeichern. Außerdem können Informationen aus Diagnoseprotokollen, wie dem Protokoll Diagnostics over IP (DoIP) genutzt werden, um den Erfolg von Angriffen zu evaluieren.

Erstellung zusätzlicher Module, um weitere Angriffsvektoren abzudecken
Um weitere Sicherheitsanalysen auf SOME/IP zu starten, müssen die nötigen Angriffsmodule implementiert werden. So kann eine volle Abdeckung der in Kapitel 4 erfassten Angriffsvektoren erlangt werden. Um den Man-in-the-Middle Angriff auf SOME/IP Ebene durchführen zu können, wird zum Beispiel noch ein Offer-Modul benötigt, das für die Angriffe Offer (siehe 4.2.5.2) und Stop-Offer (siehe 4.2.5.3) genutzt werden kann. Außerdem können noch Fuzzing-Module entwickelt werden, um Sicherheitslücken zu entdecken, die anderweitig nicht gefunden werden. Zusätzlich können Module entwickelt werden, die den Funktionsumfang des Frameworks über SOME/IP hinaus erweitern können. So kann SOME/IP in Verbindung unterschiedlicher Sicherheitsmechanismen, wie den in der Arbeit von Jochen Kreissl [18] vorgestellten TESLA und DTLS, kombiniert werden. Zusätzliche Module könnten dann auch die Integration und Umsetzung dieser Protokolle testen.

Kombination unterschiedlicher Protokolle
SOME/IP basiert auf anderen Netzwerkprotokollen. Angriffe auf diese wurden in dieser Arbeit nicht genauer betrachtet, auch wenn diese unter Umständen durchgeführt werden müssen, um SOME/IP-Angriffe zu ermöglichen. Eine weiterführende Arbeit kann sich mit den Kombinationsmöglichkeiten solcher protokollübergreifender Angriffe beschäftigen und unter Umständen weitere Angriffsvektoren finden. Auch solche Angriffe könnten als zusätzliche Angriffsmodule umgesetzt werden.

Da viele moderne Fahrzeuge, in denen SOME/IP eingesetzt wird, auch noch das Controller Area Network (CAN) einsetzen, könnten kombinierte Angriffe mit einem Modul vorstellbar sein. So könnte ein Modul entwickelt werden, das eine SOME/IP-Bridge zum CAN nutzt, um Angriffe auf dieses Netzwerk zu realisieren.

APPENDIX

Im Appendix befinden sich die für die Evaluation des Information Gathering genutzten Dokumente und die Ergebnisse dieser Studie.

Evaluationsaufgaben

Im Folgenden sind die Aufgaben, die den Probanden gestellt wuden.

Einleitung

Im Folgenden werden verschiedene Aufgaben gestellt, Informationen aus vorliegenden Netzwerk-Dumps zu beziehen. Dabei soll jede Aufgabe sowohl mit Wireshark, als auch mit dem Information Gathering Modul des SOMEI-Pilot gelöst werden. Bei den Paketen im Pcap File mit dem Namen 1.pcapng handelt es sich um ein OEM spezifisches SOME/IP Format. Das hat zur folge, dass bei Wireshark ein entsprechender Dissector benötigt wird, der leider nicht vorliegt. Für den SOMEI-Pilot liegt ein Paket Modul bereit. Das andere Pcap File enthält von einem Tool generierte Pakete, die dem Standard entsprechen.

Bei SOME/IP handelt es sich um ein Netzwerkprotokoll, dass in Fahrzeugen eingesetzt werden kann und fahrzeugspezifische Anforderungen auf dem Netzwerk erfüllt. Die einzelnen Steuermodule in einem Fahrzeug besitzen IP-Adressen und Services, mittels denen Sie miteinander kommunizieren können. Die SOME/IP-Pakete werden per UDP/IP zugestellt.

Ist dieser Test erfolgt, wird ausgewertet, wie aussagekräftig die gewonnene Informationen sind und wie aufwendig es war, diese zu erlangen. Dazu wird das Vorgehen der Testpersonen mitprotokolliert und die benötigte Zeit zum Lösen der Aufgaben ermittelt.

Übersicht - 1.pcapng

1. Wie viele unterschiedliche Geräte haben SOME/IP Pakete im Netzwerk versandt?

2. Welches Gerät hat in Summe die meisten Pakete verarbeitet?

3. Welche MAC-Adresse hat dieses Gerät?

4. Welche UDP-Ports wurden zum Übertragen der SOME/IP Pakete genutzt?

5. Über welchen Weg kann bestimmt werden, wie viel Pakete im Netzwerkdump aufgezeichnet werden?

6. Welches Gerät hat die meisten Kommunikationspartner?

7. Zwischen welchen Geräten werden die meisten Pakete ausgetauscht?

Details über das Gerät mit der IP-Adresse x.x.x.x - 1.pcapng

1. Wie viele SOME/IP Pakete wurden mit diesem Gerät empfangen / versendet?

2. Welcher SOME/IP Nachrichten-Typ hatte die Nachricht, die an das Gerät mit der IP x.x.x.x versendet wurde?

3. Wer versendete das 74. IP Paket im Dump und wie groß ist dieses?

4. Wie viele SOME-IP Pakete wurden mit diesem Paket versandt?

Übersicht - 2.pcap

1. Welches Gerät sendete Daten?

2. Wie viele IP Pakete wurden versendet?

3. Wie viele SOME-IP Pakete waren in jedem IP Paket?

4. Welche Daten wurden versendet?

5. Welche SOME/IP Service-IDs werden auf dem Gerät mit der IP 255.255.255.255 genutzt?

Evaluation Bewertung

Im Folgenden ist der Bewertungsbogen, für die Probanden dargestellt.

Einleitung

Nach dem Vergleichstest zwischen Wireshark und dem SOMEI-Pilot soll ermittelt werden, welche Aufgaben mit der jeweiligen Software einfacher gelöst werden konnten.

Angaben der Person

Im Folgenden sollen die Fragen in einer Skala von 1(wenig) bis 5(viel) bewertet werden, sofern nicht anders angegeben.

1. Wie viele Jahre Erfahrung haben Sie bereits in der IT-Sicherheit (als Pentester in Jahren)?

2. Wie viel Erfahrung haben Sie im Bereich Automotive Security (1-5)?

3. Wie viel Erfahrung haben Sie im Bereich SOME/IP (1-5)?

4. Wie viel Erfahrung haben Sie in der Arbeit mit Wireshark (1-5)?

Aufwandseinschätzung Aufgaben

Wie aufwendig war das Lösen der jeweiligen Aufgabe auf einer Skala von 1(leicht) bis 5(aufwendig)?

1 Übersicht - 1.pcapng

	Wireshark	SOMEI-Pilot
1		
2		
3		
4		
5		
6		
7		

2 Details über das Gerät mit der IP-Adresse x.x.x.x - 1.pcapng

	Wireshark	SOMEI-Pilot
1		
2		
3		
4		

3 Übersicht - 2.pcap

	Wireshark	SOMEI-Pilot
1		
2		
3		
4		
5		

Fazit

1. Welches Tool konnte besser genutzt werden, um die Aufgaben zu lösen?

2. Welche Funktionen fehlen dem SOMEI-Pilot zum Lösen solcher Aufgaben?

3. Weitere Anmerkungen / Verbesserungsvorschläge?

Im folgenden werden die Resultate der Evaluation des Information Gathering Moduls gelistet

id	Pentestererfahrung in Jahren	automotive	someip	wireshark
1	9	4	3	3
2	2	1	1	2
3	3	1	1	3
4	1	1	1	2
5	3	4	3	4

Tabelle A.1: Selbsteinschätzung der Personen

ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5
1	3	4	1	2	2	3	1	1	2	1	3	1	1	3	3	2
2	5	3	1	2	1	2	1	2	2	1	1	1	1	3	2	4
3	4	2	2	2	5	3	2	5	2	1	2	1	1	2	1	4
4	2	2	1	1	1	4	1	2	1	2	1	2	2	4	1	3
5	3	1	1	1	5	2	2	1	1	1	1	1	1	2	1	2

Tabelle A.2: Bewertung Wireshark

ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5
1	330	120	60	60	60	180	60	30	60	30	60	30	30	30	30	30
2	120	30	30	60	150	300	30	30	120	30	6	20	6	10	20	45
3	180	90	30	30	90	120	30	60	60	30	60	30	30	30	30	30
4	300	90	60	60	120	180	120	60	60	120	120	60	60	240	30	30
5	180	30	30	60	420	360	120	90	60	30	10	20	5	10	20	25

Tabelle A.3: Zeitbedarf Wireshark in Sekunden

ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5
1	2	2	2	4	2	1	1	1	1	2	1	1	1	2	2	1
2	2	1	2	3	2	1	3	1	1	1	1	1	1	1	2	1
3	1	3	2	5	2	2	2	1	2	2	1	1	1	1	1	1
4	2	1	1	2	1	1	3	1	2	2	2	1	1	1	1	1
5	2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1

Tabelle A.4: Bewertung Framework

ID	1.1	1.2	1.3	1.4	1.5	1.6	1.7	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5
1	90	60	180	180	90	60	90	30	30	90	30	30	30	60	30	30
2	120	30	120	90	240	60	90	30	30	30	5	10	2	12	25	20
3	180	120	45	240	60	60	60	30	60	30	30	30	30	30	30	30
4	60	90	120	120	60	60	180	60	180	120	120	60	30	30	30	30
5	60	60	180	120	120	30	90	60	30	60	5	20	3	10	15	30

Tabelle A.5: Zeitbedarf Framework in Sekunden

LITERATUR

[1] Adrian Perrig et al. „The TESLA Broadcast Authentication Protocol“. In: *In CryptoBytes* (2002).

[2] Gerald Combs et al. *Wireshark*. 2020. URL: https://www.wireshark.org.

[3] Guillaume Valadon et al. *scapy*. 16. Jan. 2020. URL: https://github.com/secdev/scapy.

[4] AUTOSAR. *ARXML Serialization Rules*. 8. Dez. 2017. URL: https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_TPS_ARXMLSerializationRules.pdf.

[5] AUTOSAR. *SOME/IP Protocol Specification*. 2017. URL: https://www.autosar.org/fileadmin/user_upload/standards/foundation/1-1/AUTOSAR_PRS_SOMEIPProtocol.pdf.

[6] AUTOSAR. *Specification of Secure Onboard Communication*. 8. Dez. 2017.

[7] Bundesministerium für Bildung und Forschung. *SecForCARs. Sicherheit für vernetzte, autonome Fahrzeuge*. 2018. URL: https://www.forschung-it-sicherheit-kommunikationssysteme.de/projekte/sicherheit-fuer-vernetzte-autonome-fahrzeuge.

[8] Ernst Christmann. „Datenkommunikation im Automobil Teil 1: Architektur, Aufgaben und Vorteile serieller Bussysteme. Architektur, Aufgaben und Vorteile serieller Bussysteme“. In: *Technische Fachartikel zur Entwicklung von Embedded Electronics* (2016).

[9] Ernst Christmann. „Datenkommunikation im Automobil Teil 2: Sicherer Datenaustausch mit CAN. Architektur, Aufgaben und Vorteile serieller Bussysteme“. In: *Technische Fachartikel zur Entwicklung von Embedded Electronics* (2016).

[10] Die Qt Community. *Qt5 - One framework. One codebase. Any platform.* 2019. URL: https://www.qt.io/.

[11] Die SQLite Community. *SQLite Version 3 Overview*. 2004. URL: https://www.sqlite.org/version3.html.

[12] Dor Green. *PyShark - Python packet parser using wireshark's tshark*. 2019. URL: http://kiminewt.github.io/pyshark/.

[13] IEEE. *IEEE Draft Standard for Ethernet Amendment: Physical Layer Specifications and Management Parameters for 100 Mb/s Operation over a Single Balanced Twisted Pair Cable (100BASE-T1)*. Jan. 2015, S. 1–85.

[14] IEEE. *IEEE Standard for Ethernet*. Aug. 2018, S. 1–5600. DOI: 10.1109/IEEESTD.2018.8457469.

[15] Ágnes B Juhász. „The Regulatory Framework and Models of Self-Driving Cars“. In: *Zbornik Radova* 52 (2018), S. 1371.

[16] James F. Kurose und Keith W. Ross. *Computernetzwerke: Der Top-Down-Ansatz (Pearson Studium - IT)*. Pearson Studium, 3. März 2014. ISBN: 3868942378. URL: https://www.amazon.com/Computernetzwerke-Top-Down-Ansatz-James-F-Kurose/dp/3868942378?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=3868942378.

[17] Mohd Ehmer Khan, Farmeena Khan u. a. „A comparative study of white box, black box and grey box testing techniques". In: *Int. J. Adv. Comput. Sci. Appl* 3.6 (2012).

[18] Jochen Kreissl. „Absicherung der SOME/IP Kommunikation bei Adaptive AUTOSAR". Master Thesis. Universität Stuttgart, 2017. URL: https://elib.uni-stuttgart.de/handle/11682/9482.

[19] Aljoscha Lautenbach und Mafijul Islam. „HEAVENS - HEAling Vulnerabilities to Enhance Software Security and Safety". In: *VINNOVA* (18. März 2016). URL: https://autosec.se/wp-content/uploads/2018/03/HEAVENS_D2_v2.0.pdf.

[20] Graeme Newman. „Car safety and car security: an historical comparison". In: *Understanding and Preventing Car Theft. Crime Prevention Studies* 17 (2004), S. 217–248.

[21] Title Image: pixabay. *Gray Scale Photo of Gears*. 2016. URL: https://www.pexels.com/photo/gray-scale-photo-of-gears-159298/ (besucht am 07. 12. 2020).

[22] E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3 - RFC 8446*. 2018.

[23] E. Rescorla und N. Modadugu. *Datagram Transport Layer Security Version 1.2 - RFC 6347*. 2012.

[24] Philippe Biondi und die Scapy Community. *Scapy Documentation. Release 43*. 10. Dez. 2019. URL: https://readthedocs.org/projects/scapy/downloads/pdf/latest/.

[25] Philipp Schmied. „Reverse-engineering von proprietären Fahrzeugprotokollen". Bachelor Thesis. Hochschule Aalen, 2017.

[26] Daniel Schüller. „Bussysteme im Automobil". Seminararbeit. Universität Koblenz-Landau, 20. Jan. 2005. URL: https://userpages.uni-koblenz.de/~zoebel/ws2004/Bussysteme.pdf.

[27] Adam Shostack. *Threat Modeling. Designing for Security*. 12. Feb. 2014. ISBN: 978-1118809990.

[28] Michael Stahn. *pypacker*. 2019. URL: https://gitlab.com/mike01/pypacker.

[29] Michael Stahn. *pypacker - SOME/IP tests*. 2019. URL: https://gitlab.com/mike01/pypacker/-/blob/master/tests/packets_someip.pcap.

[30] Kristian Trenkel und Patrick Wunner. „Network topology for chassis–potential of ethernet-based systems". In: *6th International Munich Chassis Symposium 2015*. Springer. 2015, S. 51–77.

[31] Dr. Lars Völker. *Scalable service-Oriented MiddlewarE over IP (SOME/IP)*. 2019. URL: http://some-ip.com/.

[32] Robert Wagner. „Address resolution protocol spoofing and man-in-the-middle attacks“. In: *The SANS Institute* (2001).

[33] Ralf Zimmermann Werner und Schmidgall. *Bussysteme in der Fahrzeugtechnik. Protokolle, Standards und Softwarearchitektur*. 2014.