Experimental Evaluation of Cyber-Physical Attacks on Automotive Ethernet Network

by

Jyothi Karnati

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Engineering
(Computer Engineering)
in the University of Michigan-Dearborn
2025

Master's Committee:

Chair: Professor Dr. Hafiz Malik,
Associate Professor: Dr. Xuan Zhou
LEO Lecturer IV: Dr. Azeem Hafeez


DEARBORN

Jyothi Karnati
jyothikn@umich.edu
ORCID iD: 0009-0004-9469-6746

© Jyothi Karnati 2025

Dedication

This Master's Thesis is dedicated, with the deepest love and respect, to my parents. From the very beginning of my life, they have been my greatest teachers, role models, and unwavering supporters. Their sacrifices, many of which were silent and unseen, paved the way for every opportunity I have been fortunate to embrace. Their encouragement has been the foundation upon which I built my academic and personal growth, and their faith in my potential has continually inspired me to strive for excellence even in moments when I doubted myself.

Throughout this academic journey, I have encountered challenges that seemed insurmountable. Each time, I drew strength from their words of wisdom, their patience, and their steady belief that perseverance would lead me forward. In times of long nights, failed experiments, and endless revisions, I remembered the values they instilled in me: resilience, hard work, and humility. Those values carried me through the most difficult moments of this endeavor.

This achievement is not mine alone, it is a reflection of their constant love and guidance. My parents have been my guiding light, reminding me that true success is not measured merely by milestones but by the integrity, dedication, and compassion we bring to every effort. Their faith in me has been a constant reminder that I am never walking this path alone.

With immense gratitude, love, and respect, I dedicate this thesis to my parents. May this work stand as a small token of appreciation for the countless ways they have shaped my life, supported my dreams, and believed in the person I could become.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Dr. Hafiz Malik, for his unwavering support, insightful guidance, and invaluable mentorship throughout the course of my master's thesis. His expertise and encouragement have been instrumental in shaping my research and academic growth, and I am truly fortunate to have worked under his supervision.
I would also like to extend my sincere appreciation to my committee members, Dr. Xuan Zhou and Dr. Azeem Hafeez, for generously contributing their time, expertise, and constructive feedback. Their thoughtful insights have greatly enriched the quality of this work and provided me with new perspectives to strengthen my research.
My heartfelt thanks go to Intrepid Control Systems for their generous support in providing the hardware devices that made the experimental part of my thesis possible. Their contribution was invaluable in translating theoretical research into practical validation.
I am deeply grateful to Abdul Kareem Shaik for being a constant source of technical support and emotional encouragement throughout this journey. His collaboration, thoughtful discussions, and unwavering support have been essential in overcoming challenges and staying motivated.
Finally, I would like to acknowledge the Department of Electrical and Computer Engineering at the University of Michigan-Dearborn for providing an inspiring academic environment and the necessary resources to complete this research. I am profoundly thankful for the opportunities this institution has given me to learn, grow, and pursue my aspirations.

Preface

My academic journey has been shaped by a deep interest in the intersection of networks, security, and embedded systems. From my undergraduate studies through my master's program at the University of Michigan–Dearborn, I have been fascinated not only by how computing systems operate but also by how they can be compromised, defended, and made resilient. This interest grew steadily as I explored cyber-physical systems, where the consequences of security vulnerabilities extend beyond data loss to real-world safety and reliability.

The motivation for this thesis emerged from my exposure to the growing use of Automotive Ethernet in modern vehicles. While the technology offers high bandwidth and scalability for advanced driver assistance systems and autonomous features, its openness and lack of inherent security mechanisms make it vulnerable to cyber-attacks. Questions such as "How can an attacker exploit communication between Electronic Control Units?" and "What happens when real-time vehicular data is intercepted or manipulated?" became central to my curiosity. These questions guided me toward investigating Man-in-the-Middle attacks on Automotive Ethernet networks and the methodologies to study vulnerabilities.

Working on this thesis gave me the opportunity to build a testbed with industry-grade hardware such as Intrepid Control Systems devices and Raspberry Pi-based ECUs. I was able to design experiments that simulated cyber-attacks, and analyze vulnerabilities. This hands-on work went far beyond theoretical study and allowed me to experience the challenges of bridging academic research with real-world automotive environments.

I am grateful for the mentorship of my advisor, Dr. Hafiz Malik, whose expertise and encouragement pushed me to take bold steps in defining and pursuing this research. I also extend my appreciation to my committee members, Dr. Xuan Zhou and Dr. Azeem Hafeez, whose insights and feedback enriched the quality of this work. Their guidance has been invaluable in shaping this thesis into a meaningful academic contribution.

This work represents more than the completion of a research project. It reflects a transformation in my perspective, from being a student of computer engineering to becoming a contributor to the field of automotive cybersecurity. It is a journey that began with curiosity about networks and systems, and has evolved into a commitment to designing safer, more secure technologies for the future of intelligent vehicles.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

PREFACE iv

LIST OF FIGURES xi

LIST OF TABLES xiii

LIST OF ACRONYMS xiv

ABSTRACT xviii

CHAPTER

1 Introduction 1

1.1 Motivation 1
1.2 Problem Statement 2
1.3 Research Objectives 3
1.4 Proposed Approach 3

2 Evolution of In-Vehicle Networking 5

2.1 Overview of In-Vehicle Networks 5
2.1.1 LIN 5
2.1.2 CAN 6
2.1.3 CAN-FD 6
2.1.4 FlexRay 7
2.1.5 MOST 8
2.2 Why the Shift Toward AE? 10
2.2.1 Unsuitability for Emerging Automotive Architectures 10
2.2.2 Growing Data Demands in Modern Vehicles 11
2.2.3 Rise of Centralized, Domain, and Zonal Architectures 11
2.2.4 Technical Enablers, Ecosystem and Standards 14
2.3 Security-Relevant Design Aspects of AE 15
2.3.1 Predictable and Static Topologies 15
2.3.2 ECU Integrity and Software Immutability 15
2.3.3 Communication Constraints and Protocol Selection 16

2.3.4 Physical Medium and Link Stability . . . 16
2.3.5 Environmental Behavior as a Fingerprint . . . 16
2.4 Foundations and Design Principles of AE . . . 16
2.4.1 Evolution from Bus Systems to Switched Ethernet . . . 17
2.4.2 Information Theory and Full-Duplex Communication . . . 17
2.4.3 Time Synchronization and Global Clock Coordination . . . 17
2.4.4 Layered Modularity and OSI Model Alignment . . . 18
2.5 Architectural Applications and Advantages of AE . . . 19
2.5.1 Network Topologies and Architectural Flexibility . . . 19
2.5.2 Integration with IP-Based Systems and Cloud Computing . . . 20
2.5.3 Scalability, Security, and Future-Readiness . . . 20
2.6 OSI Model and AE Protocol Stack . . . 21
2.6.1 Layer 1: Physical Layer . . . 21
2.6.2 Layer 2: Data Link Layer . . . 22
2.6.3 Layer 3: Network Layer . . . 22
2.6.4 Layer 4: Transport Layer . . . 23
2.6.5 Layer 5–7: Session, Presentation, and Application Layers . . . 23
2.7 AE Frame and Packet Structure . . . 24
2.8 Types of AE . . . 24
2.8.1 10BASE-T1S . . . 25
2.8.2 100BASE-T1 . . . 26
2.8.3 1000BASE-T1 . . . 26
2.8.4 MultiGig AE . . . 27
2.9 Functional and Design Advantages of AE . . . 27
2.9.1 Unified Backbone . . . 27
2.9.2 Zonal Architecture Support . . . 27
2.9.3 TSN Real-Time Communication . . . 28
2.9.4 Software-Defined Enablement . . . 28
2.10 Challenges in Implementing AE . . . 29
2.10.1 TSN Complexity . . . 29
2.10.2 Higher Initial Cost . . . 29
2.10.3 Cybersecurity Requirements . . . 29
2.10.4 Legacy Coexistence . . . 30
2.11 Standard Ethernet vs. AE: Key Differences . . . 30
2.11.1 Environmental and Operational Constraints . . . 30
2.11.2 Topology and Cabling . . . 31
2.11.3 Protocol Adaptation and Application Layers . . . 31
2.11.4 Safety and Functional Certification . . . 31
2.12 Ethernet vs. AE . . . 32
2.13 Security in AE . . . 32
2.13.1 Need for Security in AE . . . 32
2.13.2 Impact on Safety-Critical Systems . . . 32
2.13.3 Current Solutions and Gaps . . . 33
2.13.4 Compliance and Standards . . . 33
2.13.5 Towards Secure-by-Design Architectures . . . 33

3 Cyber Threats to Automotive Ethernet . . . 34
3.1 DoS Attack . . . 34
3.1.1 Overview . . . 34
3.1.2 How It Works in AE . . . 35
3.1.3 Known Attack Methods . . . 35
3.1.4 Related Works . . . 36
3.2 Replay Attack . . . 37
3.2.1 Overview . . . 37
3.2.2 Attack Characteristics . . . 38
3.2.3 How It Works . . . 39
3.2.4 Known Attack Methods . . . 40
3.2.5 Related Works . . . 41
3.2.6 Observed Replay Attack Patterns . . . 42
3.3 MitM Attack . . . 42
3.3.1 Overview . . . 42
3.3.2 How It Works . . . 43
3.3.3 Known Attack Methods . . . 43
3.3.4 Related Works . . . 44
3.3.5 Security Goals Violated . . . 45
3.3.6 Impact on AE Network . . . 45
3.4 Message Injection Attack . . . 46
3.4.1 Overview . . . 46
3.4.2 How It Works in AE . . . 46
3.4.3 Known Attack Methods . . . 46
3.4.4 Related Works . . . 47
3.4.5 Security Goals Violated . . . 48
3.4.6 Impact on AE Network . . . 48
3.5 Spoofing Attack . . . 49
3.5.1 Overview . . . 49
3.5.2 How It Works in AE . . . 49
3.5.3 Known Attack Methods . . . 50
3.5.4 Related Works . . . 50
3.5.5 Security Goals Violated . . . 51
3.5.6 Impact on AE Network . . . 51
3.6 Eavesdropping Attack . . . 52
3.6.1 Overview . . . 52
3.6.2 How It Works in AE . . . 52
3.6.3 Known Attack Methods . . . 53
3.6.4 Related Works . . . 53
3.6.5 Security Goals Violated . . . 54
3.6.6 Impact on AE Network . . . 54
3.7 Sniffing Attack . . . 55
3.7.1 Overview . . . 55
3.7.2 How It Works in AE . . . 55

3.7.3 Known Attack Methods . . . 56
3.7.4 Related Works . . . 56
3.7.5 Security Goals Violated . . . 56
3.7.6 Impact on AE Network . . . 57
3.8 Tampering Attack . . . 57
3.8.1 Overview . . . 57
3.8.2 How It Works in AE . . . 57
3.8.3 Known Attack Methods . . . 58
3.8.4 Related Works . . . 58
3.8.5 Security Goals Violated . . . 58
3.8.6 Impact on AE Network . . . 58

4 Focused Investigation on MitM Attacks in AE . . . 60

4.1 Rationale for Choosing the MitM Attack . . . 60
4.2 Severity of the MitM Attack in AE . . . 61
4.3 Accessibility of the MitM Attack in AE . . . 62
4.4 Related Work I: MitM in AE Protocols . . . 62
4.5 Related Work II: Detection and Simulation of MitM . . . 63
4.6 Methodologies for Implementing MitM in AE . . . 64
4.6.1 Related Work: Active-Tap (Inline) MitM in AE . . . 65
4.6.2 Related Works on ARP-Spoofing MitM in AE . . . 67
4.6.3 Research Gap . . . 68

5 Methodologies for MitM Attacks in AE . . . 70

5.1 Introduction . . . 70
5.2 Active Tapping . . . 70
5.2.1 Principle of Operation . . . 70
5.2.2 Distinction Between Active and Passive Tapping . . . 71
5.2.3 Active Tapping as a MitM Strategy . . . 71
5.3 ARP Spoofing . . . 72
5.3.1 Principle of Operation . . . 72
5.3.2 ARP Spoofing as a MitM Strategy . . . 73
5.3.3 Importance of ARP Spoofing for Intrusion Testing . . . 74
5.4 Selected Methodologies . . . 74

6 Experimental Setup . . . 76

6.1 Introduction . . . 76
6.2 Experimental Testbed Architecture . . . 76
6.2.1 Communication Topology . . . 77
6.2.2 Synchronization and Timing Considerations . . . 79
6.2.3 Advantages of Modular Design . . . 79
6.3 RAD-Moon: Media Converter for AE . . . 79
6.3.1 Overview and Functionality . . . 79
6.3.2 Role in the Testbed . . . 81
6.3.3 Practical Considerations . . . 81

6.3.4 Advantages of RAD-Moon in AE Vulnerability Testing 81
6.4 RAD-Jupiter: AE Switch 82
6.4.1 Key Features 82
6.4.2 Applications in the Test Bench 83
6.5 RAD-Galaxy: Active Network Tap and Gateway 84
6.5.1 Overview and Functionality 84
6.5.2 Key Features of the RAD-Galaxy 85
6.5.3 Role in the Testbed 85
6.6 Setting up RAD-Galaxy in Active Tapping Mode with Vehicle Spy 86
6.6.1 Overview of Vehicle Spy 3 87
6.6.2 How to Use Vehicle Spy 3 with RAD-Galaxy 87
6.6.3 Advantages of Using Vehicle Spy with RAD-Galaxy 88
6.7 AE Node 89
6.7.1 Raspberry Pi 5 as an AE ECU 89
6.7.2 Integrating RAD-Moon as a Media Converter 90
6.7.3 Complete AE Node 90
6.8 Active Tap Testbench for MitM 90
6.8.1 Testbench Setup for MitM Attack: 91
6.8.2 MitM Attack Workflow: 92
6.9 Working of Testbench for Monitoring Attack 92
6.9.1 Testbench Setup for Monitoring Attack: 92
6.9.2 Monitoring Attack Workflow: 93
6.10 Working of Testbench for DoS Attack 93
6.10.1 Testbench Setup for DoS Attack: 93
6.10.2 DoS Attack Workflow: 94
6.11 Vulnerability of AE to Cyber Attacks 94
6.11.1 Key Vulnerabilities Exposed by the Attacks 94
6.12 Monitor Attack through ARP Spoofing 95
6.12.1 Attack Objective 95
6.12.2 Theoretical Background 95
6.12.3 Targeted Vulnerabilities 96
6.12.4 Test Bench Setup 96
6.12.5 Implementation and Attack Flow 97
6.12.6 Observations and Implications 97
6.13 Tamper Attack through ARP Spoofing 97
6.13.1 Attack Objective 97
6.13.2 Theoretical Background 98
6.13.3 Test Bench Setup 98
6.13.4 Implementation and Attack Flow 98
6.13.5 Observations and Implications 99

7 Results 100

7.1 Monitor Attack using MitM (Active Tap) 100
7.1.1 Overview 100
7.1.2 Experimental Configuration 100

7.1.3 Observations and Results . . . 101
7.1.4 Implications . . . 101
7.2 DoS Attack using MitM via Active Tap . . . 102
7.2.1 Overview . . . 102
7.2.2 Experimental Setup . . . 103
7.2.3 Results and Observations . . . 103
7.2.4 Attack Impact . . . 104
7.2.5 Summary . . . 105
7.3 Monitoring Attack using MitM via ARP Spoof . . . 105
7.3.1 Overview . . . 105
7.3.2 Experimental Configuration . . . 105
7.3.3 Observations and Results . . . 107
7.4 Tamper Attack using MitM via ARP Spoof . . . 109
7.4.1 Overview . . . 109
7.4.2 Experimental Configuration . . . 109
7.4.3 Observations and Results . . . 110
7.4.4 Observation . . . 111

8 Conclusion . . . 114

9 Future Work . . . 116

9.1 Standardization of Security Frameworks . . . 116
9.2 Lightweight Cryptographic Solutions . . . 116
9.3 Intrusion Detection and Prevention Systems . . . 116
9.4 Integration of Security with TSN . . . 117
9.5 Validation Using Real-World Testbeds . . . 117
9.6 Security for OTA Updates . . . 117
9.7 Cross-Layer Security Approaches . . . 117

BIBLIOGRAPHY . . . 118

LIST OF FIGURES

FIGURE

2.1 Classic CAN Bus topology connecting multiple ECUs [48]. 6
2.2 Comparison between Classical CAN (b) and CAN FD (a) frame formats [48]. . 7
2.3 FlexRay network architecture showing bus topology and communication cycle structure [48]. 8
2.4 Domain-based automotive architecture where each functional domain (Infotainment, Powertrain, Body, Climate) is connected via a central gateway. Communication protocols include 100BASE-T1 Ethernet, CAN-FD, LIN, and FlexRay, demonstrating heterogeneity and centralized routing [39, 40, 48]. 12
2.5 Zonal vehicle architecture demonstrating the use of localized ECUs per zone (e.g., Zone LR, Zone RF), each connected via 100BASE-T1 Ethernet to a central compute switch [39, 40, 43, 48]. 13
2.6 Homogeneous AE network configuration where a central switch connects ECUs over various physical layer standards including 10BASE-T1S, 100BASE-T1, 1000BASE-T1, and MultiGBASE-T1 [39, 40, 42, 43]. This architecture supports high-speed and scalable communication between zonal and domain controllers[48]. 18
2.7 Common automotive networking topologies including bus, ring, star, and zonal configurations. Adapted from Intrepid Control Systems Training, 2025 [41, 48]. 19
2.8 Message encapsulation and protocol layer interaction in Ethernet networks. The figure shows how data moves through each OSI layer with headers added at each step. Adapted from Intrepid Control Systems Ethernet Training (2023) [45, 48]. 21
2.9 AE protocol stack mapped to OSI layers. It illustrates how application, transport, network, and data link protocols align with Ethernet standards in the automotive context. Adapted from Intrepid Control Systems Ethernet Training (2023) [45, 48]. 23
2.10 Structure of a standard IEEE 802.3 Ethernet frame including all essential fields: preamble, MAC addresses, optional 802.1Q tag, ethertype/length, payload, CRC, and interframe gap [41, 45]. Total frame size ranges from 64 to 1518 octets (up to 1522 with VLAN tagging)[48]. 24

3.1 Cyber Attacks on AE Projected on OSI Layers 38

6.1 Direct communication between ECU 1 and ECU 2 without active tap[48] 77
6.2 A conceptual overview of the system architecture[48]. 78
6.3 RAD-MOON 80
6.4 RAD-JUPITER 83
6.5 RAD-GALAXY 85

6.6 RAD-GALAXY side view 86
6.7 Raspberry Pi 5 as ECU 89
6.8 Hardware test bench 91

7.1 Monitoring The Data Between ECU 1 and ECU 2 Using Vehicle Spy 3 Message Viewer 102
7.2 Flooding The Network From RAD-Galaxy and Observing Results Using Vehicle Spy 3 Message Viewer 104
7.3 ECU 1 Bidirectional Communication With ECU 2 106
7.4 ECU 2 Bidirectional Communication With ECU 1 107
7.5 ECU 3 Acting As Attacker And Monitor the Data Between ECU 1 ECU 2 . . . 108
7.6 ECU 1 Bidirectional Communication With ECU 2 Under Tamper Attack 111
7.7 ECU 2 Bidirectional Communication With ECU 1 Under Tamper Attack 112
7.8 ECU 3 Acting As Attacker And Tampering the Data Between ECU 1 ECU 2 . 112

LIST OF TABLES

TABLE

2.1 Comparison of Legacy In-Vehicle Communication Protocols 9
2.2 Legacy Network Limitations 11
2.3 Comparison of Domain and Zonal Architectures 14
2.4 AE Frame Structure 25
2.5 AE PHY Standards 26
2.6 Comparison Between Standard Ethernet and AE 32

3.1 Observed DoS Attack Patterns 37
3.2 Variants of Replay Attacks and Their Sources 42

4.1 Comparative Literature on AE Protocol Vulnerabilities 64
4.2 Comparison of MitM Detection Mechanisms in AE Networks 65

5.1 Comparison of Active and Passive Tapping Techniques 71
5.2 Comparison of Active Tapping and ARP Spoofing Techniques 73

LIST OF ACRONYMS

ACC Adaptive Cruise Control

ABS Anti-lock Braking System

AE Automotive Ethernet

AEAD Authenticated Encryption with Associated Data

ADAS Advanced Driver Assistance Systems

AEDP Automotive Ethernet Development Platform

ARP Address Resolution Protocol

AVB Audio Video Bridging

AVTP Audio Video Transport Protocol

A/V Audio/Video

BITW Bump-in-the-Wire

BS Braking System

CAN Controller Area Network

CANFD Controller Area Network Flexible Data-Rate

CEP Complex Event Processing

CIA Confidentiality, Integrity, Availability

CIAA Confidentiality, Integrity, Authenticity, Availability

DDoS Distributed Denial of Service

DoIP Diagnostics over IP

DoS Denial of Service

DSRC Dedicated Short Range Communications

DTC Diagnostic Trouble Code

ECUs Electronic Control Unit

FlexRay Automotive Communication Bus

FPGA Field-Programmable Gate Array

GCL Gate Control List

GUI Graphical User Interface

HIL Hardware-in-the-Loop

ID Identifier (Message ID)

IDS Intrusion Detection System

IEEE802.1X Port-Based Network Access Control

IP Internet Protocol

IPsec Internet Protocol Security

IVNs In-Vehicle Network

LiDAR Light Detection and Ranging

LIN Local Interconnect Network

L2 Layer 2 (OSI)

L3/4 Layers 3/4 (OSI)

L7 Layer 7 (OSI)

MAC Media Access Control

MACsec Media Access Control Security

MitM Man-in-the-Middle

ML Machine Learning

NIST National Institute of Standards and Technology

NTP Network Time Protocol

OBD-II On-Board Diagnostics II

OCSVM One-Class Support Vector Machine

OMNeT++ Objective Modular Network Testbed in C++

OSI Open Systems Interconnection

OTA Over-the-Air Update

OT Operational Technology

PCA Principal Component Analysis

PCP Priority Code Point (802.1Q)

PCU Powertrain Control Unit

PTP Precision Time Protocol

QoS Quality of Service

RAD-Galaxy Intrepid Active Tap/Converter for Automotive Ethernet

RAD-Jupiter Intrepid Managed Automotive Ethernet Switch

RAD-Moon Intrepid Automotive Ethernet Media Converter

RST Reset (TCP Flag)

Scapy Python-based Packet Crafting Tool

SOA Service-Oriented Architecture

SOME/IP Scalable service-Oriented Middleware over IP

STP Spanning Tree Protocol

STRIDE Spoofing, Tampering, Repudiation, Information Disclosure, DoS, Elevation of Privilege

TCP Transmission Control Protocol

TCP/IP Transmission Control Protocol / Internet Protocol

TCPRST TCP Reset

TLS Transport Layer Security

TSN Time-Sensitive Networking

TT Time-Triggered Traffic

TTE Time-Triggered Ethernet

UDP User Datagram Protocol

UDS Unified Diagnostic Services

UI User Interface

UTP Unshielded Twisted Pair

V2X Vehicle-to-Everything

VLAN Virtual LAN (802.1Q)

Wireshark Network Protocol Analyzer

ABSTRACT

Automotive Ethernet (AE) is rapidly establishing itself as the backbone of modern in-vehicle communication, offering the high bandwidth, scalability, and real-time data exchange essential for advanced driver assistance systems and autonomous vehicles. While AE addresses several limitations of legacy in-vehicle networks, such as low data rates and insufficient source authentication, it still lacks robust security guarantees leaving safety-critical operations vulnerable to exploitation. This thesis aims to systematically investigate the security vulnerabilities of AE under realistic operating conditions by executing high-impact cyber-physical attacks and proposing corresponding countermeasures, focusing specifically on protection against spoofing attacks.

To achieve these objectives, a comprehensive AE test bench was developed to accurately mirror real-world in-vehicle network architectures, utilizing Raspberry Pi-based ECUs, RAD-Moon media converters, and RAD-Galaxy/RAD-Jupiter tapping devices. The research involved conducting and analyzing four primary attack scenarios: Man-in-the-Middle (MitM), Denial-of-Service (DoS), Monitoring via ARP spoofing, and Tampering via ARP spoofing. Attack methodologies encompassed active and passive interception, traffic flooding, packet capture, and in-transit data modification. Vehicle Spy 3 and advanced packet inspection utilities were employed to assess both network behavior and the impacts of these attacks.

Experimental results demonstrate that all four attacks are feasible with minimal resources and capable of evading immediate detection. Monitoring and MitM attacks enabled transparent interception of network traffic, tampering attacks allowed real-time modification of messages, and DoS attacks significantly diminished network availability. These findings provide concrete evidence of AE's susceptibility to fundamental network-layer threats and underscore the pressing need for integrating robust security mechanisms into future in-vehicle network designs.

CHAPTER 1

Introduction

This thesis experimentally evaluates the feasibility and impact of MitM enabled attacks on AE networks. A modular testbed using Raspberry Pi ECUs and industry-grade RAD devices is developed to demonstrate and measure DoS, monitoring, ARP-spoofing, and in-flight tampering under realistic timing and switching conditions.

1.1 Motivation

Modern vehicles are increasingly complex cyber-physical systems in which sensors, software, and networks collaborate to provide advanced driver assistance, infotainment, and over-the-air updates. To support these features, manufacturers are transitioning from legacy buses such as CAN, LIN, and FlexRay to Automotive Ethernet, which offers higher bandwidth and architectural flexibility suitable for centralized and zonal designs.

This transition, however, introduces new risks. While legacy in-vehicle networks were designed for isolated operation, AE is open and interoperable with conventional Ethernet stacks, thereby increasing the attack surface. Adversaries with network access can observe, redirect, flood, or manipulate traffic. Such attacks are not limited to information theft; they can directly interfere with safety-critical functions such as braking, steering, and perception. With the growing integration of V2X, telematics, and cloud services, these threats become more significant.

Although regulators and manufacturers acknowledge the importance of cybersecurity for vehicle safety and certification, much existing research remains simulation-based or analytical. What is lacking is systematic, reproducible evidence from real hardware under realistic switching and timing conditions. High-impact outcomes are DoS, covert monitoring, ARP-based redirection, and in-flight tampering require the ability to interpose on traffic (MitM). The feasibility of such interposition in practical AE deployments remains insufficiently characterized.

This thesis addresses this gap by developing a testbed that integrates Raspberry Pi boards with RAD automotive networking devices. By enabling MitM as the core capability, this work evaluates the practical feasibility of representative AE attacks, their effect on ECU behavior, and provides insights into effective mitigation strategies.

1.2 Problem Statement

Automotive Ethernet delivers the bandwidth and flexibility required by software-defined vehicles but was not originally designed as a secure medium for safety-critical control. Unlike legacy buses, AE inherits the openness of standard Ethernet. This openness allows attackers who gain network access to observe, redirect, flood, or alter traffic actions that can disrupt critical vehicle functions such as braking, steering, or sensor fusion. Thus, AE advantages in scalability and performance also increase its exposure to cyber threats.

Attacks such as DoS, covert monitoring, ARP redirection, and message tampering typically become practical when an adversary can interpose on communication (MitM). While these attacks are well documented theoretically and through simulations, limited experimental studies exist that evaluate their impact on real AE hardware under realistic conditions. Without hardware-based analysis, important questions remain unanswered: which attacks are easily achievable, how they affect ECU timing and functionality, and what indicators can support early detection.

This knowledge gap is critical because AE attacks can lead to functional safety failures. Without empirical evidence on latency changes, packet loss, and the propagation of tampered messages, system designers and regulators lack the data required to prioritize mitigations such as lightweight authentication, timing-aware intrusion detection, or secure TSN integration.

This thesis therefore focuses on two central research questions:

1. How do MitM techniques enable DoS, monitoring, ARP-spoofing, and tampering in a realistic AE testbed?

2. What are the measurable effects of these attacks on communication integrity, availability, and timing, and how can these findings guide the development of practical defenses?

1.3 Research Objectives

The aim of this research is to provide experimentally grounded evidence on how MitM techniques enable high-impact AE attacks and how such attacks affect ECU communication. Emphasis is placed on hardware-based experiments to produce practical insights that move beyond theoretical models.

The technical objectives are:

1. **Testbed development:** Construct a modular AE test environment using Raspberry Pi ECUs and RAD devices (RAD-Moon, RAD-Jupiter, RAD-Galaxy) to represent modern zonal/centralized architectures.

2. **MitM capability implementation:** Realize MitM using two methods: inline tapping and ARP-based redirection, enabling interception, redirection, and modification of AE traffic.

3. **Attack execution:** Conduct four representative attacks (DoS, monitoring, ARP-spoofing, and in-flight tampering) in ways that approximate realistic adversary capabilities.

4. **Measurement and analysis:** Collect and evaluate metrics including packet loss, latency, jitter, retransmissions, integrity violations, and ECU-level functional impact.

5. **Signature identification and mitigation guidance:** Identify patterns that distinguish malicious interposition from normal traffic and translate findings into recommendations for countermeasures such as lightweight authentication, intrusion detection, and secure TSN.

The study also aims to provide reproducible procedures so that results can be independently validated and extended.

1.4 Proposed Approach

A hands-on experimental method is adopted. The testbed reflects modern automotive architectures and uses Raspberry Pi 5 ECUs with RAD-Moon converters, RAD-Jupiter switches, and RAD-Galaxy taps to emulate realistic network conditions.

Two MitM modalities are implemented:

- **Inline tapping**, which places a device physically between ECUs, allowing observation, modification, or blocking of traffic at wire speed.

- **ARP-based redirection**, which manipulates ARP tables to reroute traffic through the attacker, enabling covert interception and selective modification.

Experiments are conducted as controlled trials. For each attack, network-layer metrics (latency, jitter, loss, CRC errors), protocol-level events (retransmissions, ARP table changes), and application-level impacts (delayed commands, altered sensor values) are recorded. Traffic load, timing, and attack intensity are systematically varied to identify thresholds where system behavior degrades or detection becomes possible.

The focus is not only on proving feasibility but also on identifying measurable signatures such as distinctive latency patterns or asymmetric traffic flows that can support intrusion detection or timing-aware defenses. Findings are used to prioritize mitigation strategies suitable for AE, including lightweight authentication, anomaly detection, and secure TSN design recommendations.

The preceding discussion established the motivation, problem statement, and objectives of this thesis, emphasizing the need for hardware-based evaluation of Man-in-the-Middle enabled attacks on Automotive Ethernet. While Chapter 1 has outlined the research focus and proposed methodology, it is essential to situate this study within the broader body of existing work. A systematic review of prior research on Automotive Ethernet security, in-vehicle communication protocols, and MitM-related attack vectors provides the necessary foundation to frame the experimental approach. Accordingly, Chapter 2 surveys the relevant literature, identifies the strengths and limitations of existing efforts, and highlights the unresolved challenges that this thesis addresses.

CHAPTER 2

Evolution of In-Vehicle Networking

2.1 Overview of In-Vehicle Networks

The evolution of automotive electronics has been marked by the proliferation of ECUs, which are responsible for controlling and monitoring distinct vehicular functions such as powertrain control, body electronics, braking systems, lighting, infotainment, and driver assistance [7, 66]. As the number of ECUs in modern vehicles has grown, so has the need for reliable and efficient in-vehicle communication networks to coordinate their operation [15]. Over the past decades, several dedicated automotive communication protocols have been developed to meet these requirements, each tailored to specific use cases and system constraints [66, 25].

Among these, the **LIN**, **CAN**, **CAN-FD**, **FlexRay**, and **MOST** [20, 46, 11] have been the most widely adopted. They vary in terms of communication speed, network topology, real-time performance, implementation cost, and robustness [15]. This section explores the fundamental characteristics, operating principles, and deployment contexts of these legacy in-vehicle network technologies [20, 11].

2.1.1 LIN

The LIN is a low-cost, serial communication protocol developed to facilitate communication between sensors and actuators in automotive subsystems that do not require high bandwidth or real-time guarantees [7]. Introduced in the late 1990s, LIN complements higher-speed networks like CAN by offloading simpler, non-critical control tasks [46].

LIN uses a single-wire master-slave architecture. The master (often a gateway ECU) schedules and initiates all communication on the bus, while the slave nodes respond accordingly [7]. Communication occurs in a deterministic and cyclic manner based on predefined time slots. Each LIN frame consists of a break field, sync field, identifier field, data field, and checksum, enabling simple error checking [66].

Due to its low implementation cost and straightforward topology, LIN is ideal for controlling distributed electromechanical components in vehicle body domains [46]. Common applications include window lifts, seat adjustment motors, mirror actuators, interior lighting, and climate control [7].

2.1.2 CAN

The CAN is a robust, real-time communication protocol developed by Bosch in the 1980s to support reliable data exchange among ECUs in harsh automotive environments [66]. It uses a *multi-master broadcast-based architecture* where all nodes are connected to a common two-wire differential bus [7].

CAN frames are message-based and carry unique identifiers, allowing nodes to filter and prioritize messages based on relevance [66]. The protocol employs a non-destructive bitwise arbitration mechanism, ensuring that messages with higher priority (lower identifier value) win access to the bus without corrupting data [15].

Each standard CAN data frame includes an 11-bit identifier, control bits, a data payload (up to 8 bytes), and a CRC for error detection [66]. CAN is known for its reliability, fault confinement mechanisms, and ease of integration, and is extensively used in powertrain systems, braking, steering, and body control modules [7, 15].

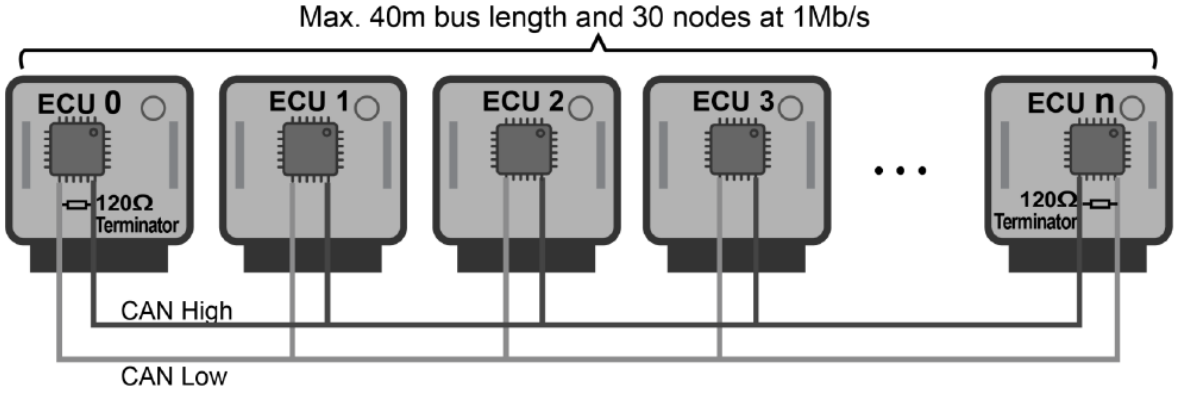

Figure 2.1: Classic CAN Bus topology connecting multiple ECUs [48].

2.1.3 CAN-FD

CAN-FD is an enhancement of the classical CAN protocol designed to overcome some of its physical and functional constraints, particularly with regard to payload size and data transmission rate [15].

Introduced in 2012 by Bosch, CAN-FD extends the maximum data payload from 8 bytes to 64 bytes and enables faster data transmission rates (up to 8 Mbps) during the data phase of communication [13]. CAN-FD maintains compatibility with traditional CAN at the arbitration level while introducing higher efficiency for data-heavy applications [15].

CAN-FD frames consist of an extended identifier field, control bits indicating frame type and payload length, data, and CRC [13]. The separation between the arbitration phase (lower speed) and the data phase (higher speed) allows CAN-FD to increase throughput while remaining stable on existing CAN infrastructure [15, 13].

CAN-FD is particularly suited for applications requiring frequent firmware updates, sensor fusion, and diagnostic communication, while maintaining the simplicity and robustness of classic CAN [66].

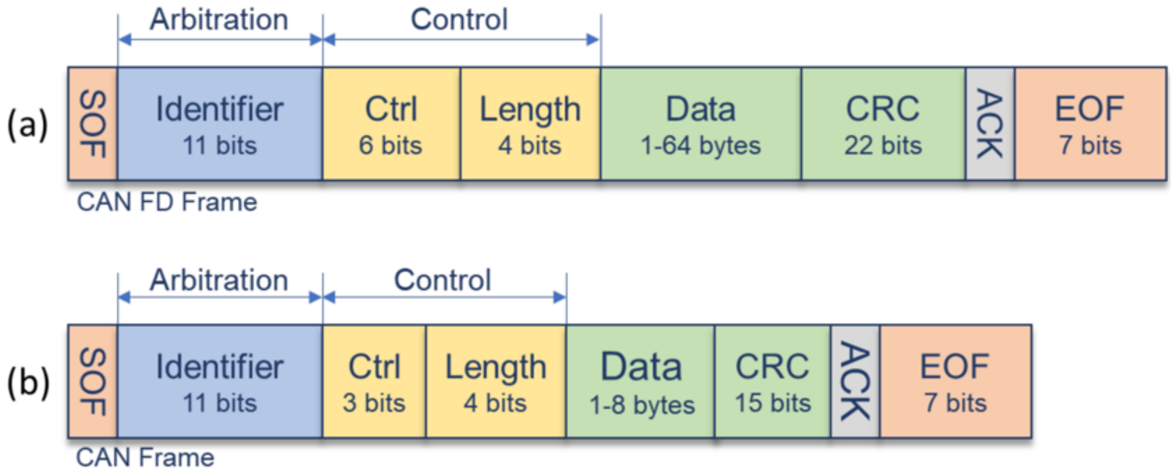

Figure 2.2: Comparison between Classical CAN (b) and CAN FD (a) frame formats [48].

2.1.4 FlexRay

FlexRay is a high-speed, deterministic, and fault-tolerant communication protocol developed by the FlexRay Consortium to meet the stringent requirements of time-critical and safety-sensitive automotive systems [46, 11, 26].

FlexRay operates with a dual-channel topology, allowing for redundant or increased bandwidth communication [20]. It uses TDMA for static communication slots, ensuring deterministic timing, and also supports event-triggered dynamic slots to handle asynchronous messages when necessary [11, 27].

The protocol supports a raw data rate of 10 Mbps per channel, and provides precise synchronization among nodes via a global time base, making it suitable for x-by-wire systems

such as steer-by-wire, brake-by-wire, and active suspension [46]. The global static schedule ensures tight coordination of ECU activities with guaranteed latency and jitter performance [20, 11].

FlexRay protocol stack is layered, consisting of a physical layer, data link layer, and protocol services, allowing for modular integration and system verification [66].

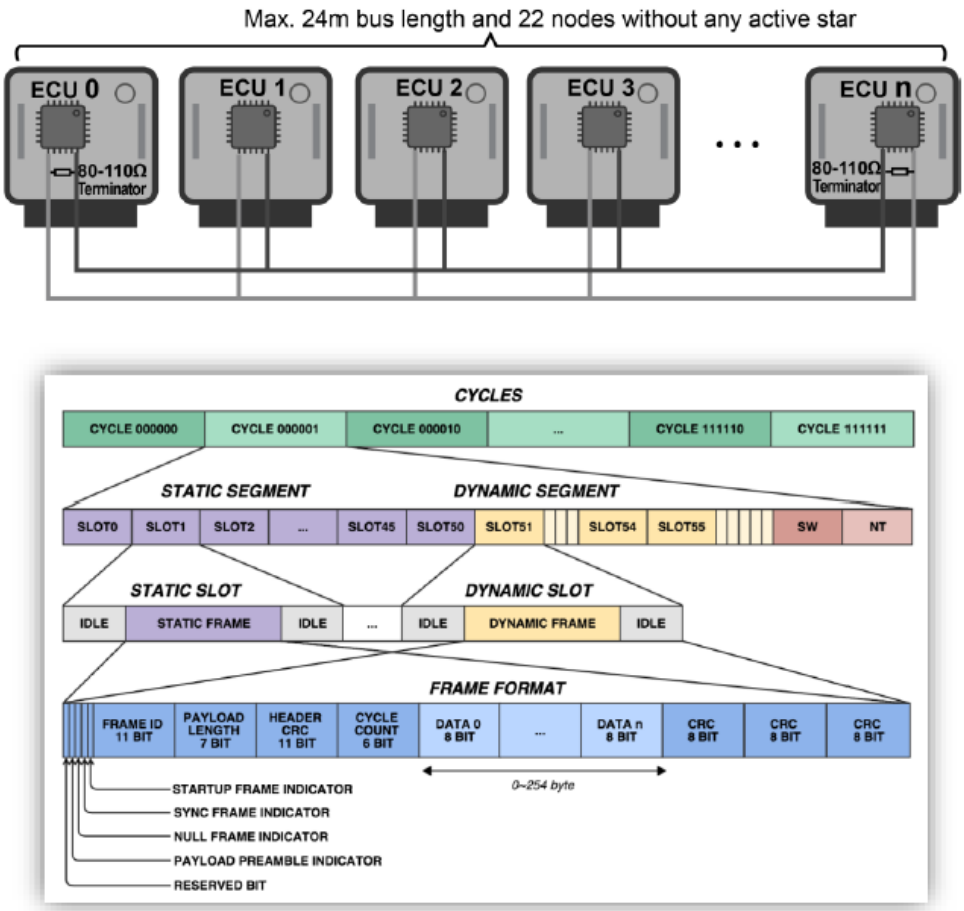

Figure 2.3: FlexRay network architecture showing bus topology and communication cycle structure [48].

2.1.5 MOST

MOST is a high-speed multimedia networking protocol designed specifically for in-vehicle infotainment and audio/video data distribution [25]. Developed by the MOST Cooperation, this protocol enables the transmission of synchronized, high-quality media streams between various entertainment and navigation modules [20].

MOST utilizes a ring or star topology and supports physical layers based on optical

fiber (MOST25, MOST50) or electrical conductors (MOST150) [25]. MOST150, the most advanced version, supports data rates up to 150 Mbps and allows the simultaneous transmission of audio, video, control, and packet data over a single network [20, 11].

The protocol is designed for isochronous and asynchronous data streaming, making it well-suited for managing tasks such as video playback, GPS routing, Bluetooth hands-free communication, and rear-seat entertainment [25]. It offers mechanisms for resource reservation, stream synchronization, and network management, making it an integral part of luxury vehicle infotainment systems [66].

Protocol	**Max Speed**	**Topology**	**Typical Use Cases**
LIN	20 kbps	Master-slave, single wire	Body electronics: mirror/seat control, HVAC [7, 46]
CAN	1 Mbps	Multi-master, broadcast bus	Powertrain, airbags, ABS, ECU coordination [66, 15]
CAN-FD	8 Mbps (data phase)	Multi-master, broadcast bus	Diagnostics, firmware updates, sensor data [13]
FlexRay	10 Mbps (per channel)	Dual-channel TDMA	Chassis control, brake/steer-by-wire systems [46, 11]
MOST	150 Mbps	Ring or star	Infotainment, navigation, video/audio streaming [25, 20]

Table 2.1: Comparison of Legacy In-Vehicle Communication Protocols

These foundational communication technologies enabled distributed control and automation in vehicles for decades [7, 66]. As automotive systems evolved, these networks provided a scalable yet dependable solution for structured vehicular communication [25, 20]. However, the advent of software-defined vehicles, connected services, and real-time data streaming has driven the need for next-generation communication protocols, paving the way for the adoption of AE and service-oriented architectures [63, 64, 15].

2.2 Why the Shift Toward AE?

The automotive industry is undergoing a paradigm shift from mechanical vehicles to software-defined, connected, and autonomous platforms [63, 64]. This transformation is driven by emerging functionalities such as ADAS, high-resolution camera and radar systems, V2X communication, and over-the-air software updates [64, 65, 55]. These use cases demand communication infrastructures that exceed the limitations of legacy protocols like CAN, LIN, and FlexRay [7, 66, 15]. AE has emerged as the most promising solution to meet these new requirements [63, 64, 41].

- **Authentication:** Any node can transmit on the bus without identity verification.
- **Encryption:** All data is transmitted in plaintext and can be intercepted or altered [51, 9].
- **Access control:** No mechanisms exist to restrict node-level permissions [77].

These gaps expose the network to common attack vectors:

- **Replay attacks:** Re-sending valid messages to manipulate vehicle behavior [9].
- **Spoofing:** Impersonating legitimate ECUs to send malicious commands [77].
- **DoS:** Flooding the bus to block critical traffic [13].

As vehicles become part of the Internet-of-Things (IoT), cybersecurity is no longer optional it is essential [55, 51].

2.2.1 Unsuitability for Emerging Automotive Architectures

As automakers shift toward centralized, domain-based, and zonal architectures, traditional buses struggle to meet the necessary performance and structural requirements [64, 65, 55]:

- **Cabling complexity:** Legacy topologies increase vehicle weight and wiring costs [7].
- **Dynamic configurability:** No support for software-defined routing or service discovery [103].
- **Incompatibility with OTA:** Inefficient for remote diagnostics and firmware updates [55].

These limitations delay implementation of advanced features like predictive maintenance, continuous software deployment, and autonomous control [7, 20].

Challenge	Legacy Protocol Weakness
Bandwidth	Max 1–10 Mbps; insufficient for ADAS and high-res video [64, 65, 7]
Scalability	Bus saturation, revalidation overhead [15, 20]
Efficiency	Broadcast model wastes bandwidth and CPU cycles [66, 13]
Security	No authentication, encryption, or access control [55, 51, 9]
Interoperability	Fragmented protocols with incompatible tools [25, 20]
Modern Use Cases	Inadequate for SDVs, V2X, OTA, and cloud integration [64, 55]

Table 2.2: Legacy Network Limitations

2.2.2 Growing Data Demands in Modern Vehicles

Modern vehicles integrate a wide range of high-bandwidth components [64, 65]:

- Multi-camera ADAS platforms for 360-degree perception [64].
- Radar and LiDAR sensors for autonomous navigation [3].
- Real-time infotainment and high-definition streaming [64].
- Telematics and continuous diagnostics streamed to the cloud [103, 55].

A single 1080p camera streaming uncompressed video can require over **1 Gbps** [64, 65]. Legacy communication buses, even in their most enhanced forms (e.g., CAN-FD or FlexRay), are incapable of handling such demands efficiently [7, 66]. Ethernet, with support for multi-gigabit data rates, becomes a natural choice [41, 39, 40, 43].

2.2.3 Rise of Centralized, Domain, and Zonal Architectures

As automotive systems evolve towards higher complexity and software-defined functionalities, the network architecture within vehicles has undergone a significant transformation [64, 103]. Traditional **domain-based architectures**, where ECUs are grouped by function (e.g., powertrain, body, infotainment), are increasingly giving way to **zonal architectures** that align ECUs based on their physical location in the vehicle [55]. This architectural evolution is enabled by the high bandwidth, determinism, and flexibility of AE [63, 65].

Domain-Based Architecture: Functional Grouping: In a domain-based setup, each group of ECUs is organized according to its function and connected to a corresponding domain controller [64]. While this approach offers organizational clarity, it introduces several drawbacks:

- High wiring complexity due to long cable runs between sensors/actuators and their domain ECUs [65].
- Increased latency and integration effort due to protocol translations between domains [55].
- Scalability challenges in introducing new ECUs or features across domains [15].

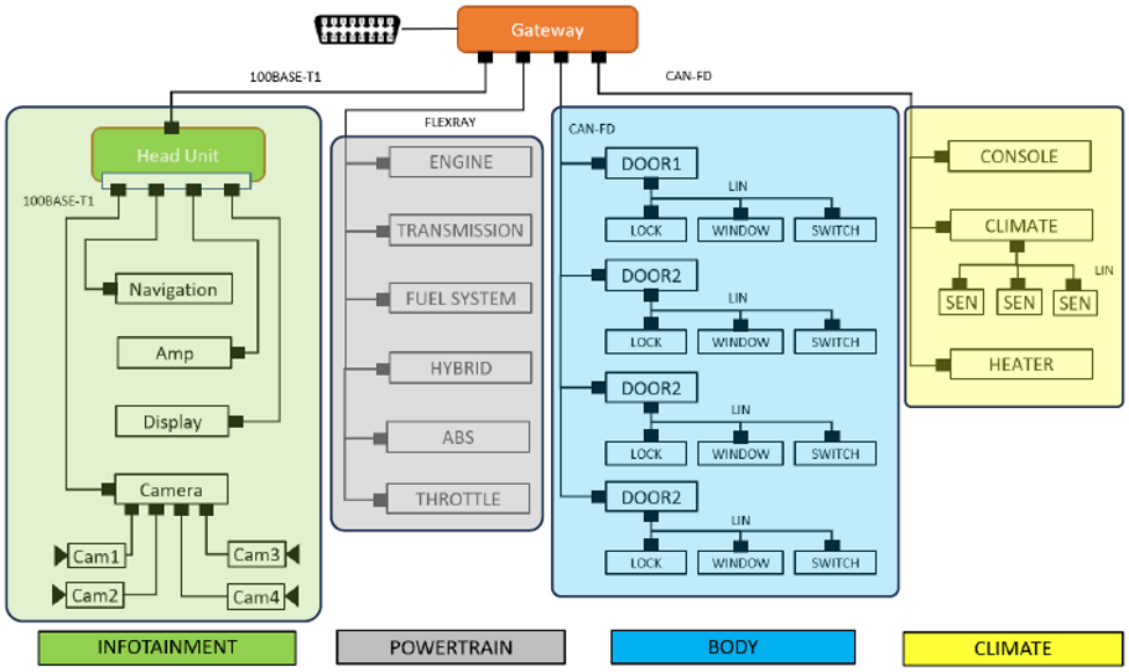

Figure 2.4: Domain-based automotive architecture where each functional domain (Infotainment, Powertrain, Body, Climate) is connected via a central gateway. Communication protocols include 100BASE-T1 Ethernet, CAN-FD, LIN, and FlexRay, demonstrating heterogeneity and centralized routing [39, 40, 48].

Zonal Architecture: Physical Grouping and Centralization In contrast, zonal architectures restructure the network by connecting ECUs within a physical zone (e.g., front-left, rear-right) to a local **Zonal Gateway ECU** [64, 103]. These zonal ECUs:

- Interface directly with nearby sensors and actuators [55].
- Preprocess and aggregate data before transmitting to a **central computing unit** [64].
- Enable shorter cable lengths, reducing both cost and vehicle weight [63].

This design enhances modularity, simplifies wiring harnesses, and improves maintainability [64, 55]. Moreover, zonal gateways typically use Ethernet-based communication to exchange data with the central processor, allowing for service-oriented and time-sensitive communication paradigms [65, 3].

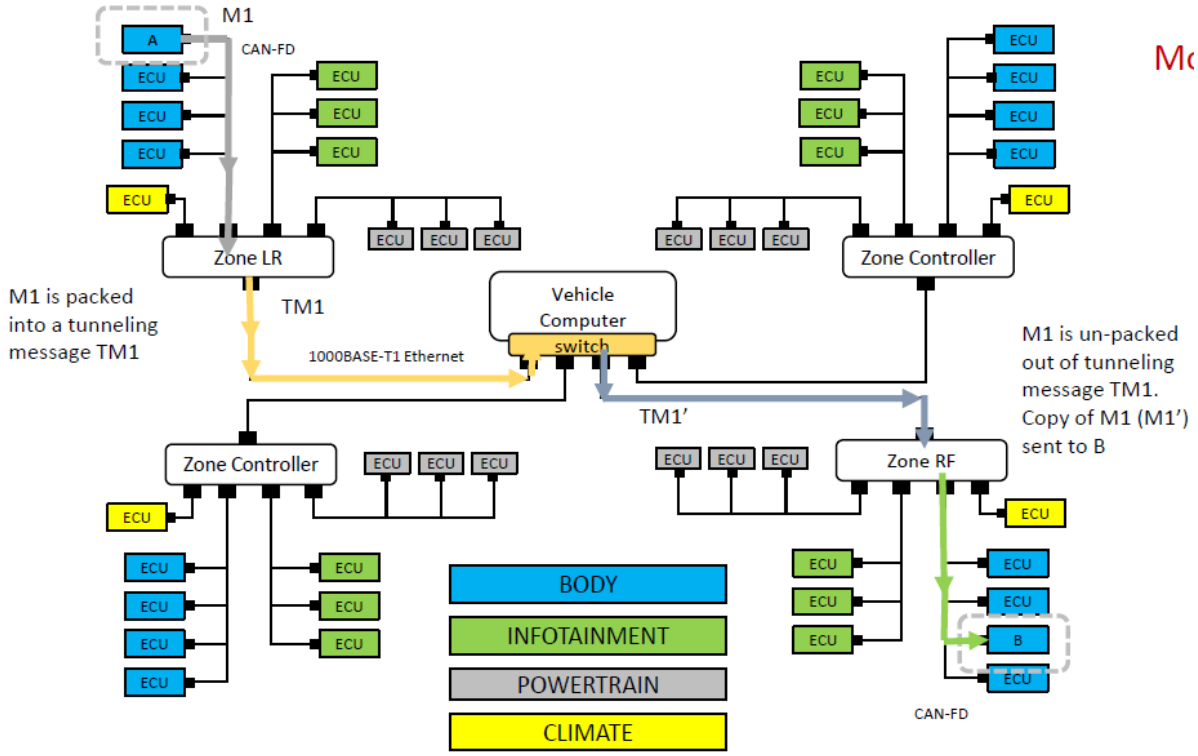

Figure 2.5: Zonal vehicle architecture demonstrating the use of localized ECUs per zone (e.g., Zone LR, Zone RF), each connected via 100BASE-T1 Ethernet to a central compute switch [39, 40, 43, 48].

Enabling Role of AE: The transition to zonal architecture is closely linked to the capabilities of AE [64, 65]. Modern Ethernet variants like **100BASE-T1** and **1000BASE-T1** provide the high-speed, low-latency, and deterministic communication needed for zonal designs [39, 40, 43]. Key features include:

- Support for ring, daisy-chain, and star topologies to optimize physical layout and fault tolerance [63].

Table 2.3: Comparison of Domain and Zonal Architectures

Feature	**Domain Architecture**	**Zonal Architecture**
Grouping Strategy	Functional (e.g., Powertrain, Body)	Physical (e.g., Front-left, Rear-right)
Wiring Layout	Centralized to each domain controller	Localized to zonal ECUs [55]
Data Aggregation	At domain controller level	Per zone, then to central processor [64]
Protocol Usage	Heterogeneous (CAN, LIN, FlexRay)	Mostly Ethernet-based [63, 64]
Gateway Complexity	High inter-domain translation overhead	Simplified and unified [55]
Scalability	Limited due to domain boundaries	High, modular and flexible [65]
Future-Readiness	Poor support for SDVs	Native support for SDVs and OTA updates [55]

- Compatibility with TSN for real-time performance [35].
- Native support for service-oriented protocols like SOME/IP and diagnostics via DoIP [16, 100, 49].

As highlighted in both industrial training materials and research literature [64, 103, 55], zonal architectures form the basis for highly modular and secure vehicles that rely on centralized compute units and edge-cloud integration. This approach not only reduces system complexity but also enhances support for over-the-air updates, predictive maintenance, and autonomous driving workloads [55, 51].

2.2.4 Technical Enablers, Ecosystem and Standards

AE adoption rests on three mutually reinforcing factors: technical enablers that make Ethernet viable in the vehicle environment, a mature ecosystem that lowers cost and complexity, and industry standardization that provides a clear migration path for OEMs and suppliers.

Technical Enablers: The development of automotive-grade physical layers (for example, 100BASE-T1 and 1000BASE-T1) and hardened connectors addressed the thermal, mechanical and EMI constraints of in-vehicle environments, making wired Ethernet hardware robust enough for vehicle deployment. [41] In parallel, TSN extensions to Ethernet (e.g., scheduled traffic and precise time synchronization) restore bounded latency and determinism for safety-critical control loops, enabling Ethernet to meet real-time requirements. [38]

IP Alignment and Protocol Support: Ethernet native support for TCP/IP and related service stacks simplifies integration with cloud backends and enterprise toolchains; support for automotive service and diagnostics protocols (for example SOME/IP and DoIP) further enables features such as FOTA and telemetry streaming. [64]

Ecosystem and Standards: A broad, mature supply chain commodity PHYs and switches, cost-effective cabling, and widely available TCP/IP stacks reduces engineering effort and shortens time to market, while standardization work (industry alliances and IEEE extensions) gives OEMs a predictable migration path. [64]

Adoption Challenges and Mitigation: Early barriers (environmental hardening, determinism, validation costs, and legacy coexistence) were progressively mitigated by automotive PHYs, TSN, tooling improvements, and phased migration strategies (hybrid topologies and gateways). Together, these factors make AE a practical backbone for zonal and centralized vehicle architectures. [41]

2.3 Security-Relevant Design Aspects of AE

AE introduces a shift from broadcast-based legacy buses to a high-speed, switched, and modular network architecture [63, 64]. This transition brings not only performance advantages but also redefines the context in which security must be addressed [14, 55]. Unlike traditional in-vehicle networks, the structure and operational characteristics of AE present both opportunities and challenges for implementing secure communication [16, 100, 103].

2.3.1 Predictable and Static Topologies

AE networks typically adopt static, predefined topologies such as daisy-chains, rings, or zonal configurations engineered during vehicle design [64, 103]. Unlike general-purpose Ethernet networks, they do not dynamically expand or adapt post-deployment. This controlled design simplifies threat modeling and anomaly detection, as any deviation from expected communication paths or timing patterns may indicate malicious behavior [55, 15].

2.3.2 ECU Integrity and Software Immutability

ECUs in AE are designed for specific functional domains and operate with fixed software stacks built on platforms like AUTOSAR, QNX, or VxWorks [6, 49]. These ECUs are typically not field-programmable post-production, which limits the risk of unauthorized software

changes [55]. While this immutability strengthens system trustworthiness, it also restricts the ability to patch vulnerabilities after deployment, emphasizing the need for secure-by-design development [50, 94, 78].

2.3.3 Communication Constraints and Protocol Selection

The network behavior in AE is shaped by real-time constraints, deterministic message scheduling, and minimal runtime negotiation [65, 3]. Lightweight protocols such as UDP are favored for their low latency, even though they offer no inherent reliability or encryption [16, 100]. This prioritization of timing and predictability over security features informs the type of safeguards that can be feasibly implemented [103, 55].

2.3.4 Physical Medium and Link Stability

AE operates primarily over single UTP links using standards like 100BASE-T1 and 1000BASE-T1 [39, 40, 43]. These physical links are point-to-point and not intended for user reconfiguration [41]. The deterministic and closed nature of these physical connections reduces exposure to plug-and-play attack vectors, but also limits the flexibility to deploy adaptive or software-defined security layers [103, 55].

2.3.5 Environmental Behavior as a Fingerprint

As a cyber-physical system, the vehicle's operational state directly affects communication patterns [55]. Actions such as braking, steering, and sensor activation produce distinct timing and message sequences [51]. These deterministic patterns offer a baseline against which deviations caused by faults or intrusions can be detected [13, 9]. This coupling between environmental stimuli and network behavior can be leveraged for real-time anomaly detection using signal-based fingerprinting [14, 55, 77].

2.4 Foundations and Design Principles of AE

The development of AE represents a significant paradigm shift in the field of in-vehicle networking, where classical real-time control protocols such as CAN, LIN, and FlexRay are increasingly being supplemented or replaced by a unified, high-bandwidth, and deterministic communication infrastructure [63, 64]. This transformation is grounded in theoretical advancements in digital communication, distributed systems, real-time computing, and deterministic networking [65, 3].

2.4.1 Evolution from Bus Systems to Switched Ethernet

Legacy automotive protocols like CAN and LIN are based on a bus-based architecture with limited bandwidth and deterministic arbitration methods [7, 66]. While suitable for traditional mechanical systems, their scalability and bandwidth constraints pose a theoretical upper bound on performance [15]. In contrast, switched Ethernet networks adopt a point-to-point topology where messages are forwarded through switches, enabling collision-free communication and QoS guarantees [63, 64].

From a theoretical standpoint, the TSN paradigm plays a crucial role in transforming Ethernet into a deterministic network [65, 3]. TSN combines multiple IEEE standards such as 802.1Qbv (time-aware shaping), 802.1AS (clock synchronization), and 802.1CB (frame replication) into a cohesive framework that allows Ethernet to exhibit behavior analogous to real-time buses, but with significantly higher throughput and flexibility [38, 34].

2.4.2 Information Theory and Full-Duplex Communication

At the physical and data link layers, AE leverages full-duplex communication, allowing concurrent bidirectional data flow [41, 39, 40]. This effectively doubles the channel capacity, reduces propagation delay, and eliminates the arbitration complexities inherent in half-duplex systems [63, 64]. The use of UTP in standards like 100BASE-T1 and 1000BASE-T1 aligns with Shannon theorem by maximizing data rate within constrained electromagnetic environments [40, 43].

Moreover, frame prioritization and VLAN tagging (IEEE 802.1Q) enable logical segmentation of communication flows, thereby enforcing theoretical models of resource allocation, latency bounds, and channel isolation [41, 38].

2.4.3 Time Synchronization and Global Clock Coordination

Distributed automotive systems rely on synchronized operations among ECUs [65]. Traditional buses use implicit synchronization through scheduling. However, in switched networks, global clock coordination becomes essential [103]. gPTP, based on IEEE 802.1AS, ensures sub-microsecond time synchronization across the network, which is fundamental for closed-loop control systems, sensor fusion, and coordinated actuation [38, 34, 55].

In theoretical terms, synchronization latency, clock skew, and jitter bounds can be modeled and minimized to ensure system-level temporal correctness [3, 103].

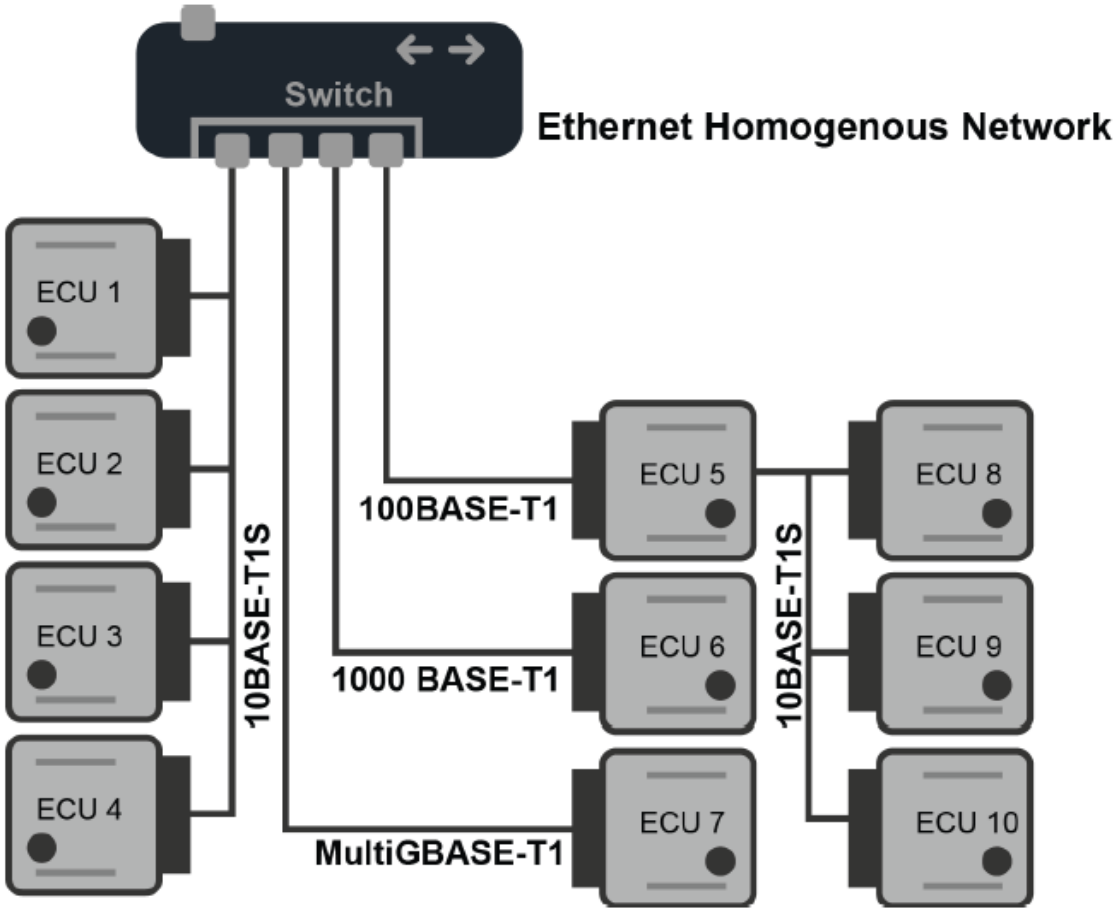

Figure 2.6: Homogeneous AE network configuration where a central switch connects ECUs over various physical layer standards including 10BASE-T1S, 100BASE-T1, 1000BASE-T1, and MultiGBASE-T1 [39, 40, 42, 43]. This architecture supports high-speed and scalable communication between zonal and domain controllers[48].

2.4.4 Layered Modularity and OSI Model Alignment

AE adopts a layered communication stack, conforming to the OSI model [41]. This modular structure facilitates protocol interoperability, abstraction, and system verification, which are key pillars of software-defined vehicle architectures [63, 64]. Each layer from physical transmission (Layer 1) to application middleware (Layer 7) can be independently developed, tested, and certified, thereby enhancing system resilience and adaptability [103, 49].

2.5 Architectural Applications and Advantages of AE

The theoretical robustness of AE translates into concrete benefits when applied across various vehicular domains [63, 64]. This section explores the architectural deployment scenarios, logical advantages, and computational efficiencies of Ethernet-based communication frameworks [65, 3].

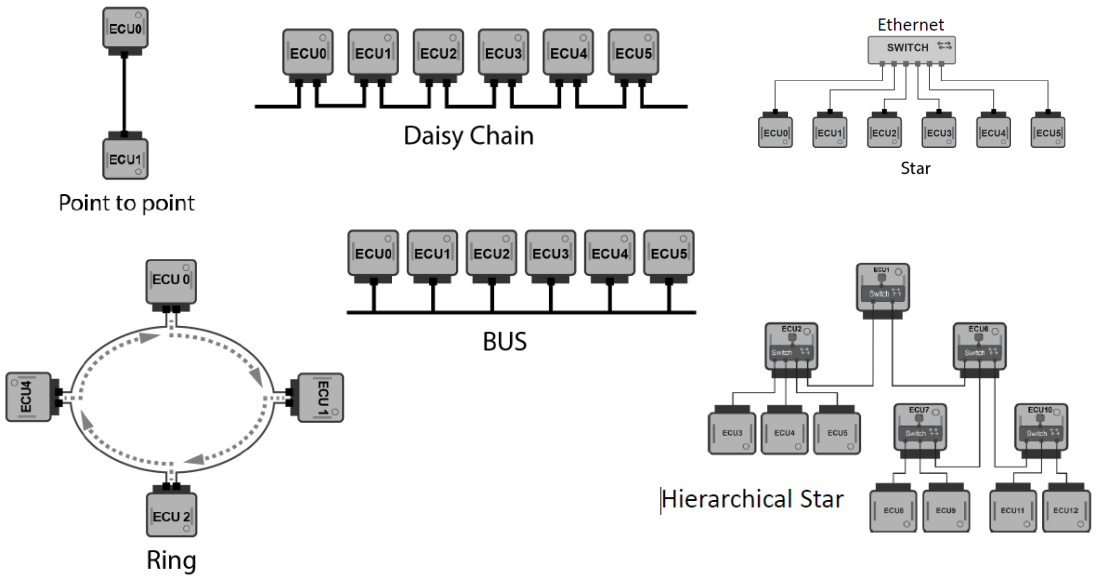

Figure 2.7: Common automotive networking topologies including bus, ring, star, and zonal configurations. Adapted from Intrepid Control Systems Training, 2025 [41, 48].

2.5.1 Network Topologies and Architectural Flexibility

AE supports a variety of topologies star, daisy-chain, ring, and zonal hybrid architectures [64, 41]. These topologies are not merely physical layouts; they correspond to graph-theoretical models of communication systems where nodes (ECUs) and edges (links) define connectivity and flow capacity [103].

- In a zonal architecture, localized ECUs are connected to a zonal gateway that aggregates data and routes it to a central processor. This reduces edge-to-core latency and enables hierarchical control [55].
- Ring topologies support redundancy through *frame replication and elimination*, enhancing fault tolerance using path diversity [36].

- Logical partitioning using VLANs models *multi-tenant systems*, where control and infotainment networks coexist without interference [41, 38].

These standards allow the implementation of real-time contracts and latency budgets, essential in the verification of automotive functional safety (e.g., ISO 26262 compliance) [103, 55].

2.5.2 Integration with IP-Based Systems and Cloud Computing

From a systems theory perspective, the ability of AE to operate within IP-based infrastructures is a pivotal strength [63, 64]. It enables seamless communication with external services, cloud analytics, and remote diagnostics, which were previously infeasible with legacy buses [103, 55].

- Protocols like SOME/IP, DoIP, and gPTP operate at higher OSI layers, enabling SOA and time-synchronized sensor networks [16, 100, 49].
- Edge-cloud coordination is facilitated by TCP/IP layers, supporting tasks like predictive maintenance, anomaly detection, and fleet-level OTA updates [55, 51].

These integrations allow vehicle communication to be modeled using distributed computing frameworks and event-driven architectures [64, 103].

2.5.3 Scalability, Security, and Future-Readiness

AE is not only a communication medium but also a platform enabler for SDVs [55, 50]. Its modular design supports:

- Incremental scalability through plug-and-play ECUs [63].
- Service discovery and reconfiguration via middleware [6, 16].
- Security primitives such as MACsec (Layer 2), IPsec (Layer 3), and TLS (Layer 4) that align with zero-trust architectures and cryptographic assurance models [55, 50, 94].

In effect, the architecture supports a secure, scalable, and software-defined future, where vehicle functionality is abstracted from hardware constraints [64, 103, 55].

2.6 OSI Model and AE Protocol Stack

AE adopts a layered communication model that aligns with the classic OSI model [41]. This standardization allows developers and system architects to understand how Ethernet-based protocols fit within the in-vehicle communication architecture and to take advantage of modularity, abstraction, and protocol interoperability [63, 64].

Each OSI layer performs specific communication functions, with defined interfaces to adjacent layers. The AE stack adapts this model for performance, reliability, fault tolerance, and real-time operation in the automotive environment [65, 3, 103]. Below, we elaborate on each layer in detail.

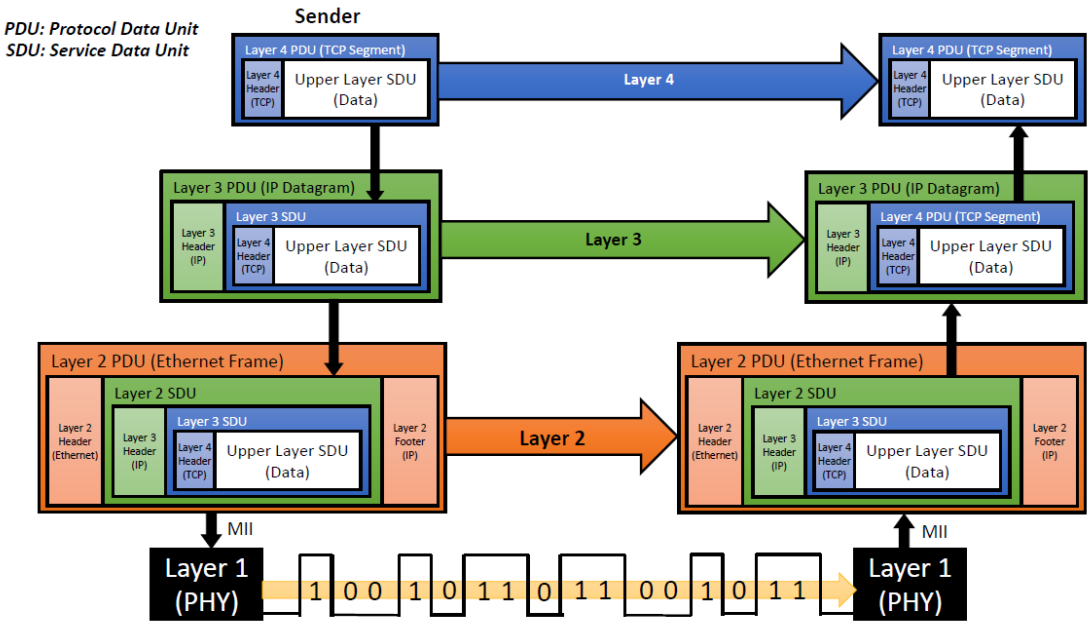

Figure 2.8: Message encapsulation and protocol layer interaction in Ethernet networks. The figure shows how data moves through each OSI layer with headers added at each step. Adapted from Intrepid Control Systems Ethernet Training (2023) [45, 48].

2.6.1 Layer 1: Physical Layer

The physical layer defines the electrical, mechanical, and signaling interface between devices. In AE, this layer supports [43]:

- **100BASE-T1 (IEEE 802.3bw)**, **1000BASE-T1 (IEEE 802.3bp)**, and **10BASE-T1S (IEEE 802.3cg)**: These standards use a single unshielded twisted pair (UTP), providing full-duplex point-to-point transmission with minimal cabling.

- **Robust PHYs** that integrate electromagnetic compatibility (EMC) filtering, fault detection, and energy-efficient Ethernet features to reduce power consumption.
- Operation in harsh environments with temperature, vibration, and EMI compliance for automotive-grade requirements.

This layer is critical in enabling lightweight cabling and long-term reliability, particularly for zonal and backbone links [103].

2.6.2 Layer 2: Data Link Layer

The data link layer ensures reliable frame delivery over the physical medium [41]. In AE, it handles:

- **MAC addressing**: Uniquely identifies nodes on the network.
- **VLAN tagging (IEEE 802.1Q)**: Enables logical traffic separation across multiple domains (e.g., infotainment, safety).
- **EtherType field**: Differentiates higher-layer protocols.
- **Frame delimitation and CRC**: Ensures integrity and accurate framing.
- **TSN standards integration**: TSN features like time-aware shaping (IEEE 802.1Qbv), clock synchronization (IEEE 802.1AS), and frame replication (IEEE 802.1CB) extend Ethernet for time-critical applications [38, 34].

These features help maintain deterministic performance and real-time communication for safety-critical systems [3, 103].

2.6.3 Layer 3: Network Layer

The network layer provides logical addressing and packet routing. AE uses **Internet Protocol (IP)** to enable [41, 49]:

- Routing between ECUs, gateways, and backend cloud systems
- Integration with V2X and smart infrastructure systems
- Support for diagnostics, OTA updates, and fleet management tools

While most AE networks are flat and localized, IP facilitates inter-domain and external communication [55].

2.6.4 Layer 4: Transport Layer

The transport layer ensures appropriate delivery of messages between applications [63]. AE supports:

- **UDP**: Low-latency, connectionless communication suitable for time-sensitive data (e.g., radar, camera feeds) [16].
- **TCP**: Reliable, ordered delivery for applications like software updates, diagnostics, and secure data logging [100].

UDP dominates safety-critical paths where timing is prioritized over reliability; TCP is used for high-integrity, non-real-time operations [103].

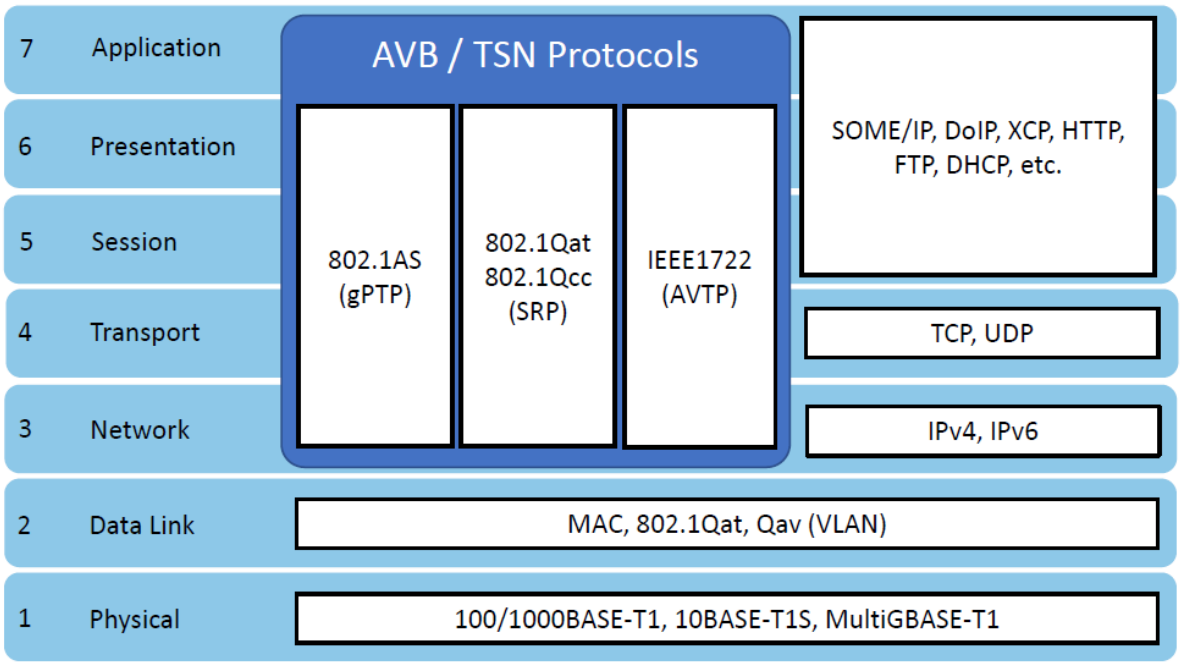

Figure 2.9: AE protocol stack mapped to OSI layers. It illustrates how application, transport, network, and data link protocols align with Ethernet standards in the automotive context. Adapted from Intrepid Control Systems Ethernet Training (2023) [45, 48].

2.6.5 Layer 5–7: Session, Presentation, and Application Layers

These layers encompass middleware, services, and applications executed over the network [16, 100, 49]. Key protocols and services include:

- **SOME/IP**: Provides service discovery, serialization, and asynchronous messaging [16, 100].

- **DoIP**: Supports UDS over Ethernet for standardized vehicle diagnostics [49].
- **gPTP**: Based on IEEE 802.1AS, used to synchronize clocks across all ECUs with sub-microsecond accuracy [38, 34].

These application-layer technologies form the foundation of modern vehicle software architectures, enabling modular deployment, centralized updates, and rich connectivity features [55, 50, 94].

2.7 AE Frame and Packet Structure

AE frames follow the IEEE 802.3 standard with extensions for in-vehicle requirements such as determinism and priority tagging [41, 45]. A typical Ethernet frame consists of several well-defined fields, each serving a distinct role in communication integrity, routing, and prioritization [103, 55].

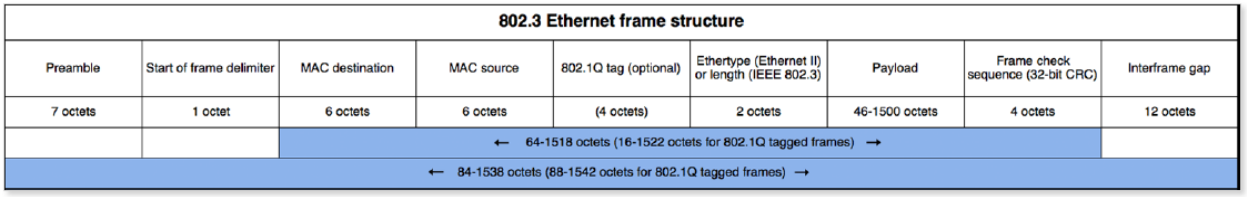

Figure 2.10: Structure of a standard IEEE 802.3 Ethernet frame including all essential fields: preamble, MAC addresses, optional 802.1Q tag, ethertype/length, payload, CRC, and interframe gap [41, 45]. Total frame size ranges from 64 to 1518 octets (up to 1522 with VLAN tagging)[48].

Note: If the payload is smaller than 46 bytes, it is padded to meet the minimum Ethernet frame size [41]. When TSN is applied, additional metadata and scheduling information may influence frame handling but not frame format [38, 34, 36].

Understanding this frame structure is essential for analyzing communication reliability, security enforcement (via MACsec), and diagnostics behavior across ECUs in Ethernet-based architectures [55, 50].

2.8 Types of AE

AE offers a range of PHY standards, each optimized for specific use cases depending on bandwidth requirements, physical layout, and network criticality [39, 40]. These PHY types define the signaling, speed, and topology supported by each class of in-vehicle communication [63, 64].

Table 2.4: AE Frame Structure

Field	Size	Description
Preamble	7 bytes	Synchronization bits for receiver clock alignment. Composed of alternating 1s and 0s to alert the receiver and prepare for data.
SFD	1 byte	Indicates the end of the preamble and the start of the frame.
Destination MAC	6 bytes	Identifies the intended recipient ECU's MAC address or a broadcast/multicast target.
Source MAC	6 bytes	MAC address of the sending ECU.
VLAN Tag (optional)	4 bytes	**IEEE 802.1Q** tag includes VLAN ID and priority code point (PCP) for TSN traffic classification [38, 33].
EtherType / Length	2 bytes	Specifies the protocol type (e.g., IPv4, IPv6, SOME/IP) or length of payload if less than 1500 [41].
Payload	46–1500 bytes	Data portion, which may contain application layer protocol frames like SOME/IP, DoIP, or gPTP [16, 100, 49].
FCS	4 bytes	CRC for frame integrity verification.

2.8.1 10BASE-T1S

10BASE-T1S (IEEE 802.3cg) is designed to enable a **cost-efficient and scalable multi-drop topology** [42]. It allows multiple devices (up to 8) to share the same twisted pair cable, significantly reducing wiring complexity. It is best suited for replacing low-speed buses like CAN or LIN in body control, lighting, or seat positioning systems [7, 66].

Table 2.5: AE PHY Standards

Standard	Speed	Max Length	Use Case
10BASE-T1S	10 Mbps	25–50m	Low-cost multi-drop communication suitable for sensor and actuator networks; can replace CAN or LIN buses in low-bandwidth domains [42].
100BASE-T1	100 Mbps	15–40m	Supports cameras, infotainment systems, and mid-speed ECUs; ideal for ADAS sensors and clusters [39].
1000BASE-T1	1 Gbps	15m	High-speed applications such as radar data, aggregated zonal processing, and centralized computing [40].
MultiGig AE (2.5G–10G)	2.5–10 Gbps	¡10m	Backbone links between high-performance domain/zonal controllers and central compute units; suitable for HD video fusion and autonomous driving workloads [43].

2.8.2 100BASE-T1

This is the most widely deployed variant in early AE implementations [39]. It supports point-to-point full-duplex communication at 100 Mbps over a single twisted pair [41]. It is commonly used for streaming data from cameras and infotainment units and is compliant with stringent automotive EMI standards [103].

2.8.3 1000BASE-T1

Targeting high-data applications such as radar processing or LIDAR inputs, 1000BASE-T1 enables 1 Gbps full-duplex transmission over a single twisted pair [40]. This PHY type is key in enabling zonal ECUs and in reducing latency across centralized systems that fuse data from multiple sensors [64, 55].

2.8.4 MultiGig AE

MultiGig standards (2.5G, 5G, 10G, and beyond) are emerging to support autonomous systems and central compute platforms requiring ultra-high throughput [43]. While currently used in premium vehicle segments, the growing need for rich sensor data fusion, HD video streaming, and real-time decision-making accelerates its adoption [63, 55]. These links generally span short distances (<10 meters) within centralized compute domains [103].

2.9 Functional and Design Advantages of AE

AE introduces a suite of architectural and operational benefits that make it indispensable for the modern vehicle [63, 64]. Its advantages span across network consolidation, scalability, determinism, and support for connected and software-defined platforms [65, 3, 103]. These benefits are foundational for the design of next-generation vehicles.

2.9.1 Unified Backbone

AE enables the replacement of multiple legacy communication buses with a single high-speed, full-duplex, IP-compatible backbone [41, 39]. This unification reduces the complexity of wiring harnesses, minimizes gateway requirements between different protocols, and streamlines network diagnostics [55]. It also enhances data integration and latency performance across various domains (e.g., infotainment, ADAS, powertrain) by allowing centralized and coordinated communication [64, 103].

2.9.2 Zonal Architecture Support

Modern vehicles are increasingly shifting from traditional domain-based architectures to zonal architectures [64, 55]. In a zonal model, ECUs are grouped based on their physical location rather than their function. AE supports this transformation by providing high-speed connectivity between zonal gateways and a central computing unit [65]. This results in:

- Substantial reduction in cabling weight and cost [55].
- Easier vehicle assembly and maintenance [103].
- Improved modularity for software updates and ECU replacement [55].

Ethernet support for multiple topologies (star, ring, daisy-chain) further facilitates this flexible, location-based structuring [41].

2.9.3 TSN Real-Time Communication

One of the groundbreaking features of AE is its support for TSN [65, 3]. TSN introduces a set of IEEE standards that enable bounded latency, jitter control, and guaranteed delivery of time-critical data [38, 34]. It ensures that even in high-load scenarios, messages essential for safety (e.g., brake commands, airbag deployment signals) are delivered within predefined time windows. Key TSN standards include:

- **IEEE 802.1AS**: Clock synchronization across all network nodes [38].
- **IEEE 802.1Qbv**: Time-aware traffic shaping to prioritize real-time flows [34].
- **IEEE 802.1CB**: Redundancy through frame replication and elimination [36].

With TSN, Ethernet achieves determinism comparable to real-time automotive buses while maintaining much higher bandwidth [103].

2.9.4 Software-Defined Enablement

The advent of SDVs demands a communication platform that can support centralized software management, OTA updates, remote diagnostics, and modular feature deployment [55]. Ethernet inherent compatibility with the TCP/IP stack enables it to seamlessly interface with backend cloud platforms [63, 64]. Benefits include:

- Efficient delivery of OTA updates across domains [55].
- Real-time telemetry streaming and fault logging [103].
- Support for service-oriented architectures through SOME/IP and DoIP protocols [16, 100, 49].

This capability empowers OEMs to continuously upgrade vehicle functionality post-production, reduce recall incidents, and implement predictive maintenance strategies [55].

In summary, AE functional and design advantages make it a cornerstone of future automotive electrical/electronic (E/E) architectures [64, 103]. It empowers OEMs to meet performance, flexibility, and safety targets, all while enabling next-generation user experiences through software-driven innovation [55, 50].

2.10 Challenges in Implementing AE

While AE presents transformative benefits, its adoption also introduces several technical and operational challenges [63, 55]. These must be carefully addressed to ensure smooth integration and long-term scalability across vehicle platforms.

2.10.1 TSN Complexity

Implementing TSN in real-world automotive systems is a non-trivial task [65, 3, 38]. TSN requires:

- Time-aware scheduling and shaping algorithms to prioritize safety-critical traffic [34].
- Precise network-wide time synchronization using protocols such as IEEE 802.1AS [38].
- Configuration tools and verification frameworks to validate scheduling, redundancy, and QoS policies [103].

Misconfiguration of TSN flows may result in communication delays or message loss, potentially affecting system safety [55].

2.10.2 Higher Initial Cost

Compared to established bus systems like CAN, AE involves a higher upfront investment in hardware and design [7, 66].

- Ethernet switches, TSN-compliant PHYs, and gateway controllers are more expensive [39, 40].
- EMI filtering, shielding, and connector standards are more demanding [43].
- Design teams require additional training and validation tools [103].

However, the long-term cost savings through wiring reduction and platform scalability often offset the initial deployment expenses [55].

2.10.3 Cybersecurity Requirements

The IP-based nature of AE introduces a broader attack surface [14, 55]. Unlike legacy protocols, Ethernet networks are vulnerable to:

- Spoofing and MitM attacks on unprotected links [16, 100, 77].

- Data interception or manipulation due to lack of encryption [51].
- Unauthorized access from internal and external vectors (e.g., V2X, telematics) [9].

To mitigate these risks, AE requires built-in security measures such as [55, 50, 94]:

- MACsec, IPsec, and TLS for data confidentiality and integrity.
- IEEE 802.1X for port-based authentication and access control.
- Intrusion detection and firewall mechanisms tailored to automotive requirements [51, 9].

2.10.4 Legacy Coexistence

In practice, Ethernet must coexist with existing protocols such as CAN, LIN, and FlexRay during transitional phases of platform development [7, 66, 15]. This hybrid coexistence introduces:

- Increased design complexity in integrating gateway controllers [20].
- Latency management when bridging between real-time and best-effort buses [15, 13].
- Software abstraction layers for harmonizing protocol behavior across ECUs [103].

Effective coexistence strategies are essential to ensure interoperability, minimize revalidation costs, and enable phased adoption of Ethernet [55].

2.11 Standard Ethernet vs. AE: Key Differences

AE extends the core functionality of IEEE Ethernet by addressing unique operational, environmental, and performance requirements of in-vehicle networks [63, 41]. This section compares standard Ethernet to AE across multiple design and application dimensions [55].

2.11.1 Environmental and Operational Constraints

Standard Ethernet is typically deployed in stable, climate-controlled environments like offices and data centers. In contrast, AE must operate under [39, 40, 43]:

- Extreme temperature variations (–40°C to +125°C).
- High electromagnetic interference.
- Continuous mechanical vibration and shock.

AE components are automotive-grade and comply with standards like **AEC-Q100/101** for reliability and safety [50].

2.11.2 Topology and Cabling

- Standard Ethernet employs STP or fiber optics, using star or bus topologies over distances exceeding 100 meters [41].
- AE uses UTP cables (e.g., 100BASE-T1, 10BASE-T1S) for compact, short-range, point-to-point or multi-drop topologies [39, 42].
- Zonal and ring configurations are supported for redundancy and localized processing [103].

2.11.3 Protocol Adaptation and Application Layers

Standard Ethernet supports general-purpose Internet protocols like HTTP, FTP, and SNMP. AE introduces in-vehicle optimized protocols [16, 100, 49]:

- **SOME/IP**: Service-oriented communication between ECUs [16].
- **DoIP**: Standardized diagnostics via Ethernet [49].
- **gPTP**: Time synchronization for TSN-enabled systems [38].
- **VLANs**: For logical separation of traffic streams [41].

2.11.4 Safety and Functional Certification

AE must meet strict functional safety standards [55, 50]:

- **ISO 26262** for safety compliance.
- **ASIL** for critical systems.
- Switches and PHYs must support frame prioritization, fault tolerance, and error detection [94].

Table 2.6: Comparison Between Standard Ethernet and AE

Feature	**Standard Ethernet**	**AE**
Environment	Controlled (e.g., office)	Harsh (temperature, EMI, vibration)
Cabling	Shielded twisted pair, fiber	Single UTP (100BASE-T1, 10BASE-T1S)
Topology	Star, bus	Point-to-point, multi-drop, ring
Real-Time Support	No native support	TSN-enabled (802.1Qbv, 802.1AS)
Protocols	TCP/IP, HTTP, FTP	SOME/IP, DoIP, gPTP
Safety Compliance	Basic EMI	ISO 26262, AEC-Q, ASIL
Use Cases	LAN, internet, enterprise	ADAS, infotainment, diagnostics

2.12 Ethernet vs. AE

While AE is based on IEEE Ethernet standards, it incorporates enhancements in protocol stack, hardware, safety, and determinism to meet the specialized needs of in-vehicle communication [63, 64, 41]. These differences underscore why AE is not simply a transport technology, but a foundational enabler for next-generation mobility [55, 50].

2.13 Security in AE

With the growing integration of connectivity features and autonomous functions, securing in-vehicle networks has become critical [14, 55]. AE, by design, is poised to handle large volumes of data, but its reliance on standard Ethernet protocols exposes it to traditional IT vulnerabilities [16, 51]. This section lays the foundation for the security-centric discussion in the next chapter.

2.13.1 Need for Security in AE

As vehicles adopt features such as V2X communication, OTA updates, and cloud diagnostics, the AE backbone becomes a target for cyberattacks [55, 9]. Unlike traditional automotive protocols like CAN, AE is built upon the IP stack, which increases the system's exposure to known internet-based threats [14]. Common attacks include DoS, replay, spoofing, message injection, and MITM attacks [13, 9, 77].

2.13.2 Impact on Safety-Critical Systems

AE is used in safety-critical applications such as steering, braking, and ADAS [55]. Therefore, a successful attack can compromise not only privacy and data integrity but also passenger

safety [51]. The delay introduced by some security mitigations (e.g., encryption, authentication) can affect real-time performance, leading to potential hazards in dynamic driving scenarios [103].

2.13.3 Current Solutions and Gaps

Existing AE security solutions include firewalls, IDS, MACsec, TLS/IPsec encryption, and access control mechanisms [55, 51]. However, many of these solutions are adaptations from traditional Ethernet and may not fully meet automotive latency and reliability constraints [9, 77]. The need for lightweight, automotive specific security frameworks such as AUTOSAR continues to grow [6].

2.13.4 Compliance and Standards

The emergence of cybersecurity regulations like ISO/SAE 21434 and UNECE WP.29 R155 mandates manufacturers to incorporate systematic risk assessments and mitigation strategies into vehicle design [50, 94]. These standards require AE to support confidentiality, integrity, and availability through its entire protocol stack [55].

2.13.5 Towards Secure-by-Design Architectures

Going forward, the automotive industry must embrace a secure-by-design approach for AE [55, 50]. This includes integrating security functions at all OSI layers, performing attack surface analysis, and applying formal verification techniques [14, 103]. Collaboration across OEMs, Tier-1 suppliers, and cybersecurity experts is essential to establish resilient in-vehicle Ethernet networks [55, 94].

Building on the foundational understanding of why security is indispensable for AE, the following chapter delves deeper into the threat landscape specific to in-vehicle Ethernet networks. It presents a detailed taxonomy of cyberattacks targeting AE infrastructures, analyzes their mechanisms and potential impacts, and explores case studies that highlight the urgency for robust defense strategies. This transition from principles to practical attack scenarios marks a critical step in developing a comprehensive security framework for future vehicle networks.

CHAPTER 3

Cyber Threats to Automotive Ethernet

As automotive systems continue to evolve, so do the security challenges, particularly with the increasing adoption of AE. Chapter 2 explored the rise of AE as the backbone of modern IVNs, highlighting its advantages over legacy protocols such as CAN and FlexRay. While AE offers significant benefits in terms of bandwidth, scalability, and integration with cloud and IT ecosystems, it also introduces a new set of security concerns that must be addressed to ensure the safe deployment of autonomous, connected, and software-defined vehicles.

This chapter examines the various cyber threats targeting AE, focusing on their mechanisms, potential impact, and real-world implications. Unlike traditional network security, which is often reactive, the automotive environment requires proactive defense strategies due to the critical nature of safety applications like ADAS, braking, and steering. The absence of built-in security features in many AE protocols, such as authentication and encryption, makes the network vulnerable to a wide array of attacks, including DoS, Replay, and Impersonation.

In the sections that follow, we dissect the anatomy of these attacks, explore their consequences on vehicle functionality, and identify the security gaps that must be addressed to safeguard the next generation of IVNs. With the growing reliance on AE for essential vehicle operations, understanding these threats is a crucial step in securing the future of automotive networking.

3.1 DoS Attack

3.1.1 Overview

A **DoS attack** is a malicious attempt to disrupt the normal functioning of a targeted system or network by overwhelming it with excessive traffic or exploiting protocol-level vulnerabilities to exhaust system resources. Within the framework of network security, DoS attacks are classified under the availability aspect of the CIA triad. The core objective of a DoS attack is

to render a service or device temporarily or permanently unavailable to its legitimate users, thereby violating the operational integrity of the system.

3.1.2 How It Works in AE

In AE systems, DoS attacks exploit architectural characteristics such as static topologies, deterministic scheduling, and real-time constraints typical of IVNs. These attacks can target multiple OSI layers (see figure 3.1):

- **Data Link Layer (L2):** Broadcast storms and MAC flooding attacks can exhaust switch forwarding tables or queues (e.g., via spoofed MAC addresses).
- **Network & Transport Layers (L3/4):** High-volume IP/UDP/TCP packets can saturate network interfaces or exploit protocol states (e.g., flooding SOME/IP or DoIP sessions).
- **Application Layer (L7):** Protocols such as DoIP can be abused to flood ECUs with repeated service requests, disrupting diagnostics or firmware flashing operations.

Due to the adoption of TSN and the critical need for low-latency, deterministic message delivery, even short-lived DoS attacks can induce severe real-time violations, potentially impacting safety-critical functionalities like braking, steering, or sensor fusion.

3.1.3 Known Attack Methods

Several DoS attack implementations and techniques have been studied in AE research and testbeds:

- **VLAN-based Flooding Attacks:** Injecting high-priority Ethernet frames (e.g., using 802.1Q PCP bits) to monopolize switch queues.
- **Bandwidth Saturation:** Generating a flood of packets that exhaust available bandwidth and prevent delivery of time-critical traffic.
- **Service Exploitation Attacks:** Targeting protocol-level services like SOME/IP or DoIP to trigger excessive request handling on ECUs.
- **Queue Starvation / Burst Injection:** Overloading TSN traffic classes with bursty low-priority traffic to delay or drop scheduled high-priority packets.
- **Protocol Abuse (e.g., STP Flooding):** Leveraging network configuration protocols to induce instability in Ethernet switch behavior.

3.1.4 Related Works

The following peer-reviewed studies and technical papers showcase the implementation and real-world implications of DoS attacks in AE environments. Each paper underscores the critical severity of these attacks in disrupting safety, reliability, and in-vehicle communication integrity.

Jeong et al. [51] Jeong and colleagues presented one of the earliest documented flooding-based DoS attacks targeting an AE switch infrastructure using AVB protocols. The experimental setup involved crafting Ethernet frames tagged with VLAN identifiers and artificially inflated PCP values. This elevated the malicious traffic above normal communication streams, effectively allowing it to monopolize switch buffers. The results were immediate and disruptive: AVTP streams critical for infotainment and sensor communication were dropped, and the target ECU became unreachable from other nodes. Despite only targeting a localized switch, the attack had system-wide effects due to Ethernet reliance on static priority queuing. The paper underscores that AE QoS mechanisms can be manipulated, making localized flooding a potent denial tactic even in well-structured IVNs.

Luo et al. [10] explored bandwidth-saturation attacks on a TSN-based backbone within an ADAS-focused architecture. Using OMNeT++ simulations, the researchers generated large volumes of traffic that were injected into TSN stream queues. As the volume of illegitimate traffic increased, the simulation revealed a clear breakdown in real-time scheduling guarantees. Time-sensitive packets, such as those from LiDAR sensors or camera feeds, began missing their deadlines due to queuing delays and link saturation. The authors describe how this form of DoS attack does not require precise targeting flooding the network indiscriminately is sufficient to degrade or completely halt deterministic message flows. This study identifies DoS as the most destructive attack in its threat model and explains how even short-duration floods can cause cascading failures in coordinated perception and control systems across multiple ECUs.

Lang et al. [91] This thesis takes a broad and analytical approach by categorizing DoS threats in AE into two types: resource exhaustion and protocol-based DoS. Resource exhaustion attacks were simulated by flooding a node (e.g., an ECU or a central gateway) with unfiltered Ethernet frames, consuming its processing cycles and memory buffers. Once saturated, the node was unable to process any other incoming packets effectively rendering it inoperable for the duration of the attack. The second category leverages STP behavior, where an attacker floods the network with STP configuration frames. Since AE switches

often rely on STP to manage redundant paths, the constant change in topology caused by the flood created routing instability, preventing any stable frame forwarding. The thesis concludes that most AE systems, even with redundancy, lack robust safeguards against these saturation and protocol-based attacks, making such threats highly feasible in practical deployments.

Talic et al. [54] Focusing on TCP-based attacks within AE diagnostic frameworks, this paper demonstrates the use of `tcpkill`, a well-known Linux utility, to perform a targeted DoS attack on DoIP sessions. The researchers established a legitimate firmware flashing process between a backend tool and an ECU, then launched the attack by injecting forged TCP RST packets. The ECU, following protocol specifications, terminated the connection instantly. The flashing process failed and displayed "Communication failed" on the diagnostic interface. This showcases how even a single well-placed packet can derail crucial OTA operations. More alarmingly, the paper asserts that no elevated privilege or deep access is required to launch such an attack just a foothold in the network is enough. This vector represents a surgical form of DoS that could cripple remote diagnostics, OTA updates, and routine ECU reprogramming workflows.

Table 3.1: Observed DoS Attack Patterns

Attack Type	**Description**	**Source(s)**
VLAN-based Flooding	Overwrites ECU streams via high-priority traffic	[1] Jeong et al.
Bandwidth Saturation	Consumes bandwidth to prevent legitimate traffic	[2] Luo et al.
TCP RST Injection	Interrupts firmware flashing or diagnostics via TCP session resets	[4] Talic et al.
STP Flooding	Protocol abuse to paralyze switch topology	[3] Lang Thesis
Diagnostic Port Exploitation	Entry point for distributed DoS across infotainment/control domains	[5] Spirent WP

3.2 Replay Attack

3.2.1 Overview

Replay attacks in AE are among the most insidious forms of network-based attacks, exploiting the deterministic and time-sensitive nature of vehicular communication systems. By capturing valid data packets and retransmitting them at a later time, attackers can disrupt

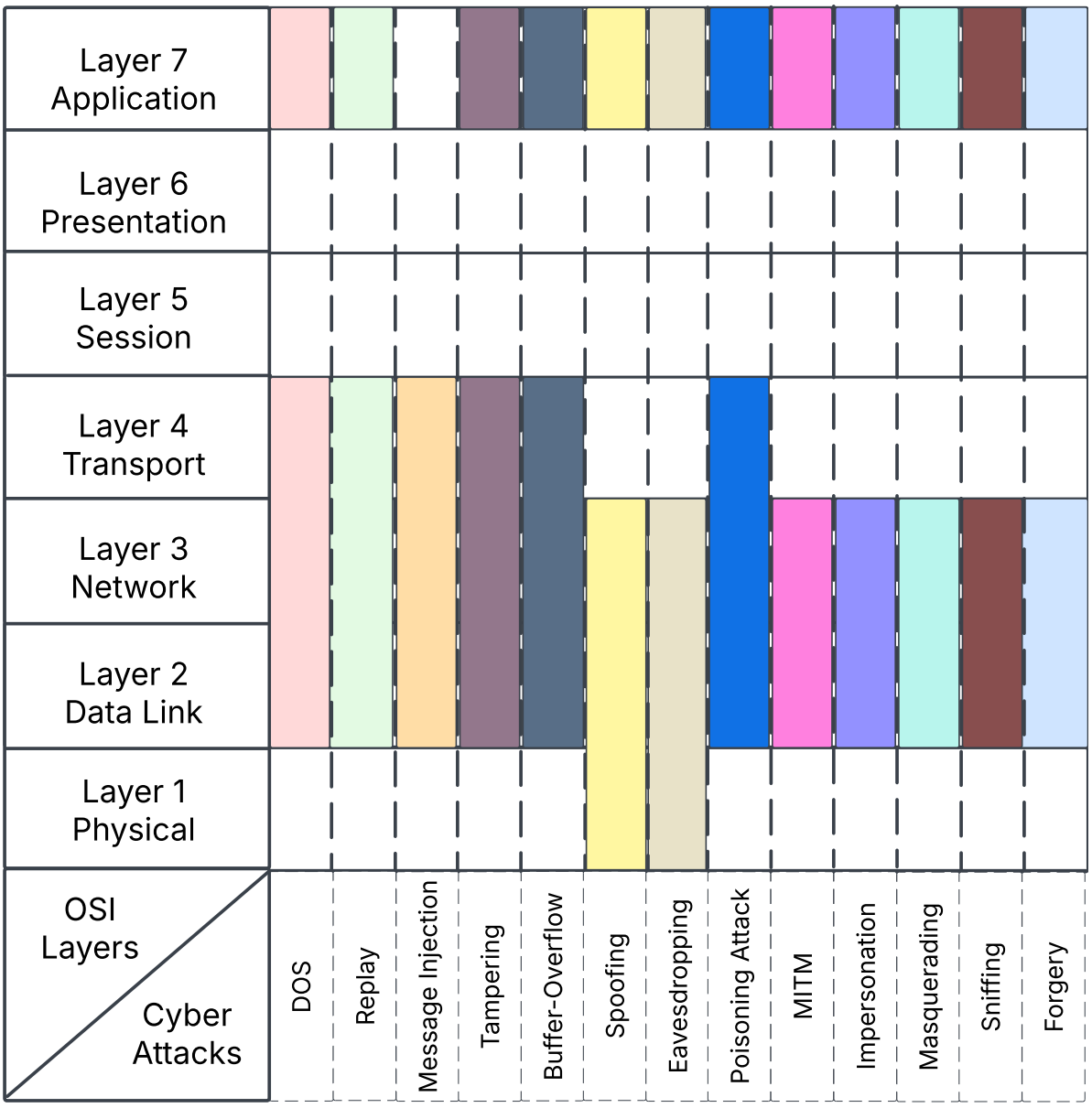

Figure 3.1: Cyber Attacks on AE Projected on OSI Layers

critical in-vehicle operations without altering message content, making detection exceptionally difficult. The replayed data appears legitimate in syntax and semantics, but its timing and context render it dangerous, potentially leading to hazardous vehicle behavior.

3.2.2 Attack Characteristics

Replay attacks target the data freshness and temporal integrity of Ethernet communications within the vehicle. In a typical scenario, an attacker captures a previously transmitted

message between two ECU and resends it to manipulate system behavior. Since most ECU act on received commands immediately without verifying freshness, a replayed message can cause the system to repeat outdated or malicious operations, posing safety-critical risks.

The following distinct characteristics make replay attacks alarming:

- **No modification of content:** Unlike injection or spoofing attacks, the replayed message is syntactically valid.
- **Difficult to distinguish from legitimate traffic:** The re-injected packets conform to protocol specifications and often pass basic validation checks.
- **Highly effective against deterministic protocols:** AE time-triggered and synchronized behavior is vulnerable to minor disruptions in the message schedule.

3.2.3 How It Works

In AE, replay attacks exploit the deterministic communication structure of IVNs, where messages are exchanged periodically between ECU using tightly scheduled protocols like TTEthernet, SOME/IP, or DoIP. These protocols often lack inherent protection against packet freshness violations, making them vulnerable to time-shifted message injection.

A replay attack typically proceeds in two stages:

- **Eavesdropping Phase:** The attacker monitors and captures valid packets transmitted between ECU. This can occur via compromised ECU, rogue diagnostic tools, or physical access to exposed interfaces (e.g., debug ports, test harnesses).
- **Re-injection Phase:** At a chosen later time, the attacker resend the captured packet without any modification. If the network or receiver lacks replay protection (e.g., sequence counters, time-based authentication), the message is accepted as valid.

This attack affects multiple OSI layers (see figure 3.1):

- **Data Link Layer (Layer 2):** If MAC-layer filtering is not enforced, rogue ECU can spoof legitimate device MAC and inject traffic.
- **Transport Layer (Layer 4):** Protocols like TCP, used in diagnostics or infotainment, may re accept stale connections if timestamps or nonces are absent.
- **Application Layer (Layer 7):** Replaying SOME/IP service discovery messages or DoIP control commands can re-trigger services or misconfigure system states.

- In real-time systems like TTEthernet, attackers can further synchronize with the global time source and precisely replay old messages within the valid window. This makes the replayed packet indistinguishable from a fresh one, allowing attackers to reactivate past operations, such as braking, without any change in payload or structure.

AE systems are especially vulnerable due to:

- Predictable traffic schedules that make target message timing easy to identify.
- Lack of temporal validation in legacy and low-level ECU.
- Multiple entry points such as diagnostic ports, infotainment bridges, and external communication gateways (e.g., V2X, cellular).

3.2.4 Known Attack Methods

Replay attacks in AE can manifest in multiple forms, depending on the source, timing, and delivery mechanism of the replayed messages. Below are notable variants observed across research studies and experimental demonstrations:

- **Simple Packet Replay:** The attacker captures a valid Ethernet frame and retransmits it at a later time without modifying any fields. This form is highly effective when ECU do not validate timestamps or message freshness. It often targets cyclic control signals (e.g., braking or steering) for repeated actuation.
- **Timestamp-Aligned Replay:** In TTEthernet-based systems, attackers with access to global time synchronization can replay packets precisely within valid transmission windows, making the replayed message appear temporally correct. This method allows the spoofed command to pass as legitimate in time-triggered environments.
- **Burst Replay Injection:** Instead of a single message, an attacker injects a burst of previously recorded packets. This form overwhelms the receiver or causes inconsistent behavior due to redundant or conflicting data being accepted over time.
- **Physical Replay via Rogue ECU:** A compromised or unauthorized ECU (e.g., test port, diagnostic gateway, or spoofed MAC address) is used to transmit captured frames, mimicking legitimate hardware sources and bypassing port-based access control mechanisms.
- **Contextual Replay Across Driving States:** An advanced form where messages from a different vehicle state (e.g., Parked) are replayed while the car is in another

state (e.g., Driving), causing state conflicts or hazardous operations (e.g., unintentional actuation of parking brake).

3.2.5 Related Works

Grimm et al. [24] conducted a detailed investigation into anomaly detection in IVN Ethernet communications using unsupervised learning techniques. Their study introduced a machine learning framework based on PCA and OCSVM to identify irregularities in time-series packet data. In one of their test cases, the authors simulated a replay attack by injecting previously captured Ethernet packets back into a live stream. The analysis revealed that replayed messages, despite having valid syntax and structure, exhibited detectable anomalies in timing distribution and statistical flow properties. The ML model flagged these messages as outliers due to their temporal misalignment. This study underscores the subtle yet disruptive nature of replay attacks: they do not violate protocol-level rules but can introduce sufficient variance to break timing expectations in synchronized automotive systems.

Xu et al. [98] In this experimental study, a controlled testbed was developed where ECU9 acted as a malicious node capable of eavesdropping and retransmitting previously captured control messages originally intended for ECU1. The replayed packets were accepted at the receiving end, but due to inconsistencies in serial number counters and MAC checks, the ECU experienced failure in proper state transitions. The consequence was not just message rejection but system-level misinterpretation of packet flows, leading to abnormal actuator states and communication breakdowns. The authors emphasize that such replay-based anomalies can easily mimic diagnostic failures or ECU malfunctions, which complicates root-cause analysis in real-world vehicular environments.

Zhang et al. [104] In this multi-vehicle evaluation, Zhang et al. executed replay attacks on both CAN and Ethernet-based systems across three commercial vehicles: Honda Accord, Kia Soul, and Nissan Leaf. They recorded real-time sensor data such as vehicle speed, acceleration, and brake status, and subsequently injected the same frames under different vehicle conditions. The study observed a range of consequences from false warning light activations to unintended actuation of steering/braking systems all without altering the packet payload. Their experiment confirmed that modern ECU largely lack mechanisms to validate message context or freshness, making them susceptible to cross-state replays, where messages captured in one operating state are re-injected in another, causing critical logic errors.

Kiravuo et al. [56] Kiravuo and colleagues presented a comprehensive taxonomy of IVN Ethernet threats and emphasized replay attacks as a top-tier concern for both functional safety and data integrity. Their research highlighted that AE systems, especially those utilizing deterministic protocols (e.g., SOME/IP and TSN), are highly sensitive to timing disruptions. In such systems, even a single out-of-sequence or outdated packet can trigger unintended system behavior. The paper specifically calls out replay attacks for their ability to bypass payload-based IDS, stressing the need for temporal validation in AE traffic. This assessment positions replay as a uniquely potent threat in the era of connected, autonomous vehicles where safety is tightly coupled with timely and context-aware communication.

3.2.6 Observed Replay Attack Patterns

Attack Type	**Description**	**Source(s)**
Simple Replay	Direct retransmission of recorded packets without any modifications	[1], [2]
Timestamp-Aligned Replay	Replay during original time slot to bypass timing filters (especially in TTE)	[3]
Burst Replay Injection	Injection of multiple stale packets to overwhelm ECU	[2]
Cross-State Message Replay	Replay of context-inappropriate messages (e.g., parking data while in drive mode)	[4]
Rogue ECU Replay	Malicious ECU mimics authorized one to send old messages	[2], [5]

Table 3.2: Variants of Replay Attacks and Their Sources

3.3 MitM Attack

3.3.1 Overview

MitM attacks represent one of the most deceptive and impactful threats in networked systems, including AE. Unlike external disruptions, MitM attacks silently intercept and potentially alter communications between two legitimate entities without either party being aware. In the context of IVNs, such attacks can compromise safety-critical data exchanges, making them especially alarming.

What makes MitM particularly dangerous is it invisibility, the attacker remains hidden from both communicating parties, all while passively listening or actively manipulating the

message stream. Unlike more overt attacks like flooding or jamming, MitM attacks are stealthy and targeted, which makes them harder to detect and often more damaging.

3.3.2 How It Works

In AE systems, messages are exchanged between ECU through a deterministic and time-sensitive Ethernet backbone. MitM attacks exploit this structure by compromising an intermediary device or port, allowing the attacker to place themselves between two ECU.

Once inside the communication path, attackers can:

- **Eavesdrop**: Record messages without modifying them, gathering sensitive or diagnostic information.
- **Tamper**: Change the message content (e.g., brake pressure value), causing misleading signals to be delivered.
- **Impersonate**: Send messages under the identity of a trusted ECU by spoofing IP or MAC addresses.
- **Replay or Delay**: Resend or delay valid messages to trigger unexpected behavior in real-time systems.

These operations affect multiple OSI layers (see figure 3.1):

- **Layer 2**: By using ARP spoofing to redirect traffic through the attacker.
- **Layer 3**: By manipulating IP addresses and routing logic.
- **Layer 7**: By attacking higher-level protocols like SOME/IP or DoIP which lack built-in security features.

Given AE tight timing constraints especially in domains like ADAS even a millisecond delay or a subtle modification can violate timing guarantees, causing misbehavior or data misinterpretation in safety-critical modules.

3.3.3 Known Attack Methods

Research and experiments in AE environments have revealed several prominent MitM attack variants:

- **Service Impersonation Attack**: The attacker poses as a valid SOME/IP service provider. ECU that request data from this service receive false or malicious responses.

- **Session Hijacking**: Involves capturing an active communication session and taking control midstream redirecting commands or injecting malicious ones under an existing session ID.
- **Event Forwarding Attack**: In a publish/subscribe system, the attacker modifies a published message before forwarding it to subscribers, altering system behavior while maintaining message format.
- **Copycat and De-Association Attacks**: These aim to sever legitimate sessions and trick ECU into establishing new ones with the attacker, giving them full control over message flow.
- **ARP Spoofing**: Exploits ARP to reroute Ethernet traffic through the attacker node by sending fake MAC responses, allowing interception without physical rewiring.

3.3.4 Related Works

Zelle et al. [101] and colleagues demonstrated a potent MitM attack targeting the SOME/IP service discovery protocol. Using vsomeip, an open-source implementation of SOME/IP, in combination with CANoe, they successfully simulated and deployed rogue services on an AE network. The attack worked by impersonating a legitimate ECU, intercepting service advertisements or requests, and injecting crafted response messages. Notably, the system lacked service-level authentication, allowing attackers to respond faster than the actual ECU, thus hijacking the communication session without raising any alarms. This study underscored the vulnerabilities in service discovery mechanisms in SOME/IP implementations, particularly in the absence of cryptographic authentication or digital signature checks.

Lastinec et al. [58] This research investigated the use of ARP spoofing as a vector for MitM attacks in IVN Ethernet networks. The authors recreated a typical AE topology and performed ARP poisoning by injecting falsified MAC-IP mappings into the ARP caches of ECU. Despite MAC filtering being enabled in some scenarios, the attacker could still reroute communication through a compromised ECU or gateway. Importantly, the latency introduced was negligible, making the attack highly feasible for real-time, low-latency systems like powertrain or chassis ECU. The study emphasizes that ARP-based address resolution remains a weak link in zonal Ethernet architectures and that network-layer authentication mechanisms (like static ARP tables or cryptographic address binding) are essential for resilience.

Ma et al. [71] While primarily focused on designing a secure SOME/IP communication protocol, this paper by Ma et al. also presents a proof-of-concept MitM scenario to evaluate the robustness of traditional message verification techniques. Their tests revealed that tampered SOME/IP payloads could bypass legacy error detection systems if the payload structure remained syntactically valid. This finding is critical because it demonstrates that message integrity can be silently breached without triggering basic validation checks. The paper further presents an AEAD-based solution that modifies the SOME/IP payload structure to add confidentiality and integrity protection without altering the core SOME/IP frame format thus facilitating backward compatibility while enhancing security.

3.3.5 Security Goals Violated

MitM attacks compromise all four essential security principles in AE:

- **Confidentiality**: By capturing messages, sensitive data such as sensor readings can be leaked.
- **Integrity**: The attacker can modify message contents, misleading safety-critical systems.
- **Authenticity**: Impersonated ECU can trick the system into accepting malicious commands.
- **Availability**: The attacker may delay or drop packets, disrupting communication.

3.3.6 Impact on AE Network

The consequences of MitM attacks include:

- Corruption of ADAS data leading to incorrect braking or steering.
- Misinformation in SOME/IP service-oriented architectures.
- Tampering of actuator/sensor instructions.
- Interference with diagnostics or OTA update processes.

3.4 Message Injection Attack

3.4.1 Overview

Message Injection Attacks are deliberate attempts by adversaries to send falsified or unauthorized messages into the in-vehicle communication network to manipulate ECU behavior. Unlike replay attacks that resend previously captured messages, injection attacks create entirely new messages to spoof, mislead, or disable functionality.

In AE systems, where high-bandwidth and real-time messaging is used for safety-critical applications (e.g., braking, steering, ADAS), injection of a malicious packet even once can cause unintended or unsafe vehicle actions. Because AE networks often lack payload-level authentication or validation, crafted messages that match expected formats may bypass standard checks.

3.4.2 How It Works in AE

In an AE environment, attackers first identify target services and message IDs through traffic sniffing (via diagnostic ports, compromised ECU, or exposed telematics units). Once the relevant traffic pattern or protocol (like SOME/IP, DoIP, or TSN) is understood, attackers craft new packets with fabricated payloads and inject them directly into the Ethernet backbone.

This attack can occur at multiple OSI layers (see figure 3.1):

- **Layer 2 (Data Link):** The attacker may spoof MAC addresses to appear as a trusted ECU.
- **Layer 3/4 (Network/Transport):** Protocols such as DoIP over TCP/IP can be exploited to send unauthorized requests.
- **Layer 7 (Application):** Malicious SOME/IP packets with crafted payloads or fake service responses may be injected to alter ECU behavior.

Critical system components like the braking ECU, steering control, or powertrain can be manipulated by injecting high-priority or high-trust commands especially if the receiving modules do not validate source identity or payload integrity.

3.4.3 Known Attack Methods

Common variants of message injection attacks in AE systems include:

- **Fabricated Command Injection:** Attacker sends forged commands (e.g., apply brakes) using known message ID and structure, causing ECU to act on non-legitimate triggers.
- **SOME/IP Method Spoofing:** Attacker mimics a valid SOME/IP method invocation (e.g., changing driving modes or system states) to alter service-level interactions.
- **Diagnostic Message Injection:** Attacker crafts UDS/DoIP messages to enter ECU into diagnostic mode or change operational parameters (e.g., firmware update triggers).
- **Flood-Based Injection:** Continuous injection of spoofed messages (e.g., service discovery floods or control commands) to overwhelm ECU and cause DoS-like effects.
- **Multicast Injection:** In service-oriented AE systems (e.g., SOME/IP-SD), the attacker injects multicast discovery or advertisement packets to impersonate services or redirect traffic.

3.4.4 Related Works

Ma et al. [71] explored vulnerabilities in the message structure of SOME/IP used in AE. They revealed that a lack of granular field validation within SOME/IP services allows for the successful injection of malicious payloads. By replicating valid service IDs and method IDs, attackers managed to craft syntactically valid yet semantically harmful messages. Their experimental setup showed that ECU processed these injected packets as if they were authentic service responses or commands. The weakness lay in the absence of encryption or per-field validation, which made the payloads indistinguishable from genuine traffic. This highlights the protocol susceptibility to message injection that exploits weak semantic checks rather than overt formatting errors.

Khatri et al. [54] performed a comprehensive analysis of injection-based threats, including those applicable to AE. A case study within their work detailed how attackers could exploit the OBD-II port to inject falsified Ethernet frames. These frames were crafted to mimic legitimate diagnostic and control messages, enabling attackers to remotely trigger actions such as activating dashboard warning lights or modifying brake behavior. Their findings showed that legacy packet forwarding mechanisms, particularly those lacking source authentication and rate-limiting, allowed for the acceptance and propagation of such spoofed packets across multiple ECU. This exposure is exacerbated by the AE architecture tendency to trust messages as long as basic structural integrity is preserved.

Gmiden et al. [23] and team created a testbed environment simulating a typical AE topology using real-time hardware components and a packet injection interface. Their findings confirmed that unprotected SOME/IP and DoIP messages could be injected into the system without detection, especially when transmitted with correct session formats. These injections triggered severe disruptions including spontaneous system reboots and ECU malfunctions. Notably, their study emphasized that many in-vehicle services fail to authenticate session-based messages after the handshake phase, allowing persistent injection throughout the connection lifetime. This points to a systemic issue where trust is established too early and not re-evaluated during session continuity.

Park et al. [80] investigated how adversaries could use DoIP to inject messages that mimic legitimate requests. They demonstrated that a malicious client could initiate DoIP sessions using correct handshake sequences and then send unauthorized commands that brought ECU into unsafe states such as bootloader mode activation. Despite DoIP being layered on top of TCP/IP (which ensures delivery and order), the absence of application-layer authentication meant ECU did not differentiate between trusted and untrusted diagnostic commands. The authors stressed that DoIP' trust model is overly reliant on IP/TCP transport correctness and overlooks deeper protocol integrity validation.

3.4.5 Security Goals Violated

Message Injection Attacks disrupt key AE security objectives:

- **Integrity:** Crafted messages alter ECU state or inject false sensor data.
- **Authenticity:** Attackers pose as trusted services without verification.
- **Availability:** Flood or burst injections overload communication channels, interrupting real-time control.
- **Safety:** Misleading control messages affect safety-critical systems like brakes, acceleration, or ADAS.

3.4.6 Impact on AE Network

The consequences of successful message injection include:

- **Sensor/Actuator Manipulation:** Sending false throttle/brake commands can endanger passengers.

- **Diagnostic Interference:** Injection of UDS/DoIP commands may halt ECU function or trigger resets.
- **Infotainment Hijacking:** Multimedia or UI misbehavior through crafted SOME/IP messages.
- **Unintentional Mode Switching:** Vehicles may be forced into incorrect operating states (e.g., maintenance mode during driving).

3.5 Spoofing Attack

3.5.1 Overview

Spoofing attacks are a form of identity deception in which an attacker masquerades as a legitimate entity on the network to gain unauthorized access or manipulate communication. In the context of AE, spoofing can involve faking MAC addresses, IP addresses, or protocol-level identifiers to trick ECU into trusting malicious data or sources.

This type of attack is particularly dangerous in AE systems because they often rely on assumptions of trust at the lower layers of the OSI model, especially when using legacy diagnostics protocols like DoIP or SOME/IP. If left unchecked, spoofing attacks can lead to manipulated control signals, unauthorized diagnostics, and severe safety risks.

3.5.2 How It Works in AE

In an AE setup, spoofing typically targets these key layers:

- **Layer 2**: MAC spoofing is used to impersonate an authorized ECU.
- **Layer 3**: IP spoofing tricks ECU or gateways by using forged IP packets that appear to come from valid sources.
- **Layer 7**: Spoofed SOME/IP or DoIP messages may claim to originate from trusted services, exploiting weak authentication.

A spoofing attack often starts with the attacker gaining access via diagnostic ports, telematic units, or compromised ECU. The attacker listens to legitimate traffic to identify critical headers and identifiers, then crafts packets that imitate authorized messages, services, or nodes.

In AE systems with diagnostic capabilities, an attacker can spoof diagnostic messages to access or modify operational parameters. For instance, mimicking a DoIP routing activation message may allow unauthorized communication with in-vehicle ECU.

3.5.3 Known Attack Methods

Key spoofing techniques observed in AE include:

- **MAC/IP Spoofing**: Imitating the hardware or IP address of a legitimate ECU to send malicious commands or disrupt routing.
- **DoIP Routing Activation Spoofing**: Faking the initial handshake and routing activation sequence to gain access to internal ECU, exploiting lack of identity verification.
- **SOME/IP Identity Spoofing**: The attacker mimics the instance ID and service ID of a valid SOME/IP service to trick subscribers into accepting malicious payloads.
- **Session Takeover**: A variant of spoofing where the attacker uses previously observed session identifiers or authentication tokens to inject data mid-session.
- **Broadcast Spoofing**: The attacker sends fake service discovery or status messages to flood or mislead the network, potentially altering multicast behavior or node associations.

3.5.4 Related Works

Ajin et al. [1] conducted a detailed analysis of DoIP and identified multiple vulnerabilities, especially during the routing activation phase. They highlighted that DoIP lacks robust authentication and integrity verification mechanisms. Specifically, attackers can forge routing activation requests and spoof legitimate test equipment. Once this spoofed message is accepted by the gateway, it enables unauthorized entities to initiate diagnostic sessions and send UDS commands to ECU. Because DoIP headers do not include cryptographic authentication, spoofed messages can pass undetected, opening the system to misuse.

Ma et al. [72] examined SOME/IP and demonstrated that identity spoofing is feasible through manipulation of service discovery fields. By crafting packets with the same service ID and instance ID as a legitimate ECU, a malicious node can impersonate trusted services. This allows the attacker to redirect traffic, override service behaviors, or insert unauthorized messages into the AE system. Their findings emphasize that SOME/IP lacks origin authentication, making it vulnerable to service-level impersonation.

Lindberg et al. [62] provided a structured evaluation of spoofing vulnerabilities in DoIP, focusing on alive-check and entity-status messages. He demonstrated that malicious test equipment or software could generate spoofed alive signals or fake status information to manipulate the diagnostic process. The study also found that ECU and gateways were often unable to verify the true source of incoming messages, especially in multi-device test environments. Lindberg concluded that the absence of end-to-end cryptographic protection and poorly defined source authentication requirements leaves DoIP-based diagnostic sessions open to manipulation.

Studnia et al. [90] conducted a broad survey of spoofing attacks through diagnostic channels such as the OBD-II interface, which increasingly leverage Ethernet-based protocols. Their research demonstrated that attackers can mimic the behavior of diagnostic tools and send falsified requests (e.g., to initiate firmware updates, erase memory, or disable systems). In Ethernet-based AE networks, spoofing is particularly dangerous because IP-based addressing and service-oriented architecture (e.g., DoIP and SOME/IP) allow impersonated sources to reach sensitive modules without physical access to ECU.

3.5.5 Security Goals Violated

Spoofing attacks violate several foundational security principles:

- **Authenticity**: Attackers pretend to be valid ECU or test tools, deceiving the receiving entity.
- **Integrity**: Malicious actors can alter or fabricate messages that are accepted as valid.
- **Confidentiality**: Spoofed responses can be crafted to leak diagnostic or operational data.
- **Availability**: Broadcast or diagnostic spoofing may disrupt legitimate operations or induce DoS conditions.

3.5.6 Impact on AE Network

Spoofing attacks can have wide-ranging effects across the system:

- Unauthorized Diagnostics: Attackers can access fault codes, initiate firmware updates, or change calibration values.

- Safety-Critical Manipulation: Mimicking safety-related control messages (e.g., torque control or steering angle) may lead to unsafe vehicle behavior.
- Network Disruption: Spoofing multicast advertisements can flood the network or cause incorrect service associations.
- Privacy Invasion: Spoofed sessions may extract personal or location-based data from connected vehicles.

3.6 Eavesdropping Attack

3.6.1 Overview

Eavesdropping attacks involve unauthorized interception and monitoring of data as it travels across the network. In AE, eavesdropping is especially critical due to the sensitivity of the data exchanged ranging from vehicle diagnostics to control signals and infotainment content. Attackers exploiting this vulnerability can gain insights into the internal functioning of ECUs, reverse-engineer protocol behaviors, or harvest private user and vehicle data.

Because Ethernet-based communication often lacks encryption in standard implementations, especially in legacy or low-cost systems, it is inherently vulnerable to passive interception. Additionally, as AE systems transition towards centralized and high-bandwidth architectures (e.g., TSN backbones), the opportunity for intercepting a larger volume of high-priority data increases significantly.

3.6.2 How It Works in AE

Eavesdropping in AE networks typically involves passive monitoring of traffic at various layers:

- **Physical Layer (Layer 1)**: Using hardware tap devices or compromised physical access (e.g., through OBD-II ports).
- **Data Link Layer (Layer 2)**: Monitoring Ethernet frames, especially when MAC address filtering is not enforced.
- **Network Layer (Layer 3)**: Intercepting IP packets to reconstruct sessions or monitor service discovery.
- **Application Layer (Layer 7)**: Decoding SOME/IP or DoIP payloads to access diagnostic requests, firmware updates, or control messages.

Attackers may use passive sniffing tools or install rogue devices (e.g., compromised telematics units) to silently log traffic. Because AE traffic often lacks payload-level encryption, an attacker can analyze patterns, extract sensitive data, or prepare for further active attacks like spoofing or injection.

3.6.3 Known Attack Methods

Some key methods of eavesdropping in Automotive Ethernet include:

- **Sniffing via Diagnostic Ports**: Using OBD-II or external dongles to capture raw traffic streams within the vehicle.
- **Remote Telemetry Exploitation**: Compromising telematics or V2X units to collect data from the central TSN backbone.
- **Switch Mirroring Exploitation**: If port mirroring is enabled or misconfigured, attackers can clone and observe real-time traffic.
- **Timing and Signature Analysis**: Long-term monitoring of traffic to extract ECU-specific behaviors or cryptographic material (side-channel-like data).
- **Passive V2X Listening**: Capturing vehicle-to-everything broadcasts or cloud communication links if not end-to-end encrypted.

3.6.4 Related Works

Luo et al. [69] conducted an in-depth security analysis of TSN-based backbone architectures using the STRIDE threat modeling framework. Eavesdropping was identified as a critical passive threat, especially in architectures lacking encryption and stream isolation mechanisms. Their work revealed that TSN streams, particularly those related to sensor data in ADAS and control functions, often transmit unencrypted payloads. An adversary with access to a tap point (e.g., compromised ECU, telematics unit, or physical port) could passively collect sensitive data without modifying traffic, enabling long-term behavioral analysis and reverse engineering. The authors also emphasized that by analyzing repetitive traffic patterns and sensor fusion timestamps, attackers could infer control strategies, deduce sensor roles, and compromise future safety decisions made by central controllers.

Luo et al. [69] further referenced scenarios where internal TSN backbones are bridged with external V2X communication systems, such as cellular or DSRC-based telematics. These setups introduce extended attack surfaces, allowing roadside units, mobile adversaries,

or rogue base stations to intercept gateway-translated Ethernet traffic. In such hybrid architectures, attackers can exploit the broadcast nature of V2X to harvest inter-vehicle or intra-vehicle communications, especially if the translation gateways fail to enforce encryption, authentication, or stream isolation. This finding underlines the necessity of boundary defense at Ethernet-V2X bridges.

Luo et al. [67] Through simulation experiments using OMNeT++, Luo et al. demonstrated that when MACsec and per-stream filtering were disabled or misconfigured, TSN backbones became highly susceptible to silent traffic monitoring. In scenarios replicating ADAS zones, attackers were shown to reconstruct data from front cameras, radar, and control commands by passively observing unprotected flows. Their work showed that stream-level eavesdropping can expose time-critical control packets (e.g., emergency braking) and payload structures (e.g., serialized object detections). The study confirmed that eavesdropping does not affect the real-time performance of the network, making detection extremely challenging unless specific auditing counters or anomaly detection logic is enabled.

3.6.5 Security Goals Violated

Eavesdropping attacks primarily undermine:

- **Confidentiality**: Unauthorized access to diagnostic, control, or sensor data.
- **Privacy**: Potential leakage of user behaviors, vehicle location, or personal identifiers.
- **Integrity (Indirectly)**: Collected data may be used in follow-up spoofing or injection attacks.

3.6.6 Impact on AE Network

Eavesdropping can have wide-ranging implications:

- **Precursor to Spoofing or Injection**: Intercepted traffic helps attackers craft valid-looking malicious packets.
- **Privacy Violations**: Sensitive personal or vehicle telemetry may be exposed to third parties.
- **Protocol Reverse Engineering**: Attackers may use passive traffic to understand proprietary implementations or discover hidden diagnostic services.

- **Fleet-Wide Surveillance**: If one vehicle is compromised, attackers may gather intelligence useful across an entire fleet.

3.7 Sniffing Attack

3.7.1 Overview

Sniffing attacks refer to the unauthorized capturing and monitoring of network traffic for analysis, typically carried out using packet sniffers or compromised nodes. In the context of AE, sniffing enables attackers to passively observe communication between ECU, potentially revealing confidential diagnostic data, control commands, sensor streams, and communication patterns. Since AE systems often operate on broadcast domains (e.g., for SOME/IP service discovery) and use unencrypted communication, they become inherently susceptible to these passive threats.

The impact of sniffing is often underestimated because it does not involve direct interference. However, it provides adversaries with critical insights into vehicle architecture, protocol behaviors, and message formats, which can later be used for active intrusions such as spoofing, injection, or replay attacks.

3.7.2 How It Works in AE

Sniffing in AE occurs primarily at the data link (Layer 2) and application (Layer 7) layers:

- **Layer 2 (Ethernet Frame Monitoring):** Sniffers collect all traffic from a broadcast-enabled switch or via mirroring ports.
- **Layer 3 (IP Packet Logging):** IP headers are analyzed to trace ECU communication or reverse network topologies.
- **Layer 7 (Payload Decoding):** SOME/IP or DoIP message contents, including diagnostic requests, responses, or firmware update signals, are inspected in cleartext.

The attacker typically installs a rogue device in the network (via test port, compromised telematics unit, or Ethernet tap) or compromises an existing ECU. Once inside, the attacker can monitor periodic messages, detect traffic patterns, and decode service-level information without modifying any packets, thereby avoiding detection.

3.7.3 Known Attack Methods

- **Broadcast Domain Monitoring:** Observing service discovery protocols (like SOME/IP-SD) that rely on multicast or broadcast behavior.
- **Switch Port Mirroring Exploitation:** Leveraging network misconfigurations to access mirrored copies of all switch traffic.
- **Compromised Node Logging:** Installing sniffing tools in a compromised ECU or telematics unit to record internal traffic.
- **OBD-II Tap Devices:** Using diagnostic port dongles to monitor network exchanges in real time, especially during servicing.
- **TSN Stream Logging:** Recording time-triggered streams without authorization, especially in systems lacking MACsec or stream encryption.

3.7.4 Related Works

Reusch et al. [85] The authors recognize sniffing as a passive but potent threat in TSN-based AE networks. Their attacker model assumes adversaries with access to the network schedule and frame content. By leveraging knowledge of GCLs, a sniffer can synchronize its packet capture to sensitive message intervals (e.g., sensor fusion data). Although their paper focuses on constraint-programmed TESLA-based protection, they highlight the inherent exposure of TT traffic in unsecured TSN systems as a major weakness against sniffing.

Zhao et al. [106] Zhao attacker model includes passive sniffing capabilities where an adversary observes deterministic communication to identify patterns or extract cryptographic material. They note that broadcast and multicast messages in TTE-like setups lack source verification and are prime targets for sniffers, especially when message authentication is not enforced by default.

3.7.5 Security Goals Violated

- **Confidentiality:** Unauthorized access to internal communication.
- **Privacy:** Exposure of sensitive user or vehicle data.
- **Integrity (Indirect Risk):** Collected traffic can be re-used in injection or replay attacks.

3.7.6 Impact on AE Network

- **Reconnaissance for Later Attacks:** Enables crafting of valid spoofed or forged messages.
- **Exposure of Diagnostics and Control Traffic:** Reveals message IDs, payload structures, and periodic behavior.
- **Surveillance of Sensor Streams:** TSN or AVB streams may be monitored to infer vehicle state or ADAS operations.
- **Potential Key Leakage:** In systems without secure key handling, sniffers might capture cryptographic material during key exchange.

3.8 Tampering Attack

3.8.1 Overview

A tampering attack refers to unauthorized alterations of data, signals, or messages in AE. The goal is to inject, modify, or manipulate communication between ECU to compromise functionality or security. These attacks are critical since tampering can lead to incorrect vehicle behavior or security breaches, potentially endangering safety-critical systems like brakes, steering, or engine control.

In AE, tampering can occur at various levels, such as message manipulation in SOME/IP, or directly modifying CAN and Ethernet frames. This may involve spoofing legitimate ECU addresses or manipulating data packets transmitted across the network.

3.8.2 How It Works in AE

Tampering attacks in AE typically involve intercepting or injecting malicious data packets to alter content or control flow within ECU:

- **Spoofing and Message Injection**: Inserting fraudulent messages while posing as an authentic ECU, disrupting operation or gaining unauthorized services.
- **Manipulating Message Content**: Modifying data within valid messages (e.g., sensor readings or control commands) to trigger faulty operations.
- **Targeting Protocols**: SOME/IP or DoIP may lack robust integrity checks; attackers can modify headers or payloads to alter service requests or responses.

3.8.3 Known Attack Methods

- **Frame Tampering**: Direct modification of packets during transmission (control messages or sensor data).
- **Injection of Malformed Messages**: Protocol-conformant but misleading data to force incorrect states.
- **Service Interception/Manipulation**: Intercepting and altering SOME/IP service calls and responses.
- **Replay Attacks**: Replaying captured messages to cause unintended behaviors or unauthorized access.

3.8.4 Related Works

Zuo et al. [107] In work on CANFD, SOME/IP gateway security, the authors discuss vulnerability to message manipulation and stress securing gateways and SOME/IP.

Seifert et al. [86] Presents a security gateway for automotive networks, showing how tampering on CAN and FlexRay can be mitigated with secure mechanisms at network transitions.

3.8.5 Security Goals Violated

- **Integrity**: Direct violation by altering messages or service requests.
- **Authenticity**: Impersonating ECU/services to execute unauthorized operations.
- **Availability**: Disruption via modified control messages causing malfunctions/failures.
- **Confidentiality (Indirect)**: Altered data may expose sensitive information.

3.8.6 Impact on AE Network

- **System Behavior Disruption**: Wrong actions affecting powertrain control or ADAS.
- **Compromise of Safety Functions**: Tampering with ABS/airbags jeopardizes safety.
- **Loss of Control/Malfunctions**: Altered messages can render ECU unresponsive.
- **V2V Exploitation**: Tampering may compromise inter-vehicle communications.

The growing awareness of these cyber threats underscores the critical need for deeper investigation into the vulnerabilities of Automotive Ethernet systems. While Chapter 2 provided a detailed examination of attack vectors capable of compromising AE networks, it is equally important to situate these threats within the broader body of existing research. Chapter 3 therefore turns to related works, with a particular emphasis on MitM attacks. By analyzing prior studies, methodologies, and findings, this chapter highlights the strengths and limitations of current approaches while exposing critical research gaps. This review not only clarifies why MitM attacks warrant focused attention in the context of AE but also establishes the foundation for the experimental analysis presented in subsequent chapters.

CHAPTER 4

Focused Investigation on MitM Attacks in AE

Building upon the taxonomy of cyber threats introduced in Chapter 2, this chapter presents an in-depth investigation into one of the most critical and versatile attack types in the domain of IVNs: the **MitM attack**. As automotive systems increasingly adopt AE for handling high-speed, time-sensitive, and safety-critical operations, the attack surfaces have expanded both in breadth and depth.

This chapter specifically focuses on **MitM attacks within AE** due to their multifaceted impact, operational stealth, and the foundational role they play in enabling other complex attack scenarios such as spoofing, replay, data tampering, and DoS. Unlike attacks that are more isolated or targeted toward specific layers, MitM enables *active manipulation and passive eavesdropping* on communication flows across multiple layers in the OSI model, making it particularly dangerous in zonal and service-oriented automotive architectures.

To frame this study, the chapter is organized into three sections. Section 4.1 explains the motivation and rationale behind choosing the MitM attack as the core focus of this thesis. Section 4.2 elaborates on the **severity** of MitM attacks in AE, analyzing their potential to compromise safety, timing, and functional behavior in modern vehicle systems. Section 4.3 discusses the **accessibility** of this attack how easily it can be carried out in real-world vehicle networks given their physical and architectural characteristics.

4.1 Rationale for Choosing the MitM Attack

In the realm of automotive cybersecurity [83, 99], the decision to focus on the MitM attack is grounded in both practical and strategic considerations. First and foremost, the MitM attack is not limited to one specific protocol or service; rather, it enables intrusion across several layers of the in-vehicle communication stack, from MAC-level packet redirection to application-level payload modification. This makes it a gateway attack capable of facilitating

a range of malicious activities, such as DoS, data spoofing, packet injection, and command hijacking.

Second, MitM attacks align well with realistic adversarial capabilities in AE systems. Attackers do not always need complex hardware or privileged software access; physical access to the AE backbone, which is often shared across multiple ECU via Ethernet switches, is sufficient for launching transparent MitM attacks. Using off-the-shelf hardware like Raspberry Pi and devices such as RAD-Galaxy, and Rad-Jupiter an attacker can intercept and manipulate real-time traffic without triggering standard diagnostics or fault detection.

Third, the experimental feasibility of MitM attacks makes them an ideal candidate for hardware-based implementation and validation, which is the focus of later chapters. By setting up a MitM device between two ECU in a testbed environment, researchers can evaluate IDS, analyze timing deviations, simulate replay attacks, and study the response of safety-critical modules such as ADAS and gateways.

Ultimately, selecting MitM as the central attack type allows this thesis to explore a broad attack spectrum under a unified threat model, while providing realistic and reproducible experimentation avenues using modern AE hardware platforms.

4.2 Severity of the MitM Attack in AE

The severity of a MitM attack in AE is underscored by its ability to violate multiple core security principles CIAA within a single compromise. Once an attacker inserts themselves into the communication path, they gain full control over the transmitted data without the sender or receiver being aware. This silent and real-time manipulation becomes particularly hazardous in time-sensitive and safety-critical systems such as adaptive cruise control, emergency braking, or lane-keeping assistance.

For instance, by tampering with SOME/IP [61, 28, 102] service discovery responses, an attacker can impersonate a valid ECU and reroute sensor data or control messages. In TSN-enabled networks, delayed or modified time-synchronized traffic can disrupt clock domains, leading to phase mismatches and control logic breakdowns. Furthermore, diagnostic sessions via DoIP [96, 73, 19] can be hijacked, allowing malicious commands to be injected directly into the vehicle's functional domain without authorization.

Several research studies have simulated these scenarios and found that even small changes in timing or payload through a MitM attacker can cause significant anomalies in vehicle behavior, ranging from warning light triggers to the complete disabling of actuation commands.

Unlike DoS attacks, which are immediately disruptive and detectable, MitM attacks often operate stealthily over extended periods, silently degrading system performance or laying the groundwork for secondary attacks like spoofing or replay.

In essence, the MitM attack poses a compound and escalating threat, especially in networks where encryption, source authentication, and stream integrity are not strictly enforced a condition commonly found in AE implementations even today.

4.3 Accessibility of the MitM Attack in AE

One of the most concerning attributes of the MitM attack is its high degree of accessibility in AE deployments. Unlike theoretical or highly resource-dependent attacks, MitM requires minimal hardware, modest technical skill, and often no software vulnerabilities to exploit[100, 59]. The nature of Ethernet switching and the lack of link-layer encryption means that any node positioned between two communicating ECU [17, 70, 12] either physically or logically can perform a MitM attack using passive or active bridging.

Many AE networks follow static topologies, where backbone links run from switch-to-ECU without encryption or flow validation. This structure enables an attacker to perform a tap using devices like RAD-Galaxy in transparent forwarding mode, effectively making the MitM device invisible to both sender and receiver. Moreover, broadcast and multicast traffic in AE, especially SOME/IP [61, 28, 102] service announcements, are inherently accessible to any connected node, making eavesdropping a trivial task.

From a practical standpoint, experimental studies and testbeds have shown that a MitM attack can be deployed using tools such as Wireshark, Scapy, and custom Python scripts, all running on inexpensive embedded devices. In fact, the same test setup used for passive monitoring can be switched into active MitM mode with a simple routing or iptables change.

Because the barriers to entry are low, and the opportunities for exploitation are wide, MitM attacks are not only plausible but likely in adversarial threat models that assume local access to the vehicle network such as during service station visits, aftermarket device installations, or even temporary ECU replacement scenarios.

4.4 Related Work I: MitM in AE Protocols

MitM attacks occupy a prominent position in the automotive cybersecurity [14, 84] landscape, particularly due to the architectural openness of AE. As the IVN transitions from

legacy protocols like CAN and LIN to high-bandwidth Ethernet, many safety-critical systems including ADAS, infotainment, diagnostics, and OTA now rely on application layer protocols such as SOME/IP [102] and DoIP [19], both of which are susceptible to packet interception, injection, and manipulation.

Lo Bello *et al.* [93] emphasize that the use of TSN [93, 81, 31] within AE, while beneficial for deterministic data transmission, introduces attack vectors that MitM exploits particularly well. TSN's periodicity and traffic predictability allow attackers to blend in malicious packets without triggering immediate disruptions. If a MitM actor tampers with time slots or injects subtle timing offsets, it can disturb synchronization mechanisms or delay control loops potentially jeopardizing braking or steering operations.

Ben Chehida Douss *et al.* [81] further outline how AE best-effort transmission design lacks authentication, encryption, and data integrity checks. These missing components are critical in thwarting MitM attacks. The authors categorize AE vulnerabilities across the CIAA axes, highlighting that MitM intrusions can exploit all four simultaneously by modifying packets in transit, replaying messages, or silently rerouting traffic.

Toghuj and Turab [8] discuss the practical feasibility of MitM attacks in zonal and domain-based architectures. They show that Ethernet's flat topology and switch-based routing make it easy for attackers to intercept communication using a passive tap or transparent bridge such as a RAD-Galaxy device. Even worse, protocols like SOME/IP [100, 59] do not validate the origin or freshness of the messages, allowing attackers to modify diagnostic messages or spoof control commands without being detected.

Furthermore, studies like those by Weiss and Steinhorst show that TSN-enabled systems, if not protected by robust stream authentication or packet integrity mechanisms, may allow graceful degradation policies to be bypassed. An attacker intercepting traffic between redundant systems could simulate faults or delays, effectively shifting failover logic and stealing control of a critical subsystem.

4.5 Related Work II: Detection and Simulation of MitM

With the proliferation of MitM threats, the research community has responded with a variety of detection mechanisms, both simulation-based and HIL. This second strand of related work focuses on how MitM attacks are simulated, detected, and mitigated through real-time and offline techniques.

Kandadi *et al.* [92] propose an AI-driven threat modeling framework that applies to heterogeneous networks including CAN, LIN, and AE. Their model identifies MitM by learning

Table 4.1: Comparative Literature on AE Protocol Vulnerabilities

Study	**Targeted Protocol**	**Identified Vulnerabilities**	**MitM Attack Method**	**Impact Severity**
Lo Bello *et al.* (2023)	TSN/Ethernet	Deterministic timing exploited	Timestamp manipulation	Desynchronization
Ben Chehida Douss *et al.* (2023)	AE/SOME/IP [61, 28, 102]	No encryption/authentication	Traffic interception	ECU impersonation
Toghuj & Turab (2023)	DoIP [96, 73, 19]	Absence of payload integrity	Payload alteration	Diagnostic misbehavior
Weiss & Steinhorst (2022)	TSN/Zonal AE	No stream verification	Packet rerouting	Failover hijacking

communication patterns and detecting anomalies in timing, message structure, or frequency. They argue that predictive models trained on normal protocol behavior can flag packet modifications, especially in SOME/IP [28, 102] and DoIP [96, 73, 19] systems where payload entropy is sensitive to manipulation.

In parallel, a study on anomaly detection for SOME/IP [28, 102] traffic [21] explored both retroactive (offline) and real-time detection modes. By capturing normal service response times and command patterns, they could detect when a MitM attacker injected slightly modified responses that altered system behavior. This approach is well-suited to AE environments where timestamp anomalies or unexpected response sequences are strong indicators of tampering.

Toghuj and Turab [8] demonstrate a practical MitM testbed where a malicious bridge alters diagnostic requests sent from one ECU to another. Their findings show that even with minimal timing delays, the attacker can manipulate service behavior (e.g., falsifying DTC codes or triggering DoIP [96, 73, 19] service restarts).

4.6 Methodologies for Implementing MitM in AE

To practically demonstrate the feasibility of MitM in AE, this thesis adopts two complementary methodologies for implementation. The first approach utilizes an active tap device, which allows transparent interception of traffic between ECU with minimal latency and high fidelity, thereby providing a controlled environment to study monitoring and manipulation of packets. The second approach employs ARP spoofing [30], a widely documented link layer technique that poisons ARP caches to redirect communication flows through the attacker, enabling both passive monitoring and active tampering of data streams. These two methods

Table 4.2: Comparison of MitM Detection Mechanisms in AE Networks

Study	**Detection Approach**	**Target Protocol**	**Detection Type**	**Key Feature Used**
Kandadi *et al.* (2023)	ML-based anomaly detection	CAN, AE	Predictive (online)	Packet timing, entropy
SOME/IP [100, 61, 28] IDS (2022)	Hybrid detection	SOME/IP [100, 59, 17]	Real-time + Retro	Message intervals
Douss *et al.* (2023)	Passive protocol monitoring	AE stack	Signature-based	Source MAC/IP
Toghuj & Turab (2023)	Simulation + behavioral filters	DoIP [96, 73, 19], AE	HIL testbed	Service response timing

together represent distinct but realistic adversarial models: one leveraging specialized hardware for precise tapping, and the other exploiting inherent protocol weaknesses in Ethernet. The following sections build upon this methodological choice by reviewing related works that have investigated MitM through active tapping and ARP spoofing [30], highlighting their implications in AE environments.

4.6.1 Related Work: Active-Tap (Inline) MitM in AE

4.6.1.1 Concept and Feasibility

Inline MitM can be realized by inserting a transparent device in the wire that forwards traffic while observing or modifying it. In the security literature this is often described as a BITW element. Authoritative guidance for OT networks explicitly describes BITW devices that sit inline on Ethernet links to add security functions when endpoints cannot be changed demonstrating their practicality and low operational friction as on-path intermediaries.

In the automotive context, a Chalmers thesis [62] evaluated a MACsec [92] enabled BITW for in-vehicle Ethernet, quantifying its latency/throughput overhead and showing that inline devices can maintain real-time properties while operating on every frame i.e., precisely the capability a MitM needs for faithful interception and transformation.

4.6.1.2 Why Inline MitM Matters in AE Protocols

Systematic AE reviews continue to flag MitM as a first-order risk because baseline deployments still mix unauthenticated discovery and best-effort data flows with emerging TSN streams. This combination produces attack surfaces where an on-path adversary can read and alter traffic before higher-layer checks occur.

For SOME/IP [100, 59, 17] specifically, formal and model-based analyses show how request/response and publish/subscribe exchanges are vulnerable to on-path manipulation and impersonation if authenticity/freshness are not enforced precisely the conditions exploited by an inline MitM. Moreover, AE adoption of TSN for deterministic delivery adds a twist: predictable periodic traffic can help an on-path attacker blend manipulations into scheduling windows unless link-layer protections are in place.

4.6.1.3 Monitoring via Active Tap

Recent academic work on vehicle IP/Ethernet analysis demonstrates that in-vehicle links can be monitored at line rate with precise timing, focusing on methods to capture and analyze traffic between ECU for protocol/latency studies. While positioned as benign monitoring, these methods rely on the same inline interception primitives that an active-tap MitM would employ, reinforcing feasibility for AE traffic observation without endpoint modification.

4.6.1.4 DoS via Active Tap

Several research efforts have highlighted that an active inline tap commonly referred to as a BITW device can also serve as a launch point for DoS in AE. By virtue of being directly in the communication path, such a device can deliberately drop, delay, or flood packets between ECU, disrupting time-sensitive communication and exhausting network resources.

Systematic reviews of AE security classify DoS as a critical outcome of on-path adversaries, particularly in TSN environments where deterministic delivery can be destabilized by even small delays or packet loss. Studies in both automotive and industrial control networks demonstrate that inline MitM nodes are capable of selectively corrupting or discarding synchronization traffic, leading to desynchronization of TSN schedules and service interruptions in higher-layer protocols like SOME/IP [59, 17, 102] and DoIP [96].

Compared to ARP-spoofing-based MitM, active tap DoS is harder to detect through signature methods, since the adversary controls all frames passing through the link and can emulate normal forwarding while silently degrading availability. This positions DoS via active tap as one of the most disruptive attack scenarios in AE, underscoring the need for robust link-layer authentication and intrusion detection mechanisms.

4.6.2 Related Works on ARP-Spoofing MitM in AE

4.6.2.1 Related Work on ARP-Spoofing–Based Monitoring Attacks

A number of studies have identified ARP spoofing as a critical enabler for MitM monitoring in both traditional and vehicular Ethernet networks. Bruschi *et al.* highlight that ARP stateless design, which updates cache entries without validation, allows attackers to poison ARP tables and redirect traffic to themselves, creating the foundation for stealth monitoring of sensitive data in industrial and vehicular contexts. Similarly, forensic analyses of IoT systems demonstrate that ARP spoofing [30] not only enables passive sniffing of TCP/UDP flows but also leaves identifiable traces that can be reconstructed during forensic investigations.

In the automotive domain, systematic reviews (Moruzzi *et al.*, 2024) categorize MitM through ARP spoofing [30] as one of the most prominent link-layer vulnerabilities in AE. These reviews emphasize that AE lack of authentication at the data-link layer makes it particularly exposed to ARP-based interception. By exploiting this weakness, an attacker gains full-duplex visibility into critical ECU-to-ECU communication such as SOME/IP [12, 61] or DoIP [96, 73] messages, without triggering immediate alarms in higher layers. Importantly, these works collectively show that monitoring attacks are not only feasible but also stealthy enough to act as precursors to more invasive manipulations.

4.6.2.2 Related Work on ARP-Spoofing–Based Tampering Attacks

Beyond passive monitoring, ARP spoofing [30] MitM has been leveraged for active tampering. Zelle *et al.* (2021) demonstrated that, once a MitM position is established, SOME/IP [61, 28, 102] request/response and publish/subscribe communications can be manipulated to inject false data or impersonate services. This finding is reinforced by threat models of DoIP [96, 73, 19], which list ARP poisoning as a direct entry point for altering diagnostic sessions and commands if encryption and mutual authentication are not enforced.

Research outside the automotive domain also confirms this escalation path. Bruschi *et al.* (2024) show how ARP-based MitM in industrial networks can lead to false data injection, command modification, and operational disruption, reinforcing the cyber-physical consequences of such tampering. Similarly, studies on ARP spoof vulnerabilities classify MitM as the most damaging variant because it enables both interception and modification, often serving as a gateway to more advanced attacks such as session hijacking, pharming, and DoS.

4.6.3 Research Gap

Although prior research has established the feasibility of MitM through both ARP spoofing [30] and active inline tapping, several critical gaps remain when these findings are applied to AE. First, most studies on ARP-spoofing MitM focus either on general IT/IoT environments or on conceptual models of in-vehicle communication, with limited empirical validation on real AE testbeds. The cyber-physical consequences of ARP-based monitoring and tampering are frequently highlighted but rarely measured against actual ECU-to-ECU traffic flows in SOME/IP [70], DoIP [96, 73], or TSN-enabled networks.

Second, while active-tap MitM has been investigated in the context of BITW security devices or industrial network evaluations, its malicious potential in AE remains underexplored. Research tends to treat inline devices as defensive gateways (e.g., MACsec [92] enabled bridges) rather than adversarial tools, resulting in a lack of systematic studies on their use for monitoring, tampering, or DoS in vehicular Ethernet. The literature acknowledges that an inline adversary can corrupt or delay TSN synchronization traffic, yet concrete experimental demonstrations in automotive environments are scarce.

Third, across both ARP-spoofing and active-tap approaches, the comparative analysis of attack vectors is missing. Existing works discuss these methods in isolation, without quantifying their relative stealthiness, detectability, or impact on safety-critical timing in AE. For instance, ARP spoofing [30] may be easier to launch but more susceptible to anomaly detection, while active-tap MitM offers precise control but requires physical insertion. The interplay of these trade-offs has not been thoroughly addressed in the automotive domain.

Finally, many proposed detection and mitigation strategies (e.g., ARP monitoring, anomaly detection, MACsec [92]) are examined at a theoretical or simulation level, but not under the latency and determinism constraints unique to AE. This creates a practical gap between security solutions suggested in the literature and their feasibility in real in-vehicle networks.

By experimentally implementing both ARP-spoofing and active-tap MitM in a controlled AE test bench, this thesis directly addresses these gaps. It provides empirical evidence of how each method affects ECU communications, evaluates their potential to enable secondary attacks such as tampering and DoS, and lays the groundwork for future studies on intrusion detection tailored to AE.

This chapter has provided an in-depth exploration of the MitM attack as a central threat to AE, supported by a review of related works on ARP-spoofing and active-tap methodologies. While ARP-spoofing offers a lightweight entry point for both monitoring and tampering, active-tap approaches reveal the potential for precise interception and even DoS at

the link layer. Together, these studies highlight both the versatility and severity of MitM attacks, while also exposing research gaps in experimental validation, comparative analysis, and cyber-physical impact assessment within real AE environments. Building on this foundation, the next chapter transitions from literature and theoretical framing to the practical methodology adopted in this thesis, detailing how MitM attacks were systematically implemented in an AE test bench to evaluate their feasibility and implications.

CHAPTER 5

Methodologies for MitM Attacks in AE

5.1 Introduction

In this research, two methodologies are selected to implement and analyze MitM attacks on AE: active tapping and ARP spoofing. Active tapping provides hardware-level inline access to live traffic and allows real-time packet manipulation, while ARP spoofing exploits weaknesses in ARP to transparently reroute communication flows through an attacker. Together, these approaches offer complementary perspectives: active tapping represents a physical-layer adversary with precise control, and ARP spoofing illustrates a link-layer adversary exploiting protocol design flaws. This dual methodology ensures that the experiments capture both hardware-based and protocol-based threats to AE systems.

5.2 Active Tapping

Active tapping is a critical technique used to enable MitM attacks in high-speed networks like AE. Unlike passive tapping, which simply monitors traffic, **active tapping places the attacker inline between two ECU**, offering full control over both observation and manipulation of data packets.

In AE environments, this technique is particularly effective due to the **transparent nature of Ethernet switching** and the **absence of built-in cryptographic protections** in key automotive protocols. Attackers using active taps can exploit these weaknesses to intercept and alter payloads without triggering alarms or detection by either communication endpoint.

5.2.1 Principle of Operation

Active tapping involves inserting a device—either software-based (e.g., a Raspberry Pi in bridge mode) or hardware-based (e.g., RAD-Galaxy)—directly between two ECU. The device

acts as a **transparent Ethernet forwarder**, relaying packets in both directions while enabling the attacker to:

- Capture and log packets in real time.
- Modify fields in transit (e.g., SOME/IP service payloads).
- Drop specific messages to simulate failures or delay traffic.
- Replay previously captured frames with modified timing.

Unlike traditional routers, an active tap preserves MAC and IP header structures unless explicitly modified. This helps the attack remain undetected from both ECU perspectives, making it highly effective for long-duration and stealthy MitM scenarios.

5.2.2 Distinction Between Active and Passive Tapping

Table 5.1: Comparison of Active and Passive Tapping Techniques

Feature	**Passive Tap**	**Active Tap**
Traffic Manipulation	Not possible	Fully supported
Placement Impact	No signal delay	May introduce slight propagation delay
Use Case	Monitoring / Forensics	Real-time MitM attacks / Injection
Tools	Ethernet TAP, SPAN port	Transparent bridge, RAD-Galaxy, Raspberry Pi
Detection Risk	Very low	Moderate if not stealthed (e.g., timing anomalies)

5.2.3 Active Tapping as a MitM Strategy

MitM attacks are a prominent cybersecurity threat in AE due to the deterministic and real-time nature of vehicular communication. Active tapping represents the most **powerful and flexible form** of MitM attack, enabling an adversary not just to eavesdrop, but to **actively participate** in the communication exchange.

With active tapping, an attacker can:

- Impersonate a legitimate ECU by injecting crafted SOME/IP or DoIP packets.

- Alter critical control messages in motion, affecting safety systems like braking or steering.
- Delay time-sensitive packets in TSN flows, resulting in synchronization failures or degraded performance.

Such real-time manipulation is essential for evaluating the robustness of AE protocols under adversarial conditions, especially when traditional software-layer intrusion methods fall short in simulating physical network compromises.

Importance of Active Tapping for Physical Intrusion Testing

Modern vehicles expose several physical access points such as OBD-II ports, diagnostics gateways, and external ECU which adversaries can exploit to reach the internal AE backbone. Once access is gained, software-based security defenses can often be bypassed. **Active tapping** becomes the most realistic and practical method to simulate and study physical-layer intrusions.

Offline simulations or packet replays fail to meet the demands of AE testing because they:

- Cannot react to live traffic changes in real time.
- Lack bidirectional synchronization.
- Do not meet the stringent timing and fidelity requirements of TSN or SOME/IP flows.

In contrast, active tapping:

- Integrates seamlessly with AE hardware using tools like RAD-Galaxy.
- Enables protocol-aware attacks while preserving Ethernet layer behavior.
- Facilitates realistic intrusion evaluation with precise control over frame content and timing.

5.3 ARP Spoofing

5.3.1 Principle of Operation

ARP spoofing is a link-layer attack technique in which the adversary sends forged ARP replies to poison the ARP caches of communicating ECU. Once poisoned, both ECU forward their traffic to the attacker, believing it to be the legitimate destination. The attacker, acting as a transparent bridge, can then:

- Passively capture ECU-to-ECU traffic for monitoring.
- Modify SOME/IP or DoIP payloads before forwarding.
- Drop or delay selected frames to disrupt services.
- Replay previously observed messages to simulate valid ECU responses.

Unlike active tapping, which requires inline hardware, ARP spoofing exploits AE lack of authentication in link-layer protocols, making it lightweight and easily deployable with minimal resources.

5.3.2 ARP Spoofing as a MitM Strategy

MitM via ARP spoofing allows an adversary to impersonate ECU at the link layer. Once traffic is rerouted through the attacker, adversaries can:

- Inject false SOME/IP responses, impersonating a legitimate ECU.
- Alter diagnostic DoIP sessions by modifying requests or responses.
- Replay captured ECU data to trigger false system states.
- Act as a stepping-stone for higher-level attacks, such as session hijacking or DoS.

This makes ARP spoofing a versatile methodology for MitM in AE, particularly because it does not require modifications to the physical network topology.

Table 5.2: Comparison of Active Tapping and ARP Spoofing Techniques

Feature	**Active Tap**	**ARP Spoofing**
Traffic Manipulation	Fully supported inline	Supported via ARP redirection
Resource Requirement	Specialized hardware (RAD-Galaxy, bridge device)	Commodity hardware + software (e.g., arpspoof, Scapy)
Stealth	High if latency is preserved	Moderate, detectable by ARP anomaly monitoring
Attack Vector	Physical-layer insertion	Protocol-layer poisoning
Use Case	High-fidelity MitM, DoS	Monitoring, Tampering, Replay

5.3.3 Importance of ARP Spoofing for Intrusion Testing

While active tapping demonstrates MitM feasibility at the physical layer, ARP spoofing highlights weaknesses in AE reliance on unauthenticated ARP. This is particularly relevant for scenarios where:

- The attacker gains access through a compromised ECU, service tool, or aftermarket device.
- Direct inline hardware insertion is impractical.
- Protocol-level weaknesses need to be evaluated alongside physical intrusions.

In such cases, ARP spoofing provides an efficient and realistic method for testing AE resilience against MitM attacks.

5.4 Selected Methodologies

Given the dual research objectives to evaluate MitM vulnerabilities under both physical and protocol layer threat models this thesis adopts both active tapping and ARP spoofing as experimental methodologies.

- **Active tapping** enables full-fidelity, bi-directional interception and modification of traffic at line speed, ideal for demonstrating packet manipulation and DoS scenarios.
- **ARP spoofing** exploits inherent protocol weaknesses to achieve MitM without specialized hardware, demonstrating the accessibility of such attacks in practice.

This combination ensures that the experimental framework reflects both high-capability adversaries with physical access and lightweight adversaries exploiting software-level vulnerabilities. Together, they provide a holistic assessment of MitM risks in AE.

This chapter established active tapping and ARP spoofing as the two selected methodologies for implementing MitM attacks in AE systems. By examining their operational principles, attack potential, and respective roles at the physical and link layers, the chapter demonstrated how each approach provides a realistic avenue for intercepting, manipulating, and disrupting in-vehicle communication. Active tapping highlights the feasibility of precise, hardware-level control over live traffic, while ARP spoofing illustrates how lightweight protocol-level weaknesses can be exploited for both monitoring and tampering. Together, these methodologies

offer a comprehensive framework for assessing AE vulnerabilities under different adversarial models. The following chapter transitions from this conceptual methodology to hands-on implementation, detailing the construction of a functional AE testbench, the configuration of both active tap and ARP spoofing setups, and their application in executing MitM, monitoring, and DoS attacks. Through these experimental setups, the chapter provides empirical evidence of AE vulnerabilities and delivers practical insights into the security posture of modern in-vehicle networks.

CHAPTER 6

Experimental Setup

6.1 Introduction

To understand the real-world impact of such intrusions, it is essential to evaluate how AE systems behave under adversarial conditions when physical access and network manipulation are possible. This chapter begins by introducing the hardware and software tools used to construct the experimental testbed, including the RAD-Galaxy Ethernet tap, Vehicle Spy analysis suite, and Raspberry Pi 5 ECU. It then explains the setup and workflow for executing each type of attack MitM, Monitoring, and DoS using active tapping as the core mechanism.

By emulating realistic attack conditions, the experiments conducted in this chapter provide valuable insights into the vulnerabilities of AE communication protocols and the critical need for robust cybersecurity frameworks in modern vehicles. The observed behaviors and outcomes serve as a foundation for the discussion and interpretation of results in the following chapter.

6.2 Experimental Testbed Architecture

To evaluate the real-world feasibility of cyber attacks on AE systems, a modular and scalable experimental testbed was designed and assembled. The core objective of this architecture is to emulate a real in-vehicle Ethernet communication environment while enabling live traffic interception, manipulation, and analysis using both standard and specialized hardware tools.

The testbed consists of three primary subsystems:

- ECU implemented using Raspberry Pi 5 boards.
- Automotive-to-Ethernet media converters, specifically RAD-Moon2 devices.
- An active tapping and monitoring unit, implemented using the RAD-Galaxy.

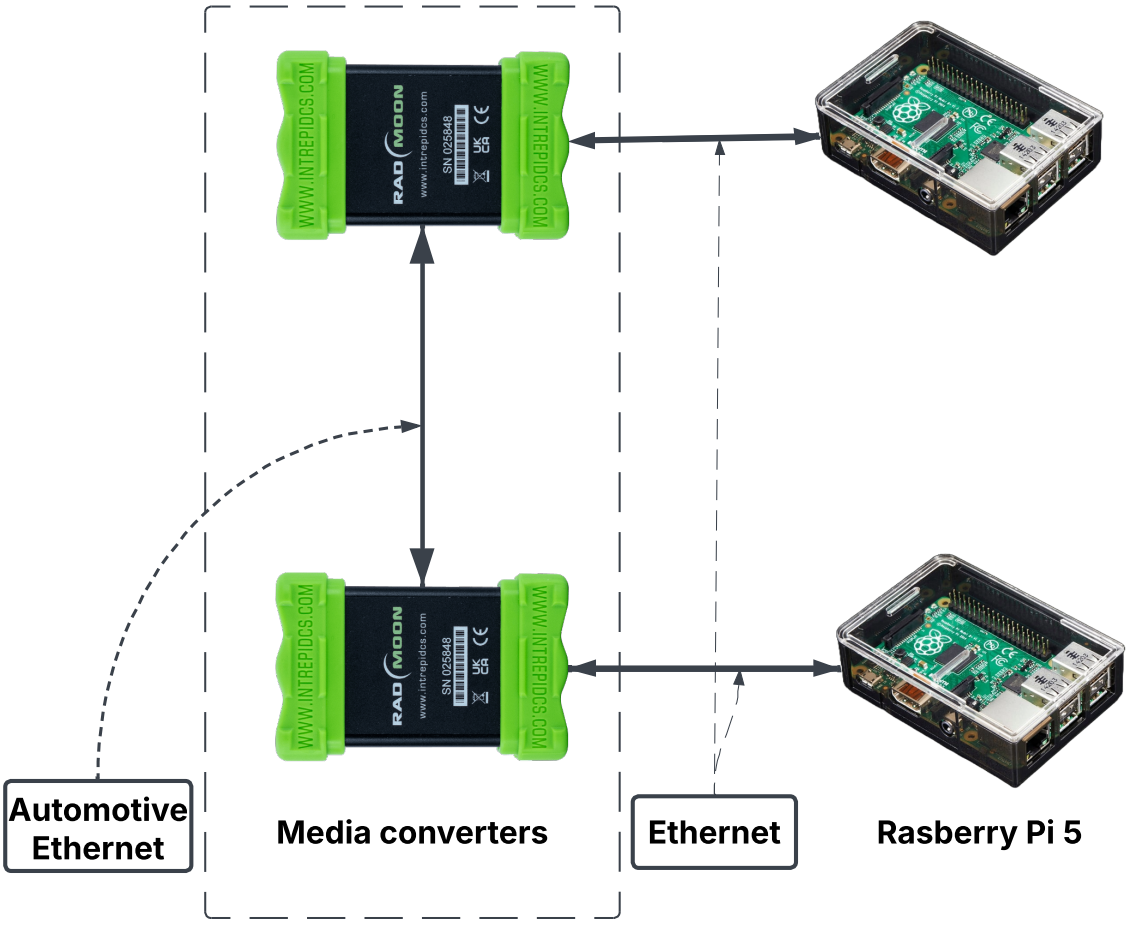

Figure 6.1: Direct communication between ECU 1 and ECU 2 without active tap[48]

Together, these components form a closed-loop communication path that mimics in-vehicle Ethernet exchange, with provisions for real-time monitoring and manipulation through a MitM setup. A conceptual overview of the system architecture is shown in Figure 3.1.

6.2.1 Communication Topology

The experimental topology is based on a point-to-point Ethernet link between two ECU, with the communication link intercepted by the RAD-Galaxy acting as an active tap. The system is logically divided into the following stages:

Stage 1: ECU to Media Converter Interface

Each Raspberry Pi 5 functions as an ECU, responsible for sending and receiving UDP packets to emulate application-level vehicle functions. Since Raspberry Pi boards support only standard RJ-45 Ethernet, a conversion to the AE standard (100BASE-T1) is required. This is accomplished using RAD-Moon2 media converters that translate standard Ethernet to

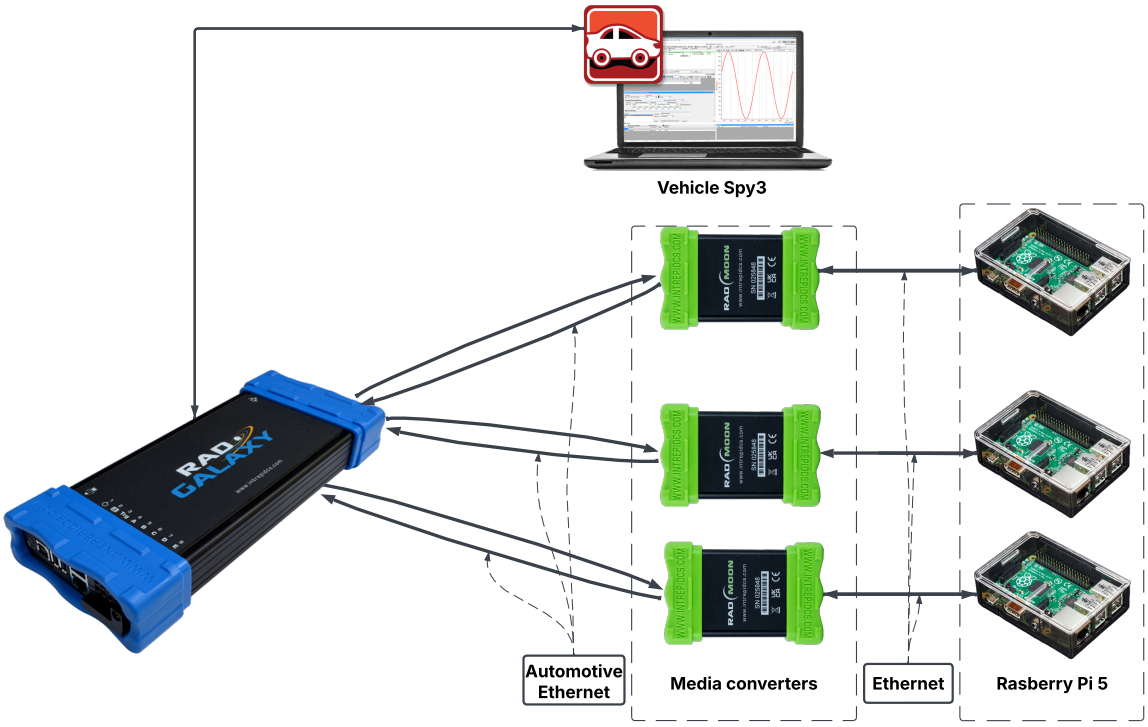

Figure 6.2: A conceptual overview of the system architecture[48].

automotive-grade signaling.

Stage 2: AE Segment with Tap

The AE signal from each RAD-Moon2 is connected to the RAD-Galaxy, a specialized Ethernet tap that supports 100BASE-T1 and provides real-time active tapping functionality. This setup enables the interception and injection of Ethernet frames with minimal latency and without disrupting the deterministic nature of AE communication.

Stage 3: Monitoring and Control Interface

The RAD-Galaxy is connected to a host machine running Vehicle Spy3, which provides a GUI-based interface for packet monitoring, editing, and logging. Through this interface, the researcher can:

- View live traffic between ECU
- Modify frame content on-the-fly
- Inject synthetic traffic
- Analyze communication patterns

This topology ensures full visibility and control over the AE communication stream, which

is critical for evaluating the effect of different cyber attacks in a deterministic automotive environment.

6.2.2 Synchronization and Timing Considerations

AE applications often require precise timing and deterministic delivery, particularly in cases involving TSN. The RAD-Galaxy is designed to maintain these constraints by operating at the physical layer without introducing jitter or frame loss.

The Raspberry Pi ECU are synchronized via NTP to ensure consistent timing for UDP communication. Packet timestamps are logged at both endpoints and compared with tapped data to measure latency, jitter, and potential manipulation effects.

6.2.3 Advantages of Modular Design

The modularity of the testbed architecture allows:

- Easy reconfiguration for different attack scenarios (DoS, Replay, ARP spoofing).
- Scalability: Additional ECU or tapping points can be added without major redesign.
- Compatibility with future tools or components, such as FPGA-based IDS units or gateway emulators.

6.3 RAD-Moon: Media Converter for AE

The RAD-Moon is a crucial component of the testbed architecture designed to interface AE (100BASE-T1) communication with standard Ethernet (100BASE-TX). As a media converter, it enables seamless communication between devices using different physical layer technologies, ensuring compatibility between legacy Ethernet-based systems and modern AE solutions. This section provides a detailed overview of the RAD-Moon and its role in the experimental setup.

6.3.1 Overview and Functionality

The RAD-Moon is designed to convert 100BASE-T1 AE to the 100BASE-TX Ethernet standard, which uses a dual twisted-pair cabling system rather than the single twisted-pair used in AE. This conversion is vital for linking AE devices, such as Raspberry Pi 5 ECU, with standard Ethernet infrastructure (e.g., the RAD-Galaxy active tap and monitoring systems).



Figure 6.3: RAD-MOON

Key features of the RAD-Moon include:

- PHY Conversion: The 100BASE-T1 PHY (NXP) is connected to the 4-wire 100BASE-TX Ethernet PHY, facilitating the transition between the two physical layers while maintaining signal integrity.
- Auto-Configuration: The RAD-Moon supports master/slave auto-configuration, automatically determining the role in the Ethernet network to simplify setup and configuration.
- Portability and Durability: The device is housed in a rugged, compact aluminum enclosure, making it ideal for field testing and easy integration into mobile setups.
- USB Powered: The RAD-Moon is powered via a standard USB connection, ensuring easy portability without the need for a dedicated power supply.
- Link Activity Indicators: LEDs are provided for both the 100BASE-T1 and 100BASE-TX PHYs, offering visual indicators of network activity, aiding in troubleshooting and verification.

- Connector Types: The Molex Mini-50 connector is used for the 100BASE-T1 PHY, and the RJ-45 connector is used for the 100BASE-TX PHY, enabling easy integration with a wide variety of Ethernet devices.

6.3.2 Role in the Testbed

In the experimental testbed, the RAD-Moon plays a pivotal role in enabling communication between the Raspberry Pi 5 units (acting as ECU) and the RAD-Galaxy active tap. Specifically, the RAD-Moon:

- Enables Communication: It converts the 100BASE-T1 signals from the Raspberry Pi ECU to the 100BASE-TX format, making the network compatible with traditional Ethernet hardware.
- Simplifies Network Setup: By providing auto-configuration features, it eliminates the need for manual configuration of network roles, which enhances the ease of setup and reduces the likelihood of configuration errors.
- Integrates into the Monitoring System: The link activity LEDs allow real-time monitoring of network status and assist in debugging potential issues during active tap-based attack testing.

6.3.3 Practical Considerations

The compact and portable nature of the RAD-Moon makes it especially useful in experimental environments that require flexibility. The rugged aluminum enclosure ensures that the device can withstand environmental stresses typically encountered in mobile test setups. Additionally, the USB-powered design allows it to be easily powered from laptops or portable power sources, making it ideal for field testing scenarios where a dedicated power supply may not be available.

Given its small form factor and robust functionality, the RAD-Moon is an essential tool in validating the performance of AE in real-world attack scenarios, particularly when integrated with RAD-Galaxy for traffic manipulation and analysis.

6.3.4 Advantages of RAD-Moon in AE Vulnerability Testing

In the context of this research, the RAD-Moon provides several advantages:

- Compatibility with Legacy Systems: It allows integration of AE devices with traditional Ethernet-based testing tools, broadening the scope of attack simulations and analysis.

- Real-Time Traffic Conversion: By supporting high-speed, real-time conversion between 100BASE-T1 and 100BASE-TX, it ensures that the testbed operates under realistic automotive communication conditions.

- Ease of Use: The USB-powered design and auto-configuration features simplify setup, allowing researchers to focus on attack execution and analysis rather than complex configuration tasks.

The RAD-Moon is a critical component in the experimental setup, bridging the gap between automotive-specific networking protocols and traditional network monitoring systems. This ensures that the testbed accurately reflects the real-world conditions of in-vehicle Ethernet communications.

6.4 RAD-Jupiter: AE Switch

The **RAD-Jupiter**, developed by Intrepid Control Systems, is a rugged 7-port managed switch designed specifically for AE (100/1000BASE-T1) bench and vehicle development. Unlike the RAD-Galaxy, which functions as an active tap, the RAD-Jupiter primarily serves as a high-performance switch with embedded gateway and protocol handling capabilities, making it an essential tool for larger or more complex AE networks.

6.4.1 Key Features

- **7-Port Managed Switch:** Provides 5× 100BASE-T1 ports, 1× 1000BASE-T/1000BASE-T1 port, and 1× 1000BASE-T or USB/GigE bridge, allowing flexible connectivity across diverse network setups.

- **AVB/TSN Protocol Support:** Includes embedded stacks for IEEE 802.1Qav, IEEE 802.1AS, and IEEE 802.1Qbv (scheduled traffic in future firmware). These features enable precise timing and deterministic communication required for ADAS and autonomous systems.

- **Media Conversion Capability:** Configurable as 2× 100BASE-T1 or 1× 1000BASE-T1 media converter without losing use of the remaining switch ports.



Figure 6.4: RAD-JUPITER

- **Integrated CAN/LIN Interfaces:** Provides 2× CAN FD channels and 1× LIN channel, supporting hybrid in-vehicle network testing.
- **Cybersecurity and Filtering:** Equipped with TCAM for advanced filtering, whitelisting/blacklisting, message forwarding rules, and firewall functionality at wire speed.
- **Programmable Embedded Processor:** Supports scripting and embedded C applications, enabling use as a gateway device between CAN/CAN FD and Ethernet.
- **Rugged Design:** Features a durable aluminum case, shock-absorbing boots, flexible power input options (DB9/USB-C/barrel jack), and wide operating temperature range ($-40°C$ to $85°C$).

6.4.2 Applications in the Test Bench

In the AE test bench, the RAD-Jupiter was used where experiments demanded higher port density and advanced protocol support. Its AVB/TSN compliance made it particularly useful for evaluating the timing sensitivity of Ethernet streams under cyber-attack conditions. Furthermore, its TCAM-based filtering and embedded gateway capabilities allowed for experiments involving traffic manipulation, whitelisting/blacklisting scenarios, and hybrid CAN–Ethernet gateway attacks. While it does not function as an active tap like the

RAD-Galaxy, its versatility as a managed switch makes it indispensable for scaling experiments involving multiple ECU and diverse network protocols.

6.5 RAD-Galaxy: Active Network Tap and Gateway

The RAD-Galaxy is a powerful, multi-functional active network tap and media converter designed specifically for AE applications. It plays a critical role in the experimental setup by enabling real-time monitoring, traffic analysis, and network manipulation on AE networks. With its ability to tap up to six links, the RAD-Galaxy supports both active tapping and media conversion features, offering a comprehensive solution for network testing and simulation.

This section provides an in-depth look at the RAD-Galaxy features, its role in the testbed, and its advantages for AE security testing.

6.5.1 Overview and Functionality

The RAD-Galaxy is a next-generation tool that integrates AE and CAN networks, offering high precision and low-latency monitoring capabilities. It features:

- **Active Tap Mode:** The RAD-Galaxy acts as an active tap, enabling the capture of full-duplex communication between up to six pairs of AE devices (ECU, switches, etc.). This setup ensures transparent network monitoring while maintaining network integrity.

- **Media Converter Mode:** The device can also be used as a media converter, facilitating the connection of standard Ethernet devices (like laptops or data loggers) to AE ECU. This feature allows for direct diagnostics, testing, and ECU flashing.

- **Gigabit Ethernet Gateway:** It provides seamless integration with standard Gigabit Ethernet networks, allowing the monitoring and manipulation of up to 12 ECU or devices using a standard Ethernet interface.

- **CAN FD and LIN Support:** In addition to Ethernet capabilities, the RAD-Galaxy supports CAN FD and LIN communication protocols, making it a versatile tool for testing hybrid networks in automotive applications.



Figure 6.5: RAD-GALAXY

6.5.2 Key Features of the RAD-Galaxy

The RAD-Galaxy feature set is highly tailored to automotive network monitoring and testing. Key features include:

- **Tapping Full-Duplex Communications:** With sub-microsecond latency, the RAD-Galaxy captures full-duplex Ethernet traffic, allowing real-time analysis without interfering with network operations.
- **Basic Filtering and Routing:** It offers basic filtering and routing capabilities for network traffic management during attack simulations.
- **PTP Support:** The device supports PTP, ensuring accurate synchronization of network devices, which is crucial for time-sensitive automotive applications.
- **AVB:** The RAD-Galaxy also supports AVB for managing audio and video data streams in automotive networks.
- **Vehicle Spy Integration:** The device includes a full version of Vehicle Spy Enterprise, which enables real-time monitoring, traffic analysis, message generation, and protocol analysis.

6.5.3 Role in the Testbed

In the context of the experimental testbed, the RAD-Galaxy serves as the primary tool for traffic monitoring and attack execution. It intercepts, copies, and forwards network traffic



Figure 6.6: RAD-GALAXY side view

from the AE links while ensuring minimal interference with the legitimate communication between ECU. The role of RAD-Galaxy in the testbed includes:

- **Real-Time Traffic Monitoring:** By tapping active Ethernet links between the ECU, the RAD-Galaxy enables real-time traffic analysis, ensuring accurate data collection for subsequent analysis of attack impacts.
- **Seamless Data Logging:** With its support for standalone data logging, the RAD-Galaxy captures all network data and stores it for later analysis. This logging is crucial for post-attack forensic analysis and report generation.

In addition to attack simulations, the RAD-Galaxy can be employed for network diagnostics, data logging, and system-level testing, ensuring that all aspects of AE network security are thoroughly validated.

6.6 Setting up RAD-Galaxy in Active Tapping Mode with Vehicle Spy

Vehicle Spy 3 by Intrepid Control Systems is a powerful software tool used for network analysis, diagnostics, and testing within automotive communication systems. It is specifically designed to work with a range of hardware devices, including the RAD-Galaxy, to provide real-time traffic analysis, data manipulation, and network testing. In this section, we will explore how to use Vehicle Spy 3 for configuring RAD-Galaxy in Active Tapping Mode and individual operating modes.

6.6.1 Overview of Vehicle Spy 3

Vehicle Spy 3 is a comprehensive software solution designed to interact with AE, CAN, LIN, and other in-vehicle communication protocols. It is equipped with several features, including:

- **Real-time network monitoring:** View traffic on the network in real-time, enabling you to analyze the flow of messages between ECUs.
- **Message generation and injection:** Allows users to send custom messages to the network for testing or simulation purposes.
- **Protocol analysis:** Supports various automotive communication protocols, providing detailed views of the data being transmitted.
- **Network configuration:** Vehicle Spy can configure devices, manage networks, and optimize the settings for automotive systems.

In the context of this thesis, Vehicle Spy 3 will be used to configure RAD-Galaxy for active tapping, enabling the capture and analysis of Ethernet traffic from an AE network.

6.6.2 How to Use Vehicle Spy 3 with RAD-Galaxy

The process of using Vehicle Spy 3 with RAD-Galaxy begins with setting up the hardware and connecting it to the software. The following steps outline how to set up RAD-Galaxy in Active Tapping Mode using Vehicle Spy 3.

Connecting the Hardware:

- Connect the RAD-Galaxy to the AE network using the BroadR-Reach (100BASE-T1) Ethernet interfaces.
- Connect the RAD-Galaxy via Gigabit Ethernet to a laptop or workstation running Vehicle Spy 3. This enables communication between the software and the device.

Launching Vehicle Spy 3:

- Open Vehicle Spy 3 and ensure that the RAD-Galaxy is correctly connected to the software. This will be done via the Network Configuration settings in Vehicle Spy.

Select Active Tapping Mode:

- In Vehicle Spy 3, go to the Configuration panel.

- Select Active Tapping Mode. This mode configures the RAD-Galaxy to monitor traffic passing between ECUs by intercepting communication without altering the data flow.
- Active Tapping Mode allows you to monitor the communication without disrupting the network, providing a transparent view of all data exchanges between the devices.

Configure Individual Modes: Vehicle Spy 3 provides options to configure the RAD-Galaxy in different modes depending on the test requirements:

- Active Tapping: This is the default mode for intercepting traffic on the network without interfering with it. The RAD-Galaxy transparently copies Ethernet frames from the communication links.
- Media Converter Mode: In this mode, the RAD-Galaxy converts AE (100BASE-T1) to standard Gigabit Ethernet (1000BASE-T) or vice versa. This allows standard Ethernet devices (such as laptops or data loggers) to communicate with ECUs that use AE.
- Monitoring Mode: This mode allows for detailed inspection of the network traffic, including packet-level analysis, which is crucial for diagnosing issues or monitoring specific behaviors.

Adjust PHY Settings: Ensure that the 100BASE-T1 PHY settings are configured properly for the devices being monitored. Vehicle Spy will allow you to customize these settings based on the specific hardware and network configuration.
Start Active Tapping: Once the configuration is complete, activate Active Tapping Mode within Vehicle Spy to start monitoring network traffic in real time. This will allow you to view data from the connected ECUs without any disruption or alteration to the data.

6.6.3 Advantages of Using Vehicle Spy with RAD-Galaxy

Using Vehicle Spy 3 with the RAD-Galaxy provides several advantages for automotive network analysis:

- **Real-Time Monitoring:** Vehicle Spy allows you to see live data from the network as it flows between devices, making it easy to identify issues such as lost packets, communication delays, or misconfigurations.
- **Protocol-Specific Configuration:** Vehicle Spy provides fine control over network configurations, allowing you to manage and simulate protocols like AE, CAN, and LIN, all through one interface.

- **Test Flexibility:** Whether you need to monitor an existing network, simulate an ECU, or perform a stress test, Vehicle Spy enables flexible configuration options that can adapt to a variety of testing requirements.
- **Ease of Use:** The intuitive GUI of Vehicle Spy simplifies the process of setting up and managing complex network tests, even for non-expert users, with drag-and-drop functionality and automatic device detection.

6.7 AE Node

The Raspberry Pi 5 serves as a realistic ECU in this experimental testbed. Each Raspberry Pi 5 is configured to function as both a traffic generator and receiver, mimicking the behavior of real ECUs in an AE network. In conjunction with the RAD-Moon media converter, the Raspberry Pi 5 enables the formation of a fully functional AE node.

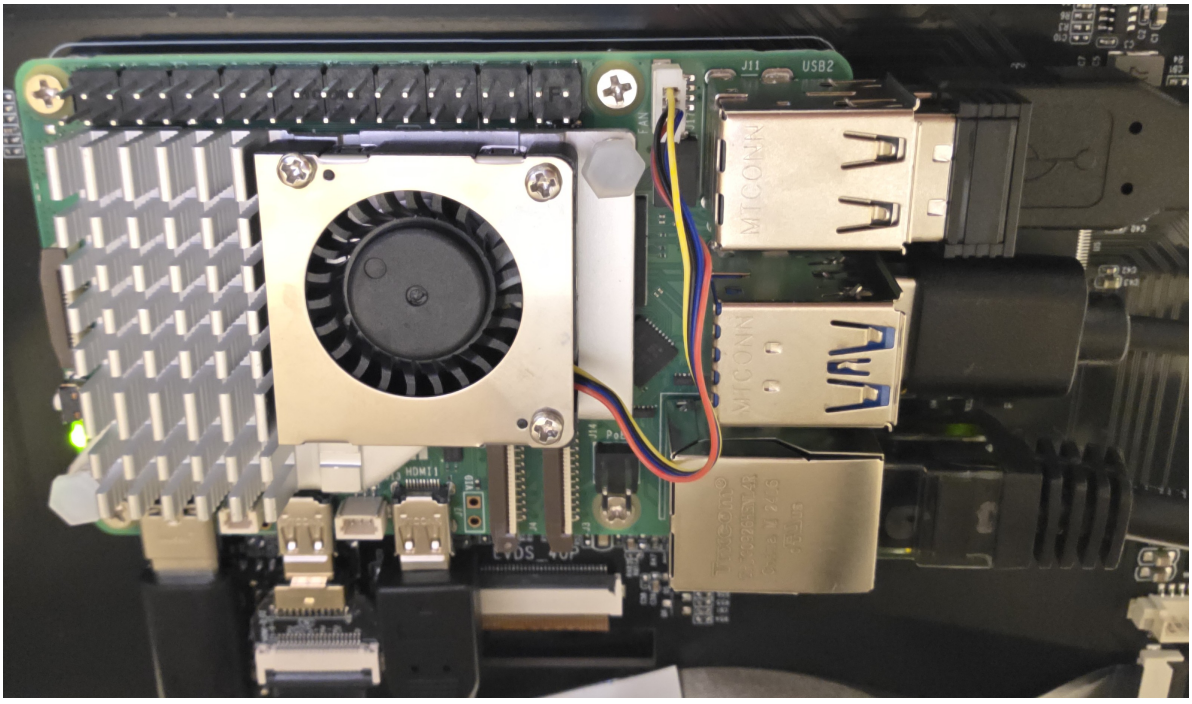

Figure 6.7: Raspberry Pi 5 as ECU

6.7.1 Raspberry Pi 5 as an AE ECU

In a typical automotive network, ECUs handle various tasks such as data processing, communication, and control functions. In this setup, each Raspberry Pi 5 emulates these functions, acting as both a data generator and receiver in an AE environment:

- **Traffic Generation:** The Raspberry Pi 5 sends simulated traffic over the network, generating sensor data, control messages, and status updates between ECUs. These messages emulate real automotive functions such as braking, steering, or vehicle diagnostics.
- **Traffic Reception:** The Raspberry Pi 5 listens for incoming messages from other ECUs, processes the data, and sends responses back. It simulates the receiving side of automotive communication, handling both real-time data reception and message processing.

The Raspberry Pi 5 serves as a highly flexible and cost-effective tool for simulating ECU functionality in a test environment.

6.7.2 Integrating RAD-Moon as a Media Converter

The Raspberry Pi 5 uses standard Ethernet for communication, but AE (100BASE-T1) is required to simulate real-world automotive communication. The RAD-Moon acts as a media converter, enabling the Raspberry Pi 5 to communicate over 100BASE-T1, the AE standard:

- The RAD-Moon converts the 100BASE-TX Ethernet used by the Raspberry Pi to 100BASE-T1 for compatibility with AE networks.
- It uses Molex Mini-50 connectors to connect the AE cable (twisted-pair) to the network, mimicking real AE infrastructure.

This combination allows the Raspberry Pi 5 to communicate seamlessly in an AE system, using real AE cables and BroadR-Reach connectors.

6.7.3 Complete AE Node

Together, the Raspberry Pi 5 and RAD-Moon form a fully functional, scalable AE node. The Raspberry Pi 5 simulates the behavior of a real ECU, while the RAD-Moon ensures real-world compatibility with the AE network. This setup enables the testing and validation of AE communication, ECU interaction, and network performance in a real-world-like testbed environment.

6.8 Active Tap Testbench for MitM

The testbench, which utilizes the RAD-Galaxy and Raspberry Pi 5 as ECUs, enables a comprehensive simulation of an active MitM attack within an AE network. In this attack,

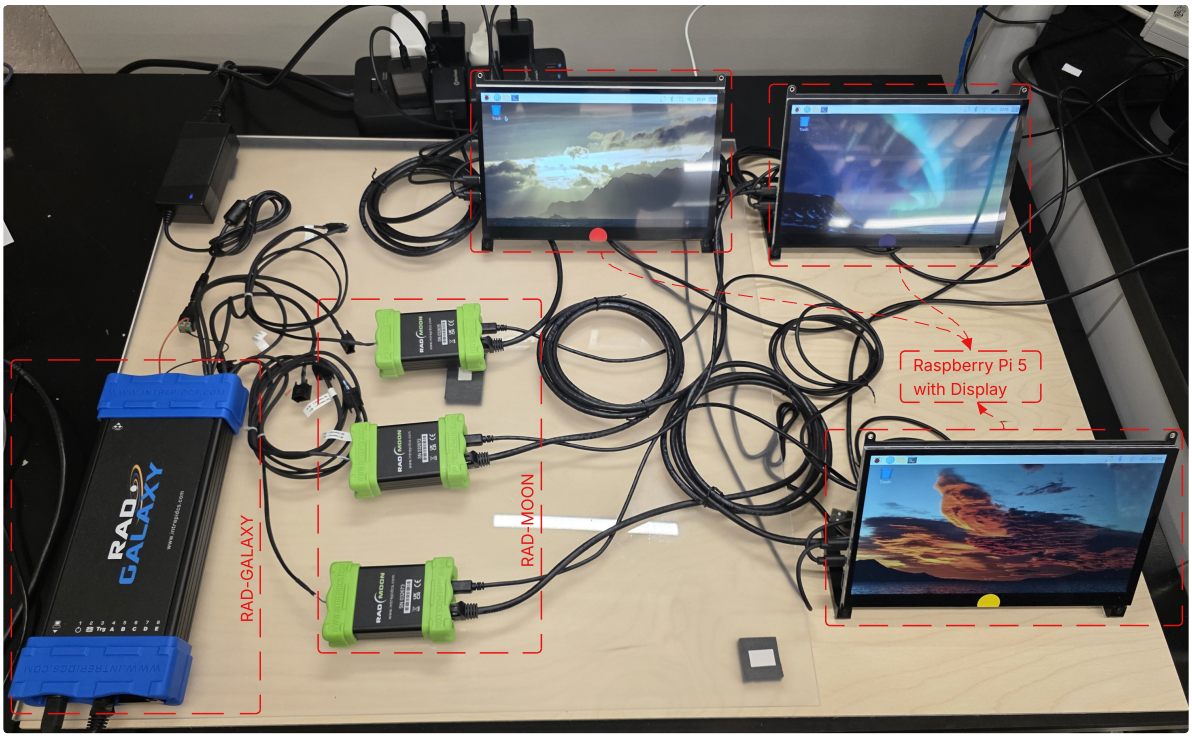

Figure 6.8: Hardware test bench

the RAD-Galaxy acts as an intermediary, intercepting and manipulating the traffic between two ECUs, while maintaining the integrity of the communication.

6.8.1 Testbench Setup for MitM Attack:

- **RAD-Galaxy Configuration:** The RAD-Galaxy is deployed as an active tap between the two Raspberry Pi 5 ECUs. It is connected to the AE network using the 100BASE-T1 interfaces, ensuring that all Ethernet frames exchanged between the ECUs pass through it.
- **Traffic Interception:** The RAD-Galaxy captures all bidirectional Ethernet frames between the ECUs without interrupting the data flow. The captured frames are then forwarded to the monitoring system, where they can be analyzed or modified.
- **Active Manipulation:** The key advantage of the active tapping is that the RAD-Galaxy does not just observe the traffic, but it can inject malicious or crafted packets into the communication stream. These packets can be tailored to:
  - Alter sensor or actuator data.

- Imitate a legitimate ECU, thereby causing the receiver to respond to fabricated commands.
- Introduce false messages into the network, disrupting control commands such as braking or steering.

6.8.2 MitM Attack Workflow:

1. The Raspberry Pi 5 ECUs start communicating over the AE network, exchanging critical vehicle data such as sensor readings or actuator states.
2. The RAD-Galaxy transparently taps into the network and intercepts the frames in real-time.
3. Depending on the attack scenario, the RAD-Galaxy manipulates the traffic, such as modifying frame content or injecting malicious packets into the communication.
4. The manipulated data is forwarded to the target ECU, which processes the malicious data as legitimate, potentially compromising vehicle safety.

By simulating this type of attack in a real-world setup, the testbench provides valuable insights into the AE network resilience against MitM attacks and helps in the evaluation of potential countermeasures.

6.9 Working of Testbench for Monitoring Attack

In the monitoring attack scenario, the goal is for the attacker (using the testbench setup) to silently capture and analyze network traffic without interacting with or altering the communication. This type of attack is particularly dangerous, as it allows the attacker to gather sensitive data such as vehicle diagnostics, control commands, and sensor readings, which can later be exploited for further attacks.

6.9.1 Testbench Setup for Monitoring Attack:

- **Passive Monitoring:** The RAD-Galaxy is configured to monitor the communication between the two Raspberry Pi ECUs without injecting any malicious packets or altering the data flow. The device taps into the network in a passive mode, where it simply observes the Ethernet frames being transmitted between the ECUs.

- **Data Logging and Analysis:** The data captured by the RAD-Galaxy is forwarded to a monitoring interface, such as Vehicle Spy 3, where it can be logged and analyzed in real-time. The attacker gains insight into the communication patterns, including control messages, status updates, and sensor data.

6.9.2 Monitoring Attack Workflow:

1. The Raspberry Pi ECUs engage in standard communication, exchanging operational data and control information.

2. The RAD-Galaxy taps into the communication link, silently capturing the Ethernet frames.

3. These captured frames are then forwarded to the monitoring system, where they can be analyzed or logged for future use.

4. The attacker does not interact with the data directly but collects enough information to carry out further malicious actions or to exploit vulnerabilities in the system at a later stage.

This type of attack underscores the importance of securing the communication channels in AE systems, especially when adversaries can potentially intercept data without detection.

6.10 Working of Testbench for DoS Attack

A DoS attack in the context of AE networks aims to overwhelm the network with a high volume of traffic, leading to the degradation of network performance or a complete loss of communication. The testbench, using the RAD-Galaxy, can simulate a variety of DoS attack scenarios to assess the resilience of the AE network under such conditions.

6.10.1 Testbench Setup for DoS Attack:

- **Flooding the Network:** The RAD-Galaxy is used to inject a large number of Ethernet frames into the AE network, simulating a DoS attack. These frames can be crafted to flood the network with excessive data, consuming available bandwidth and causing delays or packet drops.

- **Traffic Modification:** In addition to flooding, the RAD-Galaxy can be used to introduce delays into the packet flow, mimicking a situation where critical packets are

intentionally delayed to disrupt time-sensitive operations such as vehicle control systems.

6.10.2 DoS Attack Workflow:

1. The Raspberry Pi ECUs begin their communication over the AE network, performing normal operations such as exchanging sensor data and control commands.
2. The RAD-Galaxy starts injecting excessive traffic into the network, simulating a DoS attack. This traffic could consist of random or repetitive frames designed to consume bandwidth and saturate the network.
3. The flood of traffic begins to overwhelm the AE network, causing delays in communication, packet loss, and potential service disruptions.
4. The network performance is continuously monitored, and the impact of the DoS attack is analyzed in terms of increased latency, packet drops, and disrupted communication between the ECUs.

By simulating DoS attacks with the testbench, the researchers can assess how AE systems handle network congestion and disruptions, helping to design better defenses and improve network robustness under extreme conditions.

6.11 Vulnerability of AE to Cyber Attacks

The implementation of the active tapping attacks, such as the MitM attack, Monitoring attack, and DoS attack, clearly demonstrates the inherent vulnerabilities of AE networks. These tests have highlighted that AE systems, despite their technological advancements, are not immune to cyber threats. The fact that critical vehicle systems, including safety-related communications, can be intercepted, manipulated, or disrupted by malicious actors emphasizes the need for robust cybersecurity measures in modern automotive networks.

6.11.1 Key Vulnerabilities Exposed by the Attacks

- **MitM Attack:** The MitM attack demonstrated that an attacker, with physical access to the network, can intercept and manipulate the communication between ECUs. This type of attack is particularly concerning in automotive environments, where an attacker could alter critical control messages, such as braking or steering commands, posing a serious risk to vehicle safety.

- **Monitoring Attack:** The passive monitoring attack revealed that unauthorized entities could silently capture sensitive data without detection. This includes vehicle diagnostics, sensor data, and control commands. An attacker could use this information for future exploitation, leading to potential privacy breaches or the formulation of further malicious actions.
- **DoS Attack:** The DoS attack showed that the AE network can be overwhelmed by excessive traffic, causing delays, packet loss, and network disruptions. Given the time-sensitive nature of AE applications, such an attack could severely affect vehicle control systems, leading to performance degradation or failure in critical operations.

These findings underscore the critical security gaps in AE networks, where an attacker with the right tools and access could easily compromise the integrity, availability, and confidentiality of the network.

6.12 Monitor Attack through ARP Spoofing

6.12.1 Attack Objective

The primary objective of this attack is to demonstrate the feasibility of leveraging ARP spoofing to passively capture network traffic between two legitimate ECUs in an AE environment. The attack focuses on exploiting ARP inherent trust model to place the attacker in a transparent MitM position without actively disrupting or modifying traffic.

In automotive cyber-physical systems, monitoring attacks of this kind are a precursor to more invasive attacks such as data manipulation, command injection, or service disruption. By validating the possibility of silent packet interception, this experiment highlights a critical exposure in AE-based in-vehicle networks.

6.12.2 Theoretical Background

AE and ARP

AE operates predominantly on switched Ethernet architectures with 100BASE-T1 or higher physical layers, enabling high-speed deterministic communication between ECUs. At Layer 2, ARP is used to map IPv4 addresses to MAC addresses. ARP operates without authentication and relies entirely on the assumption that all network nodes provide truthful mappings.

ARP Spoofing

ARP spoofing (also referred to as ARP cache poisoning) is a Layer 2 attack wherein the

adversary sends falsified ARP replies to targeted hosts, associating the attacker MAC address with the IP address of a legitimate host. In a broadcast domain, ARP replies are generally accepted without verification, allowing an attacker to insert themselves transparently into the communication path.

Monitoring in Cybersecurity Context
From a cybersecurity standpoint, monitoring attacks are considered passive reconnaissance techniques. While they do not alter data in transit, they can provide an adversary with:

- Complete visibility into operational traffic patterns
- Access to sensitive payloads, including control commands and sensor readings
- Insight into network topology, timing behavior, and protocol stack usage

In the context of AE, these insights can be weaponized to plan advanced attacks against safety-critical vehicle subsystems.

6.12.3 Targeted Vulnerabilities

This attack specifically exploits:

- **Lack of ARP Authentication:** ARP in Ethernet does not verify the origin or legitimacy of ARP replies.
- **Flat Broadcast Domains:** AE networks often operate without ARP isolation, allowing spoofed replies to reach all nodes within a segment.
- **Absence of Packet Encryption:** Payload data is transmitted in plaintext, enabling content inspection once traffic is intercepted.

6.12.4 Test Bench Setup

- **Switch:** RAD-Jupiter automotive-grade managed Ethernet switch providing star topology connectivity.
- **Media Converters:** Three RAD-Moon units to bridge 100BASE-T1 AE to standard 100BASE-TX.
- **ECUs:**
  - ECU 1 Traffic generator

  - ECU 2 Traffic receiver
  - ECU 3 Attacker ECU (running ARP spoofing and capture tools)
- **Cabling:** UTP with automotive connectors (Board-to-Board reach).

6.12.5 Implementation and Attack Flow

1. **Initial State:** ECU 1 communicates directly with ECU 2 through the RAD-Jupiter switch. Each ECUs ARP table contains correct MAC–IP mappings.
2. **ARP Poisoning:** Attacker ECU sends falsified ARP replies:
   - To ECU 1: "ECU 2 IP is at my MAC address"
   - To ECU 2: "ECU 1 IP is at my MAC address"
3. **Path Redirection:** Both ECUs route their traffic through the attacker ECU, believing it to be the legitimate peer.
4. **Packet Capture:** Attacker ECU uses tools like `tcpdump` to capture and log all bidirectional traffic without altering it.
5. **Passive Analysis:** The attacker gains access to transport-layer headers, protocol payloads, and timing patterns.

6.12.6 Observations and Implications

The experiment confirmed that ARP spoofing in AE allows for complete traffic visibility with negligible latency impact and without alerting endpoints. In a real vehicle environment, such monitoring could be used for:

- Reverse-engineering proprietary protocols
- Tracking safety-critical data streams
- Mapping communication schedules for TSN systems

6.13 Tamper Attack through ARP Spoofing

6.13.1 Attack Objective

The aim of this attack is to escalate the capabilities of ARP spoofing from passive monitoring to active packet manipulation. By altering data in transit between two ECUs, the attacker

demonstrates the ability to compromise data integrity a critical component of the CIA triad (Confidentiality, Integrity, Availability).

6.13.2 Theoretical Background

From Monitoring to Tampering

In traditional network security taxonomy, passive attacks (monitoring) can be escalated to active attacks (tampering) once an adversary is in the data path. In the AE context, tampering could manipulate control signals, sensor data, or service messages, potentially causing unsafe vehicle behavior.

Integrity Compromise in Automotive Systems

Integrity in automotive communications ensures that the message received is exactly as intended by the sender. Without cryptographic verification (e.g., Message Authentication Codes), an attacker in a MitM position can inject or alter packets without detection.

OSI Layer Involvement:

- Layer 2 (Data Link): ARP spoofing creates MitM position.
- Layer 3 (Network): IP routing remains intact but now passes through attacker.
- Layer 4/7 (Transport/Application): Payloads can be altered to inject false sensor data, control commands, or service parameters.

6.13.3 Test Bench Setup

Identical to the monitoring attack:

- RAD-Jupiter switch, three RAD-Moon media converters
- ECU 1 (generator), ECU 2 (receiver), ECU 3 (attacker/tampering)

6.13.4 Implementation and Attack Flow

1. **ARP Spoofing Phase:** As in the monitoring attack, ECU 3 poisons ARP tables to route ECU 1 $\leftrightarrow$ ECU 2 traffic through itself.
2. **Packet Interception:** Traffic is captured using Python's `scapy` library or raw socket programming.
3. **Payload Manipulation:** The attacker modifies selected fields before forwarding:

- Example: Altering a temperature sensor reading from “85 °C” to “45 °C” in a diagnostic UDP packet.

4. **Forwarding:** Altered packets are immediately forwarded to maintain timing compliance.

5. **Verification:** tcpdump on the receiving ECU confirms the altered payload.

6.13.5 Observations and Implications

The successful tampering attack shows that AE networks lacking higher-layer authentication are susceptible to data integrity compromise. In real-world scenarios, this could:

- Mislead safety-critical ECUs (e.g., braking, ADAS)
- Cause inappropriate actuator responses
- Bypass diagnostics or fault detection by falsifying values

This chapter detailed the practical implementation of cyberattacks on an AE test bench, encompassing MitM, Monitoring, DoS, ARP spoofing, and tampering attacks. By leveraging both active tapping devices and ARP-based manipulation techniques, the experiments demonstrated how adversaries can intercept, alter, or disrupt in-vehicle communication flows in real time. These attack implementations validated their feasibility under realistic network conditions and exposed critical weaknesses in AE communication protocols and infrastructures. Building upon this foundation, Chapter 6 transitions from implementation to analysis by presenting the qualitative results and observations obtained during these experiments. It examines the behavioral effects of each attack, including how ARP spoofing enabled silent monitoring and tampering altered critical data streams, while DoS attacks disrupted service availability. Through these observations, the chapter highlights recurring patterns in system responses and underscores the broader implications for the security posture of AE networks.

CHAPTER 7

Results

7.1 Monitor Attack using MitM (Active Tap)

7.1.1 Overview

In this section, we present the results and observations from the Monitor Attack carried out using a MitM strategy facilitated by an Active Tap setup. The goal of this experiment is to validate the capability of an adversary to passively observe and log Ethernet traffic without disrupting communication between ECUs.

To achieve this, we utilized the RAD-Galaxy as an active tapping device placed transparently between two ECUs (ECU1 and ECU2), connected via RAD-Moon media converters. The monitoring interface of RAD-Galaxy was connected to a host running Vehicle Spy 3, enabling real-time packet capture and protocol inspection.

7.1.2 Experimental Configuration

- **Tap Mode:** Active Tap (RAD-Galaxy in Individual Mode)
- **ECU Setup:**
  - ECU 1 IP: `192.168.10.1`
  - ECU 2 IP: `192.168.10.2`
- **Communication Protocol:** UDP
- **Port Used:** 8888
- **Media Converters:** RAD-Moon for 100BASE-T1 to Ethernet bridging
- **Monitoring Tool:** Vehicle Spy 3 (Passive sniffing via RAD-Galaxy)

The setup ensured that the attacker (RAD-Galaxy) had visibility into all bidirectional traffic between the two ECUs without altering packet contents or introducing latency.

7.1.3 Observations and Results

From the captured trace (as shown in Figure 7.1), we were able to validate that:

- Bi-directional UDP communication between ECU1 and ECU2 was successfully intercepted.
- The attacker was able to log complete payloads of UDP frames.
- The captured packets contain identifiable IP addresses, MAC addresses, and application-layer payload data.
- Traffic characteristics such as source/destination ports, inter-packet delay, and frame lengths were fully visible to the attacker.

Example Observation (Figure 7.1):

- Packet from `192.168.10.1` to `192.168.10.2`, Port 8888, captured with full payload (e.g., `2C CF 67 9B D9...`)
- Time delta: 1.9999 s (indicating regular periodic transmission)
- Protocol: IPv4 + UDP

7.1.4 Implications

This result demonstrates that monitor-only passive attacks using MitM via Active Tap are highly effective in AE environments. Even without injecting or modifying data, an attacker can:

- Reconstruct session behavior
- Analyze control messages
- Gather sensitive information like sensor values or control commands

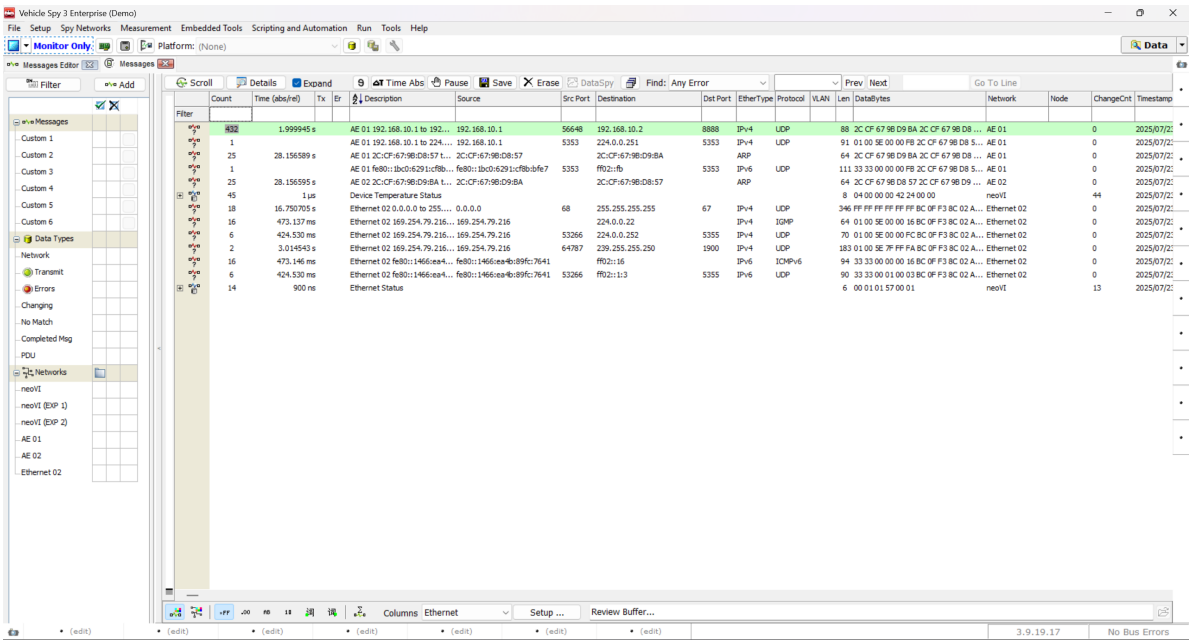

Figure 7.1: Monitoring The Data Between ECU 1 and ECU 2 Using Vehicle Spy 3 Message Viewer

Such monitoring can serve as the basis for more advanced attacks (e.g., Replay or Impersonation), highlighting the need for robust data confidentiality mechanisms such as encryption or message authentication in AE stacks.

The Monitor Attack via Active Tap MitM technique highlights a non-invasive yet powerful method for eavesdropping on AE communication. Its success underscores the inherent vulnerability of in-vehicle networks lacking encryption and access control.

7.2 DoS Attack using MitM via Active Tap

7.2.1 Overview

This section presents the results of a DoS attack executed using a MitM approach facilitated by an Active Tap. The objective of this experiment was to evaluate how vulnerable the in-vehicle AE communication is when exposed to high-volume flooding from an in-line compromised device.

The attack was implemented by injecting a continuous stream of custom IPv4 fragmented packets, tagged as `Flooding_tx`, into the AE network. This flooding aimed to overwhelm the network with excessive broadcast traffic, potentially causing message queue overflow, increased CPU load, or degraded communication reliability.

7.2.2 Experimental Setup

- **Attack Type:** DoS via Flooding
- **Attack Methodology:** MitM through Active Tap
- **Tap Device:** RAD-Galaxy in Individual Mode (Active Tap)
- **Packet Injection Tool:** Vehicle Spy 3
- **Source ECU:** ECU1 (`192.168.10.1`)
- **Flooding Destination:** Broadcast (`255.255.255.255`)
- **Flooding Message ID:** `Flooding_tx` (Custom defined)
- **Media Converters:** RAD-Moon for 100BASE-T1 to Ethernet bridging
- **Target Interface:** AE 01 (Receiving end of the flooded traffic)

RAD-Galaxy was used to both passively forward and actively inject the high-volume traffic. The attacker sent fragmented IPv4 messages to the broadcast address at a rapid rate, creating a deliberate overload scenario on the receiver or the switch.

7.2.3 Results and Observations

As shown in Figure 7.2, the `Flooding_tx` packets were observed to:

- Originate from IP: `192.168.10.1`
- Target IP: `255.255.255.255` (broadcast)
- Protocol: IPv4
- EtherType: Fragment
- Payload: Repeated hexadecimal patterns (e.g., `60 FC 70 00 02 0C...`)
- Transmission Rate: Every 60–100 milliseconds (approximately 10–16 packets per second)
- Total Messages: Over 30 packets captured in a brief time window

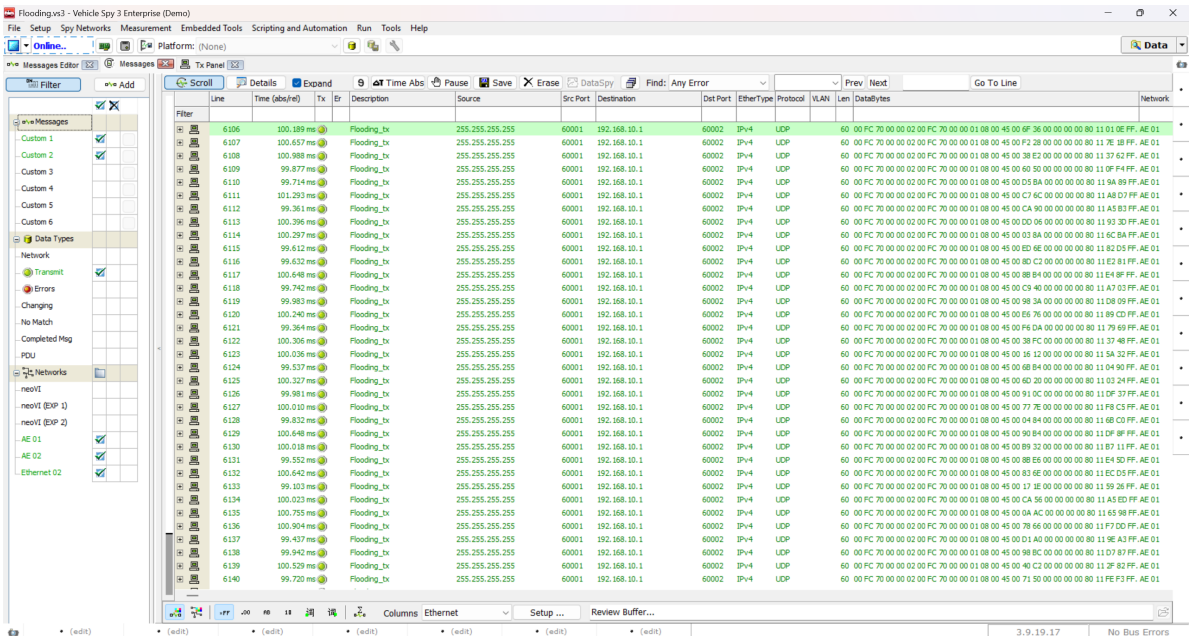

Figure 7.2: Flooding The Network From RAD-Galaxy and Observing Results Using Vehicle Spy 3 Message Viewer

Each packet was 60–64 bytes in size, and the repeated use of broadcast addressing ensured that all nodes on the subnet were forced to process the traffic. Despite appearing small, the high frequency and repetitive nature of these frames created considerable overhead for any receiving ECU or switch.

Key Point: Unlike traditional flooding with large payloads, this attack used fragmented packets with moderate size but high frequency - a stealthy and resource-intensive strategy.

7.2.4 Attack Impact

The flooding created by `Flooding_tx` has the following implications:

- **CPU Resource Drain:** ECUs or switches are forced to process unnecessary packets, consuming processing time and buffer space.
- **Queue Saturation:** Broadcast queues and message buffers can overflow, especially in low-power ECUs, resulting in message drops.
- **Timing Interference:** In time-critical systems (e.g., ADAS), even small delays can cause system faults or missed control cycles.
- **Network Congestion:** Legitimate packets may be delayed or dropped entirely, leading to partial or complete communication breakdown.

7.2.5 Summary

This experiment confirms that MitM-based active injection using `Flooding_tx` can simulate a realistic DoS attack in AE networks. By repeatedly injecting fragmented IPv4 packets at broadcast level, an attacker can effectively disrupt communication, drain resources, and destabilize critical systems.

7.3 Monitoring Attack using MitM via ARP Spoof

7.3.1 Overview

In this experiment, a Monitoring Attack was conducted using a MitM approach via ARP spoofing. The objective was to silently capture and analyze bidirectional communication between two legitimate ECUs without altering the transmitted data. This method leverages ARP's trust model, where the attacker ECU poisons the ARP tables of both victim ECUs, causing all packets to be routed through the attacker before reaching their intended destination.

In our AE test bench, this attack validates that an adversary positioned within the in-vehicle network can gain full visibility of sensitive sensor data, control messages, and operational parameters without triggering any communication errors detectable at the application level.

7.3.2 Experimental Configuration

Test Bench Setup:

- **ECU 1 (Victim):** Raspberry Pi 5 – IP: 192.168.10.1
- **ECU 2 (Victim):** Raspberry Pi 5 – IP: 192.168.10.2
- **Attacker ECU:** Raspberry Pi 5 running `arpspoof`
- **Switching & Conversion:** RAD-Jupiter AE switch with RAD-Moon media converters for 100BASE-T1 to standard Ethernet conversion
- **Monitoring Tool:** `tcpdump` and Python logging scripts running on attacker ECU

Attack Execution:

1. ARP spoofing initiated from attacker ECU using:

```
arpspoof -i eth0 -t 192.168.10.1 192.168.10.2
arpspoof -i eth0 -t 192.168.10.2 192.168.10.1
```

2. Attacker ECU began receiving traffic from both victim ECUs simultaneously.

3. Captured traffic was logged in real time without injecting any malicious packets or altering message content.

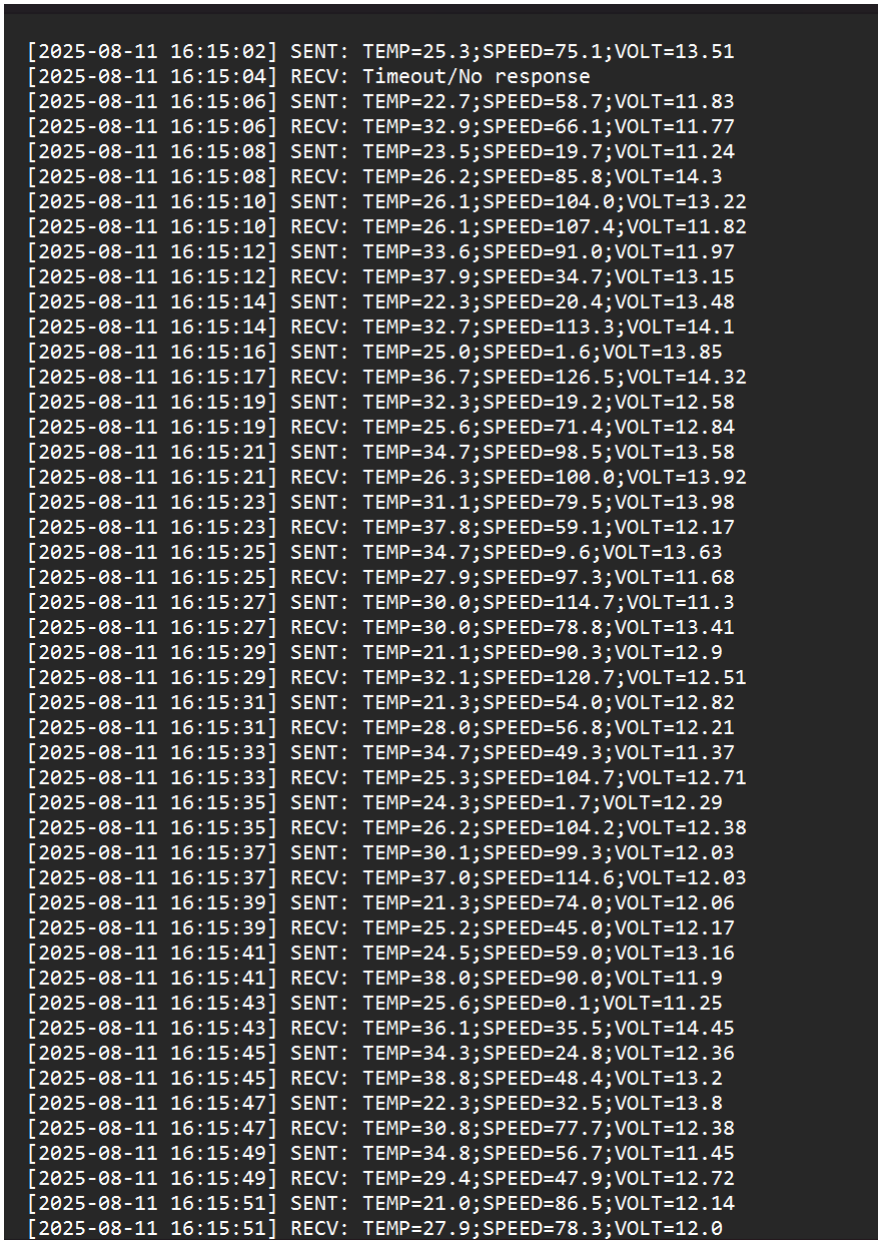

Figure 7.3: ECU 1 Bidirectional Communication With ECU 2

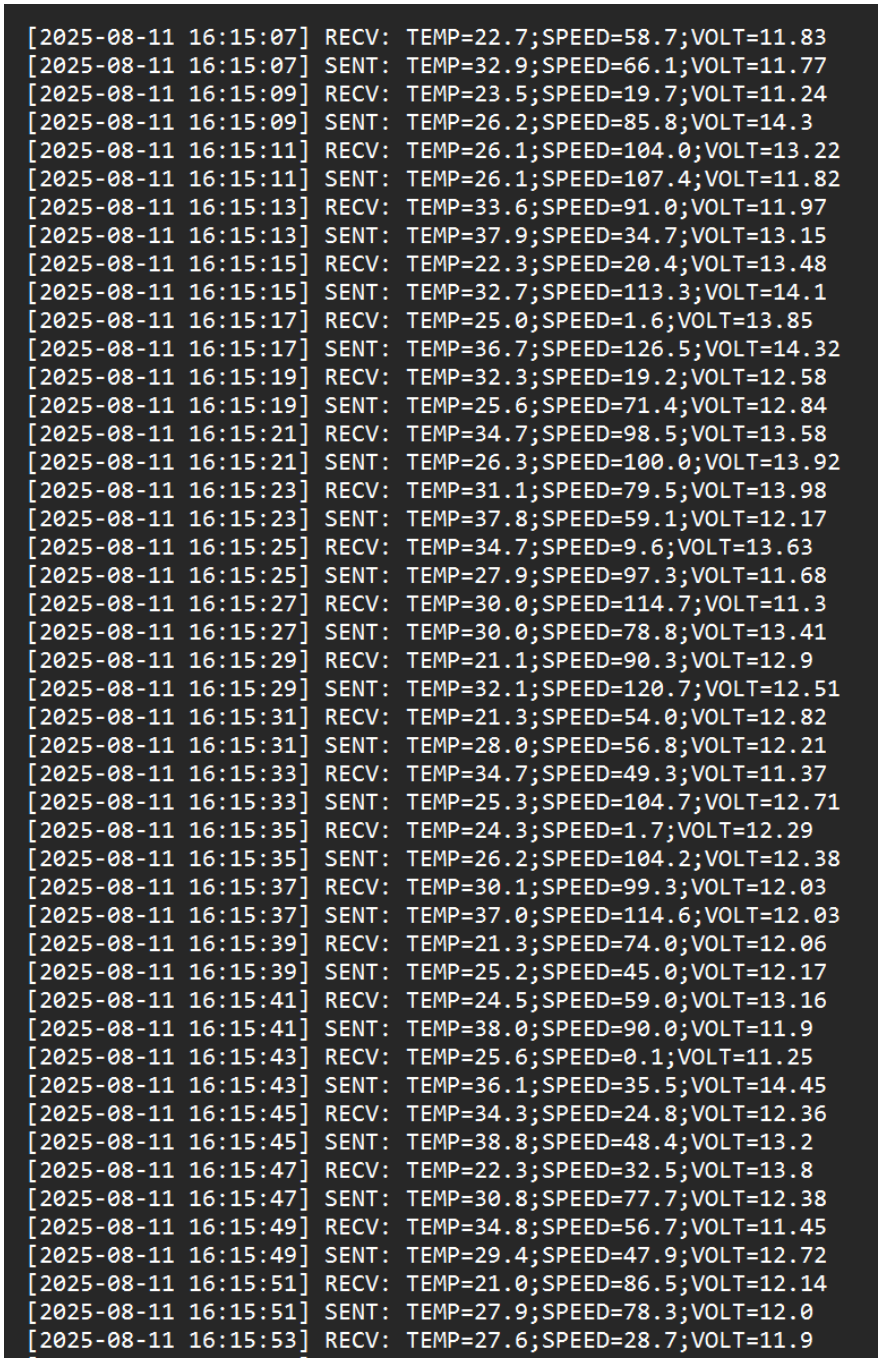

Figure 7.4: ECU 2 Bidirectional Communication With ECU 1

7.3.3 Observations and Results

The attacker ECU successfully intercepted all bidirectional communication between ECU 1 and ECU 2.

Analysis of the captured logs shows that:

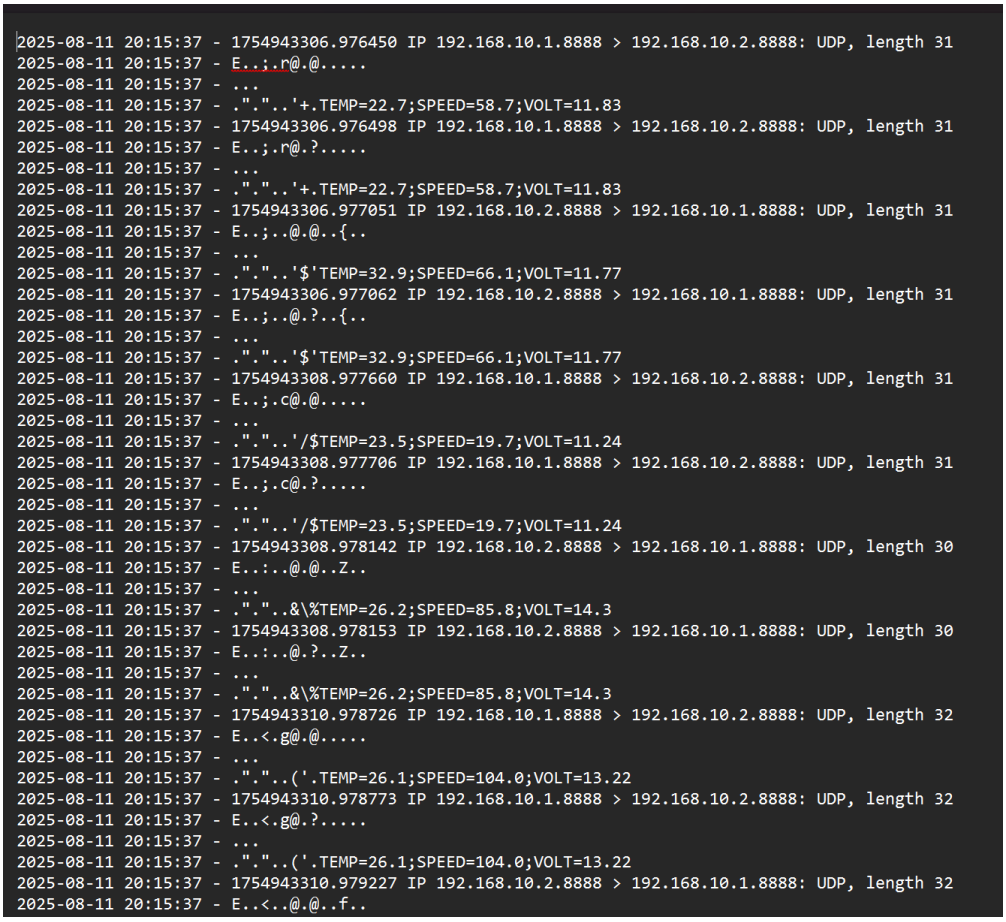

Figure 7.5: ECU 3 Acting As Attacker And Monitor the Data Between ECU 1 ECU 2

- The attacker received `SENT` and `RECV` packets from both ECUs in sequence, confirming full visibility.
- All application-level data remained unmodified, validating the "monitoring only" objective.
- The captured parameters included `TEMP`, `SPEED`, and `VOLT`, transmitted at sub-second intervals.

Example from ECU 1 logs (`ecu1_log_11_8`):

```
[2025-08-11 16:15:06] SENT: TEMP=22.7;SPEED=58.7;VOLT=11.83
[2025-08-11 16:15:06] RECV: TEMP=32.9;SPEED=66.1;VOLT=11.77
```

Example from ECU 2 logs (`ecu2_log_11_8`):

```
[2025-08-11 16:15:07] RECV: TEMP=22.7;SPEED=58.7;VOLT=11.83
[2025-08-11 16:15:07] SENT: TEMP=32.9;SPEED=66.1;VOLT=11.77
```

These entries confirm mirrored data flow, indicating that the attacker was in the direct communication path, capturing packets in both directions.

- **Figure 7.5** – `tcpdump` capture on attacker ECU showing bidirectional packet flow between ECU 1 and ECU 2.

The monitoring attack using ARP spoofing demonstrated that:

- A malicious ECU within the AE network can transparently observe all communications between two legitimate ECUs.
- The method requires no modification to the intercepted traffic, making detection by the communicating nodes highly unlikely.
- The data captured can provide valuable intelligence for subsequent targeted attacks such as message injection or replay.

7.4 Tamper Attack using MitM via ARP Spoof

7.4.1 Overview

In this experiment, a Tamper Attack was carried out using a MitM approach via ARP spoofing. Unlike the monitoring attack, where the adversary only observes data, this method actively intercepts and modifies in-transit messages before forwarding them to the intended recipient.

The attacker ECU performs ARP poisoning to insert itself between two legitimate ECUs, enabling interception of all bidirectional communication. A tampering script is then used to alter application-level parameters such as `TEMP`, `SPEED`, and `VOLT` before relaying the modified packets. This attack demonstrates the feasibility of injecting false operational data into an AE network without breaking the communication flow.

7.4.2 Experimental Configuration

Test Bench Setup:

- **ECU 1 (Victim):** Raspberry Pi 5 – IP: 192.168.10.1
- **ECU 2 (Victim):** Raspberry Pi 5 – IP: 192.168.10.2
- **Attacker ECU:** Raspberry Pi 5 running `arpspoof` and `tamper_and_forward_v2.py`

- **Switching & Conversion:** RAD-Jupiter AE switch with RAD-Moon media converters for 100BASE-T1 to standard Ethernet conversion
- **Monitoring & Tampering Tool:** Python script with packet parsing, modification, and forwarding logic

Attack Execution:

1. ARP spoofing initiated from attacker ECU:

```
arpspoof -i eth0 -t 192.168.10.1 192.168.10.2
arpspoof -i eth0 -t 192.168.10.2 192.168.10.1
```

2. Tampering script executed on attacker ECU with intentional packet delay:

```
sudo python3 tamper_and_forward_v2.py
```

3. Script intercepted sensor packets, altered values, and forwarded modified packets to the respective destination ECU.

7.4.3 Observations and Results

The attacker ECU successfully manipulated application data in transit. The logs show clear evidence of modified values for parameters such as temperature, speed, and voltage.
Tamper script output excerpt (Tamper.txt):

```
[TAMPER] 192.168.10.1 → modified → TEMP=42.7;SPEED=196.0;VOLT=23.7
[TAMPER] 192.168.10.2 → modified → TEMP=48.3;SPEED=186.3;VOLT=23.8
[TAMPER] 192.168.10.1 → modified → TEMP=38.4;SPEED=106.8;VOLT=18.4
[TAMPER] 192.168.10.2 → modified → TEMP=49.8;SPEED=194.2;VOLT=23.2
```

Comparison of ECU 2 log before and after tampering (ecu2_log.txt):

```
Before Tamper (Expected from ECU 1):
[2025-08-06 02:07:46] RECV: TEMP=28.3;SPEED=21.7;VOLT=13.45

After Tamper (Modified by Attacker):
[2025-08-06 02:14:14] RECV: TEMP=42.7;SPEED=196.0;VOLT=23.7
```

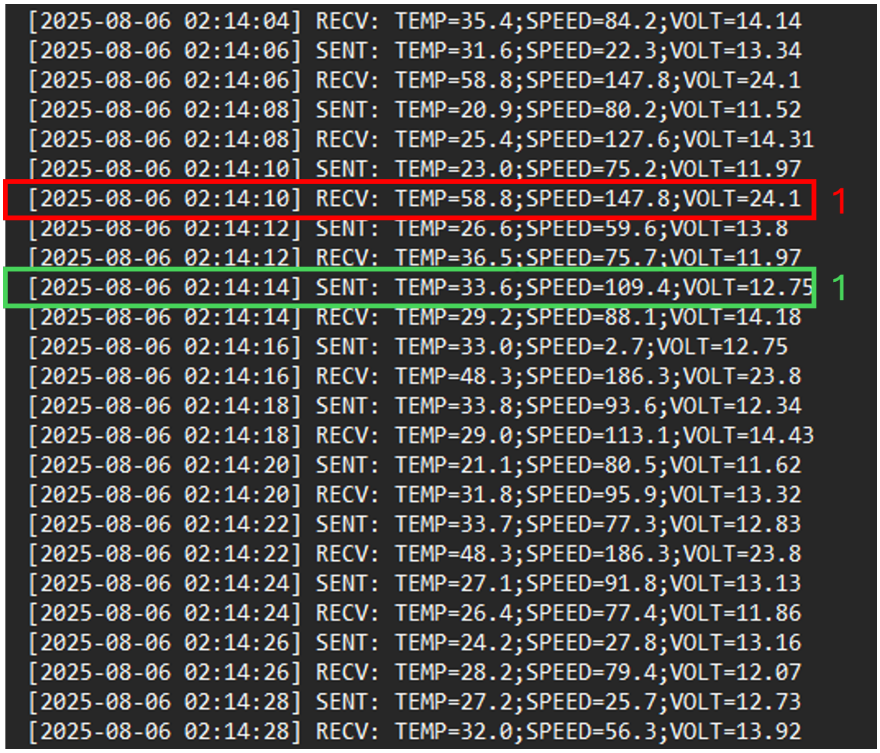

Figure 7.6: ECU 1 Bidirectional Communication With ECU 2 Under Tamper Attack

The modified data illustrates the capability of the attacker to inject abnormal values (e.g., unrealistic speed > 190 km/h and high voltage readings), potentially misleading vehicle control algorithms and diagnostic systems.

- **Figure 7.8:** Terminal output of the tamper script showing live modification of intercepted packets.
- **Figure 7.6:** ECU 2 log entries showing altered sensor data compared to original values.
- **Figure 7.7:** ECU 1 log entries reflecting injected abnormal readings.

7.4.4 Observation

The tamper attack using MitM via ARP spoofing demonstrated that:

- An attacker can not only monitor but actively alter in-transit data between ECUs.
- Tampered parameters can be set to unrealistic or unsafe values without triggering connection loss or visible communication errors.

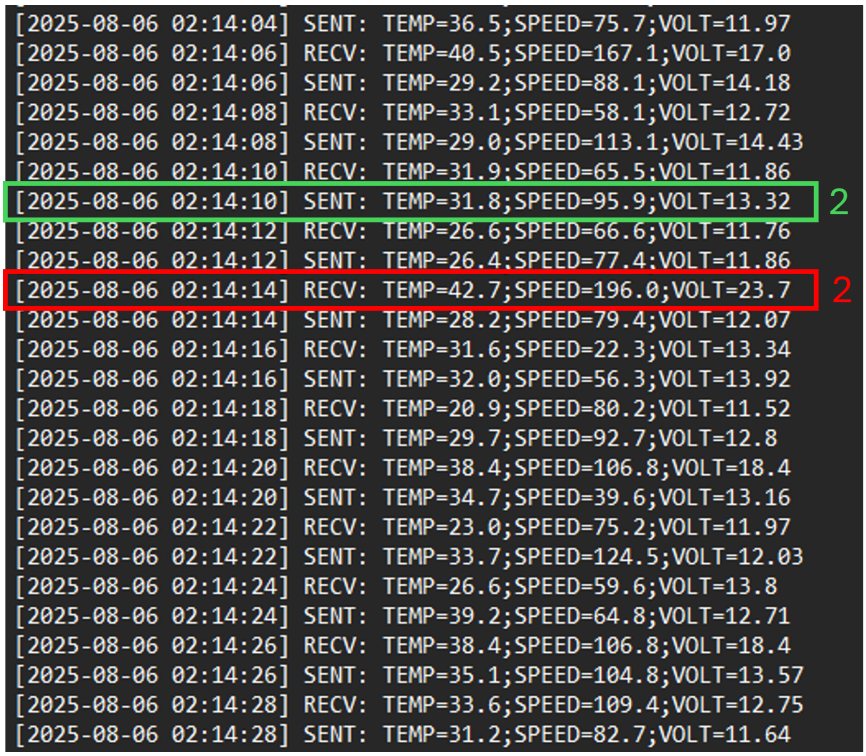

Figure 7.7: ECU 2 Bidirectional Communication With ECU 1 Under Tamper Attack

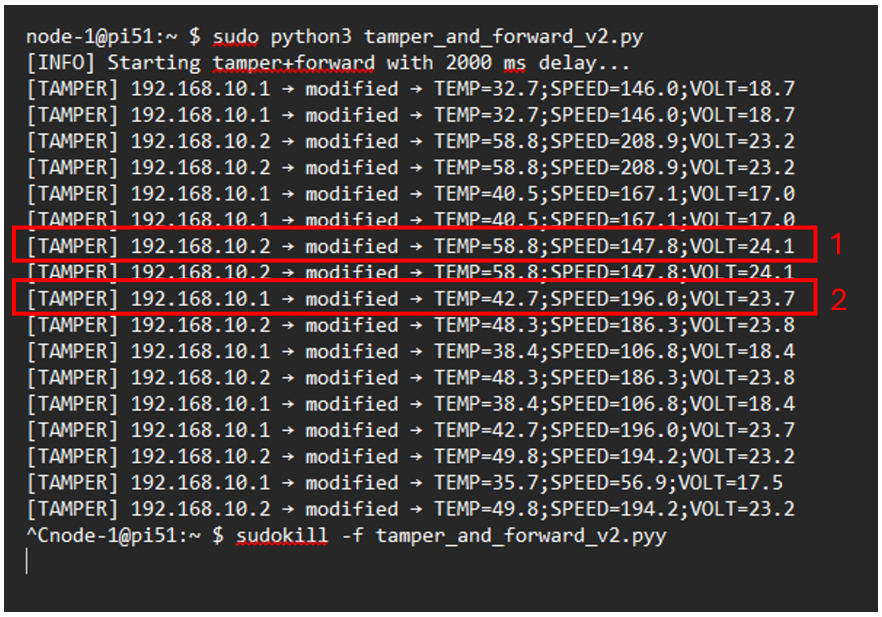

Figure 7.8: ECU 3 Acting As Attacker And Tampering the Data Between ECU 1 ECU 2

- Such an attack could be used to mislead ECUs into unsafe operational states or to cause incorrect decisions by safety-critical subsystems.

This chapter presented the qualitative results and observations derived from the execution of MitM, DoS, Monitoring, ARP spoofing, and tampering attacks on the AE test bench. The analysis revealed how each attack uniquely influenced network behavior, ranging from silent data interception and manipulation to severe communication disruption, demonstrating the absence of built-in resilience mechanisms in current AE deployments. These experimental findings highlighted recurring vulnerability patterns across different attack vectors and underscored the susceptibility of safety-critical communication to even relatively simple intrusion methods. Building on these insights, the next chapter concludes this research by synthesizing the key contributions, reflecting on the broader implications for automotive cybersecurity, and outlining directions for future work in securing Ethernet-based vehicular networks.

CHAPTER 8

Conclusion

This thesis has systematically investigated the security vulnerabilities of AE by focusing on the feasibility and impact of MitM attacks. Building on the taxonomy of cyber threats introduced earlier, the research selected two complementary methodologies active tapping and ARP spoofing to demonstrate realistic adversarial models targeting in-vehicle networks. These approaches enabled both hardware-level and protocol-level intrusions, offering a holistic perspective on how AE systems can be compromised in practice.

The experiments carried out on the constructed AE testbench validated that MitM attacks can achieve a wide spectrum of malicious objectives, from passive monitoring of critical ECU-to-ECU communication to active tampering with SOME/IP and DoIP payloads, and even DoS by selectively dropping or delaying traffic. The findings confirmed that AE's lack of built-in cryptographic protections, reliance on trust-based ARP operations, and dependence on deterministic TSN scheduling collectively expose it to significant security risks. Importantly, the results demonstrated that such attacks can be executed with modest resources and minimal detection, making them both feasible and stealthy in real-world vehicular environments.

A review of related works further reinforced these conclusions, showing that both industry and academia consistently recognize MitM as a high-severity threat for modern vehicles. While ARP spoofing exploits protocol weaknesses to achieve efficient intrusions, active tapping illustrates the consequences of direct inline interception with full control over traffic flows. Together, they represent two adversarial models that span both lightweight protocol exploitation and high-fidelity hardware compromise.

The research gap analysis highlighted that although numerous detection and prevention methods have been proposed in the literature, there is still limited empirical validation of these mechanisms under realistic AE testbeds. Furthermore, comparative studies across attack methodologies remain scarce, and little attention has been paid to the long-term cyber-physical consequences of MitM attacks in safety-critical systems such as ADAS or autonomous driving platforms.

By bridging theory and practice, this thesis contributes an experimental foundation for understanding the real-world implications of MitM attacks on AE. It underscores the urgent need for stronger link-layer authentication, time-synchronization protections, and intrusion detection systems tailored for vehicular Ethernet.

In conclusion, this work demonstrates that while AE enables the scalability and bandwidth required for next-generation vehicles, it also introduces critical vulnerabilities that cannot be ignored. Without proactive security integration, AE-based architectures remain exposed to stealthy and disruptive attacks capable of undermining both safety and reliability. The findings of this thesis therefore provide valuable insights not only for researchers but also for industry practitioners seeking to design more resilient in-vehicle communication systems.

CHAPTER 9

Future Work

While this thesis has focused on the experimental evaluation of cyber-physical attacks on AE, the work also highlights several open challenges and directions for future research. These opportunities for further investigation are crucial for advancing the security and resilience of next-generation IVNs. Based on the systematic review of existing literature, the following research directions are proposed:

9.1 Standardization of Security Frameworks

Current AE deployments lack a unified and standardized framework for authentication, encryption, and intrusion detection. Future work should aim to establish a comprehensive security architecture that can be seamlessly integrated across ECUs and vendors, ensuring consistency and interoperability in protection mechanisms.

9.2 Lightweight Cryptographic Solutions

Traditional cryptographic techniques, while effective, often introduce latency and resource overhead that are not ideal for real-time vehicular environments. Future research should explore lightweight and efficient cryptographic methods that balance security with performance, ensuring low-latency communication suitable for ADAS and autonomous driving applications.

9.3 Intrusion Detection and Prevention Systems

There is a pressing need for IDS/IPS specifically tailored for AE traffic patterns such as SOME/IP and DoIP. ML/DL based anomaly detection approaches represent a promising

direction for developing adaptive and intelligent mechanisms capable of identifying and mitigating attacks such as DoS, spoofing, and replay in real time.

9.4 Integration of Security with TSN

As AE increasingly adopts TSN to achieve deterministic communication, securing TSN protocols (e.g., 802.1AS, 802.1Qbv) remains an open challenge. Future work should investigate methods for embedding security primitives directly into TSN scheduling and synchronization mechanisms.

9.5 Validation Using Real-World Testbeds

Much of the existing research on AE security remains simulation-based. Future work should extend evaluations to real-world test benches using hardware platforms such as RAD-Moon, RAD-Galaxy, and RAD-Jupiter. Such physical setups would provide empirical insights into the practicality, overhead, and robustness of proposed countermeasures.

9.6 Security for OTA Updates

OTA software updates are increasingly critical in modern vehicles, yet they present new vulnerabilities when delivered over AE infrastructure. Future research should address secure OTA mechanisms that ensure integrity, confidentiality, and authenticity, preventing firmware manipulation or malicious payload injection.

9.7 Cross-Layer Security Approaches

Many existing countermeasures address individual OSI layers in isolation. A promising research direction is the design of cross-layer security frameworks that provide holistic protection across all layers of AE communication, from the physical to the application layer.

BIBLIOGRAPHY

[1] V. W. Ajin, L. D. Kumar, and J. Joy. Study of security and effectiveness of doip in vehicle networks. In *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*, pages 1–6. IEEE, 2016.

[2] Anonymous. Formal modeling and simulation of replay attacks in ttethernet-based automotive systems. *IEEE Access*, 10:1–12, 2022. Authors anonymized for peer review.

[3] Mohammad Ashjaei, Moris Behnam, Thomas Nolte, and et al. Time-sensitive networking in automotive embedded systems. *Journal of Systems Architecture*, 2021. Article number 102171.

[4] Multiple Authors. Study on secure communication mechanisms in automotive ethernet. *Journal of Communications*, 12(4):201–210, 2021. Paper ID: JOC3201-04.

[5] AUTOSAR. Specification of time synchronization (r22-11). Standard, 2022.

[6] AUTOSAR. AUTOSAR SOME/IP protocol specification (release r24-11). https://tinyurl.com/mr48kpp4, 2024.

[7] Omer Avatefipour and et al. State-of-the-art survey on in-vehicle network communication and security: CAN, LIN, MOST, FlexRay. *arXiv*, 2018.

[8] Krishnamoorthy Balakrishnan and Nishant Gupta. Performance of ieee 802.1as for automotive system using hardware timestamp. In *IEEE Conference on Industrial Electronics (excerpt, technical report)*, 2014.

[9] Pedro R. X. Carmo, Débora Campelo, and Paulo H. Ferreira. Machine learning-based intrusion detection system for Automotive Ethernet: Detecting cyberattacks with a low-cost platform. In *Proceedings of SBRC 2022*, pages 1–12, 2022.

[10] Hyeong Jun Cho, Boosun Jeon, and Huy Kang Kim. Anomaly detection of in-vehicle ethernet traffic based on service-level features. *Sensors*, 20(3):641, 2020.

[11] Wikipedia contributors. Flexray (overview and iso 17458 references). https://en.wikipedia.org/wiki/FlexRay, 2024.

[12] Álvaro da Luz, Pedro Ferreira, and Alysson Bessani. Multi-stage deep learning-based intrusion detection for automotive ethernet. *Journal of Intelligent & Fuzzy Systems*, 46(1):1–14, 2024.

[13] Luiz Felipe M. da Luz and et al. Multi-stage deep learning-based intrusion detection for Automotive Ethernet. In *Proceedings of a 2024 Elsevier venue (Computer Networks, in press)*, 2024. Early online.

[14] Marco De Vincenzi, Gianpiero Costantino, Ilaria Matteucci, Florian Fenzl, Christian Plappert, Roland Rieke, and Daniel Zelle. A systematic review on security attacks and countermeasures in Automotive Ethernet. *ACM Computing Surveys*, 56(6):1–38, 2024.

[15] A. B. C. Douss and et al. State-of-the-art survey of in-vehicle protocols and security with a focus on Automotive Ethernet. *Mathematical Biosciences and Engineering*, 20(9):15109–15142, 2023.

[16] Jie Du, Jie Fan, Yang Zheng, You Xie, and Wei Zhang. Security analysis and improvement of vehicle ethernet SOME/IP protocol. *Sensors*, 22(17):6527, 2022.

[17] Jun Du, Jing Tang, and Wei Feng. Security analysis and improvement of vehicle ethernet SOME/IP protocol. *Sensors*, 22(18):6792, 2022.

[18] Hesham El-Rewini, Karnam Sadatsharan, et al. Cybersecurity challenges in vehicles: A survey. *Vehicular Communications*, 23:100214, 2020.

[19] EWA Europe. Automotive doip cybersecurity analysis. Technical Article, 2025.

[20] Klaus Feichtenschlager. Analysis of multimedia networks for automotive: MOST. *Thesis/Report*, 2007.

[21] Amirmohammad Fotouhi et al. Assessing the impact of attacks on an automotive ethernet time synchronization testbed. In *Workshop/Preprint*, 2023.

[22] Edoardo Fraccaroli, Alessandro Biondi, and Giorgio C. Buttazzo. Timing predictability for SOME/IP-based service-oriented automotive systems. In *Proceedings of DATE 2023*, pages 1–6, 2023.

[23] Maher Gmiden, Sami Ayed, and Noureddine Boudriga. Experimental security analysis of automotive ethernet protocols: A hardware injection testbed for some/ip and doip. *Computers & Security*, 105:102–116, 2021.

[24] Thomas Grimm, Marcel Rumez, Reiner Kriesten, and Eric Sax. Anomaly detection in automotive ethernet using unsupervised learning. *Electronics*, 10(12):1412, 2021.

[25] Andreas (Ed.) Grzemba. MOST: The automotive multimedia network (2nd ed.). https://tinyurl.com/5ftwybuw, 2011.

[26] A. Hafeez, H. Malik, O. Avatefipour, P. Rongali, et al. Comparative study of can-bus and flexray protocols for in-vehicle communication. In *SAE Technical Paper 2017-01-0017*. SAE International, 2017.

[27] A. Hafeez, K. Topolovec, C. Zolo, and W. Sarwar. State of the art survey on comparison of can, flexray, lin protocol and simulation of lin protocol. In *SAE Technical Paper 2020-01-1293*. SAE International, 2020.

[28] Alexander Herold and Stefan Posselt. Anomaly detection for some/ip using complex event processing. In *Proceedings of the Embedded Security in Cars Conference (escar)*. escar, 2020.

[29] Nadine Herold, Stephan-A. Posselt, Oliver Hanka, and Georg Carle. Anomaly detection for some/ip using complex event processing. In *IEEE/IFIP NOMS 2016 Workshop on Security for Emerging Distributed Network Technologies (DISSECT)*, pages 1221–1226. IEEE, 2016.

[30] Vanlalruata Hnamte, Lalmuansanga Vanphawng, et al. An approach to efficient arp spoofing attacks detection within sdns. *ICT Express Plus*, 4:100015, 2024.

[31] IEEE. Ieee std 802.1as-2020: Timing and synchronization for time-sensitive applications. Standard, 2020.

[32] IEEE 1722 Working Group. IEEE Std 1722-2016: Layer 2 audio video transport protocol (AVTP), 2016.

[33] IEEE 802.1 Working Group. IEEE Std 802.1Qav-2009: Credit-based shaper. https://tinyurl.com/eun3ektt, 2009.

[34] IEEE 802.1 Working Group. IEEE Std 802.1Qbv-2015: Enhancements for scheduled traffic, 2015.

[35] IEEE 802.1 Working Group. IEEE Std 802.1Qbu-2016: Frame preemption, 2016.

[36] IEEE 802.1 Working Group. IEEE Std 802.1CB-2017: Frame replication and elimination for reliability, 2017.

[37] IEEE 802.1 Working Group. IEEE Std 802.1Qci-2017: Per-stream filtering and policing, 2017.

[38] IEEE 802.1 Working Group. IEEE Std 802.1AS-2020: Timing and synchronization for time-sensitive applications, 2020.

[39] IEEE 802.3 Working Group. IEEE Std 802.3bw-2015: 100BASE-T1 physical layer specifications and management parameters for automotive applications, 2015.

[40] IEEE 802.3 Working Group. IEEE Std 802.3bp-2016: 1000BASE-T1 ethernet over a single balanced twisted pair, 2016.

[41] IEEE 802.3 Working Group. IEEE Std 802.3-2018: IEEE standard for ethernet. https://standards.ieee.org/standard/802_3-2018, 2018.

[42] IEEE 802.3 Working Group. IEEE Std 802.3cg-2019: 10BASE-T1S and 10BASE-T1L, 2019.

[43] IEEE 802.3 Working Group. IEEE Std 802.3ch-2020: Multi-Gig automotive ethernet at 2.5/5/10 gbit/s, 2020.

[44] Vector Informatik. Time synchronization in automotive ethernet networks. Technical Article, 2016.

[45] Vector Informatik. Ethernet frame basics in automotive (Ethernet II and VLAN). https://elearning.vector.com/mod/page/view.php?id=157, 2021.

[46] National Instruments. Flexray automotive communication bus overview. https://tinyurl.com/2p9pcaue, 2025.

[47] Texas Instruments. Security in automotive networks: Ethernet and beyond. Application Note, 2025.

[48] Intrepid Control Systems. Intrepid control systems official website, 2025. Accessed: 2025-08-26.

[49] ISO. ISO 13400-2:2019: Road vehicles—diagnostics on IP (DoIP)—part 2: Transport protocol and network layer services. International Standard, 2019.

[50] ISO/SAE. ISO/SAE 21434:2021: Road vehicles—cybersecurity engineering. International Standard, 2021.

[51] Seonghoon Jeong, Boosun Jeon, Boheung Chung, and Huy Kang Kim. Convolutional neural network-based intrusion detection system for AVTP streams in Automotive Ethernet-based networks. *Vehicular Communications*, 29:100338, 2021.

[52] Bernd Jesse. Making gptp capable for secure time synchronization. In *IEEE-SA Ethernet & IP @ Automotive Technology Day (E&IP@ATD)*. IEEE, 2018.

[53] Minho Jo, Huy Kang Kim, Seonghoon Jeong, et al. A survey on security in automotive networks: Threats and countermeasures. *IEEE Access*, 9:23835–23859, 2021.

[54] Narayan Khatri, Rakesh Shrestha, and Seung Yeob Nam. Security issues with in-vehicle networks, and enhanced countermeasures based on blockchain. *Electronics*, 10(8):893, 2021.

[55] Cosmin-Vasile Kifor, Monica Ilea, and Mihai Suciu. Automotive cybersecurity: A survey on frameworks, standards, and practices. *Applied Sciences*, 14(7):3572, 2024.

[56] Timo Kiravuo, Niko Mäkitalo, and Teemu Santonen. A taxonomy of automotive ethernet threats. In *Proceedings of the 21st ACM Conference on Computer and Communications Security*, pages 1029–1036. ACM, 2016.

[57] Moez Krichen et al. Formal methods and validation techniques for ensuring trust in automotive software. *Information*, 14(12):666, 2023.

[58] Jan Lastinec and Ladislav Hudec. Approach to securing in-vehicle networks using ethernet-ip. In *Proceedings of the Colloquium for Information Systems Security Education (CISSE)*, pages 1–7, 2014.

[59] Seokjun Lee, Seungwon Lee, Huy Kang Kim, and Yongdae Kim. Protecting some/ip communication via authentication ticket. *Sensors*, 23(9):4376, 2023.

[60] Seongjae Lee, Junyeong Jeon, and Huy Kang Kim. Protecting SOME/IP communication via authentication ticket. *Sensors*, 23(14):6359, 2023.

[61] Yuekang Li, Hongxu Chen, Cen Zhang, Siyang Xiong, Chaoyi Liu, and Yi Wang. Ori: A greybox fuzzer for some/ip protocols in automotive ethernet. In *27th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2020.

[62] J. Lindberg. Security analysis of vehicle diagnostics using doip. Master's thesis, Chalmers University of Technology, Gothenburg, Sweden, 2011.

[63] Lucia Lo Bello. The case for ethernet in automotive communications. *ACM SIGBED Review*, 8(4):7–15, 2011.

[64] Lucia Lo Bello, Gaetano Patti, and Luca Leonardi. A perspective on ethernet in automotive communications—current status and future trends. *Applied Sciences*, 13(3):1278, 2023.

[65] Lucia Lo Bello and Wilfried Steiner. A perspective on IEEE time-sensitive networking for industrial communication and automation systems. *Proceedings of the IEEE*, 107(6):1094–1120, 2019.

[66] S. F. Lokman, A. T. Othman, and S. A. R. Abu-Bakar. Intrusion detection system for automotive Controller Area Network (CAN) bus: A review. *EURASIP Journal on Wireless Communications and Networking*, 2019(184), 2019.

[67] Xiang Luo, Hongyang Li, Kai Zheng, and Wei Zhang. Simulation-based security evaluation of tsn-enabled automotive ethernet backbones. In *Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 951–953. IEEE, 2019.

[68] Xiang Luo, Qiang Li, Wen Zhang, and Yu Wang. Security analysis of the tsn backbone architecture and anomaly detection. *Security and Communication Networks*, 2021:1–14, 2021.

[69] Xiang Luo, Lei Zhao, Wei Zhang, Kai Zheng, and Hongyang Li. Security analysis of in-vehicle time-sensitive networking using stride threat modeling. *Electronics*, 10(8):893, 2021.

[70] Yi Luo, Lin Yang, Wei Zhang, and Peng Li. A multi-layer intrusion detection system for some/ip-based automotive ethernet. *Sensors*, 23(9):4376, 2023.

[71] Bin Ma, Shichun Yang, Zheng Zuo, Bosong Zou, Yaoguang Cao, Xiaoyu Yan, Sida Zhou, and Jichong Li. An authentication and secure communication scheme for in-vehicle networks based on some/ip. *Sensors*, 22(647), 2022.

[72] X. Ma et al. Cryptographic defense and identity exploitation in some/ip for automotive ethernet. *Sensors*, 22(647):1–16, 2022.

[73] Akihiro Matsubayashi and Shohei Koyama. Attacks against uds on doip and countermeasures. In *Proceedings of the 7th ACM Computer Science in Cars Symposium (CSCS '23)*. ACM, 2023.

[74] Charlie Miller and Chris Valasek. Adventures in automotive networks and control units. Technical report, IOActive, 2013. Technical report demonstrating message injection and tampering attacks on production vehicles.

[75] F. R. P. Mohammed. Intrusion detection for cyber-physical attacks on ros-based collaborative robots. Master's thesis, University of Michigan-Dearborn, 2025.

[76] Patrick Moldenhauer and Ryan McCall. Automotive ethernet cyberattack defense in ground vehicles. In *NDIA GVSETS & APBI*. NDIA, 2022.

[77] Michael Mueller and et al. Secure service discovery for SOME/IP using DNSSEC and DANE. *arXiv*, 2023.

[78] Shravan Murlidharan, Varsha Ravulakole, Jyothi Karnati, and Hafiz Malik. Battery management system: Threat modeling, vulnerability analysis, and cybersecurity strategy. *IEEE Access*, 13:37198–37220, 2025.

[79] David Pannell and 802.1 TSN contributors. Choosing the right TSN tools to meet a bounded latency. In *IEEE 802.1 Public Whitepaper*, 2021.

[80] Jae Hong Park, Hyun Kim, and Sangjin Lee. Security analysis of diagnostics over ip (doip): Vulnerabilities and countermeasures. In *Proceedings of the 2020 International Conference on Information Networking (ICOIN)*, pages 234–239. IEEE, 2020.

[81] Hector Pérez Tijero, Jorge Beleno, and Juan Rios. Application-level evaluation of ieee 802.1as synchronized systems. *ACM Transactions on Embedded Computing Systems*, 23(5), 2024.

[82] Jonathan Petit and Steven E. Shladover. Potential cyberattacks on automated vehicles. In *IEEE Transactions on Intelligent Transportation Systems*, volume 16, pages 546–556. IEEE, 2015.

[83] Sravan Sai Puppala and Stefan Karlsson. On secure data-in-transit in modern and future in-vehicle networks. *Master's Thesis, Uppsala University*, 2024.

[84] Ritu Siwach Rathore, Rakesh Kumar, et al. In-vehicle communication cyber security: Challenges and solutions. *Energies*, 15(16):5900, 2022.

[85] Niklas Reusch, Paul Pop, and Silviu S. Craciunas. Work-in-progress: Safe and secure configuration synthesis for tsn using constraint programming. In *2020 IEEE Real-Time Systems Symposium (RTSS)*, pages 386–390. IEEE, 2020.

[86] Julian Seifert, Miroslav Malek, and Martin Müller. Security gateway for automotive networks. In *2014 International Conference on Embedded Security in Cars (ESCAR)*, pages 1–9. escar Europe, 2014. Focuses on CAN and FlexRay tampering defenses.

[87] Abdul Kareem Shaik. Assessment of cyber-physical vulnerabilities of industrial robotic sensing systems. Master's thesis, University of Michigan-Dearborn, 2023.

[88] Abdul Kareem Shaik, Alireza Mohammadi, and Hafiz Malik. A systematic review of sensor vulnerabilities and cyber-physical threats in industrial robotic systems. *IET Cyber-Physical Systems: Theory & Applications*, 10(1):e70023, 2025.

[89] Spirent Communications. Testing automotive ethernet phy – critical new test challenges facing developers of connected vehicles. 2020.

[90] I. Studnia et al. Survey on security threats and protection mechanisms in embedded automotive networks. In *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*, pages 1–12. IEEE, 2013.

[91] Ammar Talic. Security analysis of ethernet in cars. Master's thesis, KTH Royal Institute of Technology, 2017. Master's Thesis.

[92] Marvell Technology. Macsec in the automotive ethernet network. Whitepaper, 2020.

[93] Mehmet Topsakal and Serkan Cevher. Cyber security for ieee 802.1 time sensitive in-vehicle networking: Recent advances and impact analysis of dos attacks. *Dokuz Eylül University Faculty of Engineering Journal (DEUFMD)*, 26(76):105–118, 2024.

[94] UNECE. Un regulation no. 155: Cyber security and cyber security management system. https://tinyurl.com/k24u6j6d, 2021.

[95] Felix Wächter and Christoph Krauß. Security analysis of automotive protocols. In *4th ACM Computer Science in Cars Symposium*. ACM, 2021.

[96] Felix Wächter, Hans Löhr, and Christoph Krauß. Analysis of the diagnostics over ip (doip) protocol for security vulnerabilities. In *Proceedings of the 6th ACM Computer Science in Cars Symposium (CSCS '22)*. ACM, 2022.

[97] Simon Woo, Hyunmin Jo, Insik Kwon, and Heasoo Lee. Security analysis of vehicular networks: Attacks and countermeasures. *IEEE Communications Magazine*, 53(12):52–59, 2015.

[98] Li Xu, Qian Wang, and Rui Chen. Replay attack vulnerability analysis in automotive ethernet testbeds. *Journal of Computers*, 32(7):88–102, 2021.

[99] Jinghua Yu, Stefan Wagner, Bowen Wang, and Feng Luo. A systematic mapping study on security countermeasures of in-vehicle communication systems. *arXiv preprint arXiv:2105.00183*, 2021.

[100] Daniel Zelle, Gianpiero Costantino, Ilaria Matteucci, and Roland Rieke. Analyzing and securing SOME/IP automotive services with formal methods and practical attacks. In *Proceedings of the 2021 Annual Computer Security Applications Conference (ACSAC)*, pages 113–124, 2021.

[101] Daniel Zelle, Alexander Koch, Roland Rieke, and Florian Fenzl. Impersonation attacks on some/ip service discovery using vsomeip and canoe. In *Proceedings of the 16th International Conference on Availability, Reliability and Security (ARES 2021)*. ACM, 2021.

[102] Cen Zhang, Yuekang Li, Hongxu Chen, Chaoyi Liu, and Yi Wang. A retrospective analysis of fuzz security evaluations for some/ip (survey excerpt). In *Applied Computing and Intelligence*, 2024.

[103] Tianyu Zhang, Tianyu Gong, Xin S. Hu, Qiang Deng, Michael Lemmon, and Song Han. Time-sensitive networking for industrial automation: Challenges, opportunities, and directions. *arXiv*, 2023.

[104] Wei Zhang, Ming Li, and Yong Zhou. Cross-state replay attacks in can and automotive ethernet: A multi-vehicle study. *Electronics*, 11(3):465, 2021.

[105] Wei Zhang, Ming Li, and Yong Zhou. Hijacking vulnerabilities in some/ip service discovery for automotive ethernet. *Electronics*, 10(6):721, 2021.

[106] Rui Zhao, Guihe Qin, Ying Lyu, and Jie Yan. Security-aware scheduling for ttethernet-based real-time automotive systems. *IEEE Access*, 7:85971–85983, 2019.

[107] Zheng Zuo, Shichun Yang, Bin Ma, Bosong Zou, Yaoguang Cao, Qiangwei Li, Sida Zhou, and Jichong Li. Design of a canfd to some/ip gateway considering security for in-vehicle networks. *Sensors*, 21(23):7917, 2021.