HAW HAMBURG

MASTERTHESIS
Wilhelm Schumacher

Methoden zur Anomalieerkennung in TSN basierten Fahrzeugnetzwerken

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Wilhelm Schumacher

Methoden zur Anomalieerkennung in TSN basierten Fahrzeugnetzwerken

Masterarbeit eingereicht im Rahmen der Masterprüfung
im Studiengang *Master of Science Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Franz Korf
Zweitgutachter: Prof. Dr. Stephan Pareigis

Eingereicht am: 03. April 2023

Wilhelm Schumacher

Thema der Arbeit

Methoden zur Anomalieerkennung in TSN basierten Fahrzeugnetzwerken

Stichworte

Anomalieerkennung, Time-Sensitive Networking, TSN, Fahrzeugnetzwerk

Kurzzusammenfassung

Zukünftige Automobile involvieren eine Vielzahl an Steuergeräten für neue, hochmoderne Funktionen wie autonomes Fahren/Fahrerassistenz, Komfort und Infotainment. Zur Umsetzung dieser Funktionen werden die etablierten Automotive Kommunikationstechnologien, bestehend aus heterogenen Feldbussen, im Netzwerk Schritt für Schritt durch Ethernet-Netzwerke ersetzt. Dadurch können Anforderungen an immer weiter steigende Bandbreite und Kabelgewicht/Kabelkomplexität besser erfüllt werden. Durch die zusätzliche Integration von Time-Sensitive Networking in das zukünftige Fahrzeugnetzwerk können Anforderungen an garantierte Bandbreite, eine maximale Latenz und kein Paketverlust durch Buffer Overflow in einer Queue im Netzwerk realisiert werden. Das Verschicken von sicherheitskritischen Traffic über die gleiche Leitung wie nicht sicherheitskritischen Traffic und der vermehrten Öffnung nach Außen erfordern angepasste Sicherheitskonzepte zum Schutz der Steuergeräte. In dieser Arbeit werden Methoden der Anomalieerkennung im Rahmen eines TSN basierten Fahrzeugnetzwerks untersucht. Der aktuelle Stand der Anomalieerkennung im Fahrzeugnetzwerk insbesondere in Verbindung mit TSN wird dargestellt. Kategorisierungen von Algorithmen zur Anomalieerkennung aus der Literatur werden analysiert und Kriterien zum Vergleich von Algorithmen definiert. Im nächsten Schritt wird ein Prototyp TSN Netzwerk in Hardware aufgebaut. In verschiedenen Angriffsszenarien werden ausgewählte Methoden zur Anomalieerkennung evaluiert und an den Ergebnissen miteinander verglichen.

Wilhelm Schumacher

Title of Thesis

Methods for anomaly detection in TSN based vehicle networks

Keywords

Anomaly Detection, Time-Sensitive Networking, TSN, In-Car Networks, Automotive Security

Abstract

Future automobiles will involve a wide range of ECUs for new, state-of-the-art functions such as autonomous driving/driver assistance, comfort and infotainment. To implement these functions, the established automotive communication technologies, consisting of heterogeneous fieldbuses are being replaced step by step by Ethernet networks. This allows requirements for ever-increasing bandwidth and cable weight/cable complexity to be better met. Through the additional integration of Time-Sensitive Networking (TSN) into the future vehicle network, requirements for guaranteed bandwidth, maximum latency and no packet loss due to buffer overflow can be realized. Safety-critical traffic over the same line as non-safety-critical traffic and the increasing number of interfaces to the outside require adapted security concepts to protect the control units. In this work, anomaly detection methods are investigated in the context of a TSN based vehicular network. The current state of anomaly detection in the vehicle network, especially in connection with TSN, is presented. Categorizations of anomaly detection algorithms from the literature are analyzed and criteria for comparing algorithms are defined. The next step is to build a prototype TSN network in hardware. Selected methods for anomaly detection are evaluated in different attack scenarios and compared with each other on the basis of the results.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungen

BEF BorderEnlargementFactor

CAN Controller Area Network

CBS Credit Based Shaper

E/E Elektrik und Elektronik

ECU Electronic Control Unit

EE Elliptic Envelope

FN False Negative

FP False Positive

FPGA Field Programmable Gate Array

GCL Gate Control List

gPTP generalized Precision Time Protocol

HBO Histogram-based Outlier Detection

IO In-Output

IoT Internet of Things

IP Internet Protocol

LIN Local Interconnect Network

MOST Media Oriented System Transport

PCP Priority Code Point

PTP Precision Time Protocol

QoS Quality of Service

SDN Software-Defined Networking

SVM Support Vector Machines

TDMA Time Division Multiple Access

TP True Positive

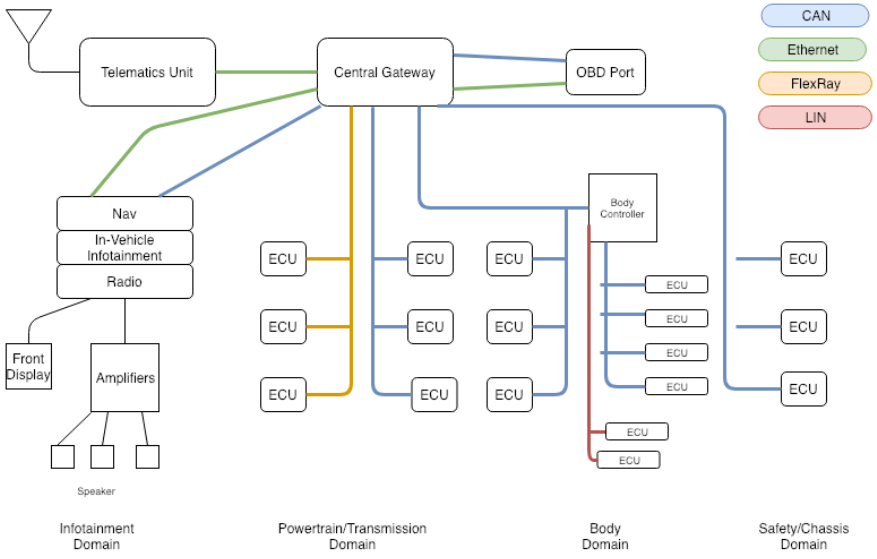
TSN Time-Sensitive Networking

V2V Vehicle to Vehicle

V2X Vehicle to Everything

1 Einleitung

In der näheren Zukunft erwarten Kunden eine Vielzahl von neuen Features in ihrem nächsten Auto zur Verbesserung der Leistungsfähigkeit, des Komforts und der Sicherheit. Das aktuelle interne Fahrzeugnetzwerk besteht aus einer Vielzahl von Sensoren und Steuergeräten (Electronic Control Unit (ECU)), die jeweils verantwortlich für bestimmte Teilfunktionen des Fahrzeuges sind [vgl. 58, Seite 2]. Als Kommunikationsmedien zwischen den Steuergeräten, die über das gesamte Fahrzeug verteilt sind, werden momentan vor allem Systembusse wie Controller Area Network (CAN), Local Interconnect Network (LIN), FlexRay, Media Oriented System Transport (MOST) und für einzelne Verbindungen Ethernet eingesetzt (siehe Abbildung 1.1).



Quelle: Basierend auf
https://www.iot-now.com/2017/02/27/59018-securing-automotive-air-updates [42]

Abbildung 1.1: Die Abbildung stellt eine typische domänenbasierte Netzwerkarchitektur von einem aktuellen Oberklasse Auto dar.

Die Weiterentwicklung hin zum autonomen Fahren, inklusive V2X (Vehicle to Everything) und Fahrerassistenzsysteme lässt die Anforderungen an die Bandbreite und Kabelkomplexität immer weiter ansteigen [vgl. 19, Seite 1] und erfordert durchgehend mit der Außenwelt verbundene Telematiksteuereinheiten sowie eine Vielzahl an zusätzlicher Sensorik wie z.B. Radarsensoren und Surround-View Kameras [vgl. 40, Seite 1]. Um diese Anforderungen besser erfüllen zu können, werden die heterogenen Feldbussysteme Schritt für Schritt durch Ethernet-Netzwerke ersetzt, hinzu einer zonalen Netzwerkarchitektur. In einer zonalen Netzwerkarchitektur wird die selbe Ethernet-Verbindung von mehreren Domänen/Funktionen gleichzeitg verwendet und zeitkritischer Traffic benötigt Quality of Service (QoS) Garantien.
Durch die zusätzliche Integration von Time-Sensitive Networking (TSN) in das Fahrzeugnetzwerk können QoS Garantien an garantierte Bandbreite, eine maximale Latenz und kein Paketverlust durch Buffer Overflow in einer Queue im Netzwerk erreicht werden [26].
Gleichzeitig ermöglicht die neue Netzwerkarchitektur und die Öffnung des internen Fahrzeugnetzwerkes die Durchführung einer Vielzahl neuer Angriffe auf das Auto. Dadurch entsteht eine erhöhte Verwundbarkeit der Informationssicherheit (Integrität, Vertraulichkeit, Verfügbarkeit) im internen Netzwerk des Fahrzeuges. Zum Schutz des Autos vor diesen Angriffen müssen neue Sicherheitskonzepte und - mechanismen eingeführt werden. Ein Bestandteil des zukünftigen Sicherheitskonzepts des Autos könnte die Entdeckung von Angriffen durch die Anomalieerkennung sein. Unterschiedliche Verfahren zur Anomalieerkennung werden bereits in anderen Domänen wie zum Beispiel bei der Erkennung von Kreditkartenbetrug, im medizinischen Bereich, für Fehlererkennungen im Raumschiff oder klassisch für Intrusion Detection Systeme, die Angriffe auf Netzwerke entdecken sollen, erfolgreich eingesetzt. Einige Algorithmen wurden dabei speziell für diese bestimmten Anwendungsbereiche entwickelt, abhängig von der Art der Daten, der Verfügbarkeit von gelabelten Trainingsdaten, den Typen von Anomalien und dem gewünschten Outputformat [vgl. 20, Seite 3-4].
Das Netzwerk des Fahrzeugs ermöglicht die Durchführung der Anomalieerkennung auf unterschiedlichen Ebenen (Layern) des OSI-Referenzmodells. Zum Beispiel betrachten Gmiden et al. [31] die Erkennung von Anomalien innerhalb des CAN-Bus-Systems auf der Sicherungsschicht (Layer 2). Van Wyk et al. [61] führen die Anomalieerkennung auf der Anwendungsschicht (Layer 7) mit den Daten aus der Sensorik durch. Weiterhin könnte sich der Einsatz von Anomalieerkennung im Auto lohnen, da die Verarbeitung auf Grund der Menge an Daten automatisiert durchgeführt werden muss und die Nachrichtenmuster genug Ähnlichkeiten und Zusammenhänge aufweisen, um sie von anormalen Daten

zu unterscheiden. Trotz allem werden bei der Anomalieerkennung auch Daten fälschlicherweise als Anomalien eingestuft (False Positive) und erkannte Anomalien erfordern entsprechende Gegenmaßnahmen. Deswegen stellt die Anomalieerkennung nur einen Teil eines gut funktionierenden Sicherheitskonzeptes dar und muss durch weitere Elemente wie zum Beispiel SDN (Software-defined Networking), Cyber Defense Center in der Cloud oder V2X Application-Level Gateways ergänzt werden.

Diese Arbeit untersucht Methoden der Anomalieerkennung im Rahmen eines TSN basierten Fahrzeugnetzwerks. Dazu wird der aktuelle Stand der Anomalieerkennung im Fahrzeugnetzwerk insbesondere in der Verbindung mit TSN aus der Literatur hervorgehoben. Anschließend werden Ansätze zur Kategorisierung von Algorithmen der Anomalieerkennung herausgearbeitet und Kriterien zum Vergleichen der Algorithmen definiert.
Zur weiteren Analyse wird ein Prototyp TSN Netzwerk in Hardware entwickelt. Anschließend werden verschiedene Angriffsszenarien im TSN Netzwerk durchgeführt und mehrere Algorithmen zur Anomalieerkennung an ihren Ergebnissen evaluiert. Abschließend werden die Ergebnisse diskutiert und mögliche Auswirkungen von TSN auf die Anomalieerkennung herausgearbeitet.

Die folgende Arbeit ist in sieben weitere Kapitel unterteilt. Die Ausarbeitung startet mit dem Kapitel **2 Grundlagen**, das fundamentale Konzepte zur Anomalieerkennung erläutert und die Funktionsweise von TSN sowie die wichtigsten TSN Standards hervorhebt. Des Weiteren wird ein Überblick über Verfahren zur Anomalieerkennung, die in dieser Arbeit verwendet werden, gegeben. Das Kapitel **3 Istanalyse** analysiert den Istzustand der Anomalieerkennung im Fahrzeug, indem der aktuelle Stand der Forschung hinsichtlich der Anomalieerkennung sowie der Anomalieerkennung im TSN Netzwerk dargestellt wird. Daraufhin folgt das Kapitel **4 Konzept**. Das Kapitel beschreibt Ansätze zur Kategorisierung von Algorithmen zur Anomalieerkennung und definiert Kriterien für einen Vergleich. Das Kapitel **5 Beschreibung des Prototyp TSN Netzwerks** erläutert den Aufbau des Prototyp TSN Netzwerkes, stellt den Nachrichtenverkehr im Netzwerk dar und beschreibt die technische Durchführung der Anomalieerkennung. Kapitel **6 Durchführung und Ergebnisse der Angriffsszenarien** betrachtet den Vergleich von sechs Algorithmen in verschiedenen Angriffsszenarien innerhalb des TSN Netzwerkes. Das Kapitel umfasst dabei die Versuchsdurchführung und die Ergebnisse. Die Evaluation der Ergebnisse und eine weitere Diskussion folgt im Kapitel **7 Evaluation der Algorithmen und Erkenntnisse**. Abschließend fasst das Kapitel **8 Fazit und Ausblick** die

wichtigsten Punkte der Arbeit zusammen, betrachtet noch offene Punkte und gibt einen Ausblick auf mögliche weiterführende Arbeiten.

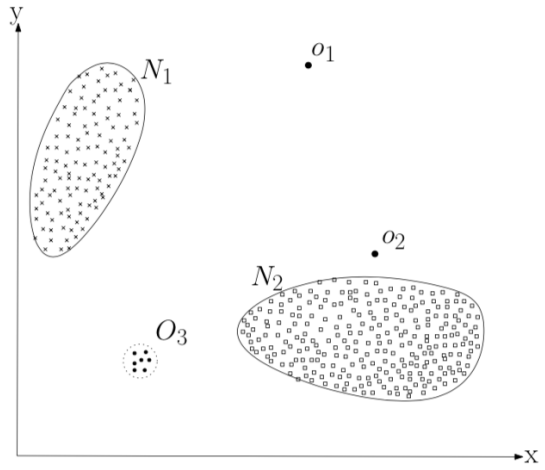
2 Grundlagen

Das Kapitel Grundlagen erklärt die fundamentalen Konzepte von Technologien, die für das weitere Verständnis der Arbeit notwendig sind. Zuerst beschreibt **2.1** wichtige Aspekte zur Anomalieerkennung und hebt die Algorithmen, die in dieser Arbeit verwendet werden, hervor. Der zweite Abschnitt **2.2** erläutert die Funktion eines TSN Services und die Sammlung von Standards, auf denen der Service basiert.

2.1 Anomalieerkennung

Die Anomalieerkennung hat als Aufgabe unregelmäßige Muster innerhalb von normalen Daten festzustellen [20]. Diese Unregelmäßigkeiten entsprechen dabei nicht dem normalen Verhalten des Systems, treten seltener auf und unterscheiden sich signifikant vom Rest der Daten. Sie werden hauptsächlich als **Anomalien** oder **Outlier** (Ausreißer) bezeichnet. Anomalieerkennung wird bereits in einer Vielzahl von unterschiedlichen Anwendungsbereichen eingesetzt. Zum Beispiel bei dem Erkennen von Kreditkartenbetrug, zum Entdecken von Tumoren in MRT-Bilder (Magnetresonanztomographie), im Cyber Security Bereich, dem Erkennen von Cheatern in Videospielen [49] oder für die Fehlererkennung bei sicherheitskritischen Systemen [28]. In vielen dieser Anwendungsbereiche spielt die Identifikation von Anomalien eine große Rolle, denn sie ermöglicht es kritische Situationen zu erkennen und daraufhin die notwendigen Gegenmaßnahmen zu veranlassen. Zum Beispiel können ungewöhnliche Nachrichtenmuster in einem Computernetzwerk darauf hinweisen, dass von einem gehackten Computer aus, sensitive Daten in das Netzwerk an einen unautorisierten Host geschickt werden oder anomales Verhalten in einem Sensor eines Raumschiffes könnte auf Fehler in einer Komponente hinweisen.

Ein konkretes Beispiel für eine Anomalie ist in Abbildung **2.1** dargestellt. Hier können innerhalb eines zweidimensionalen Datensatzs sowohl normale Bereiche (N1 und N2) als



Quelle: https://dl.acm.org/doi/10.1145/1541880.1541882 [vgl. 20, Seite 2]

Abbildung 2.1: Die Abbildung stellt die normalen Datenbereiche N1 und N2 den anomalen Beobachtungen O1,O2 und O3 gegenüber.

auch Anomalien (O1, O2 und O3), die weiter von den restlichen Daten entfernt liegen, erkannt werden. Häufig ist es aber schwierig, präzise Grenzen zwischen den normalen und anomalen Daten, wie in Abbildung **2.1**, zu definieren und stellt eine der großen Herausforderungen der Anomalieerkennung dar. Zum Beispiel können Anomalien, die dicht an der Grenze eines normalen Datenbereiches liegen, nur sehr schwierig als Anomalie oder normaler Datensatz eingeordnet werden. Beim Einordnen der Datensätze wird zwischen **False Positive** und **False Negative** unterschieden. Ein False Positive liegt vor, wenn ein Datensatz zu Unrecht als Ausreißer eingestuft wird (falscher Alarm) und ein False Negative liegt vor, wenn ein Datensatz als normal eingestuft wird, obwohl es sich dabei um einen Ausreißer handelt. Ein gutes System zur Anomalieerkennung zeichnet sich dadurch aus, dass es sowohl wenige bis zu keine False Positive erkennt und trotzdem keine Ausreißer übersieht.

2.1.1 Inputdaten in der Anomalieerkennung

Abhängig vom Anwendungsbereich kann sich die Art der Inputdaten deutlich voneinander unterscheiden. Im Allgemeinen sind die Inputdaten erstmal eine Sammlung von Daten (bezeichnet als Objekte, Vektoren, Beobachtungen, Punkte etc.), die wiederum aus einer Menge von Attributen (bezeichnet als Variablen, Felder, Feature etc.) bestehen

[vgl. 20, Seite 6-7]. Ein einzelnes Attribut stammt dabei aus einer von drei Kategorien **binär**, **kategorisch** oder **kontinuierlich**. Wenn die Daten nur aus einem Attribut bestehen, werden sie als **Univariate** bezeichnet und wenn die Daten aus mehreren Attributen bestehen als **Multivariate**. Je nachdem welche Art und Anzahl an Attributen innerhalb der Daten vorhanden sind, können andere Algorithmen zur Anomalieerkennung angewendet werden oder die Daten müssen vorab entsprechend angepasst werden.

2.1.2 Typen von Anomalien

Eine Anomalie kann in eine von drei Kategorien [30] eingeordnet werden. Die erste und einfachste Kategorie wird als **Punktanomalie** bezeichnet. Bei einer Punktanomalie wird ein einziger Datenpunkt in Anbetracht zu den restlichen Daten als Anomalie eingestuft. Ein Beispiel für eine Punktanomalie könnte die Höhe des ausgegebenen Betrages bei Bezahlung mit Kreditkarte sein. Wenn über einen längeren Zeitraum nur sehr kleine Beträge bezahlt wurden und dann ein im Vergleich zu diesen Beträgen sehr großer Betrag auftritt, handelt es sich um eine Punktanomalie. In der zweiten Kategorie **kollektive Anomalien** verhält sich eine Reihe von Datenpunkten abweichend vom Rest der Daten. Dabei sind die individuellen Datenpunkte aus der kollektiven Anomalie nicht unbedingt anormal, sondern nur deren gemeinsames Auftreten. Ein Beispiel für eine kollektive Anomalie ist das Signal eines Elektrokardiogramms [vgl. 20, Seite 9]. Wenn das Signal über einen längeren Zeitraum stabil bleibt und keine steilen Anstiege enthält wie in den anderen Intervallen, handelt es sich um eine kollektive Anomalie. Die dritte Kategorie von Anomalien sind die **kontextuellen Anomalien**. Bei einer kontextuellen Anomalie wird ein Wert nur in einem spezifischen Kontext als Anomalie betrachtet. Zum Beispiel könnte eine kontextuelle Anomalie in Zeitreihen von Temperaturen auftreten. Ein sehr niedriger Temperaturwert im Winter wäre nicht ungewöhnlich, aber im Sommer wäre es eine kontextuelle Anomalie.

2.1.3 Lernmethoden der Anomalieerkennung

Weitere wichtige Aspekte der Anomalieerkennung sind die drei unterschiedlichen Lernmethoden [17]. Beim **überwachten Lernen** erhält der Algorithmus einen Datensatz, der die Datenpunkte bereits als normal oder anormal markiert hat. Auf Basis dieses Datensatzes wird das System trainiert, um anschließend möglichst zielsicher vorauszusagen,

ob ein neuer Datenpunkt eine Anomalie darstellt oder nicht. Dagegen wird beim **semi-überwachten Lernen** nur auf Grundlage von Eingabedaten aus der normalen Klasse bestimmt, ob ein neuer Datenpunkt anormal ist oder nicht. Beim **unüberwachten Lernen** lernt der Algorithmus selbständig Muster und Zusammenhänge zu erkennen, ohne dass ein bearbeiteter Datensatz vorgegeben wird. Diese Methode nimmt an, dass normale Instanzen die am häufigsten auftretenden Muster sind. Welche Methode angewendet werden kann, hängt hautsächlich davon ab, ob gelabelte Trainingsdaten oder nur ungelabelte Trainingsdaten zur Verfügung stehen.

2.1.4 Algorithmen der Anomalieerkennung

Der folgende Abschnitt erläutert sechs Methoden zur Anomalieerkennung, die in späteren Teilen der Arbeit verwendet werden. Diese sechs Algorithmen wurden ausgewählt, um im späteren Vergleich Repräsentanten aus unterschiedlichen Kategorien von Algorithmen zur Anomalieerkennung zu haben. Insgesamt gibt es eine Vielzahl weiterer Algorithmen zur Anomalieerkennung, die in verschiedenen Kategorien eingeordnet werden können. In der Literatur existieren verschiedene Ansätze zur Kategorisierung dieser Algorithmen [17] [20] [29] . Zum Beispiel unterteilen Chandola et al. [20] Algorithmen zur Anomalieerkennung in die sechs Kategorien Statistical, Classification Based, Clustering Based, Nearest Neighbor Based, Information Theoretic und Spectral. Abhängig von der Kategorie bieten die Algorithmen unterschiedliche Vor- und Nachteile bei der Erkennung von Algorithmen wie etwa Unterschiede in der Anwendbarkeit, Rechenkomplexität oder Präzision der Einstufung.

Isolation Forest [24] basiert auf dem Prinzip des Isolierens von anormalen Daten und ist der Kategorie der Klassifikationsalgorithmen zuzuordnen. Der Algorithmus erstellt mehrere unterschiedliche Entscheidungsbäume (decision trees), die zusammen einen „Forest“ (Menge an Bäumen) bilden. Ein einzelner Baum wird aufgebaut, indem bis zur Kompletierung immer wieder ein zufälliger Trennwert aus einem zufälligen Feature ausgewählt wird. Der Baum ist komplettiert bis entweder jeder Datenpunkt isoliert dargestellt wird oder eine durchschnittliche Tiefe erreicht wird. Anomalien werden erkannt, da diese in der Regel in wenigen Schritten isoliert werden können.

Support Vector Machines (SVM) [47] basiert auf dem Prinzip des Trennens von zwei verschiedenen Klassen voneinander und ist der Kategorie der Klassifikationsalgorithmen zuzuordnen. In seiner Grundform bildet der Algorithmus durch das „Maximum Margin Hyperplane“ Verfahren eine Hyperebene zur Trennung der Daten. Das Verfahren be-

stimmt die Trennlinie (Hyperplane) so, dass der Abstand zu den Punkten aus beiden Mengen maximal ist. Zur Anomalieerkennung mit SVM wird eine leicht modifizierte Variante mit One-Class SVM [50] benutzt. One-Class SVM trennt eine Klasse von Daten ab, indem das Volumen eines Hyperballs um die Trainingsdaten minimiert wird.

Elliptic Envelope [35] beruht auf der Modellierung einer elliptischen Grenze um den Großteil der Daten und ist der Kategorie der parametrisierten statistischen Algorithmen zuzuordnen. Bei den parametrisierten statistischen Algorithmen ist die Verteilungsfunktion der Daten schon bekannt und die Parameter werden durch die gegebenen normalen Daten eingestellt. Die elliptische Grenze wird mit Hilfe einer hochdimensionalen Normal- oder Gauß-Verteilung erstellt und alle Daten außerhalb dieser Grenze werden als Anomalien eingestuft.

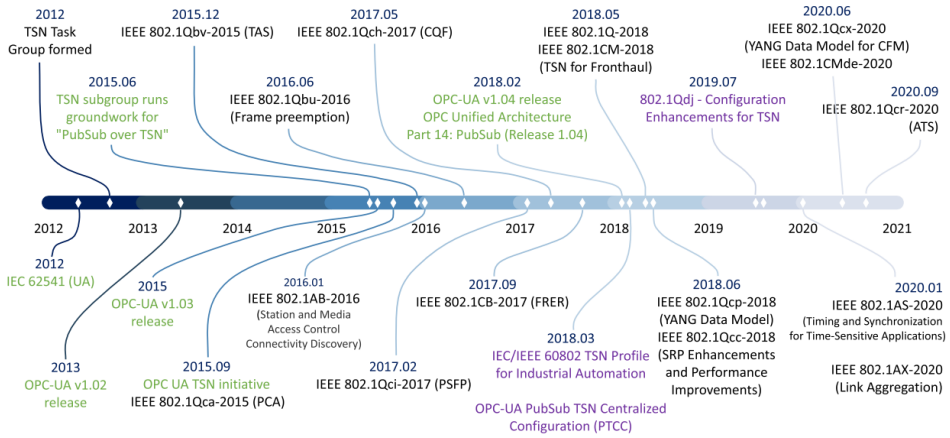
Histogram-based Outlier Detection [32] basiert auf der Erkennung von Anomalien mit Hilfe von Histogrammen und ist der Kategorie der unparametrisierten statistischen Algorithmen zuzuordnen. Unparametrisierte statistische Algorithmen verwenden statistische Modelle ohne Parameter. Bei der Anomalieerkennung mit Histogrammen wird für jedes einzelne Feature ein Histogramm erstellt und die einzelnen Histogramme werden in Behälter unterteilt. Jeder Behälter erhält durch die Menge an Daten in diesem Behälter eine bestimmte Höhe, die auch eine Wahrscheinlichkeitsangabe darstellt. Inwiefern neue Daten Ausreißer sind, unterscheidet der Algorithmus in Relation zu den Behältern.

Mean Shift [22] basiert auf der Bildung von Clustern über das Sliding-Window-Verfahren und ist ein Beispiel für die Kategorie der Clustering-Algorithmen. Ausgehend von mehreren zufälligen Startpunkten iteriert der Algorithmus durch Verschieben des Cluster-Mittelpunktes in Bereiche mit größerer Dichte bis zur Konvergenz des Algorithmus. Nach Abschluss des Verschiebens werden Cluster-Mittelpunkte mit identischer Position zusammengefasst und die übriggebliebenen Mittelpunkte bilden die finalen Cluster. Der Mean Shift Algorithmus bestimmt die Anzahl der Cluster selbst.

K-Means [39] hat als Ziel eine vorher festgelegte Anzahl an Clustern zu bilden, in die gleiche Datenpunkte eingeordnet werden können und ist der Kategorie der Clustering-Algorithmen zuzuordnen. In mehreren Iterationen werden Datenpunkte, ausgehend von zufälligen Startpunkten, wiederholt einem Clusterzentrum zugeordnet und anschließend wird das Clusterzentrum jedes Mal verschoben. Dieses Vorgehen wird so lange wiederholt bis eine bestimmte Anzahl von Iterationen erreicht worden ist oder bis keine Änderungen zwischen den Zentren der Iterationen mehr auftreten.

2.2 Time-Sensitive Networking

TSN ist ein zusätzlicher Service in einem Best-Effort-Netzwerk, der es Anwendungen ermöglicht, einen Vertrag mit dem Netzwerk einzugehen [26]. Dieser Vertrag limitiert die Anwendung auf eine maximale Bandbreite und im Gegenzug erhält die Anwendung dafür eine garantierte Bandbreite, eine maximale Latenz und kein Paketverlust durch Buffer Overflow in einer Queue im Netzwerk. Das exakte Verhalten von TSN wird in einer Reihe von Standards von der Time-Sensitive Networking (TSN) Task Group (TG) [13] als Teil der 802.1 Working Group (WG) festgesetzt. Seit der Gründung der Task Group im Jahre 2012 ist die Standardisierung ein fortlaufender Prozess, der immer noch andauert und weiterhin kontinuierlich durch neue Spezifikationen ergänzt wird. Abbildung **2.2** zeigt in einer Zeitlinie die wichtigsten TSN Standardisierungen und verdeutlicht die Vielzahl an Standards, die TSN zugeordnet werden können.



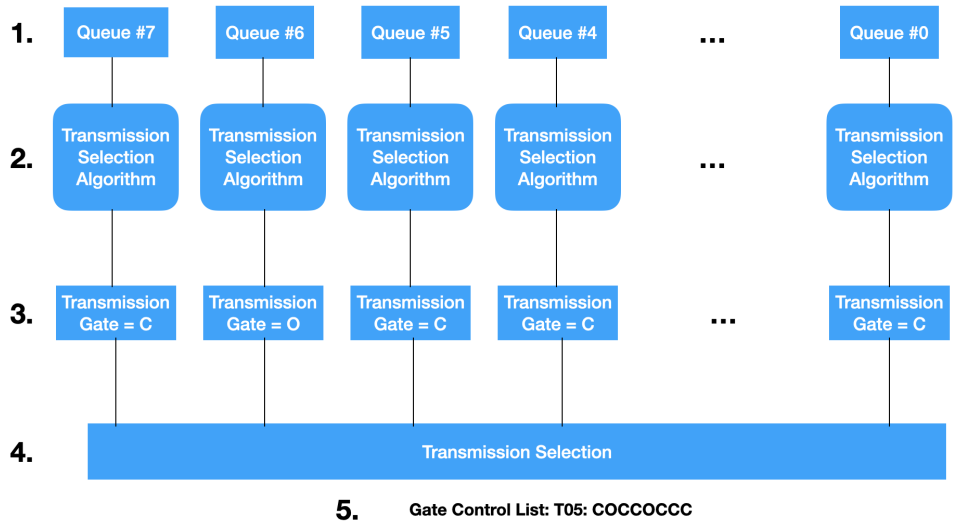
Quelle: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9576720 [55]

Abbildung 2.2: Die Abbildung stellt die Entwicklung der TSN Standardisierungen im zeitlichen Kontext dar.

TSN wird vor allem in industriellen Kontrolleinrichtungen und Kommunikationsnetzwerken, die harte Echtzeitanforderungen an den Netzwerkverkehr stellen, eingesetzt. Weiterhin wird TSN vor allem in Verbindung mit der Ethernet Technologie entwickelt und ermöglicht so eine einfache Integration von TSN in bestehende Best-Effort-Netzwerke auf Basis von Ethernet [55]. Die folgenden Abschnitte heben die maßgebenden Einzelstandards von TSN, die zum Teil auch im weiteren Verlauf der Arbeit verwendet werden, hervor.

2.2.1 IEEE 802.1Qbv – Enhancements for Scheduled Traffic

Der IEEE 802.1Qbv Standard definiert, wie ein Switch die Verarbeitung zeitgesteuerten Netzwerkverkehrs in mehreren Queues an einem Port mit Hilfe eines rotierenden Schedules (Zeitplans) vornimmt [19][26]. Abbildung **2.3** beschreibt den Mechanismus von IEEE 802.1Qbv auf einem Ausgangsport eines Ethernet Switches in mehreren Schritten. Im ersten Schritt (siehe 1) werden die Frames anhand vom VLAN-Tagging nach PCP Prioritäten auf mehrere Queues aufgeteilt. In Schritt 2 (siehe 2) entscheidet ein Transmission Selection Algorithm, ob ein Frame für die Übertragung ausgewählt wird. Eine Übertragung ist aber nur dann möglich, wenn das entsprechende Transmission Gate (siehe 3/4) im Zustand offen ist. Eine Gate Control List definiert mehrere Zustandsfolgen bestehend aus geschlossen (c) und offen (o) für jedes einzelne Gate (siehe 5). Durch das dynamische Öffnen und Schließen der Gates können zeitkritische Ethernet Frames sofort bei Ankunft übertragen werden, wenn der Schedule zu diesem Zeitpunkt das entsprechende Gate öffnet und alle anderen Gates schließt.



Quelle: Basierend auf https://www.nxp.com/docs/en/white-paper/QBVSOLUTIONSWPA4.pdf [19]

Abbildung 2.3: Die Abbildung stellt die Paketauswahl von IEEE 802.1Qbv mit Hilfe von Transmission Gates dar.

Wenn die Übertragung eines Frames am Ende eines Zeitfensters über das Zeitfenster hinaus andauert, könnte damit ein Frame aus dem nächsten Zeitfenster verzögert werden [59]. So könnte ein Best Effort Frame mit niedriger Priorität die Übertragung eines si-

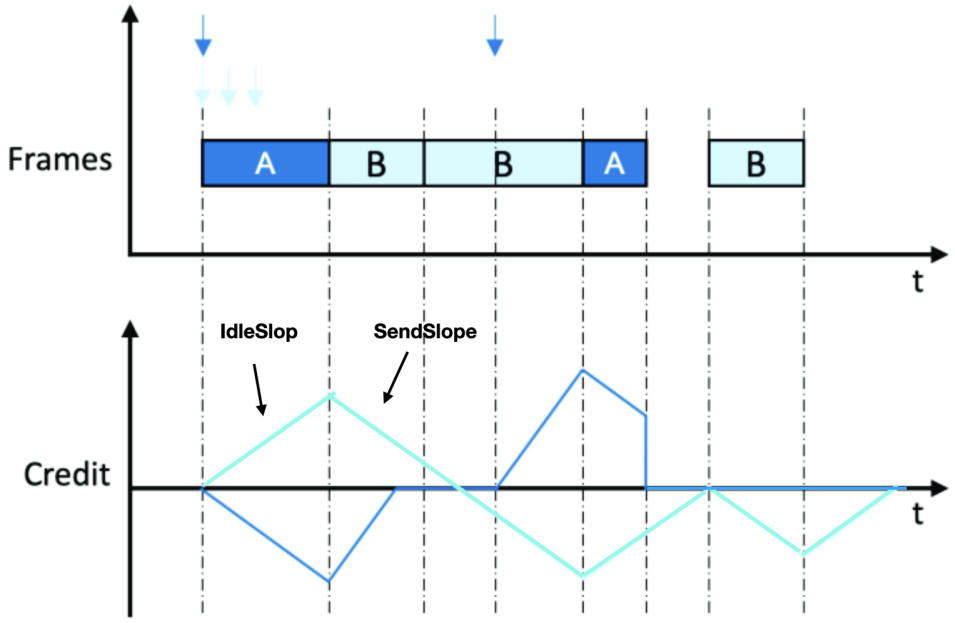
cherheitskritischen Scheduled Traffic Frames verzögern. Um dieses Problem zu vermeiden, definiert TSN, dass die Übertragung eines neuen Frames nur startet, wenn die Übertragung vor dem Schließen des Gates abgeschlossen werden kann. Dazu existiert ein Guard Band am Ende eines Zeitfensters, in dem kein weiterer Frame mehr übertragen werden kann. Ein Guard Band ist das Zeitintervall, dass für die Übertragung eines maximal großen Ethernet Frames benötigt wird.

2.2.2 IEEE 802.1AS – Clock Synchronization

Der Standard IEEE 802.1AS [2] Clock Synchronization ist eine speziell für TSN spezifizierte Version des Precision Time Protocol (PTP) aus dem Standard IEEE 1588 [6]. Er ist verantwortlich für die Zeitsynchronisation zwischen den Teilnehmern eines TSN Netzwerkes [19]. Einige andere TSN Standards wie zum Beispiel IEEE 802.1Qbv (siehe 2.2.1) setzen eine Zeitsynchronisation mit Hilfe von IEEE 802.1AS im Netzwerk voraus. Durch das gemeinsame Zeitverständnis im Netzwerk können zum Beispiel die Gates aus der Gate Control Liste so koordiniert werden, dass der zeitsynchronisierte Echtzeitverkehr ohne zusätzliche Verzögerung vor dem restlichen Netzwerkverkehr den Switch durchlaufen kann.

2.2.3 IEEE 802.1Qav – Credit Based Shaper

IEEE 802.1Qav [4] spezifiziert die Nutzung eines Credit Based Shapers. Ursprünglich wurde der Credit Based Shaper im Rahmen von AVB [3] für zeitkritischen Netzwerkverkehr aus dem Bereich Audio und Video eingeführt. Der Einsatz vom Credit Based Shaper ermöglicht das Glätten von Bursts im Netzwerk, damit die Empfangsgeräte nicht überlastet werden. Der Credit Based Shaper [34] ordnet jeder Queue einen Credit zu und der Credit wird verbraucht, sobald ein Frame dieser Queue verschickt wird. Wenn der Credit einer Queue einen Wert kleiner als 0 erreicht, ist das Verschicken weiterer Frames erstmal nicht mehr möglich. Abhängig von dem Parameter IdleSlope füllt sich der Credit einer Queue mit der Zeit wieder bis zum Wert 0 auf oder steigt als Ausgleich sogar darüber hinaus, wenn gerade ein anderes Paket versendet wird. Der Parameter SendSlope definiert den Gradienten für das Dekrementieren für die Dauer des Sendens.



Quelle: Basierend auf https://www.researchgate.net/figure/Example-of-the-operation-of-the-Credit-Based-Shaper-reproduced-as-in-27_fig1_341886210 [62]

Abbildung 2.4: Die Abbildung zeigt die Funktionsweise eines Credit Based Shapers, indem der Verlauf des Credits simultan zu dem Verschicken von Frames aus den beiden Queues A (dunkelblau) und B (hellblau) dargestellt wird.

In Abbildung **2.4** wird beispielhaft die Funktionsweise eines Credit Based Shapers anhand des zeitlichen Verlaufs von zwei Credit Linien gezeigt. Im oberen Teil der Abbildung ist zu erkennen, dass zwei verschiedene Queues A (dunkelblau) und B (hellblau) zum Start Pakete versenden möchten. Da A eine höhere Priorität als B besitzt, wird zuerst das Paket aus der Queue A versendet. Gleichzeitig zum Versenden von Frame A sinkt der Credit von A mit dem SendSlope Parameter unter den Wert 0, bis das Paket vollständig verschickt wurde. Währenddessen erhält die Queue B zusätzlichen Credit, der mit dem IdleSlope Parameter ansteigt. Nachdem der Frame A verschickt wurde und der Credit von A unter 0 liegt, wird ein Frame von B verschickt. Nun steigt der Credit von A wieder Richtung dem Wert 0 an und der Credit von B sinkt wieder. Der Credit von A erreicht im weiteren Verlauf der Abbildung wieder den Wert 0 und bleibt solange auf 0, bis ein weiteres Paket eintrifft, dass nicht sofort gesendet werden kann, weil zu dem Zeitpunkt noch ein Paket von B zu Ende übertragen wird. Anschließend muss B warten bis der Credit wieder 0 erreicht, um das dritte Paket zu senden.

2.2.4 IEEE 802.1Qbu und IEEE Std 802.3br – Frame Preemption

Die TSN Standards IEEE 802.1Qbu [15] und IEEE Std 802.3br [14] beschreiben einen Mechanismus zur Unterbrechung eines konkurrierenden Frames. Das Ziel dieser Standards ist es, die Weiterleitungsverzögerung eines zeitkritischen Frames zu reduzieren [48]. Die Frame Preemption erfordert eine Hardware-Unterstützung und definiert die zwei MAC Interfaces „express MAC Interface“ und „preemptable MAC Interface“. Jedes Ethernet Frame wird zu einem der zwei Interfaces zugeordnet und die express Frames sind in der Lage, das Senden von preemptable MAC Interface Frames zu unterbrechen.

2.2.5 IEEE 802.1CB – Frame Replication and Elimination for Reliability

Zum Schutz gegen Fehler und Ausfälle, wie beispielsweise Gerätedefekte, im Netzwerk spezifiziert die TSN Task Group den Standard IEEE 802.1CB [7]. Die essentielen Features dieses Standards sind das Hinzufügen von Sequenznummern [26] zu jedem Paket eines Flows und das einfache oder mehrfache Duplizieren von Paketen. Die duplizierten Pakete, die den Empfänger über einen anderen Pfad erreichen sollen, können, wenn ein identisches Paket den Empfänger bereits erreicht hat, entdeckt und aus dem Netzwerk entfernt werden.

2.2.6 IEEE 802.1Qci – Per Stream Filtering and Policing

Der IEEE Standard 802.1Qci [8] bietet die Möglichkeit, Frames an den Eingangsports basierend auf Ankunftszeiten, Raten und Bandbreite zu filtern [40]. Dadurch erhalten die Geräte im Netzwerk mehr Schutz gegenüber Angriffen, Fehlfunktionen oder übermäßiger Bandbreitennutzung.

3 Istanalyse

Das Kapitel Istanalyse hebt den aktuellen Stand der Informationssicherheit und Anomalieerkennung im Fahrzeugnetzwerk hervor. Zuerst spezifiziert **3.1** den Begriff der Informationssicherheit und führt wichtige Basisanforderungen auf. Anschließend betrachtet **3.2** Schnittstellen des Autos zur Außenwelt, gefolgt von einer Einordnung verschiedener Angriffe in Kapitel **3.3** und einem Blick auf Gegenmaßnahmen im Kapitel **3.4**. Das Kapitel **3.5** wendet den Begriff der Informationssicherheit auf das Fahrzeugnetzwerk an. Zum Abschluss blicken die Kapitel **3.6** und **3.7** auf den aktuellen Stand der Anomalieerkennung und Anomalieerkennung in Verbindung mit TSN im Fahrzeugnetzwerk. Kapitel **3.8** fasst aktuelle offene Forschungsaspekte zusammen und legt den Schwerpunkt dieser Arbeit fest.

3.1 Informationssicherheit

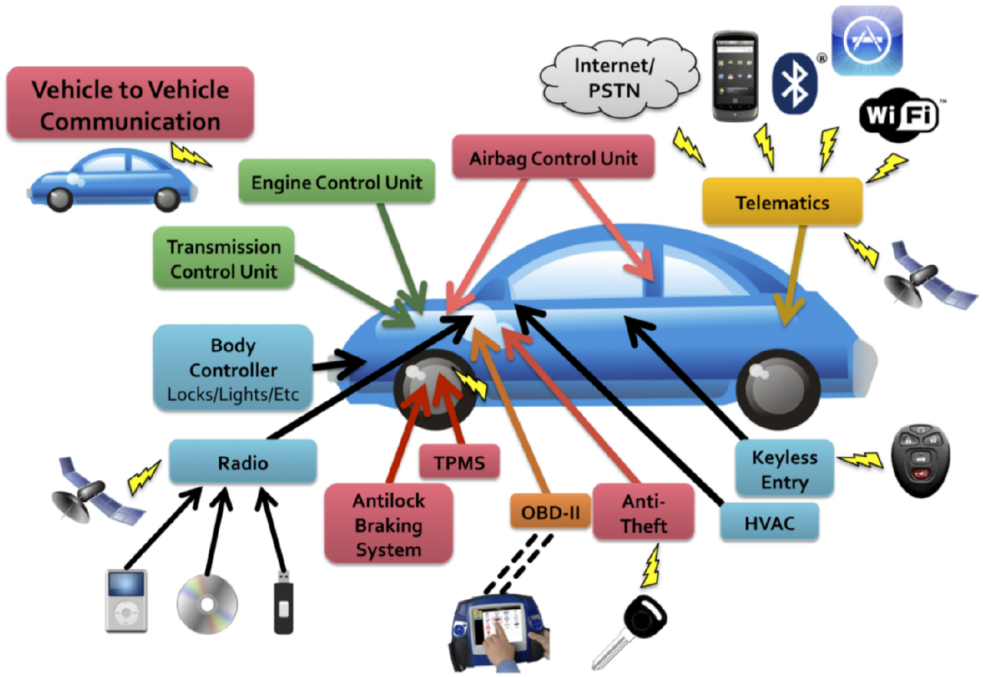
Sicherheit hat im Fahrzeug eine große Bedeutung und Systemausfälle können fatale Auswirkungen auf die Umwelt und den Menschen haben. Für den Begriff Sicherheit existieren eine Menge an Definitionen. In der Literatur wird häufig zwischen **Funktionssicherheit** (Safety) und **Informationssicherheit** (Security) unterschieden [vgl. 16, Seite 609 ff.]. Funktionssicherheit bezeichnet die Sicherheit der Technik vor Systemausfällen, die z. B. durch Natureinflüsse oder Systemfehler verursacht werden. Ein funktionssicheres System funktioniert unter allen normalen Betriebsbedingungen. Weiterhin stellt die Funktionssicherheit die Grundlage für die Informationssicherheit eines Systems dar. Informationssicherheit bedeutet für ein System, dass keine unautorisierte Veränderung und Gewinnung von Informationen zugelassen wird [vgl. 25, Seite 6 ff.].
Laut Definition von Eckert [25] müssen vier Basisanforderungen (Schutzziele) für die Informationssicherheit gewährleistet sein. Die erste Basisanforderung ist die **Informationsvertraulichkeit**. Informationsvertraulichkeit sagt aus, dass eine Information eines Systems geheim gehalten wird und nur mit entsprechender Autorisierung lesbar ist. Die

zweite Basisanforderung **Datenintegrität** bedeutet, dass unbemerkte und unautorisierte Modifikationen an Datenobjekten verhindert werden müssen. Die dritte Basisanforderung betrifft die **Systemverfügbarkeit** und besagt, dass Angriffe die Leistungsfähigkeit des Systems nicht beeinflussen dürfen. Es muss sichergestellt werden, dass die spezifizierten Funktionen des Systems zuverlässig erbracht werden. Die vierte Basisanforderung ist die **Verbindlichkeit**. Bei der Verbindlichkeit garantiert das System, dass ein Subjekt die Durchführung von Aktionen im Nachhinein nicht mehr abstreiten kann.

3.2 Angriffsoberfläche des Autos

Innerhalb der letzten Jahre hat sich das interne Netzwerk des Autos von einem komplett abgeschotteten Netzwerk hin zu einem aktiven Teilnehmer im Internet der Dinge entwickelt. Durch neue Schnittstellen und Technologien können immer mehr Funktionen umgesetzt werden, bei denen Kommunikation mit anderen Teilnehmern benötigt wird. Zum Beispiel V2V- (Vehicle to Vehicle), V2I- (Vehicle to Infrastructure) oder V2C- (Vehicle to Cloud) Kommunikation. Dabei konsumiert das Auto Dienste von außen wie zum Beispiel Updates für eine bestimmte Komponente vom Backend des Herstellers. Gleichzeitig bietet das Auto aber auch Dienste nach außen hin an. Zum Beispiel können Informationen über den Standort des Autos zur Verkehrsflusssteuerung verwendet werden. Diese neue Denkweise in Diensten benötigt aber auch eigene und neue Sicherheitsmechanismen. Das bestehende interne Netzwerk bietet zu wenig Schutz gegenüber möglichen Angreifern. Angreifer könnten Ziele wie Sabotage des Autos, Diebstahl, Erpressung, elektronisches Tuning oder das Sammeln privater Daten verfolgen.

Abbildung **3.1** gibt einen Überblick über die IO-Channel, die in einem modernen Auto vorliegen [vgl. 21, Seite 2-4]. Generell können die IO-Channel anhand ihrer Reichweite in die drei Kategorien **Indirect Physical Access**, **Short-range Wireless Access**, und **Long-range Wireless Access** unterteilt werden und bilden die Angriffsoberfläche (Surface) des Autos. Zu den Indirect Physical Access Schnittstellen gehören der Diagnoseport OBD-II, USB-Schnittstellen, Dockingstations, CD-Player oder möglicherweise sogar ein direkter Ethernet-Anschluss. Diese Schnittstellen haben alle gemeinsam, dass ein direkter Zugriff auf das Fahrzeug benötigt wird, um einen Angriff ausführen zu können. Trotzdem können die Schnittstellen zum Teil auch indirekt aus der Ferne von Angreifern verwendet



Quelle: https://www.autosec.org/pubs/cars-usenixsec2011.pdf [21]

Abbildung 3.1: Die Abbildung stellt eine typische Angriffsoberfläche eines Autos dar (sowohl Input Channel als auch Output Channel).

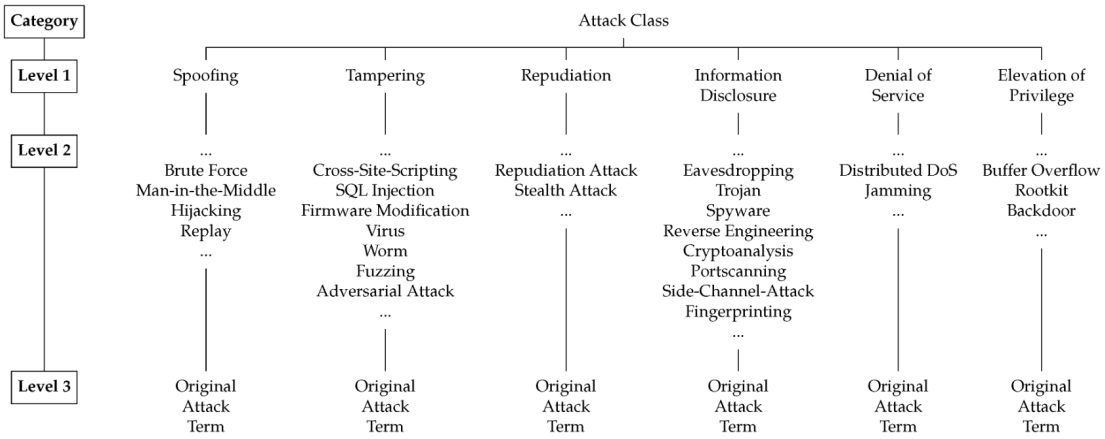
werden. Zum Beispiel, wenn das Gerät, das auf dem OBD-II Port zugreift, vom Angreifer kontrolliert wird. Die Short-range Wireless Schnittstellen bieten Zugriff innerhalb einer Reichweite von 5-300 Metern an. Dazu zählen zum Beispiel Bluetooth, Remote Keyless Entry, Tire Pressure Monitoring System (TPMS), V2V Kommunikation und Internet Hotspots. Als Letztes gibt es noch die Long-range Wireless Schnittstellen, die Kommunikation über große Reichweiten ermöglichen, wie zum Beispiel Radio-Kanäle oder Mobile Daten, die sogar individuelle Adressierung erlauben. Über Schwachstellen oder Sicherheitslücken in einem dieser Geräte könnten Angreifer Zugang ins interne Fahrzeugnetzwerk erhalten. Anschließend könnte mit einer korrumpierten Kommunikationseinheit irregulärer Nachrichtenverkehr versendet werden.

3.3 Angriffe im Fahrzeugnetzwerk

Angreifer können Schwachstellen an den Schnittstellen des Autos ausnutzen und nach anschließender Kompromittierung einzelner Komponenten Angriffe auf das interne Fahrzeugnetzwerk durchführen. Diese Angriffe können in die folgenden Angriffsvektoren unterteilt werden [vgl. 40, Seite 2-3].

- **Active Manipulation or Eavesdropping of Messages** beschreibt einen Angriff mit dem Ziel, Daten auf dem Zielrechner oder beim Transport der Daten zum Zielrechner zu manipulieren. Eavesdropping bezeichnet das Abhören der Kommunikation, um z. B. langfristig Informationen über Verschlüssung oder Details zur Authentifizierung zu erhalten.
- **Masquerading Angriffe** bezeichnet Angriffe, bei denen der Angreifer eine gestohlene Identität benutzt, um unautorisierten Zugriff auf Teile des Systems zu erhalten. Die Wirksamkeit des Angriffes hängt von dem Authentifizierungsprozess des Systems ab.
- **DoS Angriffe** (oder Flooding) sind Angriffe mit Ziel durch das Versenden zahlreicher Nachrichtenpakete, die gesamte Bandbreite ausnutzen, um die Verfügbarkeit eines Dienstes zu unterbinden. Der normale Datenverkehr im Netzwerk wird lahmgelegt und funktioniert nicht mehr korrekt.

Weiterhin haben Sommer et al. [56] eine Taxonomie für Angriffe im automotiven Bereich entwickelt. Die Taxonomie dient der Analyse und Klassifikation von Angriffen auf das Fahrzeug und soll bei frühen Entwicklungsprozessen wie der Threat Analyse oder Sicherheitstests unterstützen. Die Taxonomie besteht insgesamt aus 23 verschiedenen Kategorien. Die Abbildung **3.2** stellt die Kategorie Angriffsarten dar. Dabei besteht jede Kategorie aus insgesamt drei Abstraktionsleveln, die mit jedem Level den Angriffsarten mehr Details hinzufügen. Die Angriffsarten sind auf Level 1 in die Kategorien **Spoofing**, **Tampering**, **Repudiation**, **Information Disclosure**, **Denial of Service** und **Elevation of Privilege** unterteilt. Diese Angriffsklassen stammen aus dem STRIDE Threat Model [45] von Microsoft, das durch Modellierung versucht, Bedrohungen für ein System zu erkennen. Auf Level 2 werden für die jeweiligen Kategorien konkretere Angriffe dargestellt und Level 3 führt den konkreten Begriff eines Angriffs mit Verweis auf einen öffentlichen Angriff auf.



Quelle: https://www.mdpi.com/2078-2489/10/4/148/htm [56]

Abbildung 3.2: Die Abbildung stellt eine Ausschnitt aus der Taxonomie von Sommer et al. [56] dar. Es werden verschiedene Angriffsarten klassifiziert.

Das AutoThreat Intelligence Cyber Incident Repository [60] gibt einen öffentlichen Überblick über eine Vielzahl an IT-Sicherheitsvorfällen im Auto, die seit dem Jahre 2010 vorgefallen sind. Aktuell sind dort über 900 Fälle dokumentiert. Ein bekanntes Beispiel für einen Longrange Wireless Access Angriff auf ein Fahrzeug über Schnittstellen zum Internet stellt das Paper von Miller et al. [46] dar. In dem Beispiel haben Forscher es geschafft, einen Jeep Cherokee (Baujahr 2014) zu hacken. Dafür wurde eine Sicherheitsschwachstelle im Uconnect-System [1] des Herstellers Chrysler ausgenutzt. Anschließend konnte über die Multimedia-Schnittstelle auf das interne Kommunikationsnetz des Autos zugegriffen werden. Dadurch war es dann möglich, den Jeep fernzusteuern und Funktionen wie das Abschalten des Motors aus der Ferne auszulösen. In einem weiteren Paper [21] wurde demonstriert, wie ein in großen Massen produzierter Sedan über verschiedene Schnittstellen nach Außen komplett übernommen werden konnte. Zum Beispiel konnten Angriffe über eine Pass Thru [2] Schnittstelle (verbindet sich mit dem Diagnoseport des Autos) oder über die Update Funktionalität des CD-Players ausgeführt werden (Indirect Physical Access). Rouf et al. [53] haben bei der Analyse einer Tire Pressure Monitoring System (TPMS) entdeckt, dass die ECU, die für Verarbeitung der Signale zuständig war, zum Absturz gebracht werden konnte (Short-range Wireless Access). Auch über das Digital Radio (DAB) können Angriffe ausgeführt werden, weil DAB erlaubt, auch Daten wie

[1] Uconnect ist ein Radio Modul von Harman Kardon, das für Infotainment, WiFI, Navigation sowie Mobilfunkkommunikation verwendet wird.

[2] Pass Thru bezeichnet ein Verfahren, dass nach SAE 2534 oder ISO 22920 unabhängigen Betrieben Software-Updates für das Fahrzeug ermöglicht.

Bilder oder Texte zu übertragen [23]. Die Forschungsergebnisse all dieser ausgewählten Arbeiten zeigen, dass über die IO-Channel aller Reichweiten erfolgreich Angriffe auf das Auto durchgeführt werden können. Daher ist es notwendig, Fahrzeuge zu entwickeln, die Gegenmaßnahmen zum Schutz des internen Netzwerks enthalten.

3.4 Gegenmaßnahmen

Zum Schutz des Fahrzeugnetzwerkes vor Angriffen ist in der Literatur innerhalb der letzten Jahren die Anwendung folgender Technologien als Gegenmaßnahmen vorgeschlagen und diskutiert worden [vgl. 40, Seite 5-6]. Für ein ganzheitlich funktionierendes Sicherheitskonzept ist auch die Kommunikation mehrerer dieser Technologien möglich.

- **Intrusion Detection System** (IDS) bezeichnet ein System, dass durch aktive Überwachung des Netzwerks Angriffe auf Computersysteme und -netze erkennt. Die erkannten Angriffe werden anschließend für weitere Untersuchungen an einen zentralen Punkt gemeldet. Intrusion Detection Systeme zeichnen sich durch eine sichere Erkennung bekannter Angriffsarten aus. Es werden keine False Positives vom IDS ausgelöst. Der Nachteil eines IDS ist, dass nur bekannte Angriffe erkannt werden können.
- **Anomalieerkennung** beschreibt die Erkennung von unregelmäßigen Mustern innerhalb von normalen Daten. Die Anomalieerkennung ist in der Lage, sowohl bekannte Angriffsarten als auch unbekannte Angriffsarten zu entdecken. Dieses Verfahren geht davon aus, dass sich das Verhalten von Angreifern erkennbar vom normalen Systemverhalten unterscheidet. Je präziser die regelmäßige Kommunikation beschrieben ist, desto genauer kann die Anomalieerkennung funktionieren. Dabei kann aber auch ein untypisches Normalverhalten fälschlicherweise als Anomalie eingestuft werden (False Positive).
- **Firewalls** entscheiden anhand von Sicherheitsregeln, ob ein- und ausgehende Daten zulässig sind oder blockiert werden sollten. Es existieren verschiedene Arten von Firewalls, die auf unterschiedliche Layern, Protokollen oder Technologien basieren. Zum Beispiel könnte für das Fahrzeug die Anwendung einer Firewall auf Layer 2 interessant sein. PSFP (per-stream filtering and policing) bezeichnet eine Filtertechnick auf Layer 2 und kann Metriken wie VLAN ID, IP, Frame Length oder PCP (Priority Code Point) analysieren.

- **Cryptography** bietet die Möglichkeit durch Verschlüsselung und Authentifizierung sensible Daten zu schützen. Es gibt eine Vielzahl unterschiedlicher Verfahren zur Verschlüsselung der Daten, die sich vor allem im Aufwand und der gewährleisteten Sicherheit unterscheiden. 802.1AE MAC Security [1] stellt einen Sicherheitsstandard für Verbindungen auf dem Layer 2 des OSI-Referenzmodells dar.
- **SDN** [36] beschreibt das Paradigma von programmierbaren Switchen im Netzwerk. Die Steuerungsebene (Control Plane) wird von den Switchen in einen einzelnen zentralen Controller verschoben. Der Controller bietet die Möglichkeit der Konfiguration von Netzwerkkomponenten durch Einträge in die Flow-Tabelle des Gerätes mit Hilfe von offenen Standards wie z.B. OpenFlow [43]. Nicht identifizierbare oder unbekannte Kontrollflüsse können durch eine Differenzierung von Paketen auf der Grundlage der Header-Felder erkannt werden. Der zentrale Controller kann Weiterleitungsentscheidungen treffen oder eine Rekonfiguration des Netzwerks als Gegenmaßnahme durchführen.

Der Fokus dieser Arbeit liegt auf der Untersuchung der Gegenmaßnahme Anomalieerkennung im Fahrzeugnetzwerk.

3.5 Informationssicherheit im Fahrzeugnetzwerk

Ohne Verwendung entsprechender Gegenmaßnahmen offenbart das interne Fahrzeugnetzwerk des Autos einige Sicherheitsschwachstellen. Die vier Basisanforderungen Informationsvertraulichkeit, Datenintegrität, Systemverfügbarkeit und Verbindlichkeit können in diesem Fall nicht erfüllt werden.
Zum Beispiel verwendet die aktuelle domänenbasierte Architektur des Autonetzwerkes das CAN-Protokoll in einigen Bereichen des Netzwerkes. CAN Nachrichten können von jedem Teilnehmer auf dem Bussystem im Klartext gelesen werden. Dadurch kann die Anforderung der Informationsvertraulichkeit zur Geheimhaltung von Informationen nicht eigehalten werden. Weiterhin könnte eine korrupte ECU ein Bussystem mit Nachrichten von hoher Priorität flooden und damit die Übertragung anderer Nachrichten verhindern. Die Leistungsfähigkeit des Systems würde beeinflusst werden und die Basisanforderung der Systemverfügbarkeit wäre verletzt. Weitere Sicherheitsschwachstellen stellen die Gateway ECUs dar. Eine Gateway ECU verbindet und steuert die Kommunikation zwischen den verschiedenen Fahrzeugdomänen. Dadurch ist es ECUs aus einem Subnetz wie zum Beispiel Infotainment möglich, ohne Authentifizierung mit Steuereinheiten aus anderen

sicherheitskritischen Domänen zu kommunizieren.
Auch eine zonenorientierte E/E-Architektur mit Ethernet-Backbone als Kommunikationsmedium bietet in der Standardvariante keinen zusätzlichen Sicherheitsmechanismen zur Gewährleistung von Integrität, Authentizität und Vertraulichkeit. Unabhängig davon ob eine domänenbasierte Architektur oder eine moderne zonenorientierte E/E-Architektur vorhanden ist, sind Gegenmaßnahmen notwendig, um das Fahrzeug vor Angriffen zu schützen.

3.6 Anomalieerkennung im Fahrzeugnetzwerk

In einer Vielzahl von wissenschaftlichen Beiträgen wird bereits die Anomalieerkennung in Fahrzeugnetzwerken untersucht. Weiterhin existieren auch viele Arbeiten zur Anomalieerkennung aus anderen Bereichen, wie zum Beispiel der klassischen Erkennung von Angriffen auf Netzwerke. Rajbahadur et al. [51] zeigen in einem Übersichtspaper den aktuellen Stand (2018) zur Anomalieerkennung im Fahrzeug auf. Es wurde eine Taxonomie erarbeitet, um Lücken und Bedenken zu erkennen. Dabei wurde festgestellt, dass der Großteil der Forschung nur anhand von simulierten Daten in einer Simulationsumgebung evaluiert wurde anstatt mit realen Daten. Außerdem wurde hervorgehoben, dass fahrzeuginterne Netzwerkdaten und VANET-Daten selten zusammen betrachtet wurden, Safety weniger Aufmerksamkeit als Security erhält und der Großteil der neu vorgeschlagenen Algorithmen nicht gegen eine gemeinsame Basis verglichen wird. Einige Paper fokussieren sich auf die Erkennung von Anomalien im CAN-Bus-System. Marchetti et al. [41] schlagen einen Algorithmus zur Analyse der Sequenz von CAN-Bus-Nachrichten vor. Gmiden et al. [31] verwenden einen Ansatz zur Analyse von Zeitintervallen von CAN-Nachrichten. In einem weiteren aktuellen Paper haben Freitas et al. [27] ein IPS (Intrusion-Prevention-System) entwickelt, dass Cyberangriffe in einem CAN-Bus-System entdeckt und verhindert. Zusätzlich kann dieses System auf kostengünstiger Hardware wie dem Raspberry Pi eingesetzt werden. Da jedoch das CAN-Bus-System nicht die einzige Kommunikationstechnologie im internen Autonetzwerk ist, werden auch Anomalieerkennungssysteme für Automotive Ethernet untersucht. Grimm et al. [33] betrachtet ein hybrides Anomalieerkennungssystem innerhalb Ethernet-basierter Kommunikation. Das hybride Anomalieerkennungssystem besteht aus einer statischen Prüfung kombiniert mit drei Methoden des maschinellen Lernens (One-Class Support Vector Machine, Mahalanobis Distance und Principal Component Analysis). In der Arbeit zeigten sich Unterschiede in der Erkennungsrate und der Anzahl der Fehlalarme abhängig von der

Angriffsart zwischen den drei verwendeten Algorithmen. Rieke et al. [52] haben ein System für die Verarbeitung von Ereignisströmen, um anomales Verhalten in den Sequenzen von Ereignissen zu identifizieren, entwickelt. Alle unvorhergesehenen Ereignisse sollen zur weiteren Analyse in einem globalen Betriebszentrum verarbeitet werden. Van Wyk et al. [61] verwenden als Ansatz zur Anomalieerkennung eine Kombination von einem Convolutional Neural Network (CNN) als Deep-Learning-Methode und Kalman-Filterung mit einem X2-Detektor. Die Erkennung und Identifizierung von Anomalien wurde innerhalb von Zeitreihendaten von mehreren Sensoren durchgeführt. Ji et al. [37] ermitteln aus diskreten Signalen von Ereignissen eines Hybrid-Elektrofahrzeugs mit Hilfe des Algorithmus One-Class Support Vector Machine Anomalien. Der entwickelte Algorithmus war in der Lage, sowohl bekannte Anomalien als auch unbekannte Anomalien zu erkennen.

3.7 Anomalieerkennung mit TSN und TSN Security

Eine kleinere Anzahl an Papern konzentriert sich bereits auf die Anwendung der Anomalieerkennung im TSN Netzwerk oder führt eine Sicherheitsanalyse des TSN Protokolls durch. Meyer et. al [44] untersuchen die Anomalieerkennung mit Hilfe des TSN Standard IEEE 802.1Qci Per-Stream Filtering and Policing (PSFP). Es wurde ein System zur Anomalieerkennung auf Basis von PSFP entwickelt, das Anomalien auf Grundlage von Regeln über das vordefinierte Wissen für alle eingehenden Verkehrsströme erkennt. Innerhalb einer OMNeT++ Simulation wurde der Algorithmus an verschiedenen Angriffen evaluiert. Das Paper hat gezeigt, dass das PSFP System zur Anomalieerkennung keine False Positve Alarme auslöst. Auch bei der der Erkennung von Fehlverhalten in echten Angriffsspuren zeigten sich gute Ergebnisse. Probleme wurden insbesondere bei Erkennung von Angriffen mit der Elimination von Frames deutlich. Luo et al. [40] diskutieren den Sicherheitsmechanismus des TSN Protokolls und analysieren die Sicherheit einer TSN basierten Architektur mit Ethernet als Backbone anhand des Microsoft STRIDE-Bedrohungsmodells. Es wurde festgestellt, dass Denial-of-Service-Angriffe die größte versteckte Gefahr darstellen. Daher müssen diese besonders berücksichtigt werden. Des Weiteren wurde auch ein Algorithmus auf Basis von IEEE 802.1Qci Per-Stream Filtering and Policing vorgschlagen und in OMNeT++ in einer realen TSN Topologie evaluiert.

3.8 Forschungslücke

Zur Evaluierung der Gegenmaßnahme Anomalieerkennung im Fahrzeugnetzwerk wurden bereits eine Vielzahl an Algorithmen vorgeschlagen, die wiederum auf verschiedenen Daten oder Layern operieren. TSN verändert in der Zukunft durch QoS Garantien für sicherheitskritischen Traffic die Steuerung von Nachrichten im internen Fahrzeugnetzwerk. In der aktuellen Literatur gibt es nur wenige Paper, die ihre Algorithmen zur Anomalieerkennung in einem Netzwerk mit integrierten TSN verwenden. Der Großteil der Arbeiten evaluiert ihre Algorithmen mit simulierten Daten in einer Simulationsumgebung anstatt auf realen Daten in Hardware. Weiterhin wurde kritisiert, dass neu vorgeschlagene Algorithmen nicht gegen eine gemeinsame Basis verglichen werden. Daher liegt der Fokus dieser Arbeit auf dem Vergleich von mehreren Algorithmen zur Anomalieerkennung aus unterschiedlichen Kategorien in einem Netzwerk mit integrierten TSN in Hardware. Verschiedene Angriffsszenarien sollen Vor- und Nachteile der Algorithmen aufzeigen.

4 Konzept

Das Kapitel Konzept erarbeitet die Grundlage für einen Vergleich von Algorithmen zur Anomalieerkennung. **4.1** gibt einen Überblick zu den Kategorien von Algorithmen aus der Literatur mit den dazugehörigen Vor- und Nachteilen. Anschließend definiert **4.2** Vergleichskriterien, um Algorithmen sinnvoll miteinander vergleichen zu können. Zum Schluss des Kapitels stellt **4.3** einen Ansatz für den Aufbau eines Teilausschnitts eines modernen TSN basierten Fahrzeugnetzwerks in Hardware dar.

4.1 Kategorien der Anomalieerkennung

Algorithmen zur Anomalieerkennung können in verschiedene Kategorien eingeordnet werden. In der Literatur existieren verschiedene Ansätze zur Kategorisierung. Verfahren zur Anomalieerkennung klar voneinander abzugrenzen, ist ziemlich schwierig. Drei Beispiele für verschiedene Kategorien, in die Algorithmen eingeordnet werden können, sind in der Tabelle 4.1 dargestellt. In der Tabelle werden die Kategorien aus den Arbeiten von Bhuyan et al. [17], García-Teodoro et al. [29] und Chandola et al.[20] einander gegenübergestellt.
Gemeinsam ist bei allen drei Ausarbeitungen, dass jeweils eine Kategorie für die statistischen Verfahren existiert. Unterschiede sind vor allem in der Kategorie Clustering Algorithmen zu finden. Während bei García-Teodoro et al. Clustering-Verfahren zu der Kategorie Machine Learning zählen, haben Bhuyan et al. und Chandola et al. diese Art von Algorithmen in eine separate Kategorie eingeordnet. Zusätzlich werden bei Bhuyan et al. aus den Clustering-Verfahren nochmal die Nearest Neighbor Based Verfahren in eine extra Kategorie abgetrennt. Die Trennung basiert auf der Annahme, dass Clustering-Verfahren den Fokus auf die Bildung des Clusters und die anschließende Einordnung legen. Im Gegensatz dazu fokussieren sich die Nearest Neighbor Based Verfahren auf die lokale Nachbarschaft der Dateninstanzen. Eine Kategorie für Klassifikationsalgorithmen

(Machine Learning bei García-Teodoro et al.) ist in allen drei Arbeiten vorhanden. Ergänzend dazu sind noch weitere Kategorien wie Combination Learner, Spectral, Information Theoretic und Knowledge Based vorhanden, die speziellere Verfahren zusammenfassen.

Bhuyan et al.: Network Anomaly Detection: Methods, Systems and Tools	Chandola et al.: Anomaly Detection : A Survey	García-Teodoro et al.: Anomaly-based network intrusion detection: Techniques, systems and challenges
1. Statistical	1. Statistical	1. Statistical
2. Classification Based	2. Classification Based	2. Machine Learning Based
3. Clustering and Outlier Based	3. Clustering Based	3. Knowledge Based
4. Soft Computing	4. Nearest Neighbor Based	-
5. Knowledge Based	5. Information Theoretic	-
6. Combination Learners	6. Spectral	-

Tabelle 4.1: Die Tabelle zeigt beispielhaft Kategorien in die Algorithmen/Methoden zur Anomalieerkennung eingeordnet werden können.

4.1.1 Klassifikationsalgorithmen

Klassifikationsalgorithmen versuchen neu auftretende Dateninstanzen anhand eines zuvor gelernten Modells in verschiedene Klassen einzuordnen [17]. Dies geschieht in zwei Phasen. Zuerst erfolgt eine Trainingsphase in der gelabelte Trainingsdaten benutzt werden, um das Modell zu lernen. Anschließend wird in einer Testphase durch einen Klassifikator mit Hilfe des Modells entschieden, ob es sich um eine normale oder anormale Instanz handelt. Dabei wird zwischen Verfahren mit nur einer normalen Klasse (one-class) und Multi-Klassen (multi-class) unterschieden. Bei Multi-Klassen Verfahren wird angenommen, dass er mehrere normale Klassen gibt, in welche die Dateninstanzen eingeordnet werden können. Wenn der Klassifikator der Dateninstanz keine der normalen Klassen sicher zuordnen kann, wird diese Dateninstanz als anormal eingestuft.

Beispiele für Klassifikationsalgorithmen sind verschiedene Varianten von neuronalen Netzen, bayessche Netze, Support Vector Machines (SVM) oder regelbasierte Systeme. Bei Klassifikationsalgorithmen besitzt die Trainingsphase abhängig vom konkreten Algorithmus zwar eine hohe Rechenkomplexität, dafür ist die Testphase zum Einstufen neuer

Beobachtungen wiederum sehr schnell [vgl. 20, Seite 21]. Dies stellt einen großen Vorteil der Klassifikationsalgorithmen dar. Ein weiterer Vorteil ist, dass zum Beispiel die Multi-Klassen Verfahren es erlauben, zwischen verschiedenen normalen Klassen zu unterscheiden. Nachteile von Klassifikationsalgorithmen sind, dass viele Verfahren sehr abhängig von korrekt gelabelten Trainingsdaten sind und als Output nur angeben, ob eine Dateninstanz zu einer Klasse gehört anstatt einen Score anzugeben, der verdeutlicht wie wahrscheinlich es sich um eine anormale Instanz handelt.

4.1.2 Nearest Neighbor Based Algorithmen

Nearest Neighbor Based Verfahren [vgl. 20, Seite 28] benutzen die Distanz bzw. die Ähnlichkeit (bei kategorischen Attributen) zwischen zwei Datenpunkten als Grundlage zur Anomalieerkennung. Abhängig von den Distanzen/Ähnlichkeiten ihrer Attribute liegen Datenpunkte verschieden weit auseinander. Mit der Annahme, dass normale Daten gesammelt in kurzer Distanz zueinander liegen und anormale Daten weit von ihren Nachbarn entfernt liegen, sollen Anomalien identifiziert werden. Verschiedene Verfahren zur Distanzbestimmung können verwendet werden je nach Datenart der Attribute. Ein typisches Verfahren ist der Euklidische Abstand. Die Funktionsweise von Nearest Neighbor Based Verfahren kann in die zwei Kategorien kth Nearest Neighbor und Relative Density unterteilt werden. Bei kth Nearest Neighbor Verfahren wird für eine Dateninstanz deren Distanz zu den nächsten k-Nachbarn als Anomalie-Score betrachtet. Anhand eines Schwellenwertes (threshold) wird die Dateninstanz dann als normal oder anormal eingestuft. Bei Relative Density Verfahren wird eine Anomalie an der Dichte der Nachbarschaft erkannt. Es wird davon ausgegangen, dass Anomalien in Bereichen mit geringer Dichte an Nachbarn liegen, während normale Daten eine hohe Nachbarschaftsdichte besitzen.

Beispiele für Nearest Neighbor Based Algorithmen sind Local Outlier Factor, Varianten von K-Nearest Neighbor oder Multi-granularity Deviation Factor. Ein Vorteil von Nearest Neighbor Based Verfahren ist, dass sie von Natur aus unüberwacht funktionieren und keine direkten Annahmen über die Verteilung der Daten treffen [vgl. 20, Seite 28]. Außerdem sind Nearest Neighbor Based Algorithmen gut anpassbar in Bezug auf verschiedene Datentypen, denn es wird nur ein anderes Verfahren zur Distanzbestimmung benötigt. Nearest Neighbor Based Verfahren können auch im semiüberwachten Modus genutzt werden. Nachteile dieser Verfahren sind, dass normale Ausreißer ohne direkte Nachbarn als Anomalien eingeordnet werden könnten und Anomalien, die in Mitten von

normalen Daten liegen, als normal eingestuft werden. Auch die Komplexität der Verfahren in der Testphase ist höher als beispielsweise bei Klassifikationsalgorithmen. Weiterhin hängt die Anomalieerkennungsrate davon ab, wie gut die benutzte Methode zur Distanzbestimmung zwischen normalen und anormalen Daten differenzieren kann.

4.1.3 Statistische Algorithmen

Auch statistische Methoden können zur Erkennung von Anomalien eingesetzt werden. Als Grundlage dazu wird ein statistisches Modell benutzt [vgl. 17, Seite 9-11]. Im ersten Schritt muss das Modell mit normalen Daten befüllt werden. Mit Hilfe dieses Modells können dann Anomalien erkannt werden, indem geschaut wird, wie wahrscheinlich es ist, dass eine neue Dateninstanz von diesem Modell generiert wurde. Bei statistischen Anomalieerkennungsverfahren wird zwischen parametrisierten und nicht-parametrisierten Varianten unterschieden. Bei der parametrisierten Variante ist die Verteilungsfunktion der Daten schon bekannt und die Parameter werden durch die gegebenen normalen Daten eingestellt. Bei den nicht-parametrisierten Methoden werden statistische Modelle ohne Parameter benutzt, wie zum Beispiel Methoden, die auf Histogrammen basieren.

Beispiele für statistische Algorithmen sind Histogram-based Outlier Detection (nicht-parametrisiert), Elliptic Envelope (parametrisiert) oder Regression Model Based Verfahren. Ein Hauptvorteil von statistischen Verfahren ist, dass ein Anomalie-Score häufig auch eine Wahrscheinlichkeitsangabe als zusätzliche Information bereitstellt [vgl. 20, Seite 39-40], die angibt, wie sicher es sich um eine Anomalie handelt. Weiterhin bieten statistische Verfahren eine statistisch richtige Lösung bei der Erkennung von Anomalien an, wenn die zugrunde liegende Verteilung stimmt. Es ist auch möglich, statistische Verfahren in einem unüberwachten Modus auszuführen. Nachteile sind hingegen, dass statistische Verfahren davon ausgehen, dass die Verteilung der realen Daten auf einer bestimmten Verteilung basiert und diese Annahme nicht immer stimmen muss. Ein weiterer Nachteil ist, dass Techniken, die Histogramme benutzten, Schwierigkeiten bei der Erkennung von Anomalien in Daten mit multivariaten Attributen haben.

4.1.4 Clustering Algorithmen

Eine weitere Möglichkeit zur Anomalieerkennung bieten Clustering-Verfahren. Das Ziel von Clustering ist es, eine Menge von ähnlichen Objekten in gemeinsame Cluster einzuordnen [vgl. 20, Seite 30]. Die meisten Clustering-Verfahren operieren im unüberwachten Modus. Zur Erkennung von Anomalien können unterschiedliche Annahmen genutzt werden. Eine einfache Annahme ist zum Beispiel, dass nur normale Daten zu den Clustern gehören und Anomalien nicht zu einem Cluster gehören. Algorithmen dieser Kategorie ordnen nicht zwangsläufig jeder Dateninstanz ein Cluster zu und die übriggebliebenen Daten werden als Anomalien eingestuft. Ein weitere Annahme besagt, dass normale Daten dichter am Mittelpunkt des Clusters liegen, während anormale Daten weit vom Mittelpunkt entfernt liegen. Die letzte Annahme geht davon aus, dass normale Dateninstanzen sehr große und dichte Cluster bilden und daher nicht mit kleinen Clustern aus Anomalien zu verwechseln sind.

Beispiele für Clustering Algorithmen sind K-Means Clustering, Hierarchical-based Clustering oder Density-Based Spatial Clustering (DBSCAN). Die Vorteile von Clustering-Verfahren bei der Anomalieerkennung sind zum einen die Möglichkeit, im unüberwachten Modus zu operieren [vgl. 20, Seite 32]. Zum anderen sind Clustering-Algorithmen bei der Einordnung von neuen Dateninstanzen sehr schnell, da die neue Dateninstanz nur mit einer kleinen Menge von Clustern verglichen werden muss. Ein weiterer Vorteil besteht darin, wenn die Anzahl k an Clustern vorab bekannt ist. Dadurch wird die Bildung der Cluster und die Zuordnung sehr einfach. Ein Nachteil von Clustering-Verfahren ist, dass einige Algorithmen Probleme bekommen, wenn Anomalien selber wieder ein Cluster bilden. Ein weiterer Nachteil stellt die hohe Rechenkomplexität $O(n^2)$ einiger Clustering-Algorithmen dar und die Effektivität der Clustering-Verfahren hängt stark davon ab, ob die Cluster die Struktur der normalen Daten abbilden können.

4.2 Vergleichskriterien

Zum Vergleich von Algorithmen zur Anomalieerkennung im Fahrzeugnetzwerk mit TSN werden Kriterien benötigt, um die sechs ausgewählten Algorithmen miteinander zu vergleichen.

4.2.1 Precision Value

Als False Positive wird ein Datenpunkt bezeichnet, der zu Unrecht als Ausreißer eingestuft wird. Mit False Positives (FP) wird die Anzahl an falschen Alarmen in einem System angegeben. Ein optimales System zur Anomalieerkennung zeichnet sich dadurch aus, dass es sowohl wenige bis zu keine False Positive erkennt und trotzdem keine Ausreißer übersieht. Eine hohe Anzahl an False Positives führt zu wenig Vertrauen in das System zur Anomalieerkennung. Die U.S. National Highway Traffic Safety Administration [vgl. 57, Seite 16] definiert die Metrik des Precision Value.

$$Precison = \frac{TP}{TP + FP} \tag{4.1}$$

True Positives (TP) gibt die Anzahl an entdeckten Angriffen an. Der Precision Value ist ein Maß für die Anzahl tatsächlicher Angriffe aus der Gesamtzahl der Angriffe und gibt damit die Genauigkeit der erkannten Alarme an. Ohne False Positives (FP) ist der Precison Value gleich 1. Wenn normaler Datenverkehr als Angriff identifiziert wird (FP), sinkt dieser Wert schnell. Deswegen kann mit dem Precision Value für ein Angriffsszenario eine Aussage über normalen Datenverkehr, der fälschlicherweise als Angriff erkannt wird, getroffen werden.

4.2.2 Recall Value

Ein False Negative liegt vor, wenn ein Datensatz als normal eingestuft wird, obwohl es sich dabei um einen Ausreißer handelt. Mit False Negatives (FN) wird die Anzahl unentdeckter Angriffe angegeben. Ein System zur Anomalieerkennung, das keine Fehlalarme

auslöst, dafür aber den Großteil der echten Angriffe nicht erkennt, ist auch nicht sehr nützlich. In einem sicherheitskritischen Cyber Physical System wie dem Auto ist eine hohe Präzision bei der Erkennung notwendig, um die hohen Anforderungen an die Sicherheit gewährleisten zu können. Daher definiert U.S. National Highway Traffic Safety Administration [vgl. 57, Seite 16] die Metrik des Recall Values.

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

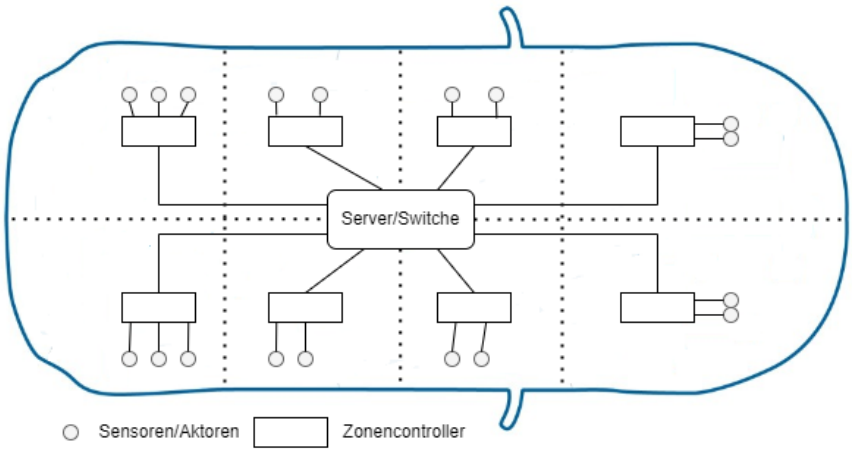
Der Recall Value wird verwendet, um zu messen, wie gut das System zur Anomalieerkennung alle tatsächlichen Angriffe erfasst hat. Ohne False Negative (FN) ist der Recall Value gleich 1. Jeder verpasste Angriff lässt diese Zahl sinken. Deshalb ist der Recall Value gut geeignet, um den Anteil der erkannten Angriffe an allen Angriffen in einem Angriffsszenario für einen Algorithmus darzustellen. Bei einem Vergleich von Algorithmen in mehreren Angriffsszenarien ist für jedes Angriffsszenario ein individueller Recall Value zum Vergleich zur Verfügung.

4.2.3 Weitere Kriterien

Neben den Erkennungsraten von Angriffen und der Anzahl an False Positives in Angriffsszenarien können noch weitere Kriterien berücksichtigt werden, um Algorithmen sinnvoll miteinander vergleichen zu können. Da die Algorithmen auf einem eingebetteten Steuergerät implementiert werden sollen, ist es wichtig, die erforderlichen Ressourcen mit einzubeziehen. Sowohl die Rechenkomplexität der Algorithmen als auch der Speicherverbrauch der Algorithmen wird in einem Vergleich betrachtet werden. Weiterhin besitzen einige Algorithmen in der Trainingsphase eine höhere Rechenkomplexität als in der späteren Testphase. Bestimmte Algorithmen können nur im supervised Modus ausgeführt werden und benötigen gelabelte Trainingsdaten für das Training. Gelabelte Trainingsdaten, die echte Angriffe enthalten, sind im Kontext des Fahrzeugnetzwerkes in großen Mengen nur schwer zu erhalten.

4.3 Verschiedene Nachrichtenströme im Fahrzeugnetzwerk

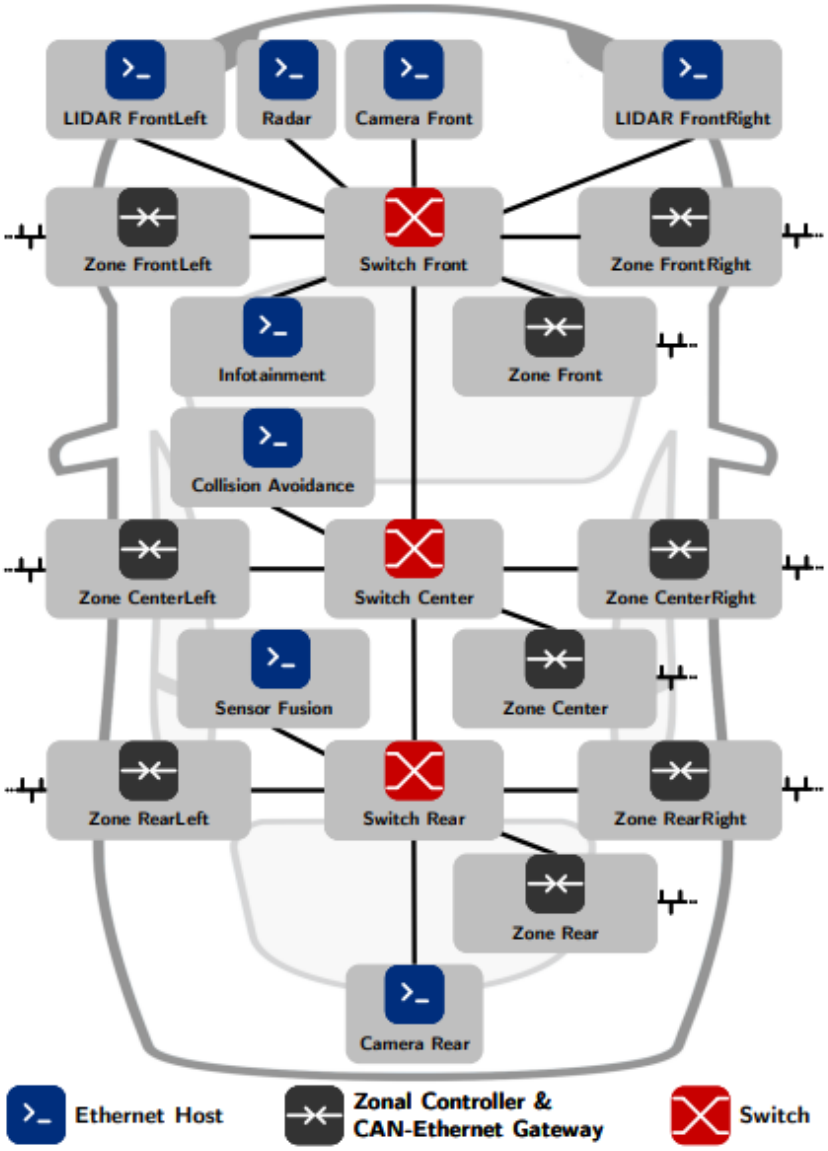
Dieses Unterkapitel beschreibt typische Netzwerkstrukturen und Kommunikation im Fahrzeugnetzwerk mit TSN. Die Beschreibungen stellen die Grundlage für das Design eines Teilausschnitts eines internen Fahrzeugnetzwerks in Hardware im nächsten Kapitel dar. Ursprünglich wurden Fahrzeugnetzwerke als isolierte Netzwerke entwickelt. Die sicherheitskritischen Kommunikationsflüsse im internen Fahrzeugnetzwerk sind bereits im Design des Netzwerks vordefiniert. Der aktuelle Stand der Technik von E/E-Architekturen im Auto hat sich von einer domänenbasierten hinzu einer zonalen Architektur [18] entwickelt. In einer zonalen Architektur werden die Komponenten in der Zukunft nach ihrer räumlichen Anordnung im Fahrzeug gruppiert. Abbildung 4.1 stellt eine zonale Architektur dar. Das Fahrzeug ist in acht Zonen unterteilt. Weiterhin enthält jede Zone einen Zonencontroller. Der Zonencontroller erlaubt den Komponenten seiner Zone über den zentralen Server mit Komponenten anderer Zonen zu kommunizieren. Außerdem hat der Zonencontroller als Aufgabe, kleinere Menge von Daten zu verarbeiten.



Quelle: Basierend auf https://ieeexplore.ieee.org/document/7986925 [18]

Abbildung 4.1: Die Abbildung zeigt eine zonale E/E Architektur eines Fahrzeugs. Die Sensoren/Aktoren werden anhand ihrer physischen Lage im Fahrzeug in Zonen unterteilt.

Die Zentren von Fahrzeugnetzwerken können sich im Aufbau voneinander unterscheiden. Ein Beispiel wird in Abbildung 4.2 dargestellt. Die Abbildung zeigt eine interne Netzwerkstruktur auf Basis einer realen Kommunikationsmatrix eines Fahrzeugs [12] [44]. Im Zentrum des Netzwerks befindet sich ein Ethernet-Backbone mit drei Switchen (Front, Center und Rear). Auch komplexere Verarbeitungsaufgaben wie z. B. Sensorfusion werden dort ausgeführt, um den Kommunikationsaufwand im Netzwerk zu reduzieren. Typische Kommunikationsflüsse aus dem abgebildeten Netzwerk aus dem SecVI Forschungsprojekt sind zum Beispiel Timed Control Traffic mit höchster Priorität (TDMA), Shaped Data Streams und Prioritized CAN Tunnel Nachrichten. Auch Chan et al. von NXP [19] modellieren in einer Arbeit zur Untersuchung von TSN die Kommunikation eines Teilausschnitts aus einem internen Fahrzeugnetzwerk. Die verwendeteten Kommunikationsflüsse sind Scheduled Traffic mit höchster Priorität, Traffic zur Zeitsynchronisation, Audio und Video Traffic und Best Effort Traffic.
Der Scheduled Traffic zeichnet sich durch eine hohe Priorität, kleine Paketgrößen, Zyklen im Millisekundenbereich und exklusive 802.1Qbv Zeitfenster in den Switchen aus. Timed Control Traffic verwendet eine hohe Priorität und umfasst z. B. Daten vom Radar oder der Sensorfusion für Aufgaben wie die Kollisionsvermeidung. Des Weiteren werden häufig Videostream Pakete oder LIDAR (Light Detection And Ranging) Pakete mit mittlerer Priorität durchs Netzwerk verschickt. Diese Art von Traffic eignet sich gut für reservierte Bandbreite durch Shaped Data Streams (802.1Qav). Weiterhin müssen auch in modernen internen Fahrzeugnetzwerken CAN-Nachrichten, die zwischen Geräten in verschiedenen Zonen ausgetauscht werden, über Ethernet getunnelt werden. Diese Art von Traffic besitzt eine niedrige bis mittlere Priorität, kleine Paketgrößen und Zyklen im Millisekundenbereich. Zum Modellieren einer realistischen Auslastung wird in diesem Versuchsaufbau ähnlich wie in der Arbeit von NXP zusätzlich ein Best Effort Traffic im Teilausschnitt verwendet. Durch den Best Effort Traffic soll der Link im Durchschnitt zu 70 % ausgelastet sein. Der Wert 70 % Auslastung basiert auf Erfahrung/Vermutung, da keine präzise Quelle für eine realistische Auslastung gefunden wurde.



Quelle: https://arxiv.org/pdf/2112.11109.pdf [44]

Abbildung 4.2: Die Abbildung stellt eine interne Netzwerkstruktur auf Basis einer realen Kommunikationsmatrix eines Fahrzeugs dar.

5 Beschreibung des Prototyp TSN Netzwerks

Dieses Kapitel stellt den Versuchsaufbau des Prototyp TSN Netzwerks und die verwendete Hardware vor. Zuerst hebt **5.1** die drei Hauptkomponenten des Netzwerks einzeln hervor. Danach überblickt **5.2** den gesamten Aufbau sowie den Nachrichtenverkehr im Netzwerk. Anschließend beschreibt **5.3** die Anwendung und Qualitätssicherung der eingesetzten TSN Feature. Zum Schluss des Kapitels zeigt **5.4**, wie das System zur Anomalieerkennung in das Netzwerk eingebunden wird.

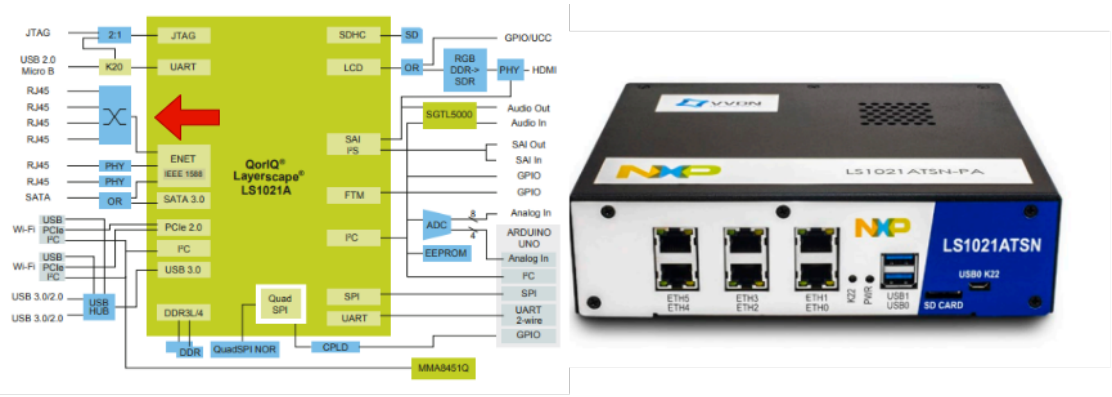
5.1 Kernkomponenten

Im Prototyp TSN Netzwerk werden insgesamt drei Switche von drei verschiedenen Herstellern eingesetzt. Die folgenden drei Kapitel stellen die im Prototyp TSN Netzwerk verwendeten Switche von NXP, InnoRoute und Kontron vor.

5.1.1 NXP Switch

LS1021A Time-Sensitive Networking Reference Design (LS1021A-TSN-RD) [11] ist der exakte Bezeichner für die im Folgenden als NXP Switch bezeichnete Plattform. Die Plattform ist ein Referenzdesign für die Entwicklung von TSN Anwendungen im industriellen IoT-Bereich. Abbildung 5.1 zeigt in einem Blockdiagramm den internen Aufbau des Systems und das Aussehen der Frontseite. Es besteht aus dem Applikationsprozessor QorIQ Layerscape LS1021A und dem TSN Switch SJA1105T (roter Pfeil). Mithilfe von einem industriellen Linux SDK können Einstellungen zur Konfiguration von TSN auf dem internen SJA1105T Switch vorgenommen werden. Der Switch unterstützt die TSN Standards 802.1Qbv (Time Aware Shaper) und 802.1Qav (Credit Based Shaper) sowie die Zeitsynchronisation nach 802.1AS. Zur Verbindung nach Außen hin bietet der Switch vier

externe Gigabit Ethernet Interfaces und zusätzlich hat die Plattform noch zwei weitere Gigabit Ethernet Interface (nicht direkt mit dem TSN Switch verbunden).

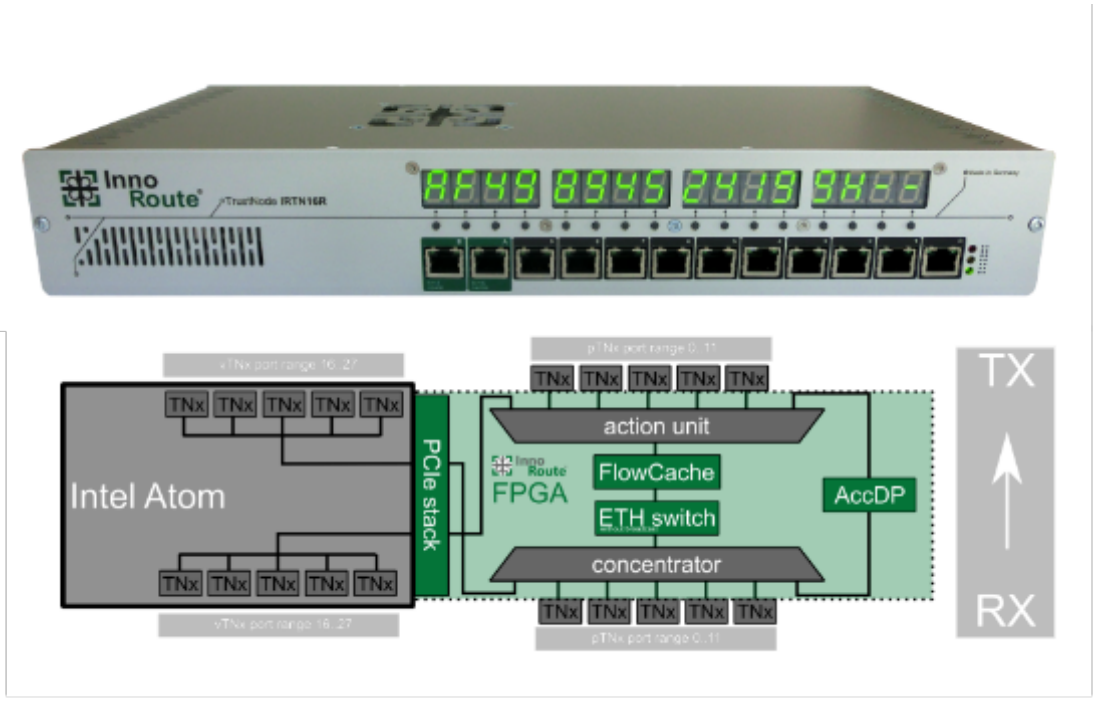


Quelle: https://www.nxp.com/docs/en/fact-sheet/LS1021ATSNRDA4FS.pdf [11]

Abbildung 5.1: Die Abbildung zeigt auf der linken Seite das Blockdiagramm des LS1021ATSN Systems mit internen SJA1105T TSN Switch. Der TSN Switch ist im Blockdiagramm mit einem roten Pfeil markiert. Auf der rechten Seite der Abbildung ist das Aussehen der Frontseite abgebildet.

5.1.2 InnoRoute Switch

Die Komponente InnoRoute Switch im Prototyp TSN Aufbau bezeichnet den TrustNode IRTN16R TSNSDN-Switch [9] der Firma InnoRoute in der Industrial Version. Dieser Switch wurde zur Forschung hinsichtlich der Netzwerkanforderungen von Industrie 4.0 bezüglich Latenz, Flexibilität und Sicherheit entwickelt. Der Switch unterstützt sowohl die Möglichkeit, SDN Features zu verwenden als auch TSN Features. Intern ist der Switch unterteilt in einen Intel Atom Prozessor und einen programmierbaren FPGA Teil (siehe Abbildung 5.2). Das FPGA-Design ist ein IPv6-Router/Switch für die Paketverarbeitung mit niedriger Latenz, der vom Atom-Prozessor gesteuert werden kann. Zur Verbindung nach Außen hin bietet der InnoRoute Switch zwölf externe Gigabit Ethernet Interfaces und unterstützt die TSN Standards 802.1Qbv sowie die Zeitsynchronisation mithilfe von 802.1AS.

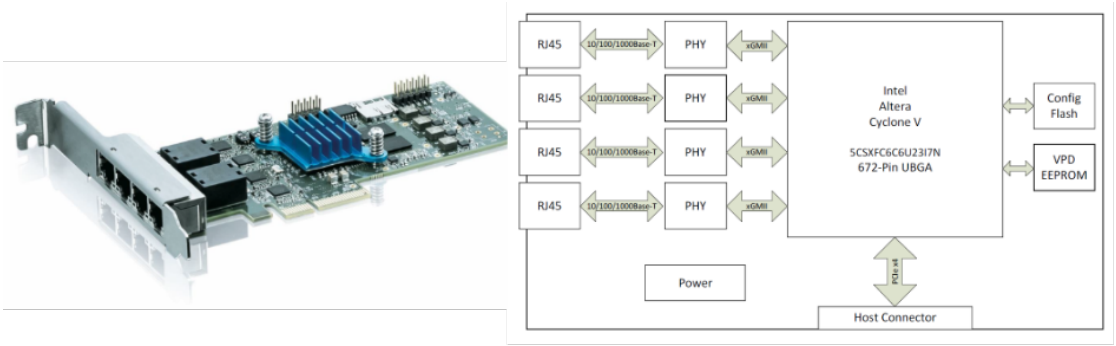


Quelle: https://innoroute.com/trustnode_router/ [9]

Abbildung 5.2: Die Abbildung zeigt im unteren Bereich das Blockdiagramm des InnoRoute Switch aufgeteilt in Atom Prozessor und FPGA. Im oberen Bereich der Abbildung ist das Aussehen der Frontseite des InnoRoute Switch abgebildet.

5.1.3 Kontron-Karte

Die dritte Kernkomponente im Prototyp TSN Netzwerk ist die PCIE-0400-TSN Network Interface Card von dem Unternehmen Kontron. Die Netzwerkkarte wird im Folgenden als Kontron Karte bezeichnet. Die Kontron Karte enthält einen integrierten TSN Switch und unterstützt den Einsatz von TSN im Netzwerk. Intern ist die Karte über PCI Express (Host Interface PCIe Gen 1 x4) mit dem Computer verbunden (siehe Abbildung 5.3). Die Konfiguration der Karte erfolgt mithilfe von Standard Linux Tools und einem Switch Treiber, der die Schnittstellen der Karte auf den Linux-Kernel/Benutzerbereich abbildet. Zur Verbindung nach Außen bietet die Kontron Karte vier externe Gigabit Ethernet Interfaces und unterstützt laut Hersteller die TSN Standards 802.1Qbv sowie die Zeitsynchronisation mithilfe von 802.1AS und den TSN Standard zur Frame Preemption IEEE 802.Qbu.



Quelle: https://www.kontron.com/de/produkte/pcie-0400-tsn-network-interface-card/p151637 [10]

Abbildung 5.3: Die Abbildung stellt das Blockdiagramm der PCIE-0400-TSN Network Interface Card von Kontron mit vier externen Gigabit Ethernet Ports und PCIe Schnittstelle dar. Des Weiteren wird auf der linken Seite der Abbildung das Aussehen der Kontron Karte gezeigt.

5.2 Aufbau und Kommunikation des Prototyp TSN Netzwerks

Das Prototyp TSN Netzwerk besteht aus den drei Automotive Ethernet Switchen, die in vorangegangenen Kapiteln eingeführt wurden. Die Abbildung 5.4 zeigt den Aufbau des Netzwerks. Den Kern des Aufbaus und der späteren Untersuchungen bildet der NXP Switch in der Mitte. Der NXP Switch ist über Ethernet mit dem InnoRoute Switch und der Kontron Karte verbunden. Die Switche unterstützen jeweils Übertragungsgeschwindigkeiten von 1 Gbit/s. Weiterhin befinden sich im Aufbau drei Raspberry Pis, die als Sender von bestimmten Nachrichtenarten mit unterschiedlichen Prioritäten fungieren (rote Vierecke). Jeweils ein Raspberry Pi versendet Videostream Pakete (Lila), Hintergrund Cross Traffic (Gelb) oder in Ethernet Pakete gekapselte CAN-Nachrichten (Orange) zur Kontron Karte. Die Kontron Karte ist auch direkt mit dem InnoRoute Switch verbunden. Zur Vermeidung der Bildung eines Kreises zwischen den drei Switchen verwirft der InnoRoute Switch alle Pakete, die vom InnoRoute Switch in Richtung Kontron Karte verschickt werden. An dem Ausgangsport des NXP Switch zur Kontron Karte bildet sich ein gemischter Nachrichtenverkehr aus dem Nachrichtenverkehr von den Raspberry Pis und dem Scheduled Traffic. Die Kontron Karte dient anschließend als Empfänger von allen Nachrichtenarten. Mithilfe von 802.1AS wird die Zeit zwischen den drei Switchen im Netzwerk synchronisiert. Zur Steuerung und Priorisierung des Nachrichtenverkehrs werden an dem NXP Switch und dem InnoRoute Switch 802.1 Qbv Schedules verwendet.

Der verwendete Schedule hat eine Gesamtdauer von 10 ms bis sich der Zyklus wiederholt. Nach ca. 5 ms wird exklusiv das Gate für die Priorität 6 für eine Mikrosekunde geöffnet. Zuvor waren für 20 Mikrosekunden alle Gates geschlossen, damit vorherige Übertragungen abgeschlossen werden konnten (mehr Details im Kapitel 5.3.2 zur Qualitätssicherung von 802.1 Qbv). Die Raspberry Pis im Netzwerk sind im Gegensatz zu den Switchen keine TSN-fähigen Sender. Daher sind die Sendezeitpunkte der Raspberry Pis nicht einsynchronisert. Abbildung 5.5 stellt den Prototyp TSN Aufbau im Labor dar.

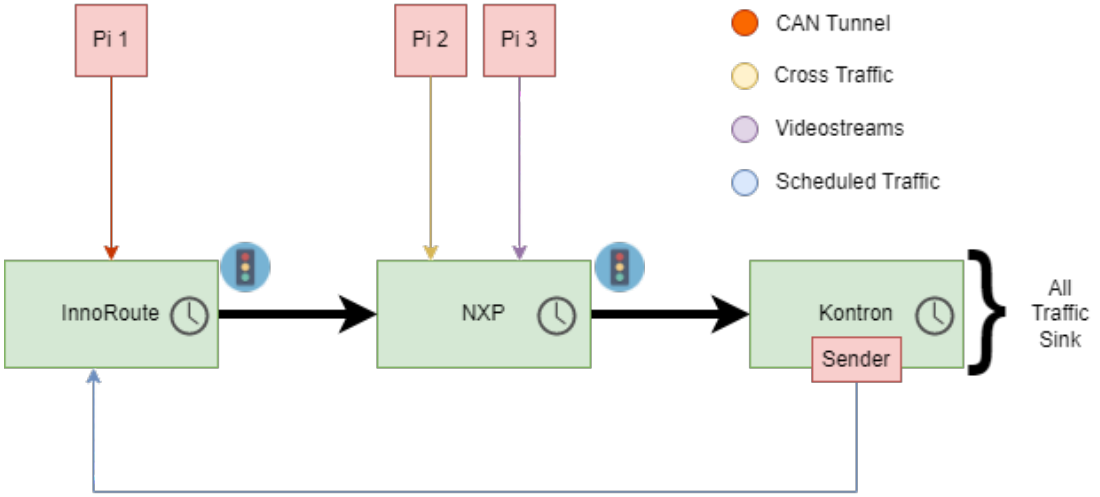


Abbildung 5.4: Die Abbildung stellt die initiale Übersicht über den Prototyp TSN Aufbau mit den verwendeten TSN Standards dar. Weiterhin sind die Sender der verschiedenen Nachrichtenarten erkennbar und die Kontron Karte als Empfänger aller Nachrichtenarten.

Tabelle 5.1 bildet die einzelnen Nachrichtenströme im Netzwerk mit Sender, Empfänger, PCP Priorität, Periode, Paketgröße und durchschnittlicher Bandbreite ab. Die Kontron Karte dient als Sender für den sicherheitskritischen Scheduled Traffic mit einem PCP Priorisierungslevel von 6 im VLAN Header. Der Scheduled Traffic wird regelmäßig in Perioden von 10 ms einmal im Kreis über den InnoRoute Switch und den NXP Switch bis zur Kontron Karte zurück versendet. Im NXP Switch ist ein IEEE 802.1Qbv Schedule eingestellt, so dass ausschließlich das Gate zur Übertragung von Queue 6 (Priorität 6) offen ist, wenn der Scheduled Traffic diesen Switch erreicht. Die Größe der Pakete beträgt 68 Bytes (minimale Größe eines Ethernet Frames und VLAN Header). Weiterhin fungiert der NXP Switch als gPTP-Grandmaster und verwendet das IEEE 802.1AS gPTP-Protokoll zur Synchronisierung der Zeit zwischen den drei Switchen, um die Qbv

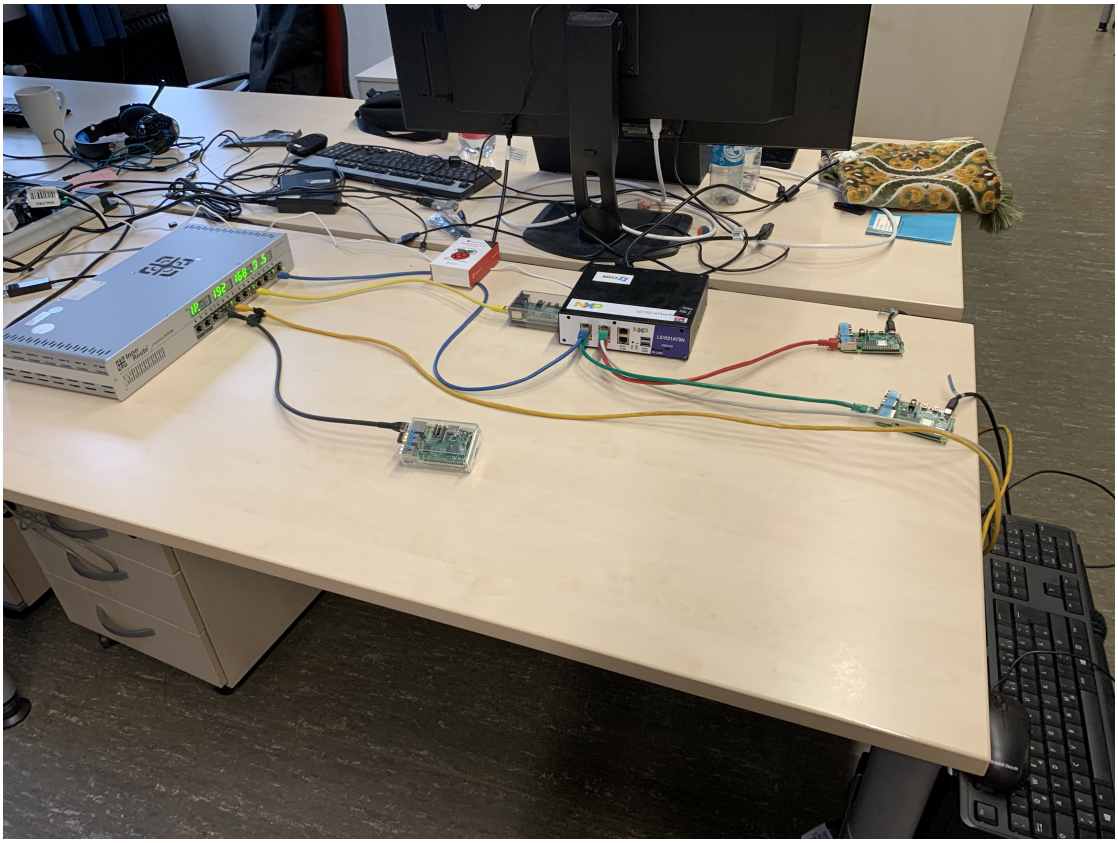


Abbildung 5.5: Die Abbildung zeigt den Prototyp TSN Aufbau im Labor mit dem Inno-Route Switch (Links), dem NXP Switch (Mitte) und der Kontron Karte (Rechts im Computer verbaut).

Schedules aufeinander abzustimmen. Der Jitter der PTP Pakete ist durch die höchste Priorität 7, die auf Treiberebene für link-lokalen Managementverkehr (STP, PTP usw.) fest auf Priorität 7 festgelegt ist, und dem sehr kleinen exklusiven Zeitfenster für Prioriät 6 vernachlässigbar gering. Ein Videostream mit einer Bandbreite von 67 Mbit/s (4k Video mit 60 fps) wird von einem Raspberry Pi an die Kontron Karte gesendet. Die Priorität des Streams ist auf 5 eingestellt. Ein weiterer Videostream wird von dem Raspberry Pi mit der Priorität 5 verschickt und der Verkehr dieses Videostreams wird durch einen Credit Based Shaper (CBS) geformt. Der CBS stellt sicher, dass der Videostream eine reservierte Bandbreite nicht überschreitet. Der nächste Raspberry Pi hat die Aufgabe, regelmäßig alle 80 ms eine CAN-Nachricht über Ethernet mit Priorität 4 an die Kontron Karte zu schicken. Der dritte Raspberry Pi im Netzwerk füllt den Link zwischen NXP Switch und Kontron Karte mit Cross Traffic, um eine höhere Auslastung des Netzwerkes

Traffic Typ	Sender	Empfänger	802.1Q Priorität	Periode	Paketgröße	Durchschnittliche Bandbreite
Scheduled Traffic (sicherheitskritisch)	Kontron Sender	Kontron Karte	6	10 ms	68 Bytes	54,4 kbit/s
gPTP Grandmaster	NXP Switch	Kontron Karte/InnoRoute Switch	7	1 s (20 Frames pro Periode)	∼ 80 Bytes	12,8 kbit/s
Videostream Traffic	Raspberry Pi	Kontron Karte	4	-	1362 Bytes	∼ 67 Mbit/s
Shaped Data Stream	Raspberry Pi	Kontron Karte	5	-	1362 Bytes	∼ 67 Mbit/s
Prioritized CAN Tunnel	Raspberry Pi	Kontron Karte	4	80 ms	68 Bytes	6,5 kbit/s
Background Cross Traffic	Raspberry Pi	Kontron Karte	0		1518 Bytes	560 Mbit/s

Tabelle 5.1: Die Tabelle gibt einen Überblick über alle Arten von Nachrichten, die im Prototyp TSN Aufbau versendet werden.

zu simulieren. Gemeinsam ist allen Arten von Nachrichten, dass sie alle über den NXP Switch in Richtung der Kontron Karte verschickt werden. Daher verwenden alle Arten den gleichen Ausgangsport und die gleiche Ethernet Verbindung. So bildet sich ein gemischter Traffic auf diesem Abschnitt.

5.3 Umsetzung der TSN Feature

Die folgenden Kapitel demonstrieren die Implementation der TSN Feature IEEE 802.1AS Clock Synchronization, IEEE 802.1Qbv Enhancements for Scheduled Traffic und IEEE 802.1Qav Credit Based Shaper in den Prototyp TSN Aufbau. Zusätzlich wird für jeden der drei Standards eine Qualitätssicherung durchgeführt, um sicherzustellen, dass die Implementation korrekt umgesetzt wurde.

5.3.1 IEEE 802.1AS

Zur Zeitsynchronisation im Hardware-Aufbau wird das Protokoll Precision Time Protocol (PTP) mit dem IEEE 802.1AS (gPTP[1]) Profil verwendet. Mit einer zusätzlichen

[1]gPTP ist die Abkürzung für Generalized Precision Time Protocol und wurde vor allem in Hinblick auf TSN Technologien entwickelt. Das Protokoll enthält eine vereinfachte Menge von PTP Optionen. Die reduzierte Komplexität von gPTP bietet statttdessen Verbesserungen in der Leistung und Interoperabilität.

Hardware-Unterstützung der Netzwerkkarten und Switche ist PTP in der Lage eine Genauigkeit im Bereich von unter einer Mikrosekunde zu erreichen [5]. Die Konfiguration und Steuerung von PTP erfolgt über die Anwendungen ptp4l und phc2sys. Ptp4l implementiert die PTP Boundary Clock and Ordinary Clock. Das Programm hat die Aufgabe, die Hardware-Uhren mit der Master-Uhr im Netzwerk zu synchronisieren. Dazu erzeugt ptp4l automatisch eine Master-Slave-Hierarchie im Netzwerk. Die übergeordnete Master-Uhr wird als Grandmaster-Uhr bezeichnet. Das Programm phc2sys synchronisiert die Systemuhr mit der PTP-Hardwareuhr auf der Netzwerkkarte. Abbildung 5.6 zeigt die Master-Slave-Hierarchie im Prototyp TSN Netzwerk. Der NXP Switch stellt die Grandmaster-Uhr und synchronisiert die Zeit auf die beiden Slaves, den Innoroute Switch und die Kontron-Karte. Weitere Geräte im Netzwerk benötigen keine Synchronisation, da nur nicht zeitkritischer Traffic versendet wird.

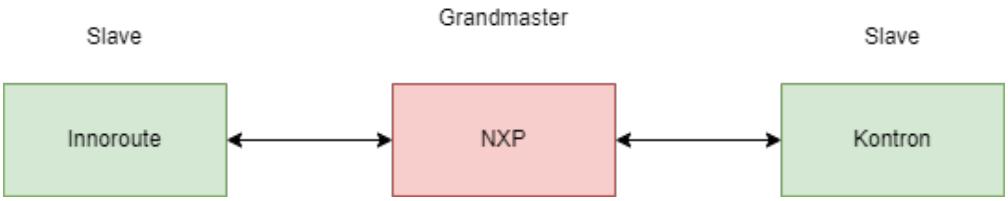


Abbildung 5.6: Die Abbildung stellt die Master-Slave-Hierarchie im Netzwerk dar.

Zur Verifikation der Zeitsynchronisation können auf allen drei Geräten die Logausgaben der Programme ptp4l und phc2sys betrachtet werden. Folgende Befehle (siehe 5.1) starten ptp4l und phc2sys auf den Geräten.

```
ptp4l -m -q -i TN1 -f /usr/share/InnoRoute/myPtp.conf
phc2sys -m -a -r -f /usr/share/InnoRoute/myPhc2sys.conf
```

Listing 5.1: Ppt4l und Phc2sys am Beispiel vom InnoRoute Switch

Die folgenden Logausgaben vom NXP Switch spiegeln den Aufbau einer Master-Slave-Hierarchie im Netzwerk wieder. In Abbildung 5.2 wird ein Ausschnitt aus dem ptp4l Log in der Initialisierungsphase vom NXP Switch gezeigt. Es ist zu erkennen, dass der NXP Switch als Grandmaster festgelegt wird. Der Bezeichner der Uhr vom NXP Switch (00049f.fffe.ef0808) ist in Rot gekennzeichnet. Die beiden Uhren von der Kontron Karte und dem InnoRoute Switch werden in Grün gekennzeichnet. Diese Uhren nehmen die Rolle des Slaves gegenüber dem NXP Switch ein und die Rolle des Masters gegenüber ihrer eigenen Systemclock ein.

```
selected /dev/ptp1 as PTP clock
port 1: link down
port 1: LISTENING to FAULTY on FAULT_DETECTED (FT_UNSPECIFIED)
port 3: link down
port 3: LISTENING to FAULTY on FAULT_DETECTED (FT_UNSPECIFIED)
selected local clock 00049f.fffe.ef0808 as best master
port 2: LISTENING to MASTER on ANNOUNCE_RECEIPT_TIMEOUT_EXPIRES
selected local clock 00049f.fffe.ef0808 as best master
port 2: assuming the grand master role
port 2: new foreign master 008082.fffe.86e64c-1
selected best master clock 008082.fffe.86e64c
port 2: assuming the grand master role
port 4: new foreign master 00534e.fffe.554c01-1
selected best master clock 00534e.fffe.554c01
port 2: assuming the grand master role
port 4: assuming the grand master role
port 4: LISTENING to GRAND_MASTER on RS_GRAND_MASTER
```

Listing 5.2: Grandmaster in der Initialisierungsphase am Beispiel vom NXP Switch

Im Gegensatz dazu zeigt das Listing 5.3 die ptp4l Initialisierungsphase vom InnoRoute Switch aus der Perspektive eines Slaves. Der Bezeichner der Clock vom NXP Switch ist wieder in Rot hervorgehoben. Die Clock vom InnoRoute Switch ist in grüner Farbe dargestellt. Es ist zu erkennen, dass ptp4l zuerst die eigene lokale Clock als Master betrachtet. Anschließend wird die Clock vom NXP Switch entdeckt, als neuer Master bestimmt und in den Slave Modus gewechselt.

```
selected /dev/ptp0 as PTP clock
port 1: INITIALIZING to LISTENING on INIT_COMPLETE
port 0: INITIALIZING to LISTENING on INIT_COMPLETE
port 1: LISTENING to MASTER on ANNOUNCE_RECEIPT_TIMEOUT_EXPIRES
selected local clock 00534e.fffe.554c01 as best master
assuming the grand master role
port 1: new foreign master 00049f.fffe.ef0808-4
selected best master clock 00049f.fffe.ef0808
port 1: MASTER to UNCALIBRATED on RS_SLAVE
port 1: UNCALIBRATED to SLAVE on MASTER_CLOCK_SELECTED
```

Listing 5.3: Ausschnitt von einem Slave in der Initialisierungsphase am Beispiel vom InnoRoute Switch

Sobald die Initialisierungsphase durchgeführt wurde, loggen die Slaves in Intervallen von einer Sekunde den Status der Zeitsynchronisation heraus. In Listing 5.4 ist ein Ausschnitt der Logs vom InnoRoute Switch abgebildet. Die ersten beiden Werte stellen den Offset zur Master Clock dar. Es wird der Durchschnitt im Intervall als rms (Root Mean Square Value) und der maximale Offset (max) angegeben. Der InnoRoute Switch hat in diesem Ausschnitt Werte im rms Bereich von 5 ns bis zu 12 ns als Offset. Anschließend zeigt jede Zeile die Frequenzkorrektur zur Synchronisation mit der Grandmaster Clock in parts per Billion (ppb). In diesem Beispiel beträgt die Frequenzkorrektur etwa 15300 ppb. Zuletzt gibt der Measured Path Delay die gemessene Zeit für ca. 70 Bytes zum Grandmaster Link an. Der Mean Path Delay beträgt 512 ns.

```
rms   12 max   20 freq +15296 +/-   6 delay   521 +/-   0
rms    8 max   15 freq +15296 +/-   9 delay   521 +/-   0
rms   13 max   17 freq +15324 +/-   5 delay   521 +/-   0
rms    5 max    9 freq +15306 +/-   5 delay   521 +/-   0
rms    9 max   11 freq +15293 +/-   5 delay   521 +/-   0
rms   12 max   17 freq +15323 +/-   6 delay   521 +/-   0
rms   11 max   18 freq +15294 +/-   7 delay   521 +/-   0
rms    6 max   11 freq +15308 +/-   7 delay   521 +/-   0
rms   12 max   20 freq +15328 +/-   7 delay   521 +/-   0
rms   10 max   15 freq +15301 +/-   8 delay   521 +/-   0
```

Listing 5.4: Ausschnitt aus den ptp4l Logs vom InnoRoute Switch, während die Zeitsynchronisation läuft.

5.3.2 IEEE 802.1Qbv

IEEE 802.1Qbv ermöglicht die zeitgesteuerte Verarbeitung des Netzwerkverkehrs in Switchen mit Hilfe eines zyklischen Schedules. Sowohl der InnoRoute Switch als auch der NXP Switch unterstützen ein Scheduling nach IEEE 802.1Qbv.
Zur Qualitätssicherung der 802.1Qbv Funktionalitäten vom NXP Switch wurde der folgende Aufbau entwickelt (siehe Abbildung 5.7). Der Aufbau besteht aus dem InnoRoute Switch, dem NXP Switch und der Kontron Karte sowie einem Raspberry Pi. Die Kontron Karte sendet einzelne Ethernet Frames mit der Größe von 1000 Byte mit einem PCP Priorisierungslevel von 5 im Vlan Header in einem Kreis über den InnoRoute Switch und dem NXP Switch zu sich selber zurück (blaue Pfeile). In ausgewählten Szenarien wird zusätzlich ein Cross Traffic im Hintergrund vom Raspberry Pi mit der PCP Priorität von 0 über den NXP Switch an die Kontron Karte versendet (gelbe Pfeile). Zur Vermeidung der Bildung eines Kreises zwischen den drei Switchen, verwirft der Innoroute Switch alle Pakete, die in die andere Richtung verschickt werden.

Da die Kontron Karte trotz Zeitsynchronisation nicht in der Lage ist, konstant zu einem exakten Zeitpunkt Pakete abzusenden, übernimmt der InnoRoute Switch die Aufgabe, den eingehenden Traffic zu synchronisieren. Weiterhin ist der Switch der Kontron Karte nicht in der Lage, Hardware Zeitstempel der ankommenden Pakete zu erhalten, daher

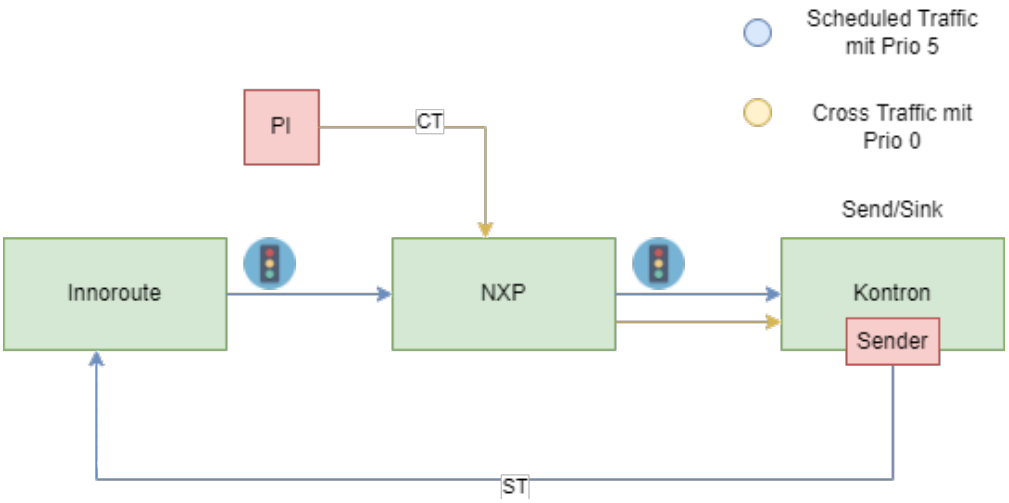


Abbildung 5.7: Die Abbildung zeigt den Versuchsaufbau für die Qualitätssicherung des IEEE 802.1Qbv Standards. Der Raspberry Pi fungiert als Sender des Cross Traffics (CT) und die Kontron Karte sendet den Scheduled Traffic (ST) mit PCP Priorität 5 einmal im Kreis.

basieren alle gemessenen Latenzen nur auf Software Zeitstempeln. Für die Ziele der Untersuchungen reichen Software Zeitstempel beim Empfangen aus, weil Eigenschaften des NXP Switches so untersucht werden, das Toleranzen im Mikrosekundenbereich keine Auswirkungen auf die Ergebnisse hätten.

Das Ziel der folgenden Szenarien ist es zu untersuchen, dass der NXP Switch Pakete mit der Priorität 5 umgehend mit Öffnen des Gates weiterleitet ohne Verzögerung durch Pakete des Cross Traffics mit der Priorität 0. Wenn der Scheduled Traffic sein Zeitfenster verpassen würden, müsste ein gesamter Zyklus gewartet werden. Zur Überprüfung dieser Eigenschaften werden folgende Messungen durchgeführt. Erstens soll der Absendezeitpunkt des InnoRoute Switch an den Qbv Schedule des NXP Switch angepasst werden, so dass die Verzögerung des Weiterleitens der Pakete im NXP Switch minimal ist. Das Einsynchronisieren der Pakete am InnoRoute Switch ist notwendig, da der Sender Kontron Karte keine Pakete zu einem exakten Zeitpunkt absenden kann. Weiterhin wird gezeigt, dass das Verpassen des Zeitslots zu einer Verzögerung der Frames von einem gesamten Zyklus führt und damit alle nachfolgenden Frames der Priorität 5 auch verzögert sind. Zweitens soll die Latenz bei der Verwendung von einem Qbv Schedule im NXP Switch in einem Szenario mit maximalen Cross Traffic im Hintergrund untersucht werden und

mit weiteren Latenzen ohne Qbv Schedule sowie ohne Cross Traffic verglichen werden.

Zum Konfigurieren des Schedules im NXP-Switch wird das Programm tc-taprio (Time Aware Priority Shaper) verwendet. Listing 5.5 zeigt den Aufbau dieser Kommandos. Die erste Zeile legt mit der Angabe swp3 (rote Markierung) das Interface fest, auf welchem der ausgehende Traffic gesteuert werden soll. Die folgenden Zeilen spezifizieren Aspekte wie die Anzahl an Queues, die Zuordnung von Prioritäten zu den Queues oder den Startzeitpunkt des Schedules. Anschließend wird der konkrete Schedule definiert (grüne Markierungen), der eine Abfolge von Gate-Zuständen für bestimmte Zeitintervalle darstellt. In einem einzelnen Eintrag wird ein Zustand in 8 Bit einer Zeitdauer in Nanosekunden zugeordnet. Ein gesetztes Bit zeigt an, dass ein Gate der jeweiligen Priorität offen ist. Zum Beispiel bedeutet 20 (0010 0000 = Bitfolge), dass das Gate für die Priorität 5 offen ist. In dem Listing 5.5 sind insgesamt vier Einträge vorhanden. Der gesamte Zyklus dauert 10 ms. Zuerst ist das Gate für allen Traffic außer der Priorität 5 für 4,99 ms offen. Anschließend werden alle Gates für 20 µs geschlossen, damit keine Übertragung mehr andauert, sobald das Gate für die Priorität 5 geöffnet wird. Die Implementation von Qbv im NXP Switch erlaubt das Fortsetzen der Übertragung von Frames, obwohl der Zeitslot schon abgelaufen ist. Daher ist im dritten Eintrag für die Priorität 5 das Gate nur für 1 µs offen, obwohl die Übertragung von 1000 Bytes in dieser Zeit nicht abgeschlossen werden kann. Zum Schluss des Zyklus sind wieder alle Gates außer das Gate für die Priorität 5 offen und danach beginnt der gesamte Zyklus von vorne.

```
tc qdisc add dev swp3 parent root taprio
num_tc 8
map 0 1 2 3 4 5 6 7
queues 1@0 1@1 1@2 1@3 1@4 1@5 1@6 1@7
base-time 0
sched-entry S df   4990000
sched-entry S 00   20000
sched-entry S 20   1000
sched-entry S df   4989000
flags 0x2
```

Listing 5.5: Tc-taprio wird zur Konfiguration von IEEE 802.1Qbv im NXP Switch verwendet. Die grün eingefärbten Zeilen stellen eine Folge von Gate-Zuständen mit Zeitintervall dar.

In der Theorie setzt sich die Latenz für den Scheduled Traffic (siehe Abbildung 5.7) einmal im Kreis von der Kontron Karte (Sender) über den InnoRoute Switch (Switch 1) und den NXP Switch (Switch 2) bis zur Kontron Karte (Empfänger) zurück aus folgenden Einzelzeiten zusammen:

$$Latenz = 3 * \underset{TransmissionDelay}{t} + 4 * (\underset{PropagationDelay}{t} + \underset{QueueingDelay}{t} + \underset{ProcessingDelay}{t}) \tag{5.1}$$

Ein 1000 Byte Paket benötigt für die Übertragung über eine Leitung (1 GBit/s) im Versuchsaufbau 8 µs (Transmission Delay). Da ein Paket, um sein Ziel zu erreichen, über drei Leitungen übertragen werden muss, ergibt sich für das gesamte Transmission Delay 24 µs. Die Queueing Delays sind annähernd nicht vorhanden, da die Queues leer sind. Die Propagation Delays können auf Grund der verwendeten Leitungslänge auch auf annähernd 0 µs abgeschätzt werden. Somit ergibt sich für die Gesamtlatenz:

$$Latenz = 24\mu s + \approx 0\mu s + \approx 0\mu s + \underset{ProcessingDelay}{t} * 4 \tag{5.2}$$

Die exakten Processing Delays der Switche sind unbekannt, liegen aber laut den Herstellern im Mikrosekundenbereich.
Zum Beginn des Versuches wurden ein paar grundlegende Latenzen (einmal im Kreis von Kontron Karte bis wieder zur Kontron Karte) für den Versuchsaufbau gemessen und in Abbildung 5.8 dargestellt. Wenn kein Qbv Scheduling und kein Cross Traffic vorhanden sind (Messung 1), ergeben sich Latenzen im Bereich von ca. 62 µs. In der folgenden zweiten Messung wurde zusätzlich ein Cross Traffic mit maximaler Bandbreite in das Netzwerk eingespielt. Es wurden größere Latenzen gemessen und die Varianz der Latenzen hat sich erhöht. Die Latenzwerte sind aber nach oben hin beschränkt, da der Scheduled Traffic eine höhere PCP Priorität (5) als der Cross Traffic besitzt. Die gemessenen Latenzen sind daher in der Theorie höchstens die Übertragungsdauer eines maximalen Ethernet Frames (ca. 12.5 µs bei 1 GBit/s) größer als die maximale Latenz ohne Cross

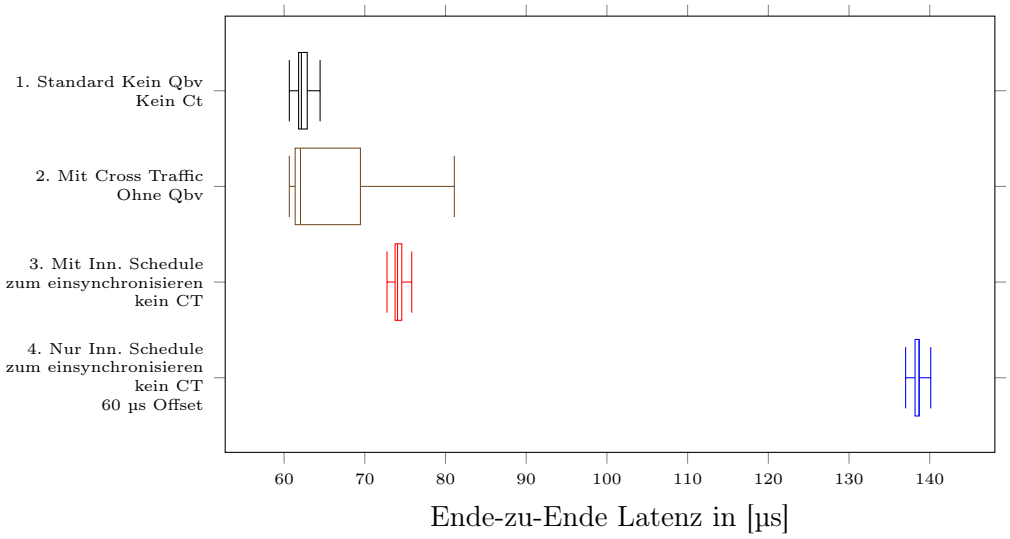


Abbildung 5.8: Diese Abbildung gibt eine Übersicht über im Versuchsaufbau gemessene Ende-zu-Ende Latenzen (Kontron zu Kontron). Auf der Y-Achse werden die verschiedenen Versuchsszenarien 1-4 dargestellt. Auf X-Achse wird die Zeit in Mirkosekunden angegeben.

Traffic. Die Ergebnisse der zweiten Messung zeigen im dargestellten Diagramm sogar eine um maximal 16 µs erhöhte Latenz. Die ersten beiden Messungen haben gezeigt, dass der Cross Traffic die maximale Latenz eines Frames des sicherheitskritischen Scheduled Traffic signifikant erhöhen kann.
Anschließend wurde in der dritten Messung ohne Cross Traffic dem Innoroute Switch ein Schedule zum synchronisierten Absenden des Traffics hinzugefügt. Da der initiale Sender (Kontron Karte) nicht in der Lage ist, den Traffic immer zum exakt gleichen Zeitpunkt abzusenden, musste ein Offset hinzugefügt werden. Das Gate vom InnoRoute Switch ist immer nach exakt 5 ms für 1 µs vom 10 ms Zyklus kurz offen für Pakete mit der Priorität 5. Daher gilt, je früher das Paket an der Kontron Karte initial abgesendet wird, desto wahrscheinlicher würde es, den entsprechenden Zeitslot am InnoRoute Switch erreichen. Um ein Verpassen des Zeitslots für die folgenden Szenarien zu auszuschließen, wurde mit der Wahl eines 60 µs früheren Absendezeitpunkt an der Kontron Karte (Messung 4) fortgefahren. Der InnoRoute hat also nur die Aufgabe den Traffic für die Zeitslots des NXP Switches zu synchronisieren.

In Abbildung 5.9 wird die Untersuchung des Ziels, den Absendezeitpunkt des InnoRoute Switch an den Qbv Schedule des NXP Switch anzupassen, dargestellt. Ein Paket soll ge-

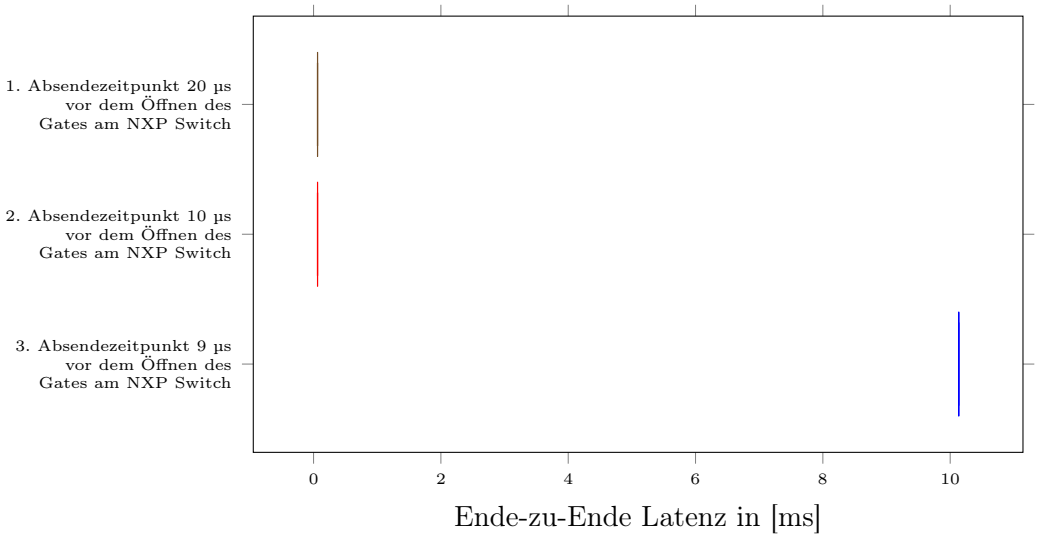


Abbildung 5.9: Diese Abbildung zeigt das knappe Verpassen des Zeitslots bei 9 µs. Dadurch entsteht für alle Pakete eine Verzögerung um einen gesamten Zyklus von 10 ms.

nau dann am NXP Switch ankommen, sobald das Gate für diese Priorität offen ist. Wenn ein Paket nicht rechtzeitig in seinem 1 µs Zeitslot am NXP Switch ankommt, werden die Gates geschlossen und es muss ein voller Zyklus (10 ms) gewartet werden. Dies bedeutet eine Latenz von über 10 ms für alle Pakete des Scheduled Traffic. In der Abbildung 5.9 ist zu erkennen, dass ein Abstand von 10 µs zwischen dem Absenden des Paketes am InnoRoute Switch und dem Öffnen des Gates am NXP Switch optimal ist. Bei 9 µs verpassen alle Pakete den Zeitslot. Das Verpassen des Zeitslots resultiert in Latenzen von über 10 ms.

Das zweite Ziel zur Qualitätssicherung der Umsetzung von Qbv im Versuchsaufbau ist es, die Latenzen mit Cross Traffic und aktiven Schedule im NXP Switch zu untersuchen. Weiterhin synchronisert der InnoRoute in jeder der folgenden Messung den Traffic ein. Abbildung 5.10 zeigt die gemessenen Ende-zu-Ende Latenzen im Vergleich. Es wird die Ende-zu-Ende Latenz (1) von nur einem aktiven Schedule im NXP ohne Cross Traffic dargestellt. Anschließend zeigt der zweite Box-Plot die gemessene Ende-zu-Ende Latenz (2) für keinen aktiven Qbv Schedule im NXP Switch mit vorhandenem Cross Traffic. Als Drittes wird ein Box-Plot sowohl mit aktiven Schedule als auch mit vorhandenem Cross Traffic dargestellt. Aus dem Vergleich der drei Box-Plots ist zu erkennen, dass die Latenzen bei aktiven Schedule und Cross Traffic nahezu den Latenzen des NXP Schedules

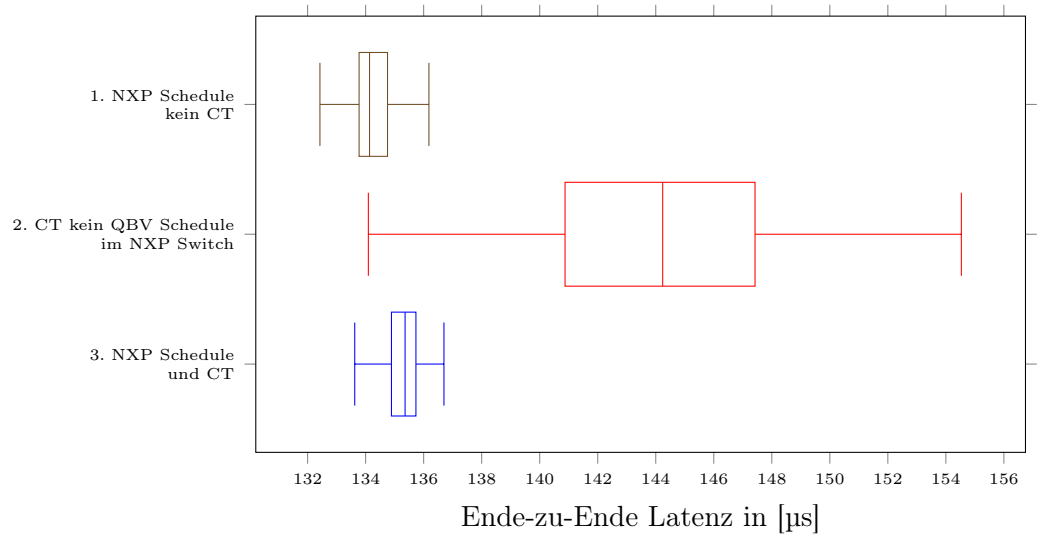


Abbildung 5.10: Diese Abbildung zeigt die Ende-zu-Ende Latenzen von nur einem NXP Schedule (1), nur Cross Traffic (2) und der Kombination von NXP Schedule und Cross Traffic (3).

ohne Cross Traffic entsprechen. Wenn kein NXP Schedule genutzt werden würde, hätte dies Auswirkungen auf die Latenzen wie im zweiten roten Box-Plot erkennbar. In diesem Versuch ist die Latenz aber durch die genutzten PCP Prioritäten nach oben hin begrenzt.

5.3.3 IEEE 802.1Qav

IEEE 802.1Qav wird im Prototyp TSN Aufbau zum Glätten des Videostream Traffic mit der Priorität 5 verwendet. Ein Credit Based Shaper (CBS) am Ausgangsport des NXP Switch zur Kontron Karte stellt sicher, dass der Videostream eine bestimmte reservierte Bandbreite nicht überschreitet. Außerdem würden Zeiträume für mehr Übertragung zur Verfügung stehen, wenn die reservierte Bandbreite durch gleichzeitigen Verkehr nicht erreicht werden würde. Mithilfe der Anwendung tc-cbs kann der Credit Based Shaper konfiguriert werden. Folgende Parameter werden bei der Verwendung vom CBS benutzt (siehe 5.6).

```
tc qdisc replace dev swp3 parent 8001:5 cbs \
locredit -1470 hicredit 30 sendslope -991000 idleslope 9000
```

Listing 5.6: Parameter vom CBS am Port des NXP Switch zur Kontron Karte

Der Parameter idleslope setzt die Bandbreite des Videostreams auf 9 Mbit/s fest (rote Makierung). Der sendslope Parameter gibt mit - 991 Mbit/s die Rate der Kredits, die aufgebraucht werden, wenn eine Übertragung durchgeführt wird, an (grüne Makierung). Die Parameter hicredit und locredit geben die maximal und minimal erreichbare Anzahl an Kredits an. Zur Qualitätssicherung der Eigenschaft, dass der Videostream eine bestimmte reservierte Bandbreite nicht überschreitet, wurde der idleslope Parameter auf 1 Mbit/s festgesetzt und kein weiterer Traffic im Netzwerk verschickt. Anschließend wurde mehrfach die Anzahl der empfangenen Videostream Pakete innerhalb von einer Sekunde betrachtet und mit den maximal erlaubten 1 Mbit/s verglichen. Es war zu erkennen, dass die Kontron Karte nicht mehr als 1 Mbit/s an Videostream Paketen empfängt, obwohl deutlich mehr Pakete gesendet wurden. Im finalen Aufbau kann das IEEE 802.1Qav Feature des NXP Switches nicht verwendet werden, da es aktuell nicht gleichzeitig mit IEEE 802.1Qbv benutzt werden kann. Im optimalen Fall wäre es besser, wenn einer der Raspberry Pis als Sender in der Lage wäre, einen Credit Based Shaper zu verwenden.

5.4 Umsetzung der Anomalieerkennung

Die Anomalieerkennung für die folgenden Versuche wird auf dem Computer, in dem die Kontron Karte eingebaut wurde, durchgeführt (siehe Abbildung 5.11). Der Computer hat Zugriff auf die Netzwerkinterfaces der Kontron Karte und kann so die notwendigen Netzwerkdaten erhalten. Eine Python Anwendung zeichnet mithilfe von PyShark [38] via Socket bestimmten Netzwerkverkehr auf und leitet diesen an ein Modul zur Anomalieerkennung. Ein großer Vorteil von PyShark gegenüber anderen Modulen zum Aufzeichnen von Netzwerkpaketen ist, dass PyShark nur ein Wrapper der Kommandozeilenanwendung Tshark ist und die notwendige Präzision in den Zeitstempeln der Pakete bietet. Aus den aufgezeichneten Netzwerkdaten erstellt die Python Anwendung dann die folgenden vier Metriken Bandbreite, Paketabstand, Paketgröße und Jitter (Differenz des größten und kleinsten Paketabstandes). Weiterhin ist es für die Algorithmen erforderlich, in einer Trainingsphase zuerst das normale Verhalten des Netzwerkverkehrs zu lernen,

bevor die Testphase zur Erkennung von Angriffen startet. Zur Umsetzung der Angriffe, in denen die Algorithmen getestet werden sollen, wurden Skripte erstellt, die gleichzeitig im Hintergrund mit dem System zur Anomalieerkennung gestartet werden. Nach einer festgelegten Zeit wird im Skript dann z. B. das zusätzliche Hinzufügen von Paketen in das Netzwerk gestartet.

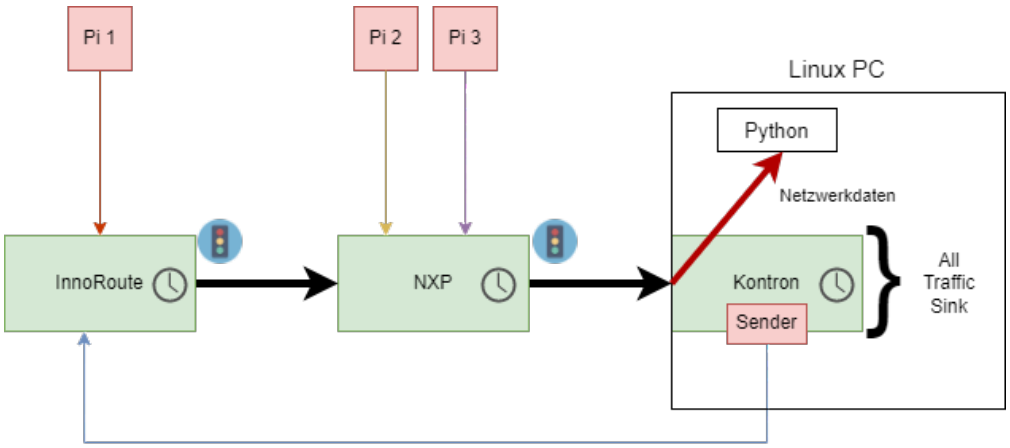


Abbildung 5.11: Die Abbildung zeigt den Punkt, an dem die Anomalieerkennung durchgeführt wird und die Geräte, die an der Anomalieerkennung beteiligt sind.

6 Durchführung und Ergebnisse der Angriffsszenarien

Das Kapitel erläutert die Durchführung sowie die Ergebnisse der Anomalieerkennung in verschiedenen Angriffsszenarien im entwickelten TSN Aufbau. Zuerst betrachtet **6.1** die Anomalieerkennung der verschiedenen Algorithmen innerhalb des Scheduled Traffic Streams, der durch das TSN Feature IEEE 802.1Qbv im NXP Switch gesteuert wird. Anschließend führt **6.2** die Anomalieerkennung auf Ebene des Videostreams durch.

6.1 Anomalieerkennung im Scheduled Traffic Stream

Mit folgender Struktur wird die Anomalieerkennung im Scheduled Traffic Stream durchgeführt. Zuerst bildet **6.1.1** die Trainingsdaten ab und erläutert wichtige Details zum Ablauf des Trainings und den Daten. Anschließend führt **6.1.2** einen Versuch zur Erkennung von Anomalien im regulären Datenverkehr durch. Das Ziel des Versuches ist es, passende Konfigurationen der Algorithmen zu finden und zu überprüfen, ob fälschlicherweise False Positives innerhalb von normalen Daten erkannt werden. Die Kapitel **6.1.3**, **6.1.4**, **6.1.5** und **6.1.6** führen spezifische Angriffsszenarien (Injection, Elimination, Modification und Rescheduling) zum Vergleichen der Algorithmen aus. Kapitel **6.1.7** diskutiert, inwiefern durch Interferenzen auf anderen Streams im Netzwerk Auffälligkeiten in dem System zur Anomalieerkennung entstehen können.

6.1.1 Trainingsdaten

Damit die Algorithmen Daten als Anomalien identifizieren können, muss zuerst das normale Verhalten des Systems in einem Training gelernt werden. Dazu werden Trainingsdaten benötigt. Abbildung **6.1** zeigt die Ergebnisse des durchgeführten Trainings. Das Training wurde über eine Gesamtdauer von 10000 s in Intervallen von 1 s durchgeführt.

Deshalb ist eine Gesamtmenge von 10000 weißen Punkten in der Abbildung erkennbar. Für die Trainingsdaten wurden die zwei Metriken Bandbreite und Jitter ausgewählt. Es wird das Verhalten der Algorithmen bei der Verwendung von zwei Dimensionen von Metriken betrachtet. Die Metrik des Jitters wurde ausgewählt, da sie bei zyklischen Verkehr Auffälligkeiten im Verschieben von Absendezeitpunkten hervorrufen kann und zum Teil die Metrik des Paketabstands mit beinhaltet. Die Metrik der Bandbreite unterscheidet sich deutlich von den Metriken des Paketabstandes/Jitter und beinhaltet zum Teil die Metrik der Paketgröße. Zum Beispiel wird die Bandbreite größer, wenn die Paketgröße erhöht wird. Die Pakete des Scheduled Traffic Streams werden in Zyklen von 10 ms gesendet und daher empfängt die Anwendung zur Anomalieerkennung im Durchschnitt 100 Pakete in einem Intervall. Die empfangenen Frames haben eine Paketgröße von 68 Bytes. Auf der Y-Achse wird die Bandbreite in mbits dargestellt. Innerhalb der Trainingsdaten liegt die durchschnittliche Bandbreite bei den erwarteten 0,054 kbits. Auf der X-Achse wird der Jitter in Sekunden dargestellt. Die Jitterwerte sind im Bereich von 15 µs bis zu 150 µs vorzufinden. Zum Ermitteln der Jitterwerte standen keine Hardware-Zeitstempel der Pakete zur Verfügung, daher sind Abweichungen im Mikrosekundenbereich möglich.

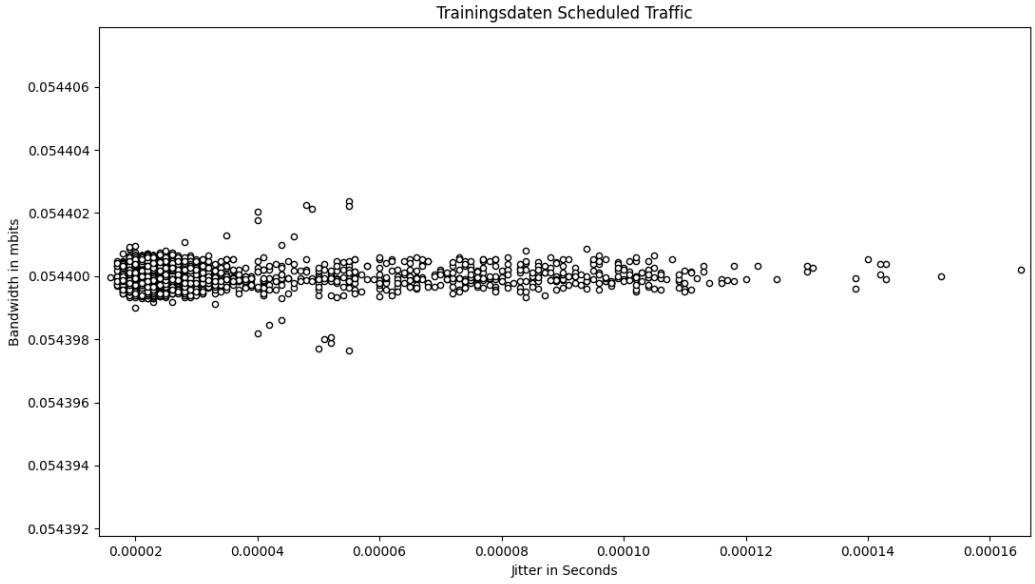


Abbildung 6.1: Die Abbildung bildet die 10000 Trainingsdaten, die für das Training aller Algorithmen verwendet werden, in weißen Punkten ab. Auf der Y-Achse wird die Bandbreite in mbits dargestellt und auf der X-Achse der Jitter in Sekunden.

6.1.2 Normaler Datenverkehr

Das Ziel des ersten Szenarios ist es, verschiedene Konfigurationen aller Algorithmen an der Erkennung von normalen Daten zu testen. Dabei sollte ein guter Algorithmus möglichst keine Daten als Ausreißer einstufen, da in diesem Versuch kein Angriff stattfindet. Aus der Erfahrung von vorherigen Projektarbeiten (Hauptprojekt [54]) konnten bereits bestimmte Konfigurationen der Algorithmen, die besonders schlechte Ergebnisse bei der Erkennung von normalen Daten erzielen, vor der Durchführung des Versuches ausgeschlossen werden. Die Testphase einer Konfiguration eines Algorithmus dauert insgesamt 200 s. In dieser Zeit bewertet ein Algorithmus 200-mal, ob in einem 1 s Intervall eine Anomalie vorliegt. Die Ergebnisse des Versuchs sind in Tabelle **6.1** abgebildet.

Die Ergebnisse zeigen, dass jeder der sechs Algorithmen mit der richtigen Konfiguration in der Lage ist, keine False Positives in einer kleinen Menge von normalen Testdaten vorzufinden. Für die folgenden Angriffsszenarien wurden von jedem Algorithmus jeweils zwei Konfigurationen ausgewählt, die besonders gute Resultate erzielt haben. Diese zwei Konfigurationen sind mit fettgedruckter Schrift in der Tabelle hervorgehoben. Nur für den Algorithmus Isolation Forest wurde eine Konfiguration mit der Precision 0.995, die einen Ausreißer in den normalen Daten erkannt hat, ausgewählt. Eine geringe Anzahl an False Postives ist für ein System zur Anomalieerkennung sehr wichtig, denn schon eine sehr geringe FP-Rate führt zu einem Verlust von Vertrauen in den Mechanismus zur Erkennung von Anomalien.

Algorithmus	Contamination	BEF	Normale Daten	Anomalien	Precision
EE	0.01	-	192	8	0.96
EE	**0.001**	-	**200**	**0**	**1**
EE	**0.0001**	-	**200**	**0**	**1**
IF	0.01	-	196	4	0.98
IF	**0.001**	-	**199**	**1**	**0.995**
IF	**0.0001**	-	**200**	**0**	**1**
SVM	0.05	-	192	8	0.96
SVM	0.01	-	199	1	0.995
SVM	**0.001**	-	**200**	**0**	**1**
SVM	0.0005	-	200	0	1
SVM	**0.0001**	-	**200**	**0**	**1**
HBO	0.01	-	191	9	0.955
HBO	**0.001**	-	**200**	**0**	**1**
HBO	**0.0001**	-	**200**	**0**	**1**
KM1	-	0.4	194	6	0.97
KM1	-	0.5	195	5	0.975
KM1	-	0.6	199	1	0.995
KM1	-	**0.7**	**200**	**0**	**1**
KM1	-	**0.8**	**200**	**0**	**1**
KM1	-	0.9	200	0	1
KM1	-	1	200	0	1
KM1	-	1.1	200	0	1
MS	-	1.3	197	3	0.985
MS	-	1.4	198	2	0.99
MS	-	1.5	199	1	0.995
MS	-	1.6	197	3	0.985
MS	-	1.7	198	2	0.99
MS	-	1.8	198	2	0.99
MS	-	1.9	198	2	0.99
MS	-	**2.0**	**200**	**0**	**1**
MS	-	**2.1**	**200**	**0**	**1**

Tabelle 6.1: Die Tabelle zeigt die Ergebnisse einer Vielzahl von Konfigurationen der Algorithmen im **False Positive Test**. Das optimale Ergebnis für einen Algorithmus wäre es keinen einzigen Ausreißer zuerkennen, da kein Angriff in diesem Szenario stattgefunden hat. Die zwei Konfigurationen der Algorithmen, die besonders gute Resultate erzielt haben und für die folgenden Versuche ausgewählt wurden, sind durch fettgedruckte Schrift hervorgehoben.

6.1.3 Packet Injection

Dieses Szenario testet, welche Konfigurationen der Algorithmen in der Lage sind, einen Angriff zu erkennen, in dem ein Angreifer zusätzliche Pakete in das Netzwerk sendet. Der Packet Injection Angriff läuft so ab, dass über einen Zeitraum von 40 s Pakete mit der PCP Priorität von 6 vor dem InnoRoute Switch zusätzlich ins Netzwerk eingespielt werden. In den ersten 20 s des Angriffsszenarios bleibt die Paketgröße der hinzugefügten Pakete identisch zu den Paketen des Scheduled Traffic. In den hinteren 20 s des Angriffs werden mit 1400 Bytes deutlich größere Frames verwendet. Die korrumpierten Frames werden in demselben Zyklus von 10 ms versendet wie die Frames des regulären Scheduled Traffic. Die Grundannahme ist, dass bei Scheduled Traffic durch das nur kurzzeitig offene Gate für Priorität 6 von 1 µs nahezu keine Anomalien bei gleicher Paketgröße entstehen dürften. Zusätzlich zu den 40 s Angriff läuft in den restlichen 160 s wieder regulärer Nachrichtenverkehr, der von den Algorithmen nicht als Anomalie erkannt werden darf. Die Ergebnisse des Angriffsszenarios Packet Injection sind in Tabelle **6.2** abgebildet.

Algorithmus	Contamination	BEF	TP	FP	FN	Recall	Precision
EE	0.001	-	40	1	0	1	0.98
EE	**0.0001**	-	**40**	**0**	**0**	**1**	**1**
IF	0.001	-	40	2	0	1	0.95
IF	0.0001	-	0	0	40	0	0
SVM	0.001	-	0	0	40	0	0
SVM	0.0001	-	0	0	40	0	0
HBO	0.001	-	40	1	0	1	0.98
HBO	**0.0001**	-	**40**	**0**	**0**	**1**	**1**
KM1	-	0.7	40	1	0	1	0.98
KM1	-	**0.8**	**40**	**0**	**0**	**1**	**1**
MS	-	**2.0**	**40**	**0**	**0**	**1**	**1**
MS	-	**2.1**	**40**	**0**	**0**	**1**	**1**

Tabelle 6.2: Die Tabelle zeigt die Ergebnisse der Algorithmen bei der Erkennung von einem **Packet Injection** Angriff. Das optimale Ergebnis für einen Algorithmus wäre, es genau 40 Ausreißer zuerkennen. Die Konfigurationen der Algorithmen, die alle Angriffe erkannt haben und keine False Positive Alarme ausgelöst haben, sind durch fettgedruckte Schrift hervorgehoben.

Die Ergebnisse des Angriffsszenarios Packet Injection zeigen, dass bestimmte Konfigurationen der Algorithmen in der Lage sind, alle Angriffe erfolgreich zu erkennen und

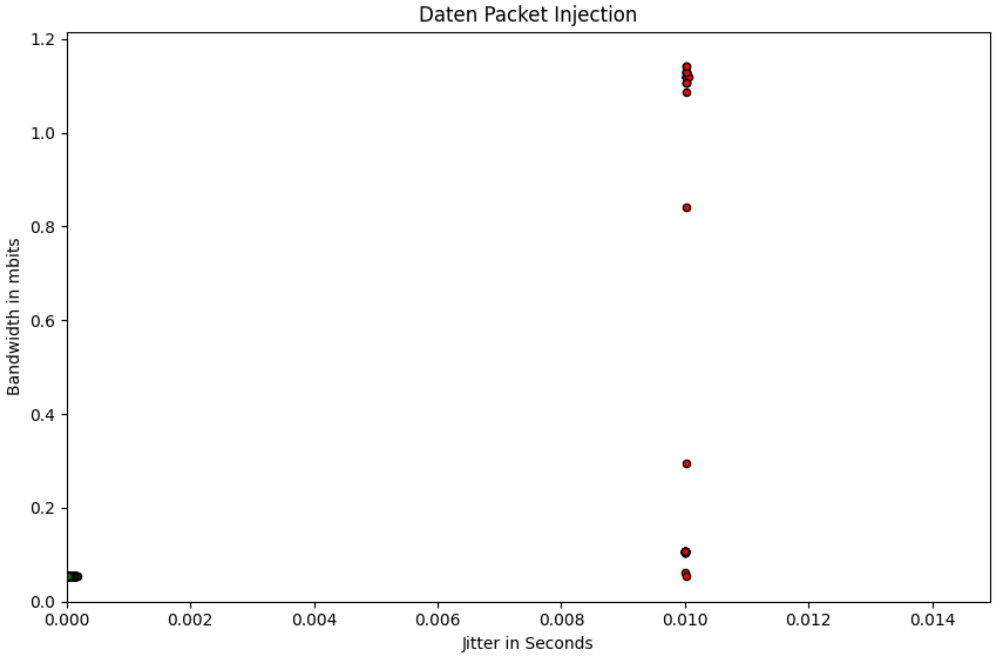


Abbildung 6.2: Die Abbildung stellt beispielhaft die Daten des Angriffsszenarios **Packet Injection** dar. Die Angriffe (rote Punkte) haben alle einen Jitter von über 10 ms und heben sich deutlich von den regulären Daten (grüne Punkte) ab.

zusätzlich alle weiteren regulären Daten nicht als Angriff einzustufen. Denn entgegen der Grundannahme ist es möglich, dass in einem regelmäßigen Zeitfenster für Gates von 1 µs sogar zwei Frames übertragen werden können. Die Implementation von IEEE 802.1Qbv im NXP Switch erlaubt die Übertragung von Paketen, auch wenn die Übertragung in diesem Zeitfenster nicht abgeschlossen werden kann. Da der reguläre Scheduled Traffic Pakete mit einer Größe von 68 Bytes verschickt, ist die Übertragung eines weiteren Paketes in dem Zeitfenster möglich. Dadurch fallen die Daten in der Metrik des Jitters auf. In Abbildung **6.2** sind die Daten des Angriffes eingezeichnet. Die regulären Daten sind als grüne Punkte hervorgehoben und die Anomalien als rote Punkte. In diesem Szenario sind die Angriffe sehr auffällig, da ein zweites Paket pro offenen Zeitfenster den Jitter auf über 10 ms ansteigen lässt. Zusätzlich steigt im zweiten Teil des Angriffes die gemessene Bandbreite durch die erhöhte Paketgröße signifikant. Die Konfigurationen der Algorithmen, die ein optimales Ergebnis erzielt haben, sind in der Tabelle in fettgedruckter Schrift hervorgehoben. Fünf Konfigurationen haben das perfekte Ergebnis erreicht und vier weitere haben alle Angriffe erkannt, haben aber auch fälschlicherweise

1-2 False Positives in den regulären Daten identifiziert. Der Algorithmus SVM und eine Konfiguration von dem Algorithmus Isolation Forest waren überhaupt nicht in der Lage, die Angriffe zu erkennen. Die SVM Konfiguration verwendet einen Polynomkern anstatt einer Radialen Basisfunktion (RBF) als Kernel, da die RBF zu viele False Postives bei regulären Daten erkannt hat. Mit Polynomkern waren zwar keine False Positves in den regulären Daten vorhanden, aber die Grenzen mit niedriger Contamination sind nur nach unten hin präzise modelliert gewesen. Isolation Forest mit der niedrigeren Contamination zeigt ein ähnliches Verhalten in der Modellierung der Grenze. Zur weiteren Betrachtung der Abhängigkeit der Anomalieerkennung von der Dauer, die das Gate für die Priorität offen ist, wurde ein Folgeversuch für das Angriffsszenario Packet Injection durchgeführt. Dabei wurde die Dauer des offenen Gates für Priorität 6 von 1 µs auf 500 ns verändert. Innerhalb von 500 ns ist die Übertragung eines 68 Bytes Paketes noch nicht abgeschlossen, daher können nie zwei Pakete in einem Zeitfenster gesendet werden.

Algorithmus	Contamination	BEF	TP	FP	FN	Recall	Precision
EE	0.001	-	21	1	19	0.525	0.95
EE	0.0001	-	21	0	19	0.525	1
IF	0.001	-	2	0	38	0.05	1
IF	0.0001	-	0	0	40	0	1
SVM	0.001	-	3	0	37	0.075	1
SVM	0.0001	-	0	0	40	0	1
HBO	0.001	-	19	3	21	0.475	0.86
HBO	0.0001	-	19	0	21	0.475	1
KM1	-	0.7	21	1	19	0.525	0.95
KM1	-	0.8	21	0	19	0.525	1
MS	-	2.0	22	0	18	0.55	1
MS	-	2.1	19	0	21	0.525	1

Tabelle 6.3: Die Tabelle zeigt die Ergebnisse der Algorithmen bei der **zweiten Ausführung** der Erkennung von einem **Packet Injection** Angriff. Das Gate im NXP Switch ist für die Priorität 6 anstatt 1 µs nur 0,5 µs offen. Das optimale Ergebnis für einen Algorithmus wäre es, genau 40 Ausreißer zuerkennen. Da keine Konfiguration der Algorithmen optimale Ergebnisse erzielt hat, ist keine Konfiguration durch fettgedruckte Schrift hervorgehoben.

Tabelle **6.3** stellt die Ergebnisse des zweiten Teils des Angriffsszenarios Packet Injection dar. Bei diesem Angriffsszenario wurde der gleiche Angriff durchgeführt und nur die Zeitdauer des offenen Gates für Priorität 6 auf 500 ns reduziert. Keine Konfiguration

der Algorithmen war in der Lage, den ersten Teil des Angriffs zu erkennen, da nur ein Paket pro Zeitfenster verschickt wird. Dadurch ist sowohl die Bandbreite als auch der Jitter der Pakete unauffällig. Der zweite Teil des Angriffs mit größeren Paketen konnte von einigen Konfigurationen der Algorithmen an der erhöhten Bandbreite identifiziert werden. In Abbildung **6.3** sind die Datenpunkte des zweiten Teils des Angriffsszenarios dargestellt. Es ist zu erkennen, dass die Jitterwerte der roten Punkte nicht auffällig sind. Die verwendeten Konfigurationen der Algorithmen haben ähnliche Ergebnisse in der Erkennung wie im ersten Angriffsszenario gezeigt. Nur Isolation Forest und SVM waren nicht der Lage, den zweiten Teil des Angriffs mit erhöhter Paketgröße zu erkennen.

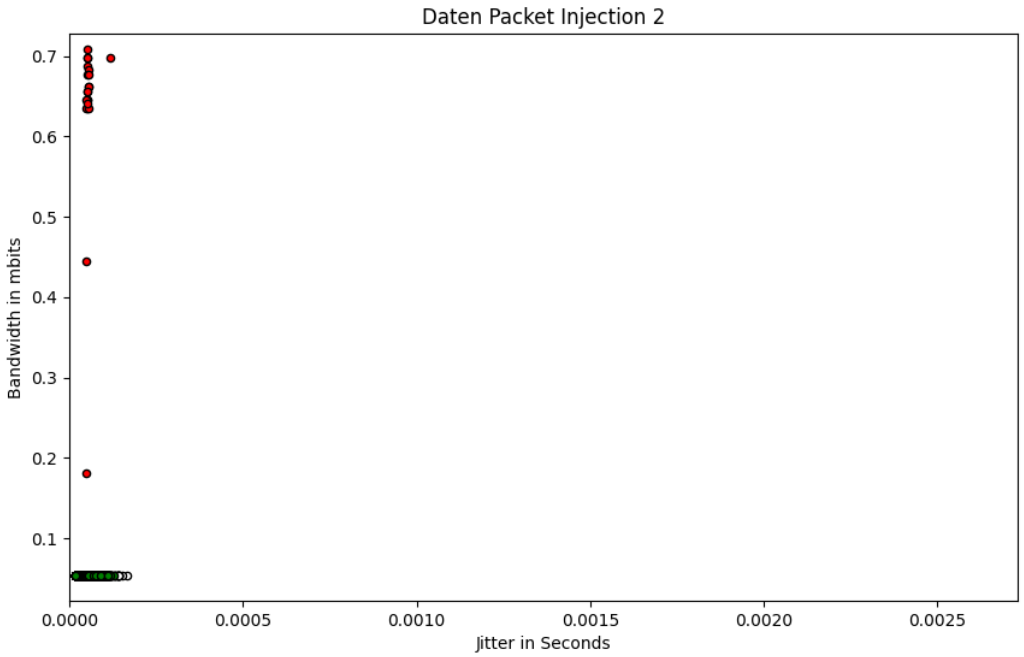


Abbildung 6.3: Die Abbildung stellt beispielhaft die Daten für den zweiten Teil des Angriffsszenarios Packet Injection dar. Die Angriffe (rote Punkte) haben zum Teil eine erhöhte Bandbreite oder befinden sich unauffällig innerhalb der grünen Punkte.

6.1.4 Packet Elimination

Das nächste Angriffsszenario betrachtet das Verhalten der Algorithmen, wenn Pakete aus dem regulären Scheduled Traffic entfernt werden (Packet Elimination). Der Packet Elimination Angriff läuft so ab, dass über einen Zeitraum von 40 s die Sendefrequenz der Pakete verändert wird, so dass insgesamt weniger Pakete gesendet werden. In den ersten 20 s des Angriffes werden die Pakete in einem Intervall von 20 ms anstatt der regulären 10 ms gesendet. Das bedeutet, nur die Hälfte aller regelmäßigen Pakete sind vorhanden und jedes zweite Paket wurde eliminiert. In den zweiten 20 s des Angriffes werden die Pakete in einem Intervall von 35 ms gesendet. Zusätzlich gehen vor und zwischen den Angriffen ein paar Pakete verloren, um die Veränderung der Sendefrequenz zu initialisieren. Die Ergebnisse des Angriffsszenarios sind in Tabelle **6.4** dargestellt.

Algorithmus	Contamination	BEF	TP	FP	FN	Recall	Precision
EE	0.001	-	40	1	0	1	0.98
EE	**0.0001**	-	**40**	**0**	**0**	**1**	**1**
IF	0.001	-	22	1	18	0.55	0.96
IF	0.0001	-	0	0	40	0	0
SVM	0.001	-	20	0	20	0.5	1
SVM	0.0001	-	15	0	25	0.375	1
HBO	0.001	-	22	2	18	0.55	0.92
HBO	0.0001	-	22	0	18	0.55	1
KM1	-	0.7	23	0	17	0.575	1
KM1	-	0.8	21	0	19	0.525	1
MS	-	**2.0**	**40**	**0**	**0**	**1**	**1**
MS	-	**2.1**	**40**	**0**	**0**	**1**	**1**

Tabelle 6.4: Die Tabelle zeigt die Ergebnisse der Algorithmen im Angriffsszenario **Packet Elimination**. Das optimale Ergebnis für einen Algorithmus wäre es, genau 40 Ausreißer zuerkennen. Die Konfigurationen der Algorithmen, die ein optimales Ergebnis erzielt haben, sind in fettgedruckter Schrift hervorgehoben.

Eine Konfiguration von Elliptic Envelope und beide Konfigurationen von Mean Shift waren in der Lage, ausnahmslos alle Angriffe zu identifizieren und keine weiteren False Positive Alarme zu generieren. Andere Konfigurationen konnten Teile des Angriffes entdecken und nur eine Konfiguration von Isolation Forest mit niedriger Contamination hat überhaupt nichts erkannt. In Abbildung **6.4** ist die Anordnung der Datenpunkte des Angriffsszenarios in einem Diagramm dargestellt. Es ist zu erkennen, dass der Großteil

der Datenpunkte durch reduzierte Bandbreite und zum Teil durch erhöhten Jitter auffallen. Die Jitterwerte von über 60 ms und 80 ms entstehen durch die Initialisierungsphase des Angriffskriptes. Mean Shift zeigt gute Ergebnisse bei der Eliminierung von Frames, da der Algorithmus mehrere Einzelcluster (19) um die Trainingsdaten gebildet hat, anstatt ein einzelnes großes Cluster. Dadurch fallen die reduzierte Bandbreitenwerte direkt außerhalb des Clusterbereiches. Auch die elliptische Form des Kreises von Elliptic Envelope trennt die Ausreißer korrekt von den Trainingsdaten. HBO und der Algorithmus K-Means haben durch die unterschiedlichen Größendimensionen der Metriken zu viel Toleranz gegenüber Ausreißern mit ausschließlich reduzierter Bandbreite. SVM modelliert die Grenze der regulären Daten nur hinsichtlich der Bandbreite präzise und nicht hin zu den Jitterwerten. Daher konnte nur ein Teil der Angriffe erkannt werden.

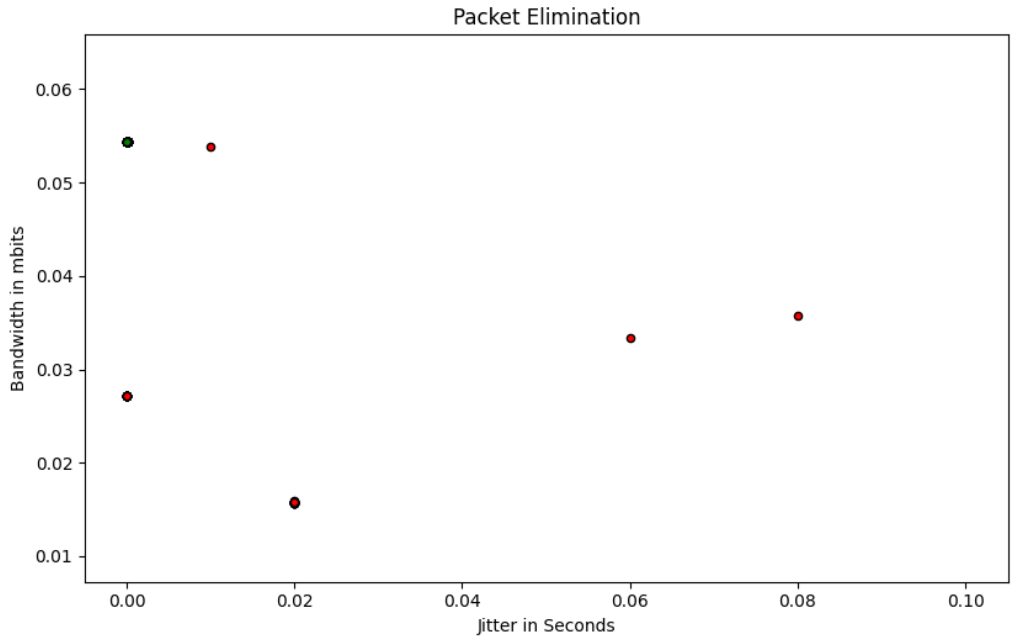


Abbildung 6.4: Die Abbildung stellt beispielhaft die Daten des Angriffsszenarios **Packet Elimination** dar. Alle roten Punkte haben eine reduzierte Bandbreite gegenüber den regulären Daten und zum Großteil einen auffälligen Jitterwert. An den roten und grünen Punkte überlappen zum Teil die Punkte ineinander, so dass nur ein einzelner Punkt durch den großen Zoom erkennbar ist.

6.1.5 Packet Modification

Das Angriffsszenario Packet Modification verändert zwischenzeitlich den Payload der Pakete des Scheduled Traffic. Beim Packet Modification Angriff wird die Gesamtgröße des Paketes im ersten Teil des Angriffes von 68 Bytes auf 80 Bytes erhöht. Im zweiten Teil des Angriffes wird die Paketgröße dann auf 1400 Bytes vergrößert. Die Ergebnisse des Angriffsszenarios sind in Tabelle **6.5** dargestellt.

Algorithmus	Contamination	BEF	TP	FP	FN	Recall	Precision
EE	**0.001**	-	**40**	**0**	**0**	**1**	**1**
EE	**0.0001**	-	**40**	**0**	**0**	**1**	**1**
IF	0.001	-	0	0	40	0	0
IF	0.0001	-	0	0	40	0	0
SVM	0.001	-	0	1	40	0	0
SVM	0.0001	-	0	0	40	0	0
HBO	0.001	-	22	2	18	0.55	0.92
HBO	0.0001	-	22	0	18	0.55	1
KM1	-	0.7	22	1	18	0.55	0.96
KM1	-	0.8	22	0	18	0.55	1
MS	-	2.0	22	0	18	0.55	1
MS	-	2.1	22	0	18	0.55	1

Tabelle 6.5: Die Tabelle zeigt die Ergebnisse der Algorithmen im Angriffsszenario **Packet Modification**. Das optimale Ergebnis für einen Algorithmus wäre es, genau 40 Ausreißer zuerkennen. Die Konfigurationen der Algorithmen, die ein optimales Ergebnis erzielt haben, sind in fettgedruckter Schrift hervorgehoben.

Der einzige Algorithmus, der in diesem Szenario alle Angriffe erkannt hat, war Elliptic Envelope. Weitere Konfigurationen bestimmter Algorithmen waren in der Lage, ein Teil des Angriffes zu entdecken. Die beiden Algorithmen SVM und Isolation Forest haben den gesamten Angriff nicht erkannt. In Abbildung **6.5** werden die Datenpunkte des Angriffes gezeigt. Es ist zu erkennen, dass ca. die Hälfte der Anomalien eine sehr deutlich erhöhte Bandbreite vorweist und die andere Hälfte nur eine minimal erhöhte Bandbreite. Die Algorithmen Mean Shift, HBO und K-Means hatten Schwierigkeiten die Punkte, die dicht oberhalb der regulären Daten lagen, als Anomalien zu identifizieren. Die Clusterkreise von K-Means und Mean Shift umschließen die anomalen Datenpunkte, da der/die Kreis/Kreise durch die Jitterwerte in den Trainingsdaten zu groß sind. Dagegen modelliert Elliptic Envelope mit der elliptischen Form eine flachere Grenze, die eine minimal

erhöhte Bandbreite nicht umschließt. Die Behälter von HBO sind ebenfalls ein wenig zu hoch und erkennen die Anomalien, die dicht an den regulären Daten liegen, nicht. SVM und Isolation Forest haben nur nach unten eine präzise modellierte Grenze und haben mit der vorliegenden Konfiguration große Probleme, selbst offensichtliche Ausreißer zu erkennen.

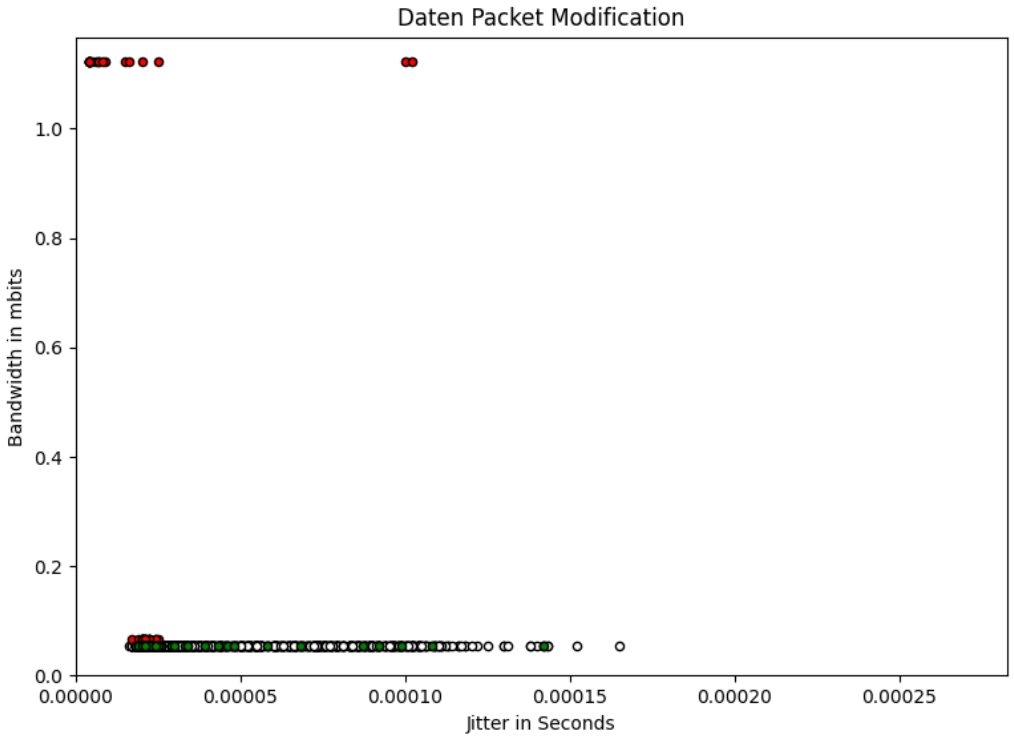


Abbildung 6.5: Die Abbildung stellt beispielhaft die Daten des Angriffsszenarios **Packet Modification** dar. Die roten Punkte des Angriffes heben sich durch eine erhöhte Bandbreite von den regulären Daten ab.

6.1.6 Packet Rescheduling

Das Angriffsszenario Packet Rescheduling verändert den Offset der Pakete des Scheduled Traffic. Für den Angriff startet ein Skript den Scheduled Traffic mit veränderten Offset Parameter zwischen 0 ms - 10 ms erneut. Bei der Ausführung des Angriffsszenarios gingen durch die Initialisierungsphase der Anwendung Pakete verloren. Dadurch beinhaltet dieser Angriff zum Teil auch Packet Elimination und einzelne Jitterwerte waren sehr

auffällig. Durch das Einsynchronisieren der Pakete am InnoRoute Switch war ansonsten nichts Relevantes zu beobachten. Daher und durch die Ähnlichkeiten zum Packet Elimination Angriff werden die Ergebnisse diese Angriffsszenarios nicht abgebildet.

6.1.7 Inteferenzen mit anderen Streams und andere unentdeckbare Angriffsmöglichkeiten

In weiteren Versuchen wurde beobachtet, ob z. B. durch Pakete mit anderen PCP Prioritäten oder maximalen Cross Traffic Inteferenzen im Scheduled Traffic Stream entstehen könnten. Dazu wurde das System zur Anomalieerkennung in einem Angriff betrachtet, der eine Vielzahl von Paketen der höheren Priorität 7 ins Netzwerk einspielt. Da Priorität 6 immer ein exklusives offenes Zeitfenster im NXP Switch hat, waren wie erwartet keine Auswirkungen auf den Scheduled Traffic sichtbar. Denn Pakete aller anderen Prioritäten sind nicht in der Lage, die Pakete des Scheduled Traffic zu verzögern. Angreifer, die den Inhalt der Pakete modifizieren würden, ohne dabei die Paketgröße zu verändern, könnten nicht durch die Systeme zur Anomalieerkennung mit den verwendeten Metriken entdeckt werden. Auch Angriffe, die nur dem Netzwerkverkehr lauschen (z. B. Eavesdropping) oder die Reihenfolge der Pakete vertauschen ohne Änderung der Timings sind nicht zu erkennen. Sobald aber ein Paket das offene Zeitfenster am NXP Switch verpasst, kann direkt durch Auffälligkeiten im Jitter und der Bandbreite sehr deutlich ein Ausreißer identifiziert werden.

6.2 Anomalieerkennung im Videostream

Das Kapitel Anomalieerkennung im Videostream fokussiert sich auf die Durchführung und Ergebnisse bei der Anomalieerkennung in verschiedenen Angriffsszenarien im Videostream. Zuerst beschreibt **6.2.1** den regulären Datenverkehr des Videostreams und den Ablauf des Trainings. Anschließend ermittelt **6.2.2** in einem False Positive Test passende Konfigurationen für alle Algorithmen, die in den folgenden Angriffsszenarien verwendet werden. Es werden Angriffsszenarien zu Packet Injection **6.2.3**, Packet Elimination **6.2.4** und Packet Modification **6.2.5** durchgeführt. Zum Abschluss des Kapitels diskutiert **6.2.6** Interferenzen mit Streams anderer Prioritäten und weitere Angriffsmöglichkeiten, die möglicherweise nicht erkannt werden können.

6.2.1 Trainingsdaten

Für die Anomalieerkennung im Videostream wurde zuerst das normale Verhalten des Systems in einem Training gelernt. Die Trainingsdaten des Videostreams sind in Abbildung **6.6** dargestellt. Das Training wurde über eine Gesamtdauer von 10000 s in Intervallen von 1 s durchgeführt. Daher sind insgesamt 10000 weiße Punkte in der Abbildung erkennbar. Als Metriken für die Anomalieerkennung im Videostream wurden die beiden Metriken Bandbreite und Paketgröße ausgewählt. Es wird das Verhalten der Algorithmen bei der Verwendung von zwei Dimensionen von Metriken betrachtet. Die Metrik der Paketgröße wurde anstatt des Jitters verwendet, da der Jitter beim Videostream über gesamten Wertebereich verteilt ist und Ausreißer nahezu nicht von normalen Daten unterschieden werden können. Dagegen ist die Paketgröße beim Videostream immer gleich und Ausreißer mit veränderter Paketgröße könnten über diese Metrik direkt identifiziert werden. Die Pakete des Videostreams werden mit PCP Priorität 5 von einem Raspberry Pi über den NXP Switch an die Kontron Karte geschickt. Bei dem Video handelt es sich um eine 4k Aufnahme mit 60 fps mit einer Gesamtdauer von 2 min. Das Video wird in Dauerschleife abgespielt. Alle Pakete haben mit 1362 Bytes die gleiche Paketgröße. Auf der X-Achse wird die Paketgröße in Bytes dargestellt. Daher sind die Punkte in der Abbildung als vertikale Linie angeordnet. Auf der Y-Achse wird die Bandbreite in mbits dargestellt. Die Bandbreite des Videostreams schwankt je nach Intervall von 20 mbits bis zu 95 mbits mit einer kleineren Lücke im Bereich zwischen 40 mbits bis 55 mbits. Die Lücke könnte durch die reduzierte Dauer und ständige Wiederholung des Videos

entstehen. Eine längere Aufnahme aus dem Straßenverkehr würde möglicherweise den gesamten Bereich der Bandbreite abbilden und die Qualität der Trainingsdaten erhöhen.

Abbildung 6.6: Die Abbildung bildet die 10000 Trainingsdaten, die für das Training aller Algorithmen verwendet werden, in weißen Punkten ab. Auf der Y-Achse wird die Bandbreite in mbits dargestellt und auf der X-Achse die Paketgröße in Bytes.

6.2.2 Normaler Datenverkehr

Das Ziel des ersten Szenarios ist es, verschiedene Konfigurationen aller Algorithmen an der Erkennung von normalen Daten zu testen, um zu betrachten, ob die Algorithmen False Positive in den regulären Daten erkennen. Ein guter Algorithmus sollte in diesem Szenario möglichst keine Datenpunkte als Ausreißer einstufen, da in diesem Versuch kein Angriff stattfindet. Aus der Erfahrung von vorherigen Projektarbeiten (Hauptprojekt [54]) konnten bereits bestimmte Konfigurationen der Algorithmen, die besonders schlechte Ergebnisse bei der Erkennung von normalen Daten erzielen, vor der Durchführung des Versuches ausgeschlossen werden. Die Testphase einer Konfiguration eines Algorithmus dauert insgesamt 200 s. In dieser Zeit bewertet ein Algorithmus 200-mal, ob in einem 1 s Intervall eine Anomalie vorliegt.

Algorithmus	Contamination	BEF	Normale Daten	Anomalien	Precision
EE	0.01	-	198	2	0.99
EE	**0.001**	-	**198**	**0**	**0.99**
EE	**0.0001**	-	**200**	**0**	**1**
IF	**0.01**	-	**200**	**0**	**1**
IF	**0.001**	-	**200**	**0**	**1**
IF	0.0001	-	200	0	1
SVM	0.01	-	199	1	0.995
SVM	0.005	-	199	1	0.995
SVM	**0.001**	-	**200**	**0**	**1**
SVM	0.0005	-	199	1	0.995
SVM	**0.0001**	-	**199**	**1**	**0.995**
HBO	0.01	-	199	1	0.995
HBO	**0.001**	-	**200**	**0**	**1**
HBO	**0.0001**	-	**200**	**0**	**1**
KM1	-	0.6	190	10	0.95
KM1	-	0.7	195	5	0.975
KM1	-	0.8	199	1	0.995
KM1	-	0.9	198	2	0.99
KM1	-	**1.0**	**200**	**0**	**1**
KM1	-	**1.1**	**200**	**0**	**1**
KM1	-	1.2	200	0	1
MS	-	0.8	162	38	0.81
MS	-	0.9	169	31	0.845
MS	-	**1.0**	**200**	**0**	**1**
MS	-	**1.1**	**200**	**0**	**1**
MS	-	1.2	200	0	1
MS	-	1.4	200	0	1
MS	-	1.6	200	0	1
MS	-	1.8	200	0	1
MS	-	2.0	200	0	1

Tabelle 6.6: Die Tabelle zeigt die Ergebnisse einer Vielzahl von Konfigurationen der Algorithmen vom **False Positive Test** im Videostream. Das optimale Ergebnis für einen Algorithmus wäre es, keinen einzigen Ausreißer zuerkennen, da kein Angriff in diesem Szenario stattgefunden hat. Die zwei Konfigurationen der Algorithmen, die besonders gute Resultate erzielt haben und für die folgenden Versuche ausgewählt wurden, sind durch fettgedruckte Schrift hervorgehoben.

In Tabelle **6.6** sind die Ergebnisse des Versuches dargestellt. Beim False Positive Test am regulären Datenverkehr hat sich gezeigt, dass jeder Algorithmus mindestens eine Konfiguration hat, die das optimale Ergebnis erzielt hat. Das optimale Ergebnis bedeutet, dass keine False Positives erkannt wurden. Für die folgenden Angriffsszenarien wurden von jedem Algorithmus jeweils zwei Konfigurationen ausgewählt, die besonders gute Resultate erzielt haben. Diese zwei Konfigurationen sind mit fettgedruckter Schrift in der Tabelle hervorgehoben. Beim Algorithmus Isolation Forest und beim Algorithmus SVM (Polynomkern) wurde jeweils eine Konfiguration ausgewählt, die fälschlicherweise einen/zwei Ausreißer in den regulären Daten identifiziert hat. Zwar ist schon eine sehr geringe FP-Rate bei einer kleinen Menge an Testdaten kein gutes Resultat, aber möglicherweise zeigt die Konfiguration vielversprechende Ergebnisse bei der Erkennung von Angriffen. Durch leichte Veränderung der Parameter könnte das Verhalten dann beim regulären Datenverkehr verbessert werden.

6.2.3 Packet Injection

Das Ziel des ersten Angriffsszenarios ist es, zu untersuchen, welche Konfigurationen der Algorithmen in der Lage sind, Angriffe mit hinzugefügten Paketen (Packet Injection) zuerkennen. Der Packet Injection Angriff läuft so ab, dass ein Skript zuerst über einen Zeitraum von 20 s in jeder Sekunde zusätzlich 50 Pakete der gleichen Paketgröße und Priorität absendet. 40 Pakete der Paketgröße 1362 Bytes bedeutet ca. 4 mbits zusätzliche Bandbreite. Im zweiten Teil des Angriffs werden für 20 s 400 Pakete pro Intervall hinzugefügt (ca. 40 mbits). Zusätzlich zu den 40 s Angriff läuft in den restlichen 160 s wieder regulärer Nachrichtenverkehr, der von den Algorithmen nicht als Anomalie erkannt werden darf. Die Ergebnisse des Angriffsszenarios Packet Injection sind in Tabelle **6.7** abgebildet.

An den Ergebnissen in der Tabelle ist zu erkennen, dass keine Konfiguration der Algorithmen in der Lage war, alle Angriffe zu erkennen. Einige Konfiguration haben einen Teil des Angriffes identifiziert. In Abbildung **6.7** sind die Datenpunkte des Angriffs dargestellt. Ein paar anomale Datenpunkte fallen durch eine erhöhte Bandbreite gegenüber den Trainingsdaten auf. Weitere anomale Datenpunkte befinden sich inmitten der Trainingsdaten. In der Abbildung sind fälschlicherweise ein paar anomale Datenpunkte als grüne Datenpunkte abgebildet, da kein Algorithmus sie korrekt als Ausreißer identifizieren konnte. Die besten Ergebnisse bei der Erkennung von Packet Injection im Videost-

Algorithmus	Contamination	BEF	TP	FP	FN	Recall	Precision
EE	0.001	-	4	0	36	0.1	1
EE	0.0001	-	0	0	40	0	0
IF	0.01	-	0	2	40	0	0
IF	0.001	-	0	1	40	0	0
SVM (Poly)	0.001	-	3	0	37	0.075	1
SVM (Poly)	0.0001	-	0	0	40	0	0
SVM (RBF)	0.001	-	20	5	20	0.5	0.8
SVM (RBF)	0.0001	-	18	0	22	0.45	1
HBO	0.001	-	14	0	26	0.35	1
HBO	0.0001	-	0	0	40	0	0
KM1	-	1.0	0	0	40	0	0
KM1	-	1.1	0	0	40	0	0
MS	-	1.0	22	0	18	0.55	1
MS	-	1.1	20	0	20	0.5	1

Tabelle 6.7: Die Tabelle zeigt die Ergebnisse der Algorithmen bei der Erkennung von einem **Packet Injection** Angriff im Videostream. Das optimale Ergebnis für einen Algorithmus wäre es, genau 40 Ausreißer zuerkennen. Da keine Konfiguration der Algorithmen optimale Ergebnisse erzielt hat, ist keine Konfiguration durch fettgedruckte Schrift hervorgehoben.

ream hat der Algorithmus Mean Shift erreicht. Es wurden 22 von 40 Angriffen korrekt identifiziert. Zur Erkennung der Ausreißer hat Mean Shift insgesamt vier Cluster gebildet. Die obere Grenze des obersten Clusters liegt direkt auf dem Punkt der Trainingsdaten mit der höchsten Bandbreite. Alle Punkte mit größerer Bandbreite ordnet Mean Shift als Anomalien ein. Beim anderen Clustering Algorithmus K-Means ist das gesamte einzelne Cluster zu groß, da der Mittelpunkt des Clusters im oberen Teil beim Großteil der Trainingsdaten liegt. Deswegen muss das Cluster groß genug sein, um reguläre Daten mit niedriger Bandbreite nicht als Ausreißer einzustufen. Dies führt aber dazu, dass für Werte mit größerer Bandbreite der gleiche Radius zur Verfügung steht. Der Algorithmus SVM wurde in den Konfigurationen mit einem Polynomkern und einer Radialen Basisfunktion (RBF) als Kernel ausgeführt. Die Konfiguration mit Radialer Basisfunktion konnte ein Teil des Angriffes erkennen, indem eine Kreisform um die Daten modelliert wurde. In der Konfiguration mit Contamination von 0.001 sind aber fünf False Positive aufgetreten. Die Konfiguration mit Polynomkern hat das Problem, dass mit einer Trennlinie nur der untere Teilbereich präzise modelliert ist. HBO konnte in der Konfiguration mit niedriger Konfiguration auch ein Teil des Angriffs erkennen. Die andere Konfiguration von HBO,

der Algorithmus Isolation Forest und eine Konfiguration vom Algorithmus Elliptic Envelope konnten keinen Angriff identifizieren. Die Grenzen dieser Algorithmen liegen zu weit oberhalb der Trainingsdaten. In dem Angriffsszenario ist aufgefallen, dass eine minimale Erhöhung der Bandbreite mit wenigen zusätzlichen Paketen nahezu von keinem Algorithmus entdeckt werden kann.

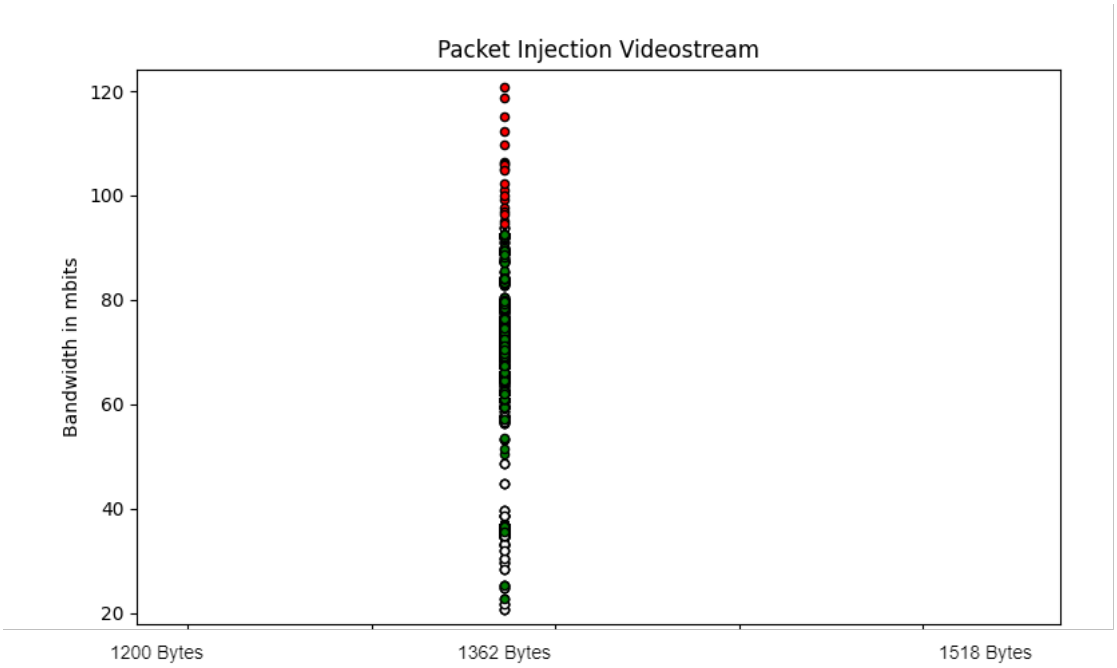


Abbildung 6.7: Die Abbildung stellt beispielhaft die Daten vom Angriffsszenario **Packet Injection** im Videostream dar. Zum Teil befinden sich Ausreißer innerhalb der grünen Punkte und sind schwer von den regulären Daten zu unterscheiden.

6.2.4 Packet Elimination

Das nächste Angriffsszenario betrachtet das Verhalten der Algorithmen, wenn mehrere Pakete aus den regulären Videostream Daten entfernt werden (Packet Elimination). Bei dem Packet Elimination Angriff werden im ersten Teil des Angriffs die Hälfte aller Pakete über eine Dauer von 20 s eliminiert. Im zweiten Teil des Angriffs werden über 20 s 2/3 aller Pakete vom Videostream entfernt. Zusätzlich zu den 40 s Angriff läuft in den restlichen 160 s wieder regulärer Nachrichtenverkehr, der von den Algorithmen nicht als Anomalie erkannt werden darf. Die Ergebnisse des Angriffsszenarios Packet Elimination sind in Tabelle **6.8** abgebildet

Algorithmus	Contamination	BEF	TP	FP	FN	Recall	Precision
EE	0.001	-	25	0	15	0.625	1
EE	0.0001	-	6	0	34	0.15	1
IF	0.01	-	29	1	11	0.725	0.97
IF	0.001	-	25	1	15	0.625	0.96
SVM (Poly)	0.001	-	24	1	16	0.6	0.96
SVM (Poly)	0.0001	-	4	0	36	0.1	1
SVM (RBF)	0.001	-	13	0	27	0.325	1
SVM (RBF)	0.0001	-	4	0	36	0.1	1
HBO	0.001	-	5	0	35	0.125	1
HBO	0.0001	-	0	0	40	0	0
KM1	-	1.0	4	0	36	0.1	1
KM1	-	1.1	1	0	39	0.025	1
MS	-	1.0	3	0	37	0.075	1
MS	-	1.1	3	0	37	0.075	1

Tabelle 6.8: Die Tabelle zeigt die Ergebnisse der Algorithmen bei der Erkennung von einem **Packet Elimination** Angriff im Videostream. Das optimale Ergebnis für einen Algorithmus wäre es, genau 40 Ausreißer zuerkennen. Da keine Konfiguration der Algorithmen optimale Ergebnisse erzielt hat, ist keine Konfiguration durch fettgedruckte Schrift hervorgehoben.

Auch im Angriffsszenario Packet Elimination konnte keine Konfiguration eines Algorithmus alle Angriffe korrekt erkennen. Viele Konfigurationen waren in der Lage, einen kleinen Teil des Angriffs zu erkennen. Abbildung **6.8** stellt die Datenpunkte des Angriffsszenarios dar. Ein paar Ausreißer fallen durch eine Bandbreite, die deutlich niedriger als die Bandbreite der regulären Daten ist, auf. Der andere Teil der anomalen Datenpunkte

befindet sich innerhalb der regulären Daten und ist schwierig zu identifizieren. In der Abbildung sind fälschlicherweise ein paar anomale Datenpunkte als grüne Datenpunkte abgebildet, da kein Algorithmus sie korrekt als Ausreißer identifizieren konnte. Das beste Ergebnis ohne False Positives hat eine Konfiguration des Algorithmus Elliptic Envelope erreicht. Auch der Algorithmus Isolation Forest sowie eine Konfiguration von SVM hat bei der Erkennung trotz eines False Positives gute Ergebnisse gezeigt. SVM mit Polynomkern und Isolation Forest bilden am unteren Ende der Daten eine präzise Grenze und sind deswegen besser bei der Erkennung von Packet Elimination Angriffen als Packet Injection Angriffen geeignet. Die beiden Clustering Algorithmen K-Means und Mean Shift konnten nur einzelne Punkte als Anomalie identifizieren. Nur wenige Angriffspunkte unterschreiten die untere Clustergrenze und die Anomalien sind eher durch die Vielzahl an Werten mit niedriger Bandbreite hintereinander zu erkennen. Außerdem fehlen während der 40 s des Angriffs größere Bandbreitenwerte über die gesamte Dauer des Angriffs. Dadurch haben die Algorithmen, die auch innerhalb des modellierten Bereiches für normale Daten bei Häufung an der unteren Grenze Ausreißer erkennen, mehr Angriffe identifizieren können.

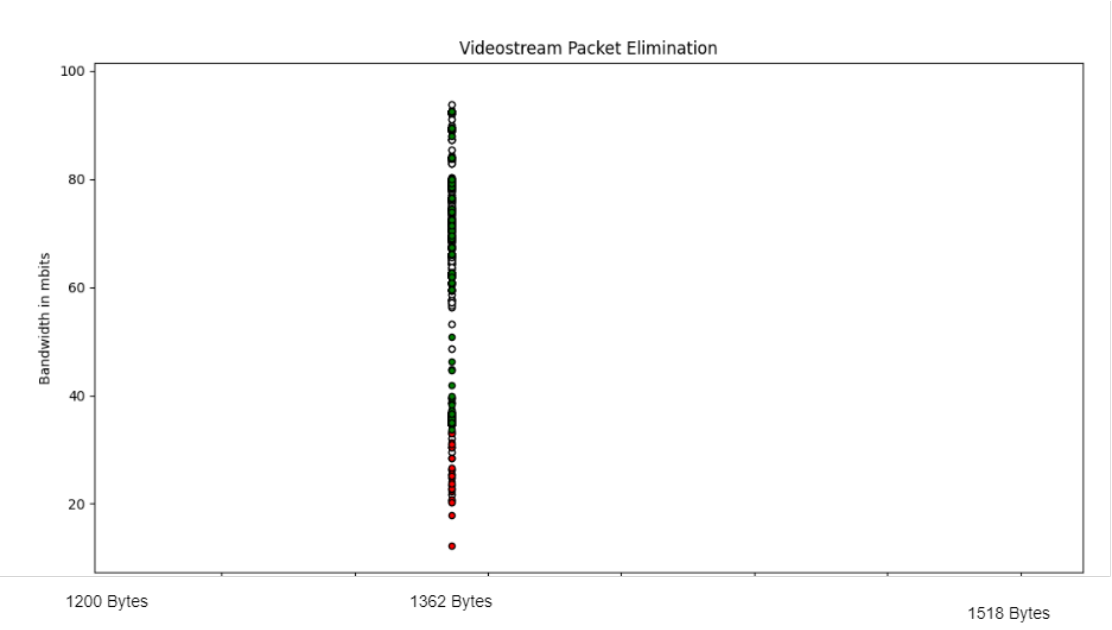


Abbildung 6.8: Die Abbildung stellt beispielhaft die Daten vom Angriffsszenario **Packet Elimination** im Videostream dar. Zum Teil befinden sich Ausreißer innerhalb der grünen Punkte und sind schwer von den regulären Daten zu unterscheiden.

6.2.5 Packet Modification

Das Angriffsszenario Packet Modification betrachtet das Verhalten der Algorithmen, wenn sich die Paketgröße der Frames gegenüber den regulären Frames des Videostreams verändert. Der Angriff läuft in zwei Teilen ab. Zuerst wird über eine Dauer von 20 s die Paketgröße der Frames um 20 Bytes auf 1382 Bytes erhöht. Im zweiten Teil des Angriffs wird die Paketgröße um 200 Bytes auf 1162 Bytes reduziert. Zusätzlich zu den 40 s Angriff läuft in den restlichen 160 s wieder regulärer Nachrichtenverkehr, der von den Algorithmen nicht als Anomalie erkannt werden darf. Die Ergebnisse des Angriffsszenarios Packet Modification sind in Tabelle **6.9** abgebildet.

Algorithmus	Contamination	BEF	TP	FP	FN	Recall	Precision
EE	0.001	-	1	0	39	0.025	1
EE	0.0001	-	0	0	40	0	0
IF	0.01	-	2	1	38	0.05	0.33
IF	0.001	-	1	0	39	0.025	1
SVM (Poly)	0.001	-	20	0	20	0.5	1
SVM (Poly)	0.0001	-	20	0	20	0.5	1
SVM (RBF)	0.001	-	23	0	17	0.575	1
SVM (RBF)	0.0001	-	22	0	18	0.55	1
HBO	0.001	-	4	0	36	0.1	1
HBO	0.0001	-	0	0	40	0	0
KM1	-	1.0	20	0	20	0.5	1
KM1	-	1.1	20	0	20	0.5	1
MS	-	**1.0**	**40**	**0**	**0**	**1**	**1**
MS	-	**1.1**	**40**	**0**	**0**	**1**	**1**

Tabelle 6.9: Die Tabelle zeigt die Ergebnisse der Algorithmen bei der Erkennung von einem **Packet Modification** Angriff im Videostream. Das optimale Ergebnis für einen Algorithmus wäre es, genau 40 Ausreißer zuerkennen. Die Konfigurationen der Algorithmen, die alle Angriffe erkannt haben und keine False Positive Alarme ausgelöst haben, sind durch fettgedruckte Schrift hervorgehoben.

Die Ergebnisse zeigen, dass nur der Algorithmus Mean Shift das optimale Ergebnis erzielt hat. Weitere Konfigurationen der Algorithmen haben ein Teil des Angriffes erkannt. Abbildung **6.9** stellt die Datenpunkte aus dem Angriffsszenario Packet Modification dar. Die Datenpunkte des ersten Teils des Angriffes sind mit leicht erhöhter Bandbreite (1382 Bytes) erkennbar und die Datenpunkte des zweiten Teils des Angriffes sind mit deut-

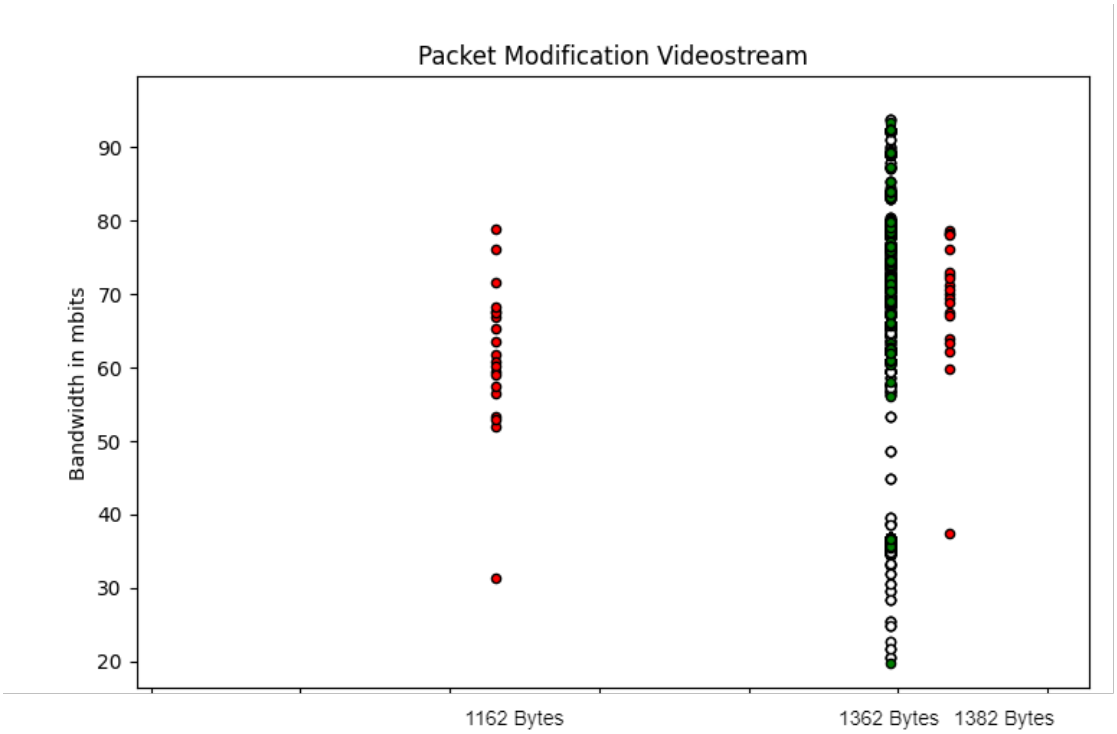


Abbildung 6.9: Die Abbildung stellt beispielhaft die Daten vom Angriffsszenario **Packet Modification** im Videostream dar. Die Ausreißer unterscheiden sich durch die veränderte Paketgröße von den regulären Daten.

lich reduzierter Bandbreite (1162 Bytes) sichtbar. Der Algorithmus Mean Shift war in der Lage, auch eine kleine Veränderung der Paketgröße zu bemerken, da durch die vier kleineren Cluster um die Trainingsdaten herum die Kreise in Richtung der Paketgröße nicht so ausgedehnt sind wie bei anderen Algorithmen. Zum Beispiel bildet K-Means ein einzelnes großes Cluster, dass die Datenpunkte mit leicht erhöhter Paketgröße mit umschließt. Deswegen erkennt K-Means nur die Hälfte aller Datenpunkte von Angriffen als Anomalien. Auch der Algorithmus SVM erkennt ca. die Hälfte der Anomalien. Die Konfiguration mit einer Radialen Basisfunktion (RBF) modelliert einen Kreis, der die leicht erhöhten Datenpunkte mit beinhaltet. Der Kreis ist ein wenig enger und zwei Datenpunkte mit größerer Bandbreite werden bei der erhöhten Paketgröße korrekterweise als Anomalien identifiziert. SVM mit Polynomkern bildet eine Trennlinie, die nur eine reduzierte Paketgröße als Ausreißer erkennt. Die Algorithmen Elliptic Envelope, HBO und Isolation Forest konnten kaum Teile des Angriffes entdecken. Alle drei Algorithmen verwenden Grenzen, die nur sehr große Abweichungen in der Paketgröße oder Kombina-

tionen von abweichender Paketgröße und Bandbreite als anomale Datenpunkte einstufen.

6.2.6 Interferenzen mit anderen Streams und andere unentdeckbare Angriffsmöglichkeiten

In weiteren Versuchen wurde beobachtet, ob z. B. durch Pakete mit anderen PCP Prioritäten oder maximalen Cross Traffic, Interferenzen im Videostream entstehen könnten. Dazu wurde zuerst die Anomalieerkennung im Videostream betrachtet mit zusätzlich maximalen Cross Traffic im Netzwerk. Da die PCP Priorität vom Videostream höher als vom Cross Traffic ist, konnten wie erwartet keine Auswirkungen auf die Anomalieerkennung beobachtet werden. Im nächsten Versuch wurde die Priorität des Cross Traffics auf 7 erhöht. Dadurch ist die Priorität des Cross Traffics höher als vom Videostream. Da der Link nicht voll ausgelastet war und keine zeitkritischen Metriken betrachtet wurden, konnte auch bei Cross Traffic mit PCP Priorität 7 keine Auswirkungen auf die Anomalieerkennung beobachtet werden. Die Zeitsynchronisation verwendet auch die Priorität 7 und durch eine Vielzahl an Warnmeldungen war erkennbar, dass Cross Traffic der Priorität 7 zuerst Auswirkungen auf den Jitter der Zeitsynchronisation hat. Angreifer, die den Inhalt der Pakete des Videostreams verändern würden, ohne dabei die Paketgröße zu verändern, könnten nicht durch die Systeme zur Anomalieerkennung mit den verwendeten Metriken entdeckt werden. Auch Angriffe, die nur dem Netzwerkverkehr lauschen (z. B. Eavesdropping), die Reihenfolge der Pakete vertauschen oder das Timing der Pakete verändern, sind nicht zu erkennen. Anomalien in einem Videostream Stream zu erkennen, scheint im Vergleich zum Scheduled Traffic deutlich schwieriger zu sein, da sensible Änderungen im Timing oder minimale Veränderungen in der Anzahl der Pakete kaum auffallen.

7 Evaluation der Algorithmen und Erkenntnisse

Das Kapitel evaluiert und diskutiert übergeordnet die Ergebnisse der Durchführung der Anomalieerkennung im entwickelten TSN Aufbau. Zuerst fasst **7.1** die Ergebnisse der Angriffsszenarien zusammen und führt übergeordnet den Vergleich der Algorithmen durch. Anschließend arbeitet **7.2** die wichtigsten Erkenntnisse aus den Versuchen heraus und listet diese als Erkenntnisse nacheinander auf.

7.1 Vergleich der Algorithmen

Zum Vergleich der Algorithmen werden zuerst die Ergebnisse aus den Angriffsszenarien im Scheduled Traffic und Videostream ausgewertet. In Tabelle **7.1** sind die Ergebnisse der Algorithmen zusammengefasst und einander gegenübergestellt. Die Algorithmen wurden auf einer Skala von 1 (bestes Ergebnis) - 5 (schlechtestes Ergebnis) abhängig von erzielten Recall/Precision Wert bewertet. Anschließend wurde sowohl für den Scheduled Traffic als auch für den Videostream eine Gesamtzahl zusammen addiert. Es wurde jeweils die beste Konfiguration von einem Algorithmus für die Bewertung in dem Angriffsszenario ausgewählt.

Der Algorithmus Elliptic Envelope hat das beste Gesamtergebnis bei der Erkennung von Anomalien im Scheduled Traffic erreicht. Das zweitbeste Ergebnis hat der Algorithmus Mean Shift gezeigt, gefolgt von HBO und K-Means. Sowohl Isolation Forest als auch SVM hatten große Schwierigkeiten, selbst deutlich erkennbare Ausreißer zu identifizieren. Elliptic Envelope ist gut für die Erkennung von Anomalien im Scheduled Traffic geeignet, weil eine Ellipse um die zentralen Datenpunkte herum angepasst wird. Die Form der Ellipse ist sehr passend zu den Trainingsdaten und die Annahme, dass die Ausreißerdaten nach einer Gauß-Verteilung angeordnet sind sowie die Mahalanobis-Distanz zum

Szenario	Isolation Forest	SVM	EE	HBO	K-Means	Mean Shift
Reguläre Daten ST	1	1	1	1	1	1
Injection ST	2	5	1	1	1	1
Injection ST 2	4	4	3	3	3	3
Elimination ST	4	3	1	3	3	1
Modification ST	5	5	1	3	3	3
Gesamt ST	**16**	**18**	**7**	**11**	**11**	**9**
Reguläre Daten Video.	1	1	1	1	1	1
Injection Video.	5	3	4	3	5	3
Elimination Video.	3	3	2	4	4	4
Modification Video.	4	3	4	4	3	1
Gesamt Video	**13**	**10**	**11**	**12**	**13**	**9**

Tabelle 7.1: Die Tabelle stellt die übergeordneten Ergebnisse der Algorithmen in einer Tabelle dar. Es wird weitestgehend die beste Konfiguration eines Algorithmus in einem Angriffsszenario bewertet. Die Bewertung wird in einer Skala von 1-5 durchgeführt. (1) = optimales Ergebnis, (2) = überdurchschnittlich gutes Ergebnis, das den Großteil des Angriffes identifiziert, (3) = ca. die Hälfte des Angriffs wurde erkannt (4) = ein kleiner Teil des Angriffes wurde erkannt, (5) = der Angriff wurde nicht erkannt

Zentrum ermöglichen eine gute Erkennung der Ausreißer. Mean Shift konnte ebenso gute Ergebnisse erzielen, da die gebildeten Einzelcluster (19 Cluster) mit reduzierten Radien weniger Platz umschließen als ein einzelnes großes Cluster wie bei K-Means. Nur eine minimal erhöhte Bandbreite konnte von Mean Shift nicht erkannt werden, da die Radien der Einzelcluster höher als die elliptische Form vom Elliptic Envelope Algorithmus sind. SVM mit Polynomkern war nicht in der Lage, den Großteil der Angriffe zu erkennen, da ein Polynomkern mit Grad 3 nur eine einfache Trennlinie bildet, die die Daten nur an einer Seite präzise abgrenzt. Daher konnten nur Ausreißer mit niedriger Bandbreite identifiziert werden. Die Wahl einer anderen Kernelfunktion oder ein höherer Grad könnten durchaus bessere Ergebnisse erzielen (RBF-Kernel hatte zu viele False Positives). Isolation Forest hatte auch große Schwierigkeiten, offensichtliche Ausreißer zu erkennen. Die Gründe dafür sind die niedrige Contamination und ein Entscheidungsraum, der nur Werte mit niedriger Bandbreite oder Jitter direkt als Ausreißer eingestuft hat. Mit weiter optimierter Konfiguration ist es vermutlich möglich, dass Ergebnis aller Algorithmen in den Angriffsszenarien des Scheduled Traffics zu verbessern.

Bei der Anomalieerkennung im Videostream hat sich gezeigt, dass die Anomalien bei Packet Injection Angriffen oder Packet Elimination Angriffen schwieriger zu erkennen sind als im Scheduled Traffic. Das beste Ergebnis in den Angriffsszenarien im Videostream hat der Algorithmus Mean Shift erreicht. Anschließend folgen die Algorithmen SVM, Elliptic Envelope und HBO. Die wenigsten Angriffe im Videostream haben die Algorithmen Isolation Forest und K-Means identifiziert. Mean Shift ist gut für die Erkennung von Ausreißern im Videostream geeignet, da der Algorithmus insgesamt vier Cluster mit keiner Abhängigkeit vom zentralsten Punkt der Daten bildet. Dadurch sind die Cluster in Richtung der Paketgröße nicht so ausgedehnt. So ist es möglich, Ausreißer mit minimaler Erhöhung der Paketgröße zu identifizieren. SVM wurde aufgrund der schlechten Ergebnisse bei der Erkennung mit einem Polynomkern zusätzlich in der Konfiguration mit einer Radialen Basisfunktion als Kernel ausgeführt. Bei dieser Konfiguration hat SVM als Ergebnis einen Kreis modelliert, der die obere Grenze an den höchsten Wert der Bandbreite aus den Trainingsdaten und die untere Grenze an dem niedrigsten Wert der Bandbreite anpasst. Dadurch waren bei allen drei Angriffsszenarien offensichtliche Ausreißer identifizierbar. Der Clustering Algorithmus K-Means hat nicht so gute Ergebnisse erzielt, weil ein einzelnes Cluster gebildet wird mit einem Zentrum an der Stelle mit den meisten Trainingsdaten. Daher muss der Radius des Kreises groß genug sein, um keine False Positives mit niedriger Bandbreite zu erkennen. Dadurch bietet der Kreis aber zu viel Platz für Ausreißer mit größerer Bandbreite. Elliptic Envelope konnte im Vergleich zu den sehr guten Ergebnissen bei der Erkennung von Ausreißern im zyklischen Scheduled Traffic nur mittelmäßige Ergebnisse bei der Erkennung von Ausreißern im Videostream erzielen. Die elliptische Form des Algorithmus war nicht so gut geeignet um eine konstante, immer gleiche Paketgröße und eine Bandbreite, die einen größeren Wertebereich ausfüllt, effizient zu umschließen. Mit weiteren Optimierungen der Konfigurationen der Algorithmen wäre es vermutlich möglich, die Ergebnisse aller Algorithmen in den Angriffsszenarien des Videostreams zu verbessern.

Bei den vorherigen Untersuchungen verschiedener Angriffsszenarien im CAN-Bus Stream [54] konnte auch der Algorithmus Elliptic Envelope sehr gute Ergebnisse erreichen. Der Traffic im CAN-Bus Stream wurde ähnlich wie der Scheduled Traffic zyklisch alle 80 ms durch das Netzwerk verschickt. Die Metrik des Jitters und der Bandbreite wurde ebenso verwendet. Durch die Metrik des Jitters war es für einen Angreifer nur schwer möglich gewesen, ein einzelnes Paket hinzuzufügen oder zu entfernen, ohne dass ein guter Algorithmus den Angriff erkennen würde. Im Gegensatz zum Scheduled Traffic waren

die Auffälligkeiten im Jitter nicht so deutlich wie beim Verpassen eines 802.1Qbv Gates im Scheduled Traffic. Auch bei der Durchführung der Anomalieerkennung in diesem Netzwerk im Videostream haben sich ähnliche Tendenzen gezeigt. Die beiden Clustering Algorithmen Mean Shift und K-Means sowie SVM konnten die meisten Angriffe von allen verwendeten Algorithmen erkennen. In diesem Versuch wurde ein anderes längeres Video (ca. 30 min) mit deutlich kleinerer Auflösung und Bildrate verwendet.

7.2 Erkenntnisse

- Die Anomalieerkennung in einem TSN Netzwerk mit IEEE 802.1Qbv in einem geschedulten Stream hängt von der Länge des offenen Gates und der Art und Weise der Implementation vom TSN Standard im Switch ab. Ein TSN Switch kann die Übertragung von Frames, deren Übertragung in dem Zeitfenster nicht mehr abgeschlossen werden kann, noch durchführen oder verhindern (Guard Band am Ende eines Zeitfensters). Der TSN Switch von NXP und der Switch von InnoRoute haben trotz Überschreiten des Zeitfensters einen weiteren Frame abgeschickt. Im offiziellen TSN Standard oder innerhalb von OMNeT++ Simulationen von TSN Netzwerken wäre die Übertragung eines Frames, die über das Zeitfenster hinaus andauert, nicht möglich gewesen. Je nachdem, ob ein zusätzlicher Frame in dem offenen Zeitfenster übertragen werden kann, führt dies zu sehr deutlichen Auffälligkeiten beim Jitter. Diese Auffälligkeiten sind für ein System zur Anomalieerkennung einfach zu identifizieren. Wenn im Gegensatz dazu z. B. nur ein Frame pro Zeitfenster übertragen werden kann, ist bei gleicher Paketgröße durch die Metriken Jitter und Bandbreite für ein System zur Anomalieerkennung nahezu nichts zu erkennen.
- Der Algorithmus Elliptic Envelope ist gut für Anomalieerkennung von zyklischen Daten mit den Metriken Jitter und Bandbreite geeignet. Sowohl bei den Angriffsszenarien im Scheduled Traffic als auch bei vorherigen Untersuchungen zu Angriffsszenarien im CAN-Bus Stream wurden die besten Ergebnisse vom Algorithmus Elliptic Envelope erzielt. Die elliptische Form mit Anpassung an die zentralen Datenpunkte des Algorithmus passte gut zu der Anordnung der Daten. Durch diese Form und die Annahme, dass die Ausreißerdaten nach einer Gauß-Verteilung angeordnet sind sowie die Mahalanobis-Distanz zum Zentrum ermöglichten eine gute Erkennung der Ausreißer.

- Die Metrik des Jitters gibt die Differenz des größten und kleinsten Paketabstandes in einem Intervall an. Für die Anomalieerkennung im zyklischen Nachrichtenverkehr hat sich die Metrik des Jitters als sehr nützlich herausgestellt. Durch den Jitter sind häufig Änderungen im Timing einzelner Pakete deutlich als Anomalie erkennbar gewesen. Sowohl das Hinzufügen als auch das Entfernen von Paketen konnten in den durchgeführten Szenarien durch die Metrik des Jitters zuverlässig erkannt werden.

- Anomalien mit den Metriken Bandbreite und Paketgröße im Videostream sind schwieriger zu entdecken als im Scheduled Traffic. Wenige hinzugefügte Pakete oder entfernte Pakete sind durch die große Anzahl an Paketen kaum über die Bandbreite zu erkennen. Die besten Ergebnisse bei der Erkennung hat der Clustering Algorithmus Mean Shift gezeigt und der Klassifikationsalgorithmus SVM. Mean Shift ist gut für die Erkennung von Ausreißern im Videostream geeignet, da insgesamt vier Cluster gebildet werden. Die vier Cluster haben im Gegensatz zu anderen Algorithmen keine direkte Abhängigkeit zum zentralsten Punkt der Trainingsdaten. Dadurch war der Algorithmus in der Lage, auch Anomalien mit nur minimal veränderter Paketgröße zu identifizieren.

- Der verwendete Algorithmus muss auf die Daten des Streams und die Metriken angepasst sein. Je mehr über die vorliegenden Daten bekannt ist, desto genauer kann der Algorithmus auf die Anomalieerkennung angepasst werden. Algorithmen, die in bestimmten Angriffsszenarien große Schwierigkeiten haben, offensichtliche Ausreißer zu erkennen, sind teilweise unerwartet präzise in anderen Angriffsszenarien gewesen.

8 Fazit und Ausblick

Dieses Kapitel schließt die Arbeit mit einem Fazit und Ausblick ab. Zuerst fasst **8.1** den Inhalt der Arbeit zusammen. Anschließend präsentiert **8.2** die Ergebnisse der Arbeit und **8.3** diskutiert offene Fragen sowie weiterführende Arbeitsmöglichkeiten.

8.1 Zusammenfassung

Durch die Integration von Time Sensitive Networking in das zukünftige Fahrzeugnetzwerk können Anforderungen an garantierte Bandbreite, eine maximale Latenz und kein Paketverlust durch Buffer Overflow in einer Queue im Netzwerk realisiert werden. Durch das Verschicken von sicherheitskritischen Traffic über die gleiche Leitung wie nicht sicherheitskritischen Traffic und der vermehrten Öffnung nach Außen werden angepasste Sicherheitskonzepte zum Schutz der Komponenten im Netzwerk benötigt. Ein möglicher Bestandteil des zukünftigen Sicherheitskonzepts des Fahrzeuges könnte die Entdeckung von Angriffen durch die Anomalieerkennung sein.
In dieser Arbeit wurden Methoden der Anomalieerkennung im Rahmen eines TSN basierten Fahrzeugnetzwerks untersucht. Dazu wurde der aktuelle Stand der Anomalieerkennung im Fahrzeugnetzwerk in Verbindung mit TSN herausgearbeitet. Es wurden verschiedene Kategorisierungen von Algorithmen zur Anomalieerkennung aus der Literatur dargestellt und Kriterien zum Vergleich von Algorithmen definiert. Anschließend wurde ein Teilausschnitt eines TSN basierten Fahrzeugnetzwerks in Hardware konstruiert. Sechs Algorithmen zur Anomalieerkennung wurden in verschiedenen Angriffsszenarien in einem Scheduled Traffic und einem Videostream untersucht. Die Ergebnisse der Algorithmen wurden miteinander verglichen und ausgewertet.

8.2 Ergebnisse

Bei der Anomalieerkennung im Scheduled Traffic Stream wurde festgestellt, dass die Anomalieerkennung von der Länge des offenen Gates (GCL) und der Art und Weise der IEEE 802.1Qbv Implementation im Switch abhängt. Der verwendete TSN Switch von NXP und der Switch von InnoRoute haben trotz Überschreiten des Zeitfensters einen weiteren Frame abgeschickt (kein Guardband). Je nachdem, ob ein zusätzlicher Frame in dem offenen Zeitfenster übertragen werden kann, ist es sehr einfach oder kaum möglich, Anomalien beim Einfügen von Paketen zu entdecken. Anomalien im Scheduled Traffic in Angriffsszenarien, in denen Pakete entfernt werden, sind sehr auffällig. Der Wert des Jitters beim Entfernen eines einzelnen Paketes steigt direkt um mehrere Dimensionen auf mindestens die Dauer bis zum erneuten Öffnen des Gates an.
Der Algorithmus Elliptic Envelope ist gut für Anomalieerkennung von zyklischen Daten mit den Metriken Jitter und Bandbreite geeignet. Die elliptische Form mit Anpassung an die zentralen Datenpunkte und die statischtische Annahme des Algorithmus, dass die regulären Daten aus einer Gauß-Verteilung stammen, passte gut zur Anordnung der Daten.
Die Metrik des Jitters bildet die Differenz des größten und kleinsten Paketabstandes in einem Intervall ab. Der Wert des Jitters war sehr auffällig bei der Erkennung von Packet Injection oder Packet Elimination im zyklischen Traffic. Dies ermöglichte das direkte Identifizieren vieler Ausreißer.
Bei der Anomalieerkennung im Videostream mit den verwendeten Metriken Bandbreite und Paketgröße ist es in vielen Szenarien signifikant schwieriger gewesen, Anomalien zu entdecken als im Scheduled Traffic. Die besten Ergebnisse bei der Erkennung hat der Clustering Algorithmus Mean Shift gezeigt.
Je mehr über die vorliegenden Daten des Streams und verwendeten Metriken bekannt ist, desto genauer kann der passende Algorithmus ausgewählt und auf die Anomalieerkennung angepasst werden.

8.3 Ausblick

Folgende weiterführende Tätigkeiten ergeben sich aus den Untersuchungen dieser Arbeit. Viele der Algorithmen bieten eine Vielzahl an Möglichkeiten zur spezifischen Konfiguration der Algorithmen. Durch eine bessere Konfiguration könnten möglicherweise die

Ergebnisse aller Algorithmen in den Angriffsszenarien optimiert werden. Einzelne Algorithmen wie Elliptic Envelope oder Mean Shift, die besonders vielversprechende Ergebnisse gezeigt haben, könnten spezifisch weiterbetrachtet werden. Es könnten mehr als zwei Dimensionen von Metriken zur Anomalieerkennung verwendet werden. Weiterhin könnte untersucht werden, ob die Algorithmen auch in der Lage sind, aufgezeichnete Angriffe aus Angriffsdatenbanken zuverlässig zu erkennen. Auch das Verhalten der Anomalieerkennung bei Verwendung anderer TSN Feature als IEEE 802.1Qbv könnte analysiert werden. Zur Einordnung der Ergebnisse müssten die Algorithmen direkt gegen andere Ansätze wie z. B. die Anomalieerkennung über Per-Stream Filtering and Policing (PSFP) im TSN Netzwerk verglichen werden.

Literaturverzeichnis

[1] *802.1AE: MAC Security (MACsec).* https://1.ieee802.org/security/802-1ae/. – Accessed: 2021-12-6

[2] *802.1AS - Timing and Synchronization.* https://www.ieee802.org/1/pages/802.1as.html. – Accessed: 2021-11-25

[3] *802.1BA - Audio Video Bridging (AVB) Systems.* https://www.ieee802.org/1/pages/802.1ba.html. – Accessed: 2021-12-1

[4] *802.1Qav - Forwarding and Queuing Enhancements for Time-Sensitive Streams.* https://www.ieee802.org/1/pages/802.1av.html. – Accessed: 2021-12-1

[5] *Configuring PTP Using ptp4l.* https://docs.fedoraproject.org/en-US/fedora/latest/system-administrators-guide/servers/Configuring_PTP_Using_ptp4l/. – Accessed: 2022-18-8

[6] *IEEE 1588-2008 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.* https://standards.ieee.org/standard/1588-2008.html. – Accessed: 2021-11-25

[7] *IEEE 802.1CB-2017 - Frame Replication and Elimination for Reliability.* https://standards.ieee.org/standard/802_1CB-2017.html. – Accessed: 2021-12-6

[8] *IEEE 802.1Qci-2017 - Per-Stream Filtering and Policing.* https://standards.ieee.org/standard/802_1Qci-2017.html. – Accessed: 2021-12-6

[9] *InnoRoute.* https://innoroute.com/trustnode_router/. – Accessed: 2022-23-11

[10] *Kontron PCIE-0400-TSN Network Interface Card.* https://www.kontron.com/de/produkte/pcie-0400-tsn-network-interface-card/p151637. – Accessed: 2022-3-12

[11] *LS1021A Time-Sensitive Networking Reference Design.* https://www.nxp.com/docs/en/fact-sheet/LS1021ATSNRDA4FS.pdf. – Accessed: 2022-23-11

[12] *Security for Vehicular Information.* https://secvi.inet.haw-hamburg.de/. – Accessed: 2023-7-1

[13] *Time-Sensitive Networking (TSN) Task Group.* https://1.ieee802.org/tsn/. – Accessed: 2021-11-5

[14] IEEE Standard for Ethernet Amendment 5: Specification and Management Parameters for Interspersing Express Traffic. In: *IEEE Std 802.3br-2016 (Amendment to IEEE Std 802.3-2015 as amended by IEEE St802.3bw-2015, IEEE Std 802.3by-2016, IEEE Std 802.3bq-2016, and IEEE Std 802.3bp-2016)* (2016), S. 1–58

[15] IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks – Amendment 26: Frame Preemption. In: *IEEE Std 802.1Qbu-2016 (Amendment to IEEE Std 802.1Q-2014)* (2016), S. 1–52

[16] Abts, Dietmar ; Mülder, Wilhelm: *Informationssicherheit.* S. 601–635. In: *Grundkurs Wirtschaftsinformatik: Eine kompakte und praxisorientierte Einführung.* Wiesbaden : Springer Fachmedien Wiesbaden, 2017. – URL https://doi.org/10.1007/978-3-658-16379-2_17. – ISBN 978-3-658-16379-2

[17] Bhuyan, Monowar H. ; Bhattacharyya, D. K. ; Kalita, J. K.: Network Anomaly Detection: Methods, Systems and Tools. In: *IEEE Communications Surveys Tutorials* 16 (2014), Nr. 1, S. 303–336

[18] Brunner, Stefan ; Roder, Jurgen ; Kucera, Markus ; Waas, Thomas: Automotive E/E-architecture enhancements by usage of ethernet TSN. In: *2017 13th Workshop on Intelligent Solutions in Embedded Systems (WISES)*, 2017, S. 9–13

[19] Chan, Lulu ; Vermeulen, Bart ; Concer, Nicola ; Automotive, Philip Axer-NXP: Architecting Network Latencies for Mixed Criticality In-Vehicle Applications. (2018)

[20] Chandola, Varun ; Banerjee, Arindam ; Kumar, Vipin: Anomaly Detection: A Survey. In: *ACM Comput. Surv.* 41 (2009), Juli, Nr. 3, S. 15:1–15:58. – URL https://dl.acm.org/doi/10.1145/1541880.15418822. – ISSN 0360-0300

[21] Checkoway, Stephen ; McCoy, Damon ; Kantor, Brian ; Anderson, Danny ; Shacham, Hovav ; Savage, Stefan ; Koscher, Karl ; Czeskis, Alexei ; Roesner, Franziska ; Kohno, Tadayoshi: Comprehensive Experimental Analyses of Automotive Attack Surfaces. In: *Proceedings of the 20th USENIX Conference on Security*. USA : USENIX Association, 2011 (SEC'11), S. 6

[22] Comaniciu, D. ; Meer, P.: Mean shift: a robust approach toward feature space analysis. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), Nr. 5, S. 603–619

[23] Davis, Andy: Broadcasting your attack: security Testing DAB Radio in Cars. In: *Black Hat USA, Las Vegas, Nevada, USA* (2015)

[24] Duan, Xuting ; Yan, Huiwen ; Tian, Daxin ; Zhou, Jianshan ; Su, Jian ; Hao, Wei: In-Vehicle CAN Bus Tampering Attacks Detection for Connected and Autonomous Vehicles Using an Improved Isolation Forest Method. In: *IEEE Transactions on Intelligent Transportation Systems* (2021), S. 1–13

[25] Eckert, C.: *IT-Sicherheit: Konzepte – Verfahren – Protokolle*. De Gruyter, 2018 (De Gruyter Studium). – URL https://books.google.de/books?id=kI1uDwAAQBAJ. – ISBN 9783110563900

[26] Finn, Norman: Introduction to Time-Sensitive Networking. In: *IEEE Communications Standards Magazine* 2 (2018), Nr. 2, S. 22–28

[27] Freitas, Paulo ; Pinheiro, Antônio ; Kaddoum, Georges ; Campelo, Divanilson ; Soares, Fabio: An Efficient Intrusion Prevention System for CAN: Hindering Cyber-Attacks With a Low-Cost Platform. In: *IEEE Access* PP (2021), 12, S. 1–1

[28] Fujimaki, Ryohei ; Yairi, Takehisa ; Machida, Kazuo: An Approach to Spacecraft Anomaly Detection Problem Using Kernel Feature Space. In: *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. New York, NY, USA : ACM, 2005 (KDD '05), S. 401–410. – URL http://doi.acm.org/10.1145/1081870.1081917. – ISBN 1-59593-135-X

[29] Garcia-Teodoro, P.: Anomaly-based network intrusion detection: Techniques, systems and challenges. In: *Computers and Security* 28 (2009), Nr. 1, S. 18–28. – URL https://www.sciencedirect.com/science/article/pii/S0167404808000692. – ISSN 0167-4048

[30] GmbH, Wirecard T.: *DATA and ANALYTICS – ANOMALIEERKENNUNG IN ZEITREIHEN*. 2018. – URL https://www.wirecard.de/data-analytics/data-stories/anomalieerkennung/. – Zugriffsdatum: 2021-12-18

[31] Gmiden, Mabrouka ; Gmiden, Mohamed H. ; Trabelsi, Hafedh: An intrusion detection method for securing in-vehicle CAN bus. In: *2016 17th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, 2016, S. 176–180

[32] Goldstein, Markus ; Dengel, Andreas: Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm, 09 2012

[33] Grimm, Daniel ; Weber, Marc ; Sax, Eric: An Extended Hybrid Anomaly Detection System for Automotive Electronic Control Units Communicating via Ethernet - Efficient and Effective Analysis using a Specification- and Machine Learning-based Approach, 01 2018, S. 462–473

[34] Hirano, Kohei ; Ito, Yoshihiro: Study on Appropriate IdleSlope Value of Credit Based Shaper for QoS Control on In-Vehicle Ethernet. In: *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*, 2020, S. 686–687

[35] Hoyle, Ben ; Rau, Markus M. ; Paech, Kerstin ; Bonnett, Christopher ; Seitz, Stella ; Weller, Jochen: Anomaly detection for machine learning redshifts applied to SDSS galaxies. In: *Monthly Notices of the Royal Astronomical Society* 452 (2015), 08, Nr. 4, S. 4183–4194. – URL https://doi.org/10.1093/mnras/stv1551. – ISSN 0035-8711

[36] Häckel, Timo ; Schmidt, Anja ; Meyer, Philipp ; Korf, Franz ; Schmidt, Thomas C.: *Strategies for Integrating Controls Flows in Software-Defined In-Vehicle Networks and Their Impact on Network Security*. 2020. – URL https://arxiv.org/abs/2010.03839

[37] Ji, Yonghyeok ; Lee, Hyeongcheol: Event-Based Anomaly Detection Using a One-Class SVM for a Hybrid Electric Vehicle. In: *IEEE Transactions on Vehicular Technology* 71 (2022), Nr. 6, S. 6032–6043

[38] KimiNewt: *PyShark*. https://kiminewt.github.io/pyshark/. 2021

[39] Lima, Moisés F. ; Zarpelão, Bruno B. ; Sampaio, Lucas D. H. ; Rodrigues, Joel J. P. C. ; Abrão, Taufik ; Proença, Mario L.: Anomaly detection using

baseline and K-means clustering. In: *SoftCOM 2010, 18th International Conference on Software, Telecommunications and Computer Networks*, 2010, S. 305–309

[40] Luo, Feng ; Wang, Bowen ; Fang, Zihao ; Yang, Zhenyu ; Jiang, Yifan: Security Analysis of the TSN Backbone Architecture and Anomaly Detection System Design Based on IEEE 802.1Qci. In: *Security and Communication Networks* 2021 (2021), Sep, S. 6902138. – URL https://doi.org/10.1155/2021/6902138. – ISSN 1939-0114

[41] Marchetti, Mirco ; Stabili, Dario: Anomaly detection of CAN bus messages through analysis of ID sequences. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, S. 1577–1583

[42] Mckenna, Daniel: *Securing automotive over-the-air updates*. https://www.iot-now.com/2017/02/27/59018-securing-automotive-air-updates/. – Accessed: 2022-16-12

[43] McKeown, Nick ; Anderson, Tom ; Balakrishnan, Hari ; Parulkar, Guru ; Peterson, Larry ; Rexford, Jennifer ; Shenker, Scott ; Turner, Jonathan: OpenFlow: Enabling Innovation in Campus Networks. In: *SIGCOMM Comput. Commun. Rev.* 38 (2008), mar, Nr. 2, S. 69–74. – URL https://doi.org/10.1145/1355734.1355746. – ISSN 0146-4833

[44] Meyer, Philipp ; Häckel, Timo ; Reider, Sandra ; Korf, Franz ; Schmidt, Thomas C.: Network Anomaly Detection in Cars: A Case for Time-Sensitive Stream Filtering and Policing. (2021), Dezember

[45] Microsoft: The STRIDE Threat Model, URL https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20), 12 2009

[46] Miller, Charlie ; Valasek, Chris: Remote exploitation of an unaltered passenger vehicle. In: *Black Hat USA* 2015 (2015), Nr. S 91

[47] Noble, William S.: What is a support vector machine? In: *Nature Biotechnology* 24 (2006), Dec, Nr. 12, S. 1565–1567. – URL https://doi.org/10.1038/nbt1206-1565. – ISSN 1546-1696

[48] Nsaibi, Seifeddine: Comparison between the deterministic communication mechanisms of TSN: Scheduled Traffic and Frame-Preemption Gegenüberstellung der de-

terministischen Kommunikationsmechanismen von TSN: Netzwerkplanung und Telegrammunterbrechung, 06 2017

[49] Patterson, Andrew: *Outlier detection methods for detecting cheaters in mobile gaming*. – URL https://www.youtube.com/watch?v=Q2HLPCBStLQ&t=1709s

[50] Pedregosa, F. ; Varoquaux, G. ; Gramfort, A. ; Michel, V. ; Thirion, B. ; Grisel, O. ; Blondel, M. ; Prettenhofer, P. ; Weiss, R. ; Dubourg, V. ; Vanderplas, J. ; Passos, A. ; Cournapeau, D. ; Brucher, M. ; Perrot, M. ; Duchesnay, E.: Scikit-learn: Machine Learning in Python. In: *Journal of Machine Learning Research* 12 (2011), S. 2825–2830

[51] Rajbahadur, Gopi K. ; Malton, Andrew J. ; Walenstein, Andrew ; Hassan, Ahmed E.: A Survey of Anomaly Detection for Connected Vehicle Cybersecurity and Safety. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, S. 421–426

[52] Rieke, Roland ; Seidemann, Marc ; Talla, Elise K. ; Zelle, Daniel ; Seeger, Bernhard: Behavior Analysis for Safety and Security in Automotive Systems. In: *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, 2017, S. 381–385

[53] Rouf, Ishtiaq ; Miller, Rob ; Mustafa, Hossen ; Taylor, Travis ; Oh, Sangho ; Xu, Wenyuan ; Gruteser, Marco ; Trappe, Wade ; Seskar, Ivan: Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study. In: *19th USENIX Security Symposium (USENIX Security 10)*. Washington, DC : USENIX Association, August 2010. – URL https://www.usenix.org/conference/usenixsecurity10/security-and-privacy-vulnerabilities-car-wireless-networks-tire-pressure

[54] Schumacher, Wilhelm: Modellbasierter Vergleich von Methoden zur Anomalieerkennung im Fahrzeugnetzwerk des SecVI Demonstrators - Hauptprojekt / CoRE Research Group, Hochschule für Angewandte Wissenschaften Hamburg. Oktober 2021. – Forschungsbericht

[55] Seol, Youhwan ; Hyeon, Doyeon ; Min, Junhong ; Kim, Moonbeom ; Paek, Jeongyeup: Timely Survey of Time-Sensitive Networking: Past and Future Directions. In: *IEEE Access* 9 (2021), S. 142506–142527

[56] Sommer, Florian ; Dürrwang, Jürgen ; Kriesten, Reiner: Survey and Classification of Automotive Security Attacks. In: *Information* 10 (2019), Nr. 4. – URL https://www.mdpi.com/2078-2489/10/4/148. – ISSN 2078-2489

[57] Stachowski, Stephen ; Gaynier, Ron ; LeBlanc, David J.: *An Assessment Method for Automotive Intrusion Detection System Performance*. Apr 2019. – URL https://rosap.ntl.bts.gov/view/dot/41006. – Tech Report

[58] Studnia, Ivan ; Nicomette, Vincent ; Alata, Eric ; Deswarte, Yves ; Kaâniche, Mohamed ; Laarouchi, Youssef: Survey on security threats and protection mechanisms in embedded automotive networks. In: *2013 43rd Annual IEEE/IFIP Conference on Dependable Systems and Networks Workshop (DSN-W)*, 2013, S. 1–12

[59] Stüber, Thomas ; Osswald, Lukas ; Lindner, Steffen ; Menth, Michael: *A Survey of Scheduling in Time-Sensitive Networking (TSN)*. 2022. – URL https://arxiv.org/abs/2211.10954

[60] Upstream: AutoThreat Intelligence Cyber Incident Repository, URL https://upstream.auto/research/automotive-cybersecurity/?id=null), 2022

[61] Wyk, Franco van ; Wang, Yiyang ; Khojandi, Anahita ; Masoud, Neda: Real-Time Sensor Anomaly Detection and Identification in Automated Vehicles. In: *IEEE Transactions on Intelligent Transportation Systems* 21 (2020), Nr. 3, S. 1264–1276

[62] Álvarez, Inés ; Moutinho, Luis ; Pedreiras, Paulo ; Bujosa, Daniel ; Proenza, Julian ; Almeida, Luís: Comparing Admission Control Architectures for Real-Time Ethernet. In: *IEEE Access* PP (2020), 06, S. 1–1

Glossar

Anwendungsschicht Die Anwendungsschicht, auch Application Layer genannt, ist die 7. Schicht des OSI-Referenzmodells. Ihr Aufgabenbereich ist die Bereitstellung von anwendungsorientierten Grunddiensten mit entsprechenden Datenstrukturen und Protokollen.

BorderEnlargementFactor BEP ist die Abkürzung für BorderEnlargementFactor und legt den Radius des Clusters ausgehend vom Clusterzentrum fest.

Buffer Overflow Buffer Overflow, auch Pufferüberlauf genannt, ist eine von Angreifern ausnutzbare Sicherheitslücke um z. B. Schadcode auszuführen oder Programmabstürze zu verursachen. Der Buffer Overflow entsteht, wenn es gelingt, mehr Daten in einen Speicher zu schreiben, als der dafür vorgesehene Bereich aufnehmen kann.

Contamination Contamination beschreibt den prozentualen Anteil an Ausreißern im Datensatz. Der Parameter wird im Training benutzt, um die Grenzen für die normalen Daten festzulegen.

False Positive Als False Positive wird ein Datenpunkt bezeichnet, der zu Unrecht als Ausreißer eingestuft wird (falscher Alarm).

OSI-Referenzmodell Das OSI-Referenzmodell ist ein Schichtenmodell für Netzwerkprotokolle, das aus insgesamt sieben Schichten besteht.

Sicherungsschicht Die Sicherungsschicht, auch Data Link Layer genannt, ist die zweite Schicht des OSI-Referenzmodells. Ihr Aufgabenbereich ist ein Zugangsverfahren für das Kommunikationsmedium bereitzustellen.

Software-defined Networking Software-defined Networking (SDN) ist ein Netzwerkkonzept, das die zentrale Verwaltung und Steuerung einzelner Komponenten im Netzwerk mithilfe von Software ermöglicht.

VLAN VLAN ist ein Akronym für Virtual Local Area Network. Ein Virtual Local Area Network bildet ein logisches Teilnetz Segment/Netzwerk innerhalb eines gesamten physischen Netzwerks.

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original