WIP: On the Real-World Applicability of Automotive CAN Intrusion Detection Systems

Kristijan Lazeski, **Michael Zohner**,
Jannis Hamborg, Christoph Krauß, Alexander Wiesmaier

Applied Cyber Security Darmstadt
Department of Computer Science
Darmstadt University of Applied Sciences

August 10, 2026

Motivation & Research Question

Context: The CAN bus has **no encryption and no authentication** — a well-known attack vector in every modern vehicle.

State of the art: Dozens of ML-based CAN IDS report precision and recall **near or at 100 %** on public benchmark datasets.

Research Question

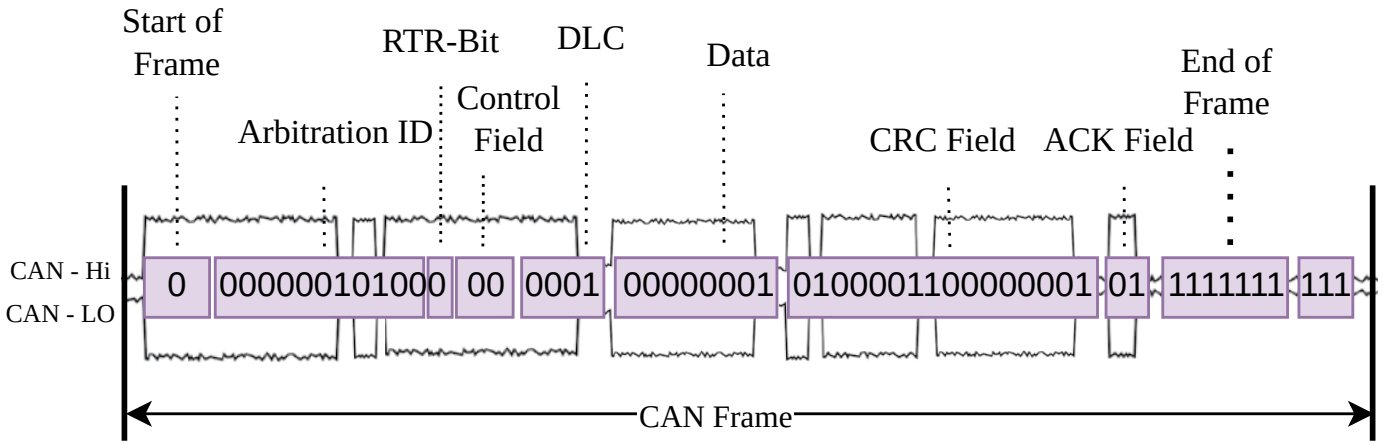
Do these near-perfect scores reflect **robust behavioral invariants** — or merely **dataset-specific artifacts**?

Our contributions (preliminary, WIP):

1. Systematic analysis of **five** widely used CAN IDS datasets
2. Training **interpretable Decision Trees** to expose what is actually learned
3. Evaluation of the learned rules against a **realistic attacker model**

Scope: per-frame, binary intrusion detection.

Attacker Model



Two attacker classes (Hamborg et al.)

- **Standard-compliant:** observe, inject, replay, DoS — within the CAN specification
- **Physical-layer:** additionally manipulates electrical signalling — frame suppression, bus-off

Assumed in This Work

Physical bus access; can **observe, inject, replay** frames and **adapt** CAN IDs, payload values and timing to observed traffic — **no keys or proprietary specifications needed**.

Enabled attacks: DoS · fuzzing · spoofing · replay

Methodology

Two-step approach:

1. **Structural dataset analysis** — trace organization, recorded fields, attack strategies, class distribution, CAN-ID/payload overlap between benign and malicious traffic
2. **Decision Tree training & rule inspection** — one DT per dataset and attack type; inspect *which features drive the decision*

Setup: Orange; ≥ 2 instances per leaf; no splits below 3 traces; 99 % stopping criterion (100 % for imbalanced/fuzzing subsets). Added **interval feature** (time since CAN ID last seen); payload as integer *and* per byte. Dedicated test files where available (Car Hacking Challenge), otherwise **10-fold random sampling** (66/34, averaged).

Why Decision Trees?

Native interpretability — they **expose structural shortcuts** directly. Any shortcut a DT exploits will be exploited at least as aggressively by an opaque model (RF, CNN, GAN) — just invisibly.

What the Datasets Actually Contain

Attack	How it is implemented across the five analyzed datasets
DoS	Static, unique CAN-ID/payload pair — almost always 0x000 with all-zero payload, **absent from benign traffic** (CT&T: 0x000/0x002, 3 constant variants)
Spoofing	Fixed CAN-ID/payload combinations not present in benign traffic (RPM/gear: static payload; CT&T speed: exactly 0, 10, 60 mph)
Fuzzing	Random IDs/payloads that **do** overlap benign traffic ⇒ models fall back on secondary artifacts (recording structure, timestamps)
Replay	Only in Car Hacking Challenge; replayed benign frames, consistent timestamps ⇒ genuinely hard

Consequence

Where malicious CAN-ID/payload combinations **never** occur in benign traffic, detection reduces to **recognizing a constant** — classification by memorization, not by semantic inconsistency.

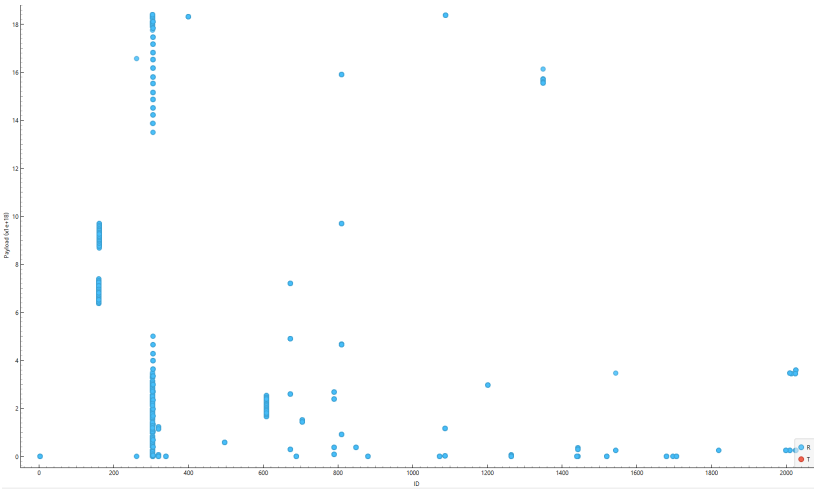
Structural Artifacts in the Datasets

- **Incomplete benign coverage** — in Car Hacking, **11 benign CAN IDs appear only in the fuzzing file** (ECU responses to injected frames); the benign recording does not cover all observable traffic
- **Missing labels** — OTIDS attack files have **no frame-level labels**, preventing reliable per-frame supervised learning
- **Class imbalance** — DoS constitutes 2 % of traces in CT&T, 4 % in Survival ⇒ accuracy rewards majority-class prediction
- **Visible recording boundaries** — attack scenarios form **contiguous time blocks** with absolute timestamps, a recording artifact unrelated to vehicle behaviour

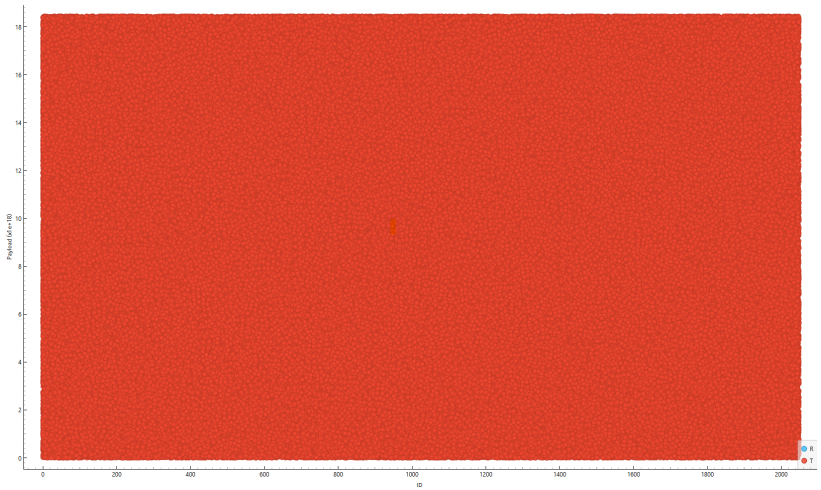
Why This Matters

None of these properties are things an **adaptive attacker would reproduce** on a live bus — yet they are exactly what the models can most easily latch onto.

Separability Is Structural, Not Semantic



Benign traces concentrate on a fixed set of CAN IDs with characteristic payload ranges.



Fuzzing spreads over the whole ID/payload space — but overlaps benign traffic where it matters.

Take-away

If malicious CAN-ID/payload combinations **never** occur in benign traffic, classification degenerates into **lookup of a constant** — no behavioral model is required, and none is learned.

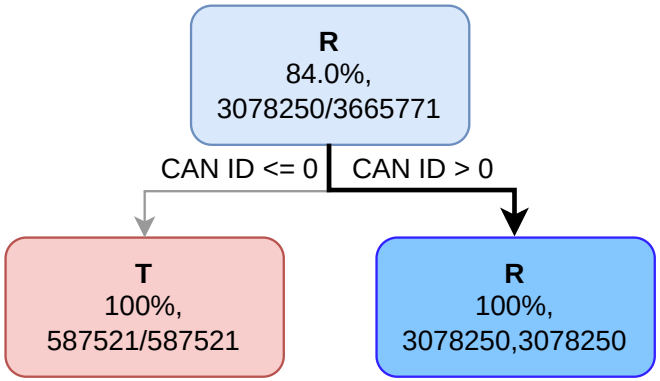
Decision Tree Results

Dataset	Attack	F1	Nodes	Comment
Car Hacking	DoS	1.000	1	1 unique pattern
Car Hacking	Spoofing (Gear/RPM)	1.000	1	1 unique pattern
Car Hacking	Fuzzing (100 %)	0.998	389	interval + byte features
Survival	DoS	1.000	1	1 unique pattern
CT&T	Fuzzing (100 %)	0.937	235	lowest fuzzing score
CT&T	Speed-Spoofing	1.000	3	3 unique attacks
CarChall.	Spoofing Prelim/Final	**0.000**	25	unseen pattern
CarChall.	Replay (100 %)	0.764 / **0.547**	10 707	train / test

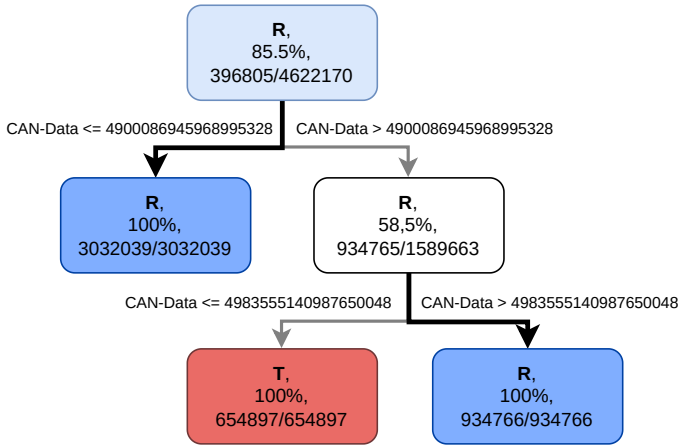
The Pattern Behind the Numbers

F1 = 1.000 with a single-node tree is not detection — it is memorization. Where memorization is impossible (unseen spoofing patterns, replay), performance **collapses to 0.00–0.55** and tree sizes explode to >10 000 nodes.

The Learned Rules Themselves



DoS: one split on a static CAN ID.



RPM spoofing: two splits on a static payload value.

- **DoS / spoofing:** rules reduce to *"is this the one ID/payload from the recording?"*
- **Fuzzing:** interval feature first, then byte patterns — some nodes split on **absolute UNIX timestamps**
- **Replay:** timestamp used already at the **second level** of the tree ⇒ pure overfitting on recording boundaries

Assessment Under the Attacker Model

Attack	Learned rule	Trivial evasion
DoS	CAN ID `0x000` + all-zero payload	Use **any other high-priority ID** or vary the payload — DoS effect preserved
Spoofing	Fixed payload outside benign range	Inject **within-range** values — still influences vehicle behavior
Fuzzing	Absolute timestamp boundaries	**Shift injection timing**; also mislabels benign traffic recorded later
Replay	Recording-specific artifacts	Replay a selected subset **aligned to the live bus schedule**

Result

The majority of learned decision rules reduce to recognizing **static, dataset-specific artifacts that no adaptive adversary would reproduce**. An attacker can additionally trigger rare-but-legitimate signals (missing from benign recordings) to **inflate false positives** and mask activity.

Conclusion & Future Work

Three central limitations:

1. Attacks implemented as **static constants**
2. **Incomplete benign coverage**
3. **Structural artifacts** (file separation, timestamps, labeling)

Architectural argument:
ML on raw CAN traffic mostly **reverse-engineers the DBC file**. Better: **rule-based pre-filtering first**, then ML on the residual traffic — this also scales to multi-bus, multi-gateway networks.

Future work:

- Criteria for **security-relevant datasets**: realistic attacker capabilities, frame-level labels, representative benign coverage, no structural separability, multi-bus topologies
- Extend to **X-CANIDS** and **window-based** detection
- Move toward **system-level** security modelling

Core Message

High benchmark performance does not imply practical security — the bottleneck is dataset realism, not model choice.

Thank You — Questions?

Takeaways

1. Near-perfect F1 on public CAN IDS datasets is largely **memorization of constants**
2. Learned rules **do not survive** an adaptive, standard-compliant attacker
3. We need **better datasets and specification-informed detection**, not bigger models

Decision trees and analysis artifacts:

`https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/5167`

Supported by the German Federal Ministry of Research, Technology and Space (BMFTR), project **PARFAIT** (Grant: 16KIS2143).

Backup: Dataset Selection Criteria

Dataset	Environment	Attacks	Labels	Access	Analyzed
OTIDS	1 vehicle	D, F	✗	✓	✓
Survival	3 vehicles	D, F	✓	✓	✓
CarChall.	1 vehicle	D, F, R, S	✓	✓	✓
CT&T	4 vehicles; 1 testbed	D, S, F, M, R, Su	✓	✓	✓
HCRL	2 vehicles	D, S, F, RPMt	✓	✓	✓
ROAD	1 vehicle	S, F, M	✗	✓	✗
SynCAN	1 simulation	S, Su, M	✓	✓	✗
TTIDS	1 vehicle	M	✗	✓	✗

D – DoS, F – Fuzzing, S – Spoofing, M – Masquerade, R – Replay, Su – Suppression, RPMt – RPM Tampering.

Backup: Full Fuzzing & Replay Results

Dataset	Attack	FPR	Precision	Recall	F1
Car Hacking	Fuzzing (99 %)	0.059	0.994	0.941	0.967
Car Hacking	Fuzzing (100 %)	>0.001	0.996	1.000	0.998
Survival	Fuzzing	0.024	0.997	0.976	0.987
Survival	Malfunction	0.003	0.998	0.995	0.997
CT&T	Fuzzing (100 %)	0.014	0.893	0.986	0.937
CT&T	Systematic (100 %)	0.010	0.999	0.990	0.995
CarChall.	Fuzzing Final (100 %)	0.006	0.871	0.994	0.928
CarChall.	Replay Train (100 %)	0.255	0.784	0.745	0.764
CarChall.	Replay Prelim (100 %)	0.555	0.709	0.445	0.547