POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Physical-Layer Intrusion Detection for the CAN Bus: A Voltage-Based Approach Against Transport-Layer Attackers

Laurea Magistrale in Computer Science Engineering - Ingegneria Informatica

Author: Pierantonio Mauro

Advisor: Prof. Stefano Longari

Academic year: 2025-26

1. Introduction

The **Controller Area Network (CAN)** bus is the backbone of in-vehicle communication in modern automobiles, interconnecting up to one hundred Electronic Control Units (ECUs) that govern everything from engine management and antilock braking to infotainment and park-assist. Introduced by Robert Bosch GmbH in 1986 and standardised as ISO 11898, CAN was designed for robustness and real-time performance in noisy automotive environments, not for security: the protocol provides no encryption, no message authentication and no access control. These deficiencies have become a significant vulnerability as vehicles gained wireless connectivity. In 2015, Miller and Valasek remotely exploited a Jeep Cherokee's infotainment system to gain control over steering and braking [11]; similar attacks followed on other platforms [15]. As a result, the security community has increasingly focused on **Intrusion Detection Systems (IDSs)** as a practical defensive layer, since they require no modification of the underlying protocol.

Most CAN security research has been conducted under the **Conventional Remote Attacker Model (CRAM)** [14], where a compromised ECU accesses the bus only through the intact CAN controller. However, recent work has demonstrated that software-only techniques — peripheral clock gating (CANnon [8]) and pin conflict exploitation (CANflict [7]) — allow a compromised ECU to bypass the CAN controller entirely and manipulate individual bus bits in real time. This **Enhanced Remote Attacker Model (ERAM)**, formalised by Tang et al. [14], undermines existing defences including physical-layer IDSs, through fingerprint corruption, partial-frame attacks and short-pulse injections.

This thesis addresses the question: *can a physical-layer IDS maintain its effectiveness against an ERAM attacker?* The key insight is that an ERAM attacker, while gaining full data-link layer control, still uses the victim ECU's physical CAN transceiver, whose analogue output characteristics cannot be altered through software alone. A lightweight IDS is proposed that monitors the CAN bus differential voltage in real time, deriving a detection threshold ($\mu + 3\sigma$) from a baseline capture of clean traffic and correlating voltage anomalies with CAN error flags detected by a protocol-aware frame counter operating at 1 MSa/s.

2. Background

The CAN specification [12] defines the physical and data-link layers. Noise resilience comes from differential signalling over two twisted wires, CAN High and CAN Low. A logical zero (the **dominant bit**) corresponds to a differential voltage of ≈2 V (CAN-H at ≈3.5 V, CAN-L at ≈1.5 V), while a logical one (the **recessive bit**) corresponds to ≈0 V, with both lines at ≈2.5 V. The bus is a wired-AND: whenever any node drives a dominant bit, the bus becomes dominant regardless of other nodes' output.

Each ECU consists of three components: a **microcontroller** (MCU), a **CAN controller** implementing the data-link layer (frame assembly, bit stuffing, CRC, arbitration, error handling), and a **CAN transceiver** implementing the physical layer (converting digital bits into differential voltages and vice versa). The transceiver's analogue output characteristics — slew rate, output impedance, parasitic capacitance — depend on the silicon process and PCB layout, making them a potential hardware fingerprint.

The protocol employs priority-based arbitration: simultaneous transmitters compare their ID bits against the bus level, and the node with the lowest numerical ID always wins. Error handling relies on two per-node counters: the **Transmit Error Counter (TEC)** and the **Receive Error Counter (REC)**, which drive the ECU through three states: *Error Active* (TEC, REC ≤ 127), *Error Passive* (TEC or REC > 127) and *Bus Off* (TEC > 255), at which point the node ceases all communication.

3. Threat Model and Existing Defences

3.1. Attacker Models

Under the **CRAM**, a compromised ECU can read and transmit well-formed CAN frames through the intact controller. This suffices for fabrication, replay, flooding, and bus-off attacks via simultaneous transmission [3]. WeepingCAN [2] is a stealthier variant that evades timing-based detection by carefully controlling error counter increments.

The **ERAM** [14] subsumes all CRAM capabilities and adds: (i) ubiquitous link-layer visibility (real-time bit-level reading), (ii) omnipotent link-layer write-ability (injection of arbitrary signals, including error frames, partial frames and pulses), and (iii) link-layer rule non-compliance (ignoring arbitration, error-state transitions and stuffing rules). This enables frame hijacking, Janus and counterfeit frames, synchronisation disruption, double-receive, freeze doom loop and physical fingerprint corruption — attacks undetectable by every IDS class designed against the CRAM.

3.2. Existing IDSs and Their Limitations

Payload-based IDSs (e.g., CANnolo [9], CANdito [10]) analyse CAN traffic at the data-link layer using neural networks or statistical models. They are blind to the identity of the transmitting node and ineffective against masquerade or ERAM Janus attacks.

Timing-based IDSs (e.g., CIDS [4]) exploit near-constant inter-message intervals and clock-skew fingerprints. They can be evaded by attackers imitating the victim's transmission frequency [13], and ERAM frame hijacking preserves the original timing.

Voltage-based IDSs (e.g., Viden [5], VoltageIDS [6]) monitor the analogue bus waveform to extract per-node hardware fingerprints. While promising, they are vulnerable to three ERAM-specific attacks: *fingerprint corruption* (gradual model poisoning via pulse injection), *partial-frame attacks* (leaving the fingerprinted field intact while modifying the payload), and *short-pulse injections* (invisible to IDSs that process only complete frames). These gaps motivate the present work.

4. System Design

The proposed IDS is designed around four requirements: *passive monitoring* (no bus topology alteration), *real-time detection* (millisecond-scale alerts), *statistically derived thresholds* (adapting to the actual bus rather than using fixed values), and a *low false-positive rate* (the $\mu + 3\sigma$ threshold corresponds to <0.13% false-alarm probability under a Gaussian assumption).

The system is organised into two sequential pipelines (Figure 1):

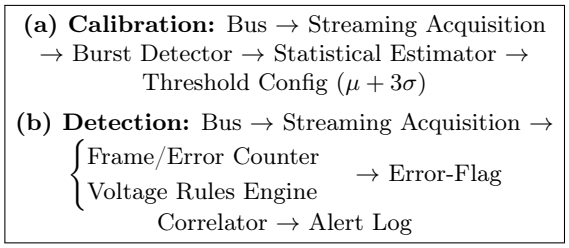

Figure 1: Logical architecture of the IDS: (a) calibration pipeline; (b) detection pipeline.

Calibration pipeline. Run once with only the target ECU active. The *Burst Detector* segments the continuous voltage stream into per-frame events; the *Statistical Estimator* computes the mean μ and standard deviation σ of the burst-peak CANH−CANL differential and outputs the upper bound $\mu + 3\sigma$. On the test bed, this yielded $\mu + 3\sigma \approx 1876\,\text{mV}$.

Detection pipeline. Run continuously during operation. Two parallel processing paths receive the differential signal $V_{\text{diff}} = V_{\text{CAN-H}} - V_{\text{CAN-L}}$:

- The **Frame/Error Counter**: a state machine identifying SOF, error flags (>5.5 bit-times dominant run) and EOF events from dominant/recessive run lengths.
- The **Voltage Rules Engine**: evaluates two detection rules on every dominant-phase sample:
  - R_1 (DIFF spike): $V_{\text{diff}} > 1.876\,\text{V}$ for $\geq$1 sample.
  - R_2 (DIFF sustained): $V_{\text{diff}} > 1.876\,\text{V}$ for $\geq$3 consecutive samples.

The **Error-Flag Correlator** checks whether any rule fired within a 500-sample lookback window (500 μs) before each error flag, classifying it as *confirmed* or *missed*. The detection rate is the ratio of confirmed to total error flags.

5. Implementation

The laboratory test bed follows the hardware configuration of the CANdemonium prototype [1].

Legitimate traffic nodes. Two Arduino development boards, each equipped with a CAN transceiver module, transmit periodic data frames with fixed identifiers at 500 kbit/s, simulating two independent ECUs.

Attacker node. An STM32 Nucleo board with a CAN transceiver serves as the ERAM-class attack injector. The firmware implements the CANflict technique [7]: SPI peripherals are redirected to the CAN-TX and CAN-RX pins, bypassing the bxCAN controller entirely. SPI1 monitors bus bits in real time, and SPI2 injects dominant bits at arbitrary positions within ongoing frames.

Signal acquisition. A PicoScope 2205A MSO oscilloscope captures both CAN-H and CAN-L through passive BNC probes at $f_s = 1\,\text{MSa/s}$ in streaming mode, with a $\pm 5\,\text{V}$ range (8-bit ADC, $\approx$39 mV resolution). All three nodes and the probes are connected to a short bus terminated with 120 Ω resistors at both ends.

The software pipeline is implemented in Python: calibration uses NumPy for burst-peak statistics; detection uses a multithreaded architecture separating the streaming acquisition callback from the live monitor display. The Frame/Error Counter uses a vectorised run-based approach (NumPy identifies transitions across each buffer), while the Voltage Rules Engine evaluates rules only on dominant-phase samples ($\approx$0.2% of total), making the CPU cost negligible.

6. Experiments and Results

The only attack class evaluated experimentally is the ERAM bus-off attack via CANflict. The Nucleo firmware targets frames with CAN identifier 0x48C: each time such a frame is detected, 6 consecutive dominant bits are injected via SPI2, causing a stuff violation or bit error that increments the victim's TEC by 8. Approximately 40 injections drive TEC above 255 and achieve bus-off.

Four evaluation sessions were recorded:

1. A **baseline session** (126.1 s, 65 M samples, 1315 frames) with the Nucleo silenced.
2. A **sustained attack session** (127.4 s, 64.5 M samples, 575 error flags).
3. Two **dedicated bus-off sessions** (69.5 s and 45.1 s), producing 16 and 19 error flags respectively.

6.1. Results

The baseline session produced **zero rule fires and zero error frames**, confirming that the $\mu + 3\sigma$ threshold passes all legitimate two-ECU traffic without spurious alerts.

Table 1: Error-flag correlation results.

Metric	Sustained	Bus-off 1	Bus-off 2
Duration (s)	127.4	69.5	45.1
Error flags	575	16	19
Confirmed	515	16	19
Missed	60	0	0
Detection rate	**89.6%**	**100%**	**100%**
R_1 per-flag rate	89.6%	100%	100%
R_2 per-flag rate	65.2%	87.5%	78.9%

The dedicated bus-off sessions achieved **100% error-flag detection rates** (16/16 and 19/19): when the attacker's injections consistently overlap with the victim's dominant bits, every error flag is preceded by a detectable voltage anomaly. The sustained session achieved **89.6%** overall (515/575 confirmed). The 60 missed flags (10.4%) correspond to injections where the attacker's differential contribution did not exceed the threshold, occurring when the injection happens during a recessive phase and the transient pulse is too brief to be sampled above threshold at 1 MSa/s.

Rule R_1 (spike) is the primary detection mechanism; R_2 (sustained) is more restrictive and does not contribute additional detections beyond R_1 in any session. During the sustained session, R_1 also fired in 680 frames — more than the 575 error frames — because some injection attempts raised the voltage above threshold without producing a protocol-level error. These "false positives" are informative: they indicate ongoing attack activity even when individual injections did not corrupt a frame.

6.2. Example Voltage Traces

Figures 2–5 show representative voltage traces captured by the Detection Pipeline for each classification category. Each trace displays the CAN-H, CAN-L and differential waveforms; the horizontal dashed line marks the 1.876 V detection threshold.

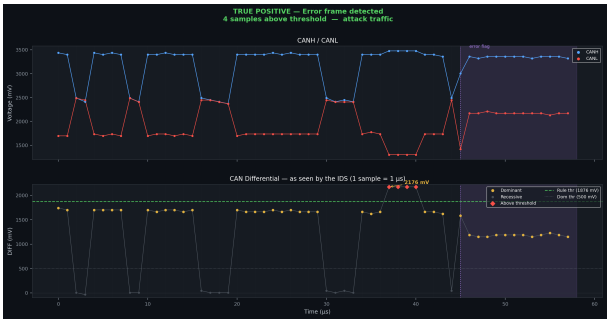

Figure 2: **True Positive** — an error frame preceded by a differential voltage excursion above the 1.876 V threshold.

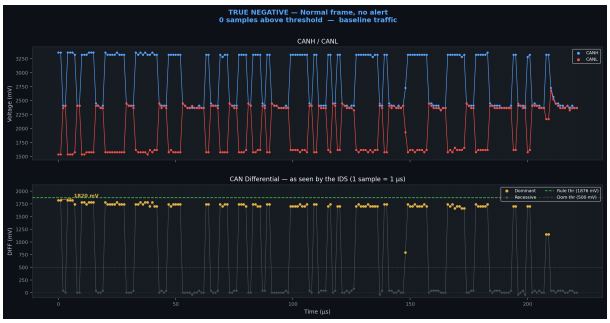

Figure 3: **True Negative** — a normal frame with no voltage anomaly. The differential remains within the calibrated baseline.

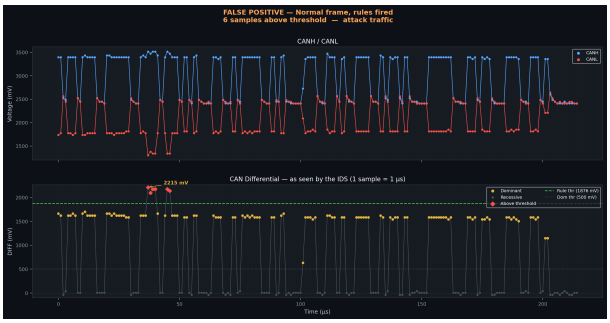

Figure 4: **False Positive** — a normal frame in which the differential momentarily exceeded the threshold due to an injection attempt that did not corrupt the frame.

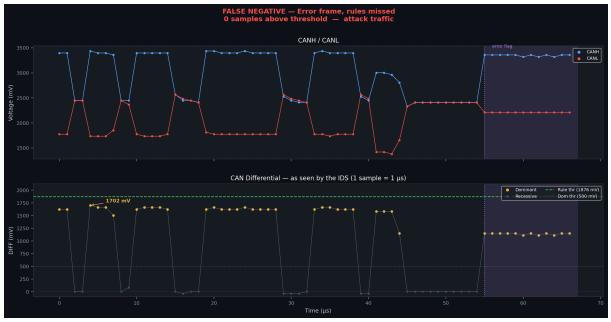

Figure 5: **False Negative** — an error frame not preceded by a voltage anomaly: the attacker's pulse did not overlap with the victim's dominant bits, so the differential stayed within the calibrated range.

7. Conclusions

This thesis investigated whether a physical-layer IDS can detect attacks launched under the Enhanced Remote Attacker Model. Three contributions were made: (i) a systematic analysis of how ERAM undermines existing physical-layer defences, identifying fingerprint corruption, partial-frame attacks and short-pulse injections as key vulnerabilities; (ii) a lightweight IDS prototype based on error-flag-correlated voltage monitoring, using a $\mu + 3\sigma$ statistical threshold derived from clean-traffic calibration; and (iii) an experimental evaluation against an ERAM-class bus-off attack via the CANflict technique, achieving 100% detection on dedicated sessions and 89.6% on a sustained attack with zero false positives in baseline.

The system has limitations: it depends on a laboratory oscilloscope, was tested on a minimal two-node bench with a single attack class, and uses static thresholds that may drift with temperature and battery voltage. Future work should address embedded implementation on low-cost hardware (e.g., STM32 with on-chip ADC), multi-condition calibration across the automotive operating range, extension to additional attack classes (fabrication, masquerade, fuzzing), evaluation on a more realistic multi-ECU test bed following the CANdemonium requirements framework [1], and a defence-in-depth combination with payload-based and timing-based IDSs to cover a broader fraction of the ERAM threat envelope.

References

[1] Daniele Azzarà. CANdemonium: An evaluation test bed for objective CAN intrusion detection systems benchmarking. Master's thesis, Politecnico di Milano, 2024.

[2] Gedare Bloom. WeepingCAN: A stealthy CAN bus-off attack. In *Workshop on Automotive and Autonomous Vehicle Security (AutoSec), co-located with NDSS 2021*, AutoSec '21, 2021.

[3] Kyong-Tak Cho and Kang G. Shin. Error handling of in-vehicle networks makes them vulnerable. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 1044–1055. ACM, 2016.

[4] Kyong-Tak Cho and Kang G. Shin. Fingerprinting electronic control units for vehicle intrusion detection. In *25th USENIX Security Symposium*, USENIX Security '16, pages 911–927. USENIX Association, 2016.

[5] Kyong-Tak Cho and Kang G. Shin. Viden: Attacker identification on in-vehicle networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17. ACM, 2017.

[6] Wonsuk Choi, Kyungho Joo, Hyo Jin Jo, Moon Chan Park, and Dong Hoon Lee. VoltageIDS: Low-level communication characteristics for automotive intrusion detection system. *IEEE Transactions on Information Forensics and Security*, 13(8):2114–2129, 2018.

[7] Alvise De Faveri Tron, Stefano Longari, Michele Carminati, Mario Polino, and Stefano Zanero. CANflict: Exploiting peripheral conflicts for data-link layer attacks on automotive networks. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, pages 711–723. ACM, 2022.

[8] Sekar Kulandaivel, Shalabh Jain, Jorge Guajardo, and Vyas Sekar. CANNON: Reliable and stealthy remote shutdown attacks via unaltered automotive microcontrollers. In *2021 IEEE Symposium on Se-*

curity and Privacy, SP '21, pages 195–210. IEEE, 2021.

[9] Stefano Longari, Damiano H. Nova Valcarcel, Mattia Zago, Michele Carminati, and Stefano Zanero. CANnolo: An anomaly detection system based on LSTM autoencoders for controller area network. *IEEE Transactions on Network and Service Management*, 18(2):1913–1924, 2021.

[10] Stefano Longari, Cosimo Andrea Pozzoli, Andrea Nichelini, Michele Carminati, and Stefano Zanero. CANdito: Improving payload-based detection of attacks on controller area networks. In *Cyber Security, Cryptology, and Machine Learning (CSCML 2023)*, volume 13914 of *Lecture Notes in Computer Science*. Springer, 2023.

[11] Charlie Miller and Chris Valasek. Remote exploitation of an unaltered passenger vehicle. Black Hat USA 2015 technical white paper, 2015. Presented at Black Hat USA and DEF CON 23, Las Vegas, NV, August 2015.

[12] Robert Bosch GmbH. *CAN Specification Version 2.0*. Robert Bosch GmbH, Stuttgart, Germany, 1991.

[13] Sang Uk Sagong, Xuhang Ying, Andrew Clark, Linda Bushnell, and Radha Poovendran. Cloaking the clock: Emulating clock skew in controller area networks. In *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems*, ICCPS '18, pages 32–42. ACM/IEEE, 2018.

[14] Zhaozhou Tang, Khaled Serag, Z. Berkay Celik, Saman Zonouz, Dongyan Xu, and Raheem Beyah. ERACAN: Defending against an emerging CAN threat model. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, CCS '24. ACM, 2024. Distinguished Paper Award.

[15] Tencent Keen Security Lab. Car hacking research: Remote attack Tesla cars. Technical blog post, Tencent Keen Security Lab, 2016.