Technical report · IF-TR-2026-02 · version 1.0

# CanLab: offline counter, checksum and field-boundary detection for CAN captures, with a gated transmit path

Sherin Joseph Roy 10 November 2026 CanLab v2.0.0 P3 · Rigs and calibration

Funded entirely by BluRabbit Mobility Private Limited.

Abstract

CanLab is a desktop workstation for reverse-engineering Controller Area Network traffic from recorded captures. This report describes release v2.0.0: the offline detectors that propose rolling counters, checksum bytes and field boundaries, the DBC builder that checks a signal definition against real frames before export, and the single transmit gate that every send path passes through. The author reproduced the repository's unit suite (540 tests passed on CPU, with no CAN hardware) and ran the command-line detector on the bundled generated sample. The checksum result matches the generator source exactly. The counter total matches the README, but one of the reported counters is a checksum byte that the generator makes identical to the counter byte, so the README's statement that the detector finds exactly the counters the generator wrote is not supported. Acceptance checks on public real-vehicle recordings are reported from the repository and were not reproduced. The analysis proposes candidates; it does not identify signals.

## 1 Problem

A sensor rig on a vehicle often needs signals the vehicle already carries on its own networks, such as wheel speed, steering angle or brake state, and the message catalogue for those networks is usually not published. Working out which bytes of an undocumented CAN bus carry which signals, and writing correct definitions for them, is slow and error-prone by hand. CanLab addresses that step: it loads a recorded capture, narrows the search to candidate counters, checksums and field boundaries, lets the analyst write a definition in a bit-level editor, checks it against the captured frames, and exports a DBC that other tools can read [repo]. A byte that changes every frame may be a counter, a checksum or the low byte of a fast measurement, and the three look alike in a hex dump.

### The Classic CAN data frame

The unit of analysis is the data frame of ISO 11898-1:2015 [iso11898]. In the base format with an 11-bit identifier, commonly called CAN 2.0A, a data frame carries, in order:

- **SOF:** one dominant start-of-frame bit, on which receivers synchronise.
- **Arbitration field:** the 11-bit identifier and the **RTR** bit (dominant in a data frame, recessive in a remote frame). The lower identifier wins arbitration, so the identifier is also the frame’s priority.
- **Control field: IDE** (dominant for the base format), the reserved bit **r0** (the FDF position in the 2015 edition, dominant in a Classical CAN frame) and the 4-bit **DLC**.
- **Data field:** zero to eight bytes, 0 to 64 bits; the only field that carries application signals.
- **CRC field:** a 15-bit CRC sequence and a recessive **CRC delimiter**.
- **ACK field:** the **ACK slot**, driven dominant by any receiver that took the frame without error, and a recessive **ACK delimiter**.
- **EOF:** seven recessive end-of-frame bits.

| SOF | ID | RTR | IDE | r0 | DLC | DATA | CRC | DEL | ACK | DEL | EOF |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 11 | 1 | 1 | 1 | 4 | 0–64 | 15 | 1 | 1 | 1 | 7 |

Figure 1. Fields of a Classic CAN 2.0A data frame with their lengths in bits (ISO 11898-1:2015); bit stuffing and interframe space omitted. Signal discovery works inside the highlighted data field.

Every CanLab parser reduces a frame to the columns `Timestamp, ID, Bus, DLC, Extended, B0..B7`, widened to `B63` for CAN FD [repo]. The analysis therefore sees the identifier, its format, the DLC, the data bytes and a receive time; the controller's CRC is not in the log. Every counter or checksum in this report is one placed inside the data field.

## 2 System design

CanLab is a PyQt6 desktop application, with 16 tabs grouped into five workspaces, and a Qt-free command line, `canlab-cli` [repo]. Three components carry most of the design:

- **One receive thread** (`core/bus_hub.py`) is the only caller of `recv()` and hands frames to subscribers filtered by identifier.
- **A capped ring buffer** (`core/frame_store.py`) holds frames in preallocated NumPy arrays (500,000 by default, oldest dropped) and builds the pandas DataFrame on demand. For the in-memory capture the README describes, it reports `5 ms` to build that DataFrame and `0.5 ms` to fetch one identifier, as the shape of the cost rather than a benchmark.
- **One transmit gate** (`core/safety.py`) holds the armed flag, a blocked-identifier set and a registry of running transmit workers; every send goes through `gated_send`.

The offline detectors (`counter_checksum_detector.py`, `checksums.py`, `checksum_guesser.py`, `entropy_boundary.py`, `bit_flags.py`, `value_tables.py`) send nothing anywhere and need no API key. The release notes state that none of the 25 tools of its MCP server can transmit.

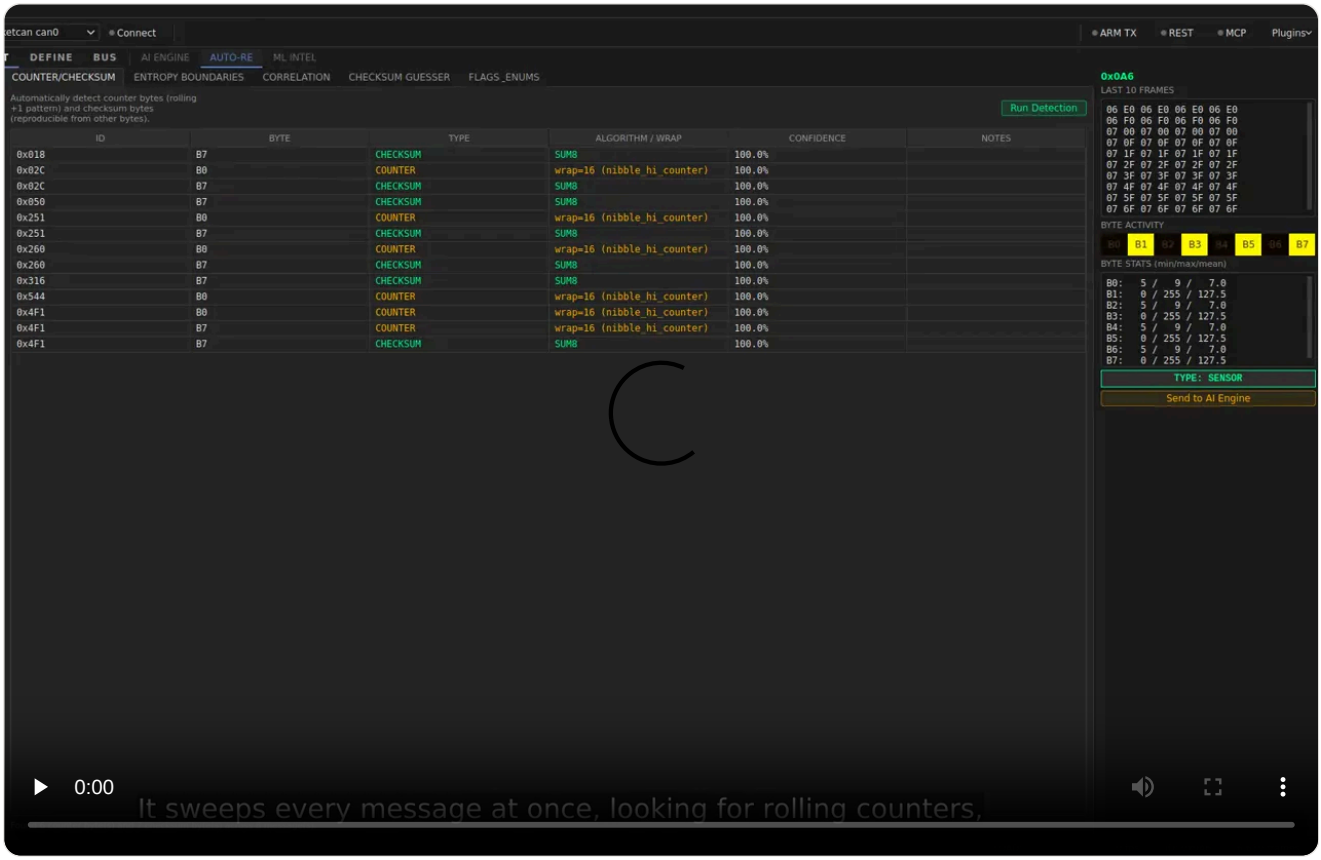



Figure 2. The AUTO-RE tab sweeping every identifier of the bundled sample capture for rolling counters and checksum bytes, then the checksum guesser on one message. Screen recording from the repository (docs/canlab-demo-part1-analysis.mp4), trimmed, cropped and muted; the subtitles are the repository's narration.

## 3 Method

### Counters

For each identifier, frames are put in time order and every data byte is scored as a whole byte, as its low nibble and as its high nibble. The score is the fraction of consecutive frames in which the value rises by one or rolls over from its observed maximum to zero, and a candidate must exceed `COUNTER_MATCH` (0.70 in the source). The modulus is read from the data and reported only when a roll-over was observed.

### Checksums

`_detect_checksums_vectorized` tests, for each byte position, whether the byte equals XOR8, SUM8 or a nibble sum of the other seven bytes, or an identifier-dependent XOR variant. Two guards apply. A relation counts only if it beats the byte's most common value, so constant padding never qualifies. A relation that holds across most of the payload is not attributed to any byte unless the varying columns do not duplicate one another, because "is byte k the XOR of the other seven" is the same question as "does the message XOR to zero"; if kept, it goes to the last matching byte. Bytes still unexplained are scored against the full registry in `checksums.py` (parametrised CRC-8 and manufacturer variants checked against opendbc and published check values) over at most the first 400 frames. The real-data README records why: the earlier sweep knew only the cheap relations and found no checksums on a real bus [realdata]. One checksum at most is reported per message. A separate guesser scores one chosen byte with a chronological fit and validation split.

### Boundaries, flags and enumerations

`entropy_boundary.py` combines per-bit entropy, per-bit activity and byte alignment into candidate segments; `bit_flags.py` and `value_tables.py` find rarely changing bits and few-valued bytes. They propose where a field starts and ends; none names a field.

### Building a DBC and checking it against frames

The DBC builder draws each byte as a row with bit 7 on the left, and `bit_coords.py` converts grid positions to DBC bit numbers, including the big-endian start-bit sawtooth (bit 7 to bit 0 within a byte, then bit 7 of the next) that the README calls easy to get wrong by hand. All decoding and encoding goes through one cantools-backed path [cantools], so the live preview decodes captured frames through the definition being edited and shows the value the exported file will produce. PLOT draws decoded values over time, where a wrong byte order appears as a sawtooth. Each export (DBC, openpilot DBC, CANdb++, ARXML, Wireshark Lua) is loaded back in the test suite by the tool that reads it. Annotated capture ranks every byte and bit against intervals the analyst marked while acting, and reference calibration fits a field to a physical reference by least squares with an $R^2$ verdict of PASS or UNCONFIRMED. `canlab-cli detect --dbc` drafts a DBC with overlapping claims resolved; `auto_dbc.py` calls its drafts starting points, not identifications.

### The transmit gate

Injection, replay, fuzzing and bridging are restricted by the README to isolated bench setups. Transmit is disarmed at start-up. `gated_send` raises `BusNotArmedError` unless ARM TX is set, and `BlockedIdError` for a blocked identifier even while armed. Disarming stops and joins every registered transmit worker. Diagnostic requests, reads included, pass the same gate, and the UDS service scan probes read-only services unless destructive ones are enabled and confirmed. `tests/test_tx_gate.py` covers the disarmed and blocked cases, worker shutdown, and the ISO-TP, injection, fuzzer and replay paths.

## 4 Validation

### Unit suite (reproduced)

On CPU, with Qt offscreen and the CI job's extras, `540 tests` passed with none failed or skipped, in `54.97 s` as reported by pytest. No CAN hardware, AI key or network service was used; buses were stand-ins or python-can's virtual bus [pythoncan]. This agrees with the README badge (`540 tests`) and Testing section (`540 tests`).

### Detectors on the bundled generated sample (reproduced)

The bundled sample is synthetic: `canlab/sample_data/generate_sample.py` writes it, seeded, over ten identifiers, so the generator source is its ground truth. The author ran `canlab-cli detect` over it.

**Checksums.** The detector reports `7 checksums`, all SUM8 in byte 7 at full match, on 0x018, 0x02C, 0x050, 0x251, 0x260, 0x316 and 0x4F1, exactly the seven identifiers into which the generator writes a SUM8 of bytes 0 to 6. This agrees with the README (`7 checksums`), including byte and algorithm.

**Counters.** The detector reports `6 counters`, all high-nibble counters wrapping at 16: byte 0 of 0x02C, 0x251, 0x260, 0x4F1 and 0x544, and byte 7 of 0x4F1. The total equals the README's figure (`6 counters`), but the attribution does not hold. The generator writes a counter only into the high nibble of byte 0, in five messages, and the five byte-0 detections match them one for one. The remaining detection, byte 7 of 0x4F1, is that message's SUM8 checksum, which the same run also reports as a checksum.

The generator source explains it. In the 0x4F1 branch, bytes 1 and 2 come from a speed variable that is reset to zero for each message and assigned only in the 0x0A6 branch, so both stay zero, the SUM8 equals byte 0, and byte 7 repeats the counter. The counter sweep scores each byte independently and does not exclude a byte already claimed as a checksum. The README's "exactly the 6 counters and 7 checksums the generator wrote" therefore holds for the checksums and the counter total, not for which bytes the counters are.

1

1. The same detail leaves the vehicle-speed field of 0x4F1 constant in the sample.

| Quantity | Value | Conditions | Basis |
|---|---|---|---|
| Rolling counters detected in the bundled sample capture | `6 counters` | bundled sample capture (6,610 frames, 10 IDs); all reported as high-nibble counters wrapping at 16: byte 0 of 02C, 251, 260, 4F1, 544 and byte 7 of 4F1 | Reproduced 2026-11-10 |
| Checksums detected in the bundled sample capture | `7 checksums` | bundled sample capture (6,610 frames, 10 IDs); all 7 reported as SUM8 in byte 7 at 100% confidence, on 018, 02C, 050, 251, 260, 316, 4F1; the sample's generator script (canlab/sample_data/generate_sample.py) writes a SUM8 into byte 7 of exactly these 7 IDs | Reproduced 2026-11-10 |
| Counters detected in the bundled sample (README claim) | `6 counters` | README: 'the detector finds exactly the 6 counters and 7 checksums the generator wrote, with the right byte and the right algorithm' | Reported 2026-09-15 |
| Checksums detected in the bundled sample (README claim) | `7 checksums` | README: the detector finds exactly the 7 checksums the generator wrote, with the right byte and algorithm | Reported 2026-09-15 |
| Unit test suite: tests passed | `540 tests` | 540 passed, 0 failed, 0 skipped, 2 warnings; extras dev,ai,rest,mcp,mdf,adapters (the CI test job's set); Qt offscreen; no CAN hardware, AI keys or network services used (adapters, AI providers and buses are stand-ins or python-can's virtual bus) | Reproduced 2026-11-10 |
| Unit test suite duration | `54.97 s` | as reported by pytest for 540 tests; whole command including interpreter start-up measured 58.780242880 s by date before/after | Reproduced 2026-11-10 |
| Tests passing (README badge) | `540 tests` | README shields.io badge 'tests 540 passing'; the README status note and docs/reference.html also say 540 | Reported 2026-09-15 |
| Tests passed (README Testing section) | `540 tests` | '540 passed' in the Testing section; 'the single skip is the MDF4 parser, which needs the optional asammdf extra' | Reported 2026-09-15 |

Table 1. Detector output on the bundled generated sample (canlab-cli detect, all offline detectors, CPU) and the unit suite (Qt offscreen, CPU, no CAN hardware), each beside the README's figure. Detector and suite rows reproduced by the author at commit 00995a4; README rows quoted from the repository. "Reproduced" rows were run by the author; "Reported" rows are quoted from the repository at the commit shown.

## Acceptance on public real-vehicle recordings (reported)

The README runs the whole application over recordings it did not produce, because synthetic data agrees with whatever the code assumes [realdata]. The first corpus is the SavvyCAN example capture (11-bit, one bus). The second is five MDF4 recordings from a third-party CAN

logger's public examples, covering 29-bit J1939, a two-channel passenger-car log and a marine NMEA 2000 bus, plus BLF and ASC files from python-can's writers with 64-byte CAN FD and error frames. The repository reports `90 checks`.

| Quantity | Value | Conditions | Basis |
|---|---|---|---|
| Real-capture acceptance checks | `90 checks` | two corpora: SavvyCAN example capture (12,974 frames, 180 IDs, 11-bit, one bus) 36 checks; third-party CAN-logger MDF4 recordings plus python-can BLF/ASC format files (2 to 154,896 frames, 11- and 29-bit, dual-bus, CAN FD, error frames) 54 checks | Reported 2026-09-15 |
| Real-capture acceptance, phase 1 (loading, panels, offline analysis): checks passed | `14 checks` | 14/14 passed on the SavvyCAN example capture (12,974 frames, 180 IDs); stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 |
| Real-capture acceptance, phase 2 (decoding, exports, codegen, TX gate, REST, MCP): checks passed | `11 checks` | 11/11 passed on the SavvyCAN example capture; every export loaded back by cantools; disarmed sent 0 frames, armed sent 1; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 |
| Real-capture acceptance, phase 3 (MDF4, project round trip, TX workers, ISO-TP): checks passed | `11 checks` | 11/11 passed on the SavvyCAN example capture; 3000 real frames survive an MDF4 round trip; replay, fuzzer and sweep each send 0 when disarmed; scripted ECU answers OBD-II and UDS over ISO-TP on a virtual bus; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 |
| Second real corpus (third-party logger MDF4 and python-can format files): checks passed | `54 checks` | 54/54 passed; 5 third-party CAN-logger MDF4 recordings (public, third-party, 2 to 154,896 frames, 29-bit and dual-bus), 14 BLF/ASC files incl. 64-byte CAN FD and error frames, 9,600 real 29-bit NMEA 2000 frames identical across csv, blf, asc, log and pcap; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 |
| Counters found on the SavvyCAN real capture | `25 counters` | 25 counters and 5 checksums across 19 messages, out of 180 IDs in 12,974 real frames; no ground truth is stated, so these are detections, not verified hits; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 |
| Checksums found on the SavvyCAN real capture | `5 checksums` | 5 checksums and 25 counters across 19 messages, out of 180 IDs in 12,974 real frames; no ground truth is stated, so these are detections, not verified hits; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 |

Table 2. End-to-end acceptance scripts in tests/real_data over public recordings, as stated in the repository's real-data README under 'What a good run looks like'. Reported, not reproduced: the corpora are downloaded from the internet, not bundled. "Reproduced" rows were run by the author; "Reported" rows are quoted from the repository at the commit shown.

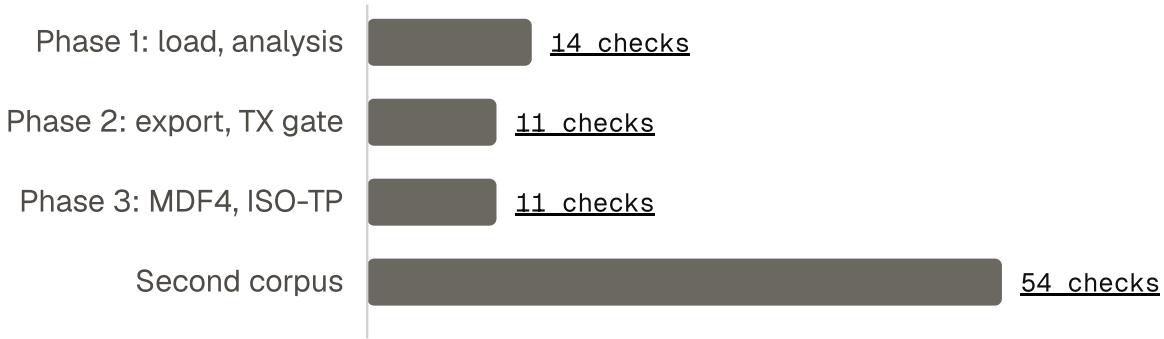



Figure 3. Acceptance checks passed per script on public real recordings; each script reports all its checks passed. Stated in the repository's real-data README as expected output, not a dated run log; not reproduced by the author.

Phases 2 and 3 are the only real-data evidence for the transmit gate: they check that the gate, replay, fuzzer and sweep send nothing while disarmed and that a send goes out once armed (counts in the table). On the SavvyCAN capture the sweep reports `25 counters` and

`5 checksums`; no ground truth is stated, so these are detections, not verified hits. The second corpus exposed two defects now pinned by tests: the openpilot DBC exporter wrote 29-bit identifiers a DBC reader refuses, and the sniffer expired every row of a loaded capture on opening.

## 5 Limitations

- **Candidates, not identifications.** The release notes' Status: "The analysis suggests candidates rather than identifying signals, and a confidence figure is a match fraction over the frames loaded, not a proof." The README adds that the tool has not been validated across a wide range of real vehicles.
- **Counter attribution.** On the only capture with known ground truth, one of the `6 counters` is a checksum byte, reported in the same run as both. The detectors do not reconcile such overlaps; the DBC drafting step does. Any checksum over a mostly constant payload tracks whichever byte changes, so the same confusion can be expected on sparse real messages.
- **Narrow synthetic ground truth.** The generator uses one checksum algorithm and one counter layout; the real-data README notes the earlier sweep looked correct on it and failed on a real bus.
- **Real-data results are reported only.** They are the README's expected output, not a dated log, and the SavvyCAN detections have no ground truth.
- **Performance figures are not benchmarks.** The four README timings, including the table refresh (`72 ms`) and plot-tree rebuild (`70 ms`), come from an unstated development machine and were not reproduced.
- **Hardware not exercised.** By the repository's own statements: no physical CAN adapter, GVRET board or FD hardware (FD was tested on a virtual bus), no gateway (it needs two hardware channels), no real ECU answering UDS or security access, no AI provider, and no interface driven by a person. The gate was tested on recording doubles and a virtual bus, never on a vehicle.
- **Bench use only.** Frames injected on a live vehicle bus can interfere with braking, steering and airbag systems (README, Safety).
- **Integration exposure.** The in-window MCP server has no authentication unless a token is set; enabled plugins run with full application privileges.

## 6 Reproducibility

CanLab v2.0.0, commit `00995a4016c68393baf4db8a10bef3ad330660f4` [repo]. Python 3.12.13 in a uv environment; Linux 7.0.0-42-generic (Ubuntu); AMD Ryzen 9 9900X, 12 cores and 24 threads, about 30 GB RAM; CPU only.

```
git clone https://github.com/Sherin-SEF-AI/CanLab && cd CanLab && git checkout v2.0.0
uv venv --python 3.12 && uv pip install -e '.[dev,ai,rest,mcp,mdf,adapters]'
QT_QPA_PLATFORM=offscreen .venv/bin/python -m pytest -q -rs
.venv/bin/canlab-cli detect canlab/sample_data/*.csv --json report.json
```

The per-message results above are read from `report.json`. The README's single skip is the MDF4 parser without `asammdf`; the author installed the `mdf` extra and saw no skip, consistent with that.

Real-data acceptance, not run by the author: download the corpora as `tests/real_data/README.md` describes [realdata], then

```
for p in 1 2 3; do QT_QPA_PLATFORM=offscreen python tests/real_data/acceptance_phase$p.py <data-dir>; done
python tests/real_data/acceptance_new_sources.py <data-dir>
```

Also not run: anything needing a CAN adapter, GVRET board, FD hardware, two bus channels, a real ECU, an AI key or network access, including the opendbc matcher's first fetch.

## References

1. ISO 11898-1:2015. Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling. International Organization for Standardization, 2015. www.iso.org/standard/63648.html
2. Sherin Joseph Roy. CanLab, release v2.0.0 (commit 00995a4). Source code, MIT. github.com/Sherin-SEF-AI/CanLab/tree/v2.0.0

3. Sherin Joseph Roy. CanLab real-data acceptance run, tests/real_data/README.md, release v2.0.0. github.com/Sherin-SEF-AI/CanLab/blob/v2.0.0/tests/real_data/README.md
4. cantools: CAN bus tools in Python (DBC, KCD, SYM, ARXML and CDD parsing, encoding and decoding). Source code, MIT. github.com/cantools/cantools
5. python-can: the CAN package for Python. Documentation. python-can.readthedocs.io

## Evidence

| Quantity | Value | Conditions | Basis | Source and reproduction |
|---|---|---|---|---|
| Rolling counters detected in the bundled sample capture | 6 counters | bundled sample capture (6,610 frames, 10 IDs); all reported as high-nibble counters wrapping at 16: byte 0 of 02C, 251, 260, 4F1, 544 and byte 7 of 4F1 | Reproduced 2026-11-10 | command: canlab-cli detect canlab/sample_data/<bundled sample>.csv --json report.json<br>commit 00995a4 (v2.0.0) · AMD Ryzen 9 9900X (12 cores / 24 threads), ~30 GB RAM, Linux 7.0.0-42-generic (Ubuntu), Python 3.12.13, CPU only<br>`git clone https://github.com/Sherin-SEF-AI/CanLab && cd CanLab && git checkout v2.0.0 && uv venv --python 3.12 && uv pip install -e '.[dev,ai,rest,mcp,mdf,adapters]' && .venv/bin/canlab-cli detect canlab/sample_data/*.csv --json report.json` |
| Checksums detected in the bundled sample capture | 7 checksums | bundled sample capture (6,610 frames, 10 IDs); all 7 reported as SUM8 in byte 7 at 100% confidence, on 018, 02C, 050, 251, 260, 316, 4F1; the sample's generator script (canlab/sample_data/generate_sample.py) writes a SUM8 into byte 7 of exactly these 7 IDs | Reproduced 2026-11-10 | command: canlab-cli detect canlab/sample_data/<bundled sample>.csv --json report.json<br>commit 00995a4 (v2.0.0) · AMD Ryzen 9 9900X (12 cores / 24 threads), ~30 GB RAM, Linux 7.0.0-42-generic (Ubuntu), Python 3.12.13, CPU only<br>`git clone https://github.com/Sherin-SEF-AI/CanLab && cd CanLab && git checkout v2.0.0 && uv venv --python 3.12 && uv pip install -e '.[dev,ai,rest,mcp,mdf,adapters]' && .venv/bin/canlab-cli detect canlab/sample_data/*.csv --json report.json` |
| Counters detected in the bundled sample (README claim) | 6 counters | README: 'the detector finds exactly the 6 counters and 7 checksums the generator wrote, with the right byte and the right algorithm' | Reported 2026-09-15 | README.md (Analysis, On checksum detection)<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>`not documented (reproducible with canlab-cli detect on the bundled sample)` |
| Checksums detected in the bundled sample (README claim) | 7 checksums | README: the detector finds exactly the 7 checksums the generator wrote, with the right byte and algorithm | Reported 2026-09-15 | README.md (Analysis, On checksum detection)<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>`not documented (reproducible with canlab-cli detect on the bundled sample)` |
| Unit test suite: tests passed | 540 tests | 540 passed, 0 failed, 0 skipped, 2 warnings; extras dev,ai,rest,mcp,mdf,adapters (the CI test job's set); Qt offscreen; no CAN hardware, AI keys or network services used (adapters, AI providers and buses are stand-ins or python-can's virtual bus) | Reproduced 2026-11-10 | command: QT_QPA_PLATFORM=offscreen python -m pytest -q -rs<br>commit 00995a4 (v2.0.0) · AMD Ryzen 9 9900X (12 cores / 24 threads), ~30 GB RAM, Linux 7.0.0-42-generic (Ubuntu), Python 3.12.13, CPU only |

| Quantity | Value | Conditions | Basis | Source and reproduction |
|---|---|---|---|---|
| | | | | `git clone https://github.com/Sherin-SEF-AI/CanLab && cd CanLab && git checkout v2.0.0 && uv venv --python 3.12 && uv pip install -e '.[dev,ai,rest,mcp,mdf,adapters]' && QT_QPA_PLATFORM=offscreen .venv/bin/python -m pytest -q` |
| Unit test suite duration | `54.97 s` | as reported by pytest for 540 tests; whole command including interpreter start-up measured 58.780242880 s by date before/after | Reproduced 2026-11-10 | command: QT_QPA_PLATFORM=offscreen python -m pytest -q -rs<br>commit 00995a4 (v2.0.0) · AMD Ryzen 9 9900X (12 cores / 24 threads), ~30 GB RAM, Linux 7.0.0-42-generic (Ubuntu), Python 3.12.13, CPU only<br>`git clone https://github.com/Sherin-SEF-AI/CanLab && cd CanLab && git checkout v2.0.0 && uv venv --python 3.12 && uv pip install -e '.[dev,ai,rest,mcp,mdf,adapters]' && QT_QPA_PLATFORM=offscreen .venv/bin/python -m pytest -q` |
| Tests passing (README badge) | `540 tests` | README shields.io badge 'tests 540 passing'; the README status note and docs/reference.html also say 540 | Reported 2026-09-15 | README.md (badge, line 11)<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>`QT_QPA_PLATFORM=offscreen python -m pytest -q` |
| Tests passed (README Testing section) | `540 tests` | '540 passed' in the Testing section; 'the single skip is the MDF4 parser, which needs the optional asammdf extra' | Reported 2026-09-15 | README.md (Testing)<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>`pip install -e '.[dev]' && QT_QPA_PLATFORM=offscreen python -m pytest -q` |
| Real-capture acceptance checks | `90 checks` | two corpora: SavvyCAN example capture (12,974 frames, 180 IDs, 11-bit, one bus) 36 checks; third-party CAN-logger MDF4 recordings plus python-can BLF/ASC format files (2 to 154,896 frames, 11- and 29-bit, dual-bus, CAN FD, error frames) 54 checks | Reported 2026-09-15 | README.md (Validated against real captures)<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>`download the corpora as described in tests/real_data/README.md, then: for p in 1 2 3; do python tests/real_data/acceptance_phase$p.py <data-dir>; done; python tests/real_data/acceptance_new_sources.py <data-dir>` |
| Real-capture acceptance, phase 1 (loading, panels, offline analysis): checks passed | `14 checks` | 14/14 passed on the SavvyCAN example capture (12,974 frames, 180 IDs); stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 | tests/real_data/README.md<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>`download the corpora as described in tests/real_data/README.md, then: QT_QPA_PLATFORM=offscreen python tests/real_data/acceptance_phase1.py tests/real_data` |
| Real-capture acceptance, phase 2 (decoding, exports, codegen, TX gate, | `11 checks` | 11/11 passed on the SavvyCAN example capture; every export loaded back by cantools; disarmed sent 0 frames, armed | Reported 2026-09-15 | tests/real_data/README.md<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated |

| Quantity | Value | Conditions | Basis | Source and reproduction |
|---|---|---|---|---|
| REST, MCP): checks passed | | sent 1; stated as 'What a good run looks like' (expected output), not a dated run log | | download the corpora as described in tests/real_data/README.md, then: QT_QPA_PLATFORM=offscreen python tests/real_data/acceptance_phase2.py tests/real_data |
| Real-capture acceptance, phase 3 (MDF4, project round trip, TX workers, ISO-TP): checks passed | 11 checks | 11/11 passed on the SavvyCAN example capture; 3000 real frames survive an MDF4 round trip; replay, fuzzer and sweep each send 0 when disarmed; scripted ECU answers OBD-II and UDS over ISO-TP on a virtual bus; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 | tests/real_data/README.md<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>download the corpora as described in tests/real_data/README.md, then: QT_QPA_PLATFORM=offscreen python tests/real_data/acceptance_phase3.py tests/real_data |
| Second real corpus (third-party logger MDF4 and python-can format files): checks passed | 54 checks | 54/54 passed; 5 third-party CAN-logger MDF4 recordings (public, third-party, 2 to 154,896 frames, 29-bit and dual-bus), 14 BLF/ASC files incl. 64-byte CAN FD and error frames, 9,600 real 29-bit NMEA 2000 frames identical across csv, blf, asc, log and pcap; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 | tests/real_data/README.md<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>download the corpora as described in tests/real_data/README.md, then: python tests/real_data/acceptance_new_sources.py <data-dir> |
| Counters found on the SavvyCAN real capture | 25 counters | 25 counters and 5 checksums across 19 messages, out of 180 IDs in 12,974 real frames; no ground truth is stated, so these are detections, not verified hits; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 | tests/real_data/README.md<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>download the corpora as described in tests/real_data/README.md, then: QT_QPA_PLATFORM=offscreen python tests/real_data/acceptance_phase1.py tests/real_data |
| Checksums found on the SavvyCAN real capture | 5 checksums | 5 checksums and 25 counters across 19 messages, out of 180 IDs in 12,974 real frames; no ground truth is stated, so these are detections, not verified hits; stated as 'What a good run looks like' (expected output), not a dated run log | Reported 2026-09-15 | tests/real_data/README.md<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · not stated<br>download the corpora as described in tests/real_data/README.md, then: QT_QPA_PLATFORM=offscreen python tests/real_data/acceptance_phase1.py tests/real_data |
| Build the full DataFrame from the ring buffer | 5 ms | capture of 150,000 frames across 16 IDs held in memory; README: 'treat these as the shape of the cost rather than a benchmark' | Reported 2026-09-15 | README.md (Performance)<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · development machine (Linux, Python 3.12); CPU and RAM not stated<br>not documented |
| Fetch one ID's frames | 0.5 ms | capture of 150,000 frames across 16 IDs held in memory; README: 'treat these as the shape of the cost rather than a benchmark' | Reported 2026-09-15 | README.md (Performance)<br>commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · development machine (Linux, Python 3.12); CPU and RAM not stated<br>not documented |

| Quantity | Value | Conditions | Basis | Source and reproduction |
|---|---|---|---|---|
| **Refresh the frame table, 1,000 rows on screen** | 72 ms | capture of 150,000 frames across 16 IDs held in memory; README: 'treat these as the shape of the cost rather than a benchmark' | Reported 2026-09-15 | README.md (Performance) commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · development machine (Linux, Python 3.12); CPU and RAM not stated not documented |
| **Rebuild the plot signal tree, 16 IDs** | 70 ms | capture of 150,000 frames across 16 IDs held in memory; README: 'treat these as the shape of the cost rather than a benchmark' | Reported 2026-09-15 | README.md (Performance) commit 5a59b43 (last change; tag v2.0.0 = 00995a4) · development machine (Linux, Python 3.12); CPU and RAM not stated not documented |

Table 3. Every measured quantity in this report, with its source, commit, hardware and the command that reproduces it. “Reproduced” rows were run by the author; “Reported” rows are quoted from the repository at the commit shown.

## Cite this report

APA

Roy, S. J. (2026). CanLab: offline counter, checksum and field-boundary detection for CAN captures, with a gated transmit path (Technical Report IF-TR-2026-02, Version 1.0). IsoFrames. https://isoframes.io/reports/if-tr-2026-02/

BibTeX

```
@techreport{isoframes_if_tr_2026_02,
  author      = {Roy, Sherin Joseph},
  title       = {CanLab: offline counter, checksum and field-boundary detection for CAN captures, with a gated
transmit path},
  institution = {IsoFrames, BluRabbit Mobility},
  number      = {IF-TR-2026-02},
  type        = {Technical Report},
  year        = {2026},
  month       = nov,
  note        = {Version 1.0},
  url         = {https://isoframes.io/reports/if-tr-2026-02/}
}
```