Application Review Report

The project: vehicle dashboard application

Document number	SQC-CR-2026-001, revision 1.0
Object of review	The project, vehicle dashboard application (C++20, Qt 6, QML)
Version tested	Git commit `17891cf7b1ca4a415b7cdc8d4c92cecca384be9b`, committed 2026-10-09, clean tree
Review date	2026-10-10
Reviewer	Erick, Squared Computing
Method	Static review, build, existing test suites, sanitizer run, targeted reproduction, headless application run
Result	**Not recommended for release** until SP-01 to SP-05 are resolved

Anonymisation: <app>, <vendor>, <board> and <kiosk-user> replace the product name, the third-party compatibility target, the embedded board and its login user in paths, code and console output. Absolute paths are shortened to <scratch> and <Qt>. All other path components and line numbers are exact. Snapshots were taken from the version tested.

1 Scope

Code base	C++20 about 54 000 lines; QML about 15 000 lines; GLSL 6 files; Java (Android USB bridge); Bash (appliance services)
Build system	CMake 3.21+, presets, Ninja; nine layered static libraries and one application
Dependencies	Qt 6.10; miniz 3.0.2 (vendored, MIT); usb-serial-for-android (vendored, MIT); MapLibre Native Qt (optional, BSD-2)
Entry points	`src/app/main.cpp`; `tools/sim/main.cpp`
Tests	74 Qt Test suites, non-UI code only
Reviewed in depth	Expression engine, value pipeline, channel table, lap timer, trip computer; zip, binary dashboard, document, CAN/OBD2 XML, DBC, ELM327, GPX and log parsers; transports; threading and lifetime; scripting and trust gate; appliance update service; Android manifest
Not reviewed in depth	QML user interface, gauge rendering and shaders, maps, media, voice, packaging scripts

2 Test environment and execution

Host	x86-64, AMD Ryzen 9 7940HS, 16 threads, 29 GB RAM
Operating system	Ubuntu 26.04.1 LTS, Linux 7.0.0
Toolchain	GCC 15.2.0, CMake 4.2.3, Ninja 1.13.2, Qt 6.10.2 (gcc_64)
Application runs	Xvfb virtual display 1920×1080, Mesa llvmpipe software rendering, built-in vehicle simulator; settings, data, state and cache directories isolated per run
Source tree	Not modified; out-of-tree build; `git status` clean before and after

```
cmake -S . -B <scratch>/build -G Ninja -DCMAKE_BUILD_TYPE=RelWithDebInfo -DCMAKE_PREFIX_PATH=<Qt>/gcc_64
cmake --build <scratch>/build -j8
QT_QPA_PLATFORM=offscreen ctest --test-dir <scratch>/build --output-on-failure -j4
# sanitizer build of 15 parser and core suites
cmake ... -DCMAKE_CXX_FLAGS="-fsanitize=address,undefined -fno-sanitize-recover=undefined -fno-omit-frame-pointer"
# application run
<app> --settings <run>/settings.ini --data <run>/user --size 1920x1080 --scenario dash \
      [--dashboard <file> | --description <file>] --shot <png> --shot-after 4000
```

Step	Result	Record
Build, project code with `-Wall -Wextra -Wpedantic -Werror`	Pass	766 steps, 0 warnings, 2 min 12 s
Test suites (74)	Pass	74/74, 6.4 s
ASan and UBSan, 15 suites	Pass	0 sanitizer reports
Application start, built-in dashboard	Pass	First frame 0.8 s, 0 QML warnings (Figure 2)
Android, Windows, embedded builds	Not performed	See section 8

```
$ cmake --build <scratch>/build -j8 | tail -2
[764/766] Linking CXX executable tests/tst_log_library
[765/766] Linking CXX executable tests/tst_replay_seek
$ grep -c 'warning:' build.log
0
$ ctest --test-dir <scratch>/build --output-on-failure -j4 | tail -4

100% tests passed, 0 tests failed out of 74

Total Test time (real) =   6.42 sec
$ ctest --test-dir <scratch>/asan -R "<15 parser and core suites>" | tail -4

100% tests passed, 0 tests failed out of 15

Total Test time (real) =  12.11 sec
```

Figure 1. Build and test record (console capture).



Figure 2. Application under test, built-in dashboard, simulated vehicle data. Reference state for Figures 4 and 7.

3 Summary

Parsers are defensive, the cross-thread channel table is correct, and the build and test suite are clean. Defects concentrate where the code meets untrusted files and driver errors. Five High findings: script execution at load, memory corruption from a CAN description file, a crash on CAN adapter open failure, an Android serial handle leak, and a lap timer that records no laps on large circuits. Four of the five were reproduced; two of those also in the running application.

	Critical	High	Medium	Low	Info	Total
Findings	0	5	4	8	2	19
Reproduced	0	4	0	2	0	6
Static analysis only	0	1	4	6	2	13

4 Findings register

ID	Title	Severity	Status	Area
SP-01	Script text executed at parse time, before the trust check	High	Reproduced	Scripting
SP-02	Offset overflow in CAN description causes out-of-bounds access	High	Reproduced	Protocol parser
SP-03	Null dereference when a Qt CAN device fails to open	High	Reproduced	Transport
SP-04	Android USB serial handle not released on source close	High	Static	Transport
SP-05	Lap timer discards track and laps beyond 2 km from start line	High	Reproduced	Lap timing
SP-06	Script trust gate is a deny-list; copies and plain documents bypass it	Medium	Static	Scripting
SP-07	Update payload hashed and extracted from two separate reads	Medium	Static	Appliance update
SP-08	Exit during import aborts the process (`qFatal`)	Medium	Static	Lifetime
SP-09	Half-open TCP or Bluetooth link never detected	Medium	Static	Transport
SP-10	Upper-case function names read as bytes in letter naming	Low	Reproduced	Expressions
SP-11	Quadratic parse time on crafted binary dashboard; no cancel	Low	Reproduced	Import
SP-12	UDP link accepts datagrams from any sender	Low	Static	Transport
SP-13	Trigger can clear vehicle fault codes without confirmation	Low	Static	Behaviour
SP-14	Root report writer follows symbolic links	Low	Static	Appliance
SP-15	Library file copy misses short writes; not atomic	Low	Static	Storage
SP-16	Choosing the stored car model deletes it	Low	Static	Storage
SP-17	Replay start offset ignored	Low	Static	Replay
SP-18	Share server on all interfaces, no authentication	Info	Static	Network
SP-19	Android backup and shell profiling enabled	Info	Static	Android

Static: static analysis, not reproduced at runtime.

5 Findings

SP-01 Script text executed at parse time, before the trust check

Severity	High
Status	Reproduced (harness and application)
Location	`src/dash/script_host.cpp:15-18`, `:40-45`, `:114-135`, `:228-231`; `src/dash/dash_behaviour.cpp:401`, `:442`

Description. Syntax check concatenates script text into `"(function () {" + code + "})"` and evaluates it. Text containing `}); ... (function () {` closes the wrapper and runs at top level. Check runs inside `DashDocument::fromJson`, on the GUI thread, before the trust gate, without watchdog. Worker `load()` repeats the evaluation; watchdog covers `call()` only. Engine has no file or network API: impact is denial of service. Chosen dashboard is persisted before load (`src/app/app_controller.cpp:706`); a hang recurs on restart (code reading).

Reproduction, harness. Minimal document with one script action passed to `DashDocument::fromJson` (Figure 3): breakout that throws is reported as a problem raised by executed code; silent breakout is accepted with no problem;

infinite loop never returns (killed by `timeout`, status 124).

Reproduction, application. Built-in dashboard plus one script action, opened with `--dashboard`. Throwing variant: application's own problem banner reports the exception raised during reading (Figure 4). Looping variant: no window content after 8 s; screenshot never taken; process killed at the 20 s limit, status 124 (Figure 5). Baseline first frame: 0.8 s.

```
$ ./script_repro throw
mode=throw valid=1 actions=1 problem='line 1: injected code ran at parse time' elapsed=0 ms
[exit status 0]
$ ./script_repro quiet
mode=quiet valid=1 actions=1 problem='' elapsed=0 ms
[exit status 0]
$ timeout 10 ./script_repro loop
[exit status 124]
```

Figure 3. SP-01. Harness output for the three script variants.

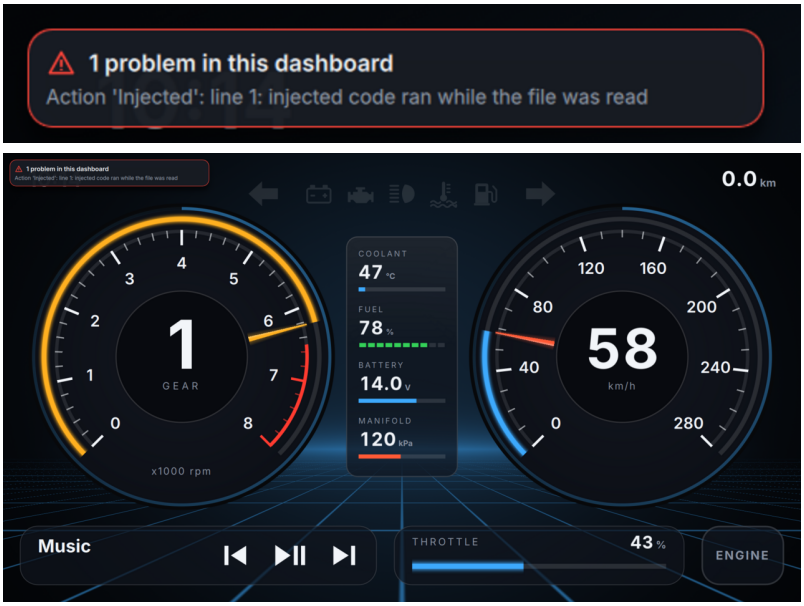

Figure 4. SP-01. Application with the throwing script variant. Top: enlarged banner, "Action 'Injected': line 1: injected code ran while the file was read". Bottom: full window; dashboard otherwise unchanged from Figure 2.



Figure 5. SP-01. Display 8 s after start with the looping script variant: no content rendered. Process terminated by timeout at 20 s (exit status 124).

Expected. Loading never executes scripts. **Actual.** Arbitrary script runs at load; application hangs.

Fix. Compile body with the `Function` constructor (parses, does not run); arm watchdog in `load()`.

```
QJSValue compileBody(QJSEngine &engine, const QString &code)
{
    return engine.globalObject().property(QStringLiteral("Function"))
        .callAsConstructor({QJSValue(code)});
}
```

SP-02 Offset overflow in CAN description causes out-of-bounds access

Severity	High
Status	Reproduced (harness and application)
Location	`src/protocols/<vendor>/rd_can_description.cpp:259`, `:315`, `:321`; composite id `:170–179`; use in `:525`, `:636`, `:766`

Description. `offset` parsed as `qint64`, checked only for `< 0`. `offset + length > limit` overflows for values near `INT64_MAX` (undefined behaviour, wraps negative) and passes. `int(offset)` truncates `0x7FFFFFFFFFFFFFF8` to -8. Decode reads and encode writes 8 bytes before the payload (`QByteArray` heap header). Such files are shared between users.

Reproduction, harness. `<value targetId="37" offset="0x7FFFFFFFFFFFFFF8" length="8"/>` in frame `0x100`; one frame of eight `0x11` bytes applied. Release build: file accepted; decoded value 8 instead of 1.23×10^{18}. UBSan build: signed overflow at line 315 (Figure 6).

Reproduction, application. Built-in description with the engine speed value changed to the offset above, passed with `--description`. File accepted with no error shown. Engine speed reads about 0 while the simulated vehicle runs at 58 km/h in first gear; baseline about 6 300 rpm (Figure 7). The simulator encodes through the same description, so the out-of-bounds write path also ran; no crash within the 4 s run. Heap corruption was not instrumented in the application run.

```
$ ./canxml_plain
accepted=1 error='' values=1
payload is 8 bytes of 0x11; decoded value = 8 (0x1111111111111111 would be 1.2297829382473034e+18)
[exit status 0]
$ ./canxml_repro
src/protocols/<vendor>/rd_can_description.cpp:315:20: runtime error: signed integer overflow: 8 + 9223372036854775800 cannot be
represented in type 'long long int'
[exit status 1]
```

Figure 6. SP-02. Harness output, release build (top) and UBSan build (bottom).



(a) Built-in description (b) Offset `0x7FFFFFFFFFFFFFF8`

Figure 7. SP-02. Engine speed gauge, same simulated state. (a) about 6 300 rpm. (b) about 0 rpm; value decoded from memory before the frame payload. No error displayed.

Expected. File rejected. **Actual.** File accepted; out-of-bounds read per frame, heap metadata written per send.

Fix. Bound before arithmetic; same for composite offset and `startbit`.

```
if (!offsetText.isEmpty() && (!parseInteger(offsetText, &offset) || offset < 0
        || offset > CanDescription::kMaxFrameBytes)) {
```

SP-03 Null dereference when a Qt CAN device fails to open

Severity	High
Status	Reproduced (harness)
Location	`src/links/qt_can_bus.cpp:135-143`; `fail :92`; `drop :100`

Description. On `open()` failure, `QCanBusDevice::connectDevice()` emits `UnconnectedState` synchronously (Qt 6.10.2 source). The connected handler calls `fail()`, which calls `drop()` and sets `m_device = nullptr`. Line 143 then calls `m_device->errorString()`.

Reproduction. `QtCanBus("socketcan:nosuchcan0", 0).open()`: SIGSEGV; frame #1 at `qt_can_bus.cpp:143` with `this=0x0` in frame #0 (Figure 8). In the application this path serves vendor drivers (PEAK, Vector; mainly Windows); those were not run. Not reachable from the Linux UI, which uses its own SocketCAN link.

```
$ gdb -batch -ex run -ex bt ./can_repro
opening socketcan:nosuchcan0
qt.canbus.plugins.socketcan: Cannot load library socketcan: socketcan: cannot open shared object file: No such file or directory
qt.canbus.plugins.socketcan: Cannot load library libsocketcan, some functionality will not be available.
Cannot load library socketcan: socketcan: cannot open shared object file: No such file or directory
Program received signal SIGSEGV, Segmentation fault.
QCanBusDevice::errorString (this=0x0) at qtserialbus/src/serialbus/qcanbusdevice.cpp:402
#0  QCanBusDevice::errorString (this=0x0) at qtserialbus/src/serialbus/qcanbusdevice.cpp:402
#1  0x0000555555561666 in QtCanBus::openNow (this=0x7fffffffcc30) at src/links/qt_can_bus.cpp:143
#2  0x00005555555619c0 in QtCanBus::open (this=<optimized out>) at src/links/qt_can_bus.cpp:81
#3  0x000055555556098e in main (argc=<optimized out>, argv=0x7fffffffce18) at can_repro.cpp:15
```

Figure 8. SP-03. Debugger backtrace of the crash.

Expected. Error state and reconnect. **Actual.** Crash; repeats if the connection is restored at start.

Fix.

```
if (!m_device->connectDevice()) {
    if (m_device)
        fail(tr("%1 on %2 cannot be opened: %3").arg(m_channel, m_driver, m_device->errorString()));
    return;
}
```

SP-04 Android USB serial handle not released on source close

Severity	High (Android)
Status	Static analysis, not reproduced at runtime
Location	`src/links/serial_link.h`; `src/links/serial_link_android.cpp:83-90`; `src/app/data_engine.cpp:224-228`; `packaging/android/src/com/<app>/app/RaceUsbSerial.java:30-49`

Description. `SerialLink` has no destructor; the Java handle is released only in `close()`/`drop()`. `DataEngine::closeSource()` deletes the source without closing the link. Each reconnect, connection change or replay leaves the Java `Line` registered: its reader thread keeps filling an undrained inbox (unbounded memory) and its USB claim is expected to block reopening the adapter. Desktop builds unaffected (`QSerialPort` is a child object).

Trigger. Any source over Android USB serial; reconnect from the Garage screen.

Expected. Port released. **Actual (code reading).** Handle and thread leak; reopen likely fails.

Fix.

```
SerialLink::~SerialLink() { drop(); }
```

SP-05 Lap timer discards track and laps beyond 2 km from start line

Severity	High
Status	Reproduced (harness)
Location	`src/services/computers/lap_computer.cpp:318-347`; `src/services/computers/lap_computer.h:45`

Description. Every 5 s `choose()` selects the track whose start line lies within `kNearestWithinM` = 2000 m. Outside that radius `nearest` is empty; the branch `nearest != m_active` clears `m_active`, `m_timing` and `m_laps`. On any circuit with a point more than 2 km from the line, no lap completes. No message is shown.

Reproduction. Real `LapComputer`, 50 m/s, 10 Hz GPS, 3.5 laps per track (Figures 9, 10).

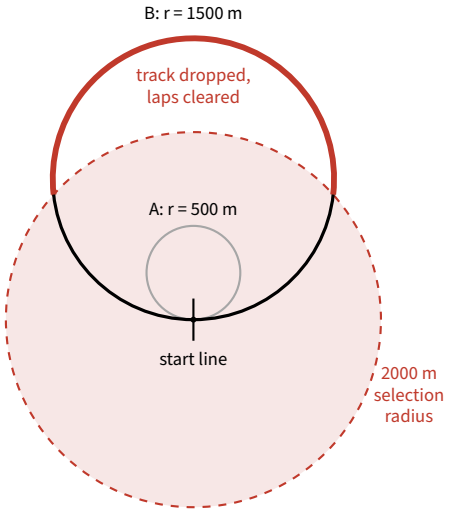

Figure 9. SP-05. Test geometry. Track A stays inside the 2000 m selection radius and records 3 of 3 laps. Track B leaves it on the highlighted arc (beyond 2000 m from the start line); the active track and its laps are discarded there on every lap, and 0 laps are recorded.

```
$ ./repro
circle radius 500 m (farthest point 1000 m from the line), 219.9 s driven, 3.5 laps of 62.8 s each: laps recorded = 3
circle radius 1500 m (farthest point 3000 m from the line), 659.7 s driven, 3.5 laps of 188.5 s each: laps recorded = 0
[exit status 0]
```

Figure 10. SP-05. Harness output for tracks A and B.

Expected. 3 laps on both. **Actual.** 0 laps on track B.

Fix. Keep the timed track until another track is found; derive the radius from track extent.

```
const bool keepTimedTrack = nearest.isEmpty() && m_timing && findTrack(m_active);
if (nearest != m_active && !keepTimedTrack) {
```

SP-06 Script trust gate is a deny-list; copies and plain documents bypass it

Severity	Medium
Status	Static analysis, not reproduced at runtime
Location	`src/app/app_controller.cpp:744-747, :738, :710-730, :948-956, :1101-1120`

Description. Scripts are trusted unless the id is in `scripts/untrusted`; only package import adds entries. Plain documents opened from the file dialog, and copies made by save-as or edit-own-copy, come out trusted. Trusted `set()` reaches `Session::setValue` and sends frames for writable channels (`src/app/data_engine.cpp:762–776`). Non-script actions are outside the gate.

Trigger. Import untrusted package; save a copy; copy runs scripts.

Fix. Allow-list (built-in, user-created, explicitly allowed); copies inherit the source's state.

SP-07 Update payload hashed and extracted from two separate reads

Severity	Medium
Status	Static analysis, not reproduced at runtime
Location	`packaging/<board>/rootfs/usr/local/lib/<app>/stick:160–161`

Description. Manifest signature verified correctly. Payload is read once for `sha256sum` and again for extraction. A malicious mass-storage device can return different data on the second read; unsigned code would be installed and run by the kiosk session (extraction runs unprivileged).

Fix. Copy to root-owned local storage once; verify and extract from the copy only.

```
copy="$root/.incoming.update"
cp -- "$file" "$copy" || { set_update refused "$version" damaged; return; }
```

SP-08 Exit during import aborts the process

Severity	Medium
Status	Static analysis, not reproduced at runtime
Location	`src/app/app_controller.cpp:773–804`, `:1073–1081`

Description. Import `QThread` is a child of `AppController`. The destructor waits for the thread pool only. Destroying a running `QThread` calls `qFatal("QThread: Destroyed while thread . . . is still running")` (Qt 6.10.2 source). Staging folder is left behind.

Fix. Call existing `RdImportJob::cancel()`, then `quit()` and `wait()` the thread in the destructor.

SP-09 Half-open TCP or Bluetooth link never detected

Severity	Medium
Status	Static analysis, not reproduced at runtime
Location	`src/links/tcp_link.cpp:55–71`; `src/links/bluetooth_link.cpp:137–157`

Description. No keep-alive, no read-idle timeout. Receive-only adapters that reboot without FIN leave the link open indefinitely; values time out but reconnect never runs.

Fix. `KeepAliveOption` plus an idle timer that fails the link after a configurable silence.

SP-10 Upper-case function names read as bytes in letter naming

Severity	Low
Status	Reproduced (harness)
Location	`src/core/expression.cpp:523–527` (before lookup at `:542`)

Description. In letter byte naming, any 1 to 3 upper-case letter word is a byte index before function lookup. Lookup is otherwise case-insensitive.

Reproduction. Byte 0 = 200 (Figure 11). Letters mode: `PI` valid, value 0 (byte 424); `ABS(A–B)` and `MAX(A,B)` rejected; `abs(A–B)` = 200. Indexed mode: `PI` = 3.14159. Corpus: 10 letter-mode expressions, none with functions; real-world impact unconfirmed.

```
$ ./repro   (expression section)
Indexed  pi             valid=1 value=3.14159 maxByteIndex=-1 error=''
Indexed  PI             valid=1 value=3.14159 maxByteIndex=-1 error=''
Letters  pi             valid=1 value=3.14159 maxByteIndex=-1 error=''
Letters  PI             valid=1 value=0 maxByteIndex=424 error=''
Letters  abs(A-B)       valid=1 value=200 maxByteIndex=1 error=''
Letters  ABS(A-B)       valid=0 value=nan maxByteIndex=-1 error='Unexpected '('
Letters  MAX(A,B)       valid=0 value=nan maxByteIndex=-1 error='Byte index 8837 is out of range'
Indexed  ABS(B0-B1)     valid=1 value=200 maxByteIndex=1 error=''
```

Figure 11. SP-10. Expression compiler output in both naming modes.

Fix. Resolve as function first when followed by `(`; define and document `PI`, `E`.

SP-11 Quadratic parse time on crafted binary dashboard; no cancel

Severity	Low
Status	Reproduced (harness)
Location	`src/rdimport/rd_file.cpp:181-199`, `:206-217`

Description. `nextBoundary` calls `trailerAt` at every zero byte; each call may read up to 4096 strings. Cancel flag polled only between resources and pages.

Reproduction. Header, one unknown object, repeated `[u32 0][u32 4096]`; release build (Figures 12, 13). Second run shown; first run 0.92, 4.04 and 17.77 s.

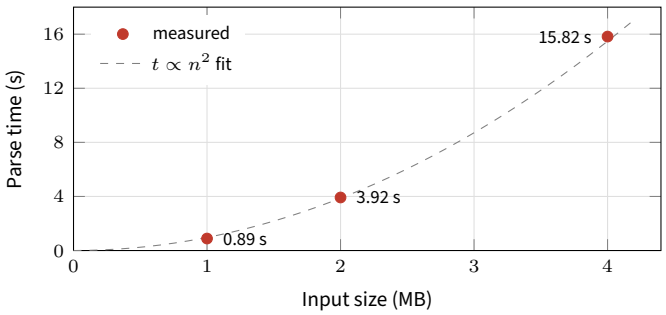

Figure 12. SP-11. Parse time against input size. Doubling the input multiplies the time by about four.

```
$ ./rd_slow 1
1 MB: 887 ms valid=1 objects=1
[exit status 0]
$ ./rd_slow 2
2 MB: 3923 ms valid=1 objects=1
[exit status 0]
$ ./rd_slow 4
4 MB: 15823 ms valid=1 objects=1
[exit status 0]
```

Figure 13. SP-11. Harness output, second run.

Fix. Locate the trailer once from end of file; poll cancel inside the object loop.

SP-12 UDP link accepts datagrams from any sender

Severity	Low
Status	Static analysis, not reproduced at runtime
Location	`src/links/udp_link.cpp:38`, `:66-80`

Description. Every datagram is decoded regardless of sender. Without a fixed peer, the last sender becomes the write destination. A host on the same network can inject readings.

Fix.

```
if (m_fixedPeer && datagram.senderAddress() != m_peer)
    continue;
```

SP-13 Trigger can clear vehicle fault codes without confirmation

Severity	Low
Status	Static analysis, not reproduced at runtime
Location	src/dash/behaviour_engine.cpp:515-516; src/app/qml/DashScene.qml:271; src/app/app_controller.cpp:640-651

Description. App action `faultsClear` sends an OBD2 clear request; actions fire from trigger conditions and are outside the trust gate. An imported dashboard can clear stored codes and reset readiness monitors silently.

Fix. Confirm, or accept only from a direct user tap.

SP-14 Root report writer follows symbolic links

Severity	Low
Status	Static analysis, not reproduced at runtime
Location	packaging/<board>/rootfs/usr/local/lib/<app>/stick:89, :92

Description. Root copies /home/<kiosk-user>/.local/state/<app>/app.log* with plain cp: a link planted by the kiosk user exfiltrates root-only files to the stick. Archive written to a predictable name on the stick; a planted link on a link-capable filesystem redirects the write. Automounter filesystem support not checked.

Fix. cp -P; build archive locally; refuse an existing or linked destination.

SP-15 Library file copy misses short writes; not atomic

Severity	Low
Status	Static analysis, not reproduced at runtime
Location	src/app/dashboard_library.cpp:49-59; caller :255

Description. `write(...) < 0` misses short writes; close errors lost; package written in place, stamp written after. Truncated package recorded as current on a full disk.

Fix. `QSaveFile`; compare written size; `commit()`.

SP-16 Choosing the stored car model deletes it

Severity	Low
Status	Static analysis, not reproduced at runtime
Location	src/app/app_controller.cpp:1382-1384

Description. `QFile::remove(target)` precedes the copy without a same-file check; selecting the file already in the `cars` folder deletes it. The map chooser has this guard.

Fix. If `QFileInfo(source) == QFileInfo(target)`, select without copying.

SP-17 Replay start offset ignored

Severity	Low
Status	Static analysis, not reproduced at runtime
Location	src/app/data_engine.cpp:479; src/services/log/replay_link.cpp:28-33, :45

Description. `seek()` called before `open()` clamps to a duration of 0; `open()` re-seeks to 0. Reachable only through `--replay-from`.

Fix. Store the requested position unclamped; clamp in `open()`.

SP-18 Share server on all interfaces, no authentication

Severity	Info
Status	Static analysis, not reproduced at runtime
Location	`src/services/share/share_server.cpp:46-48`; default off `src/services/garage/connection_store.h:103`

Description. When enabled, streams all channels including GPS to any network client; no client limit. Off by default. Recommend a pairing code and a visibility notice.

SP-19 Android backup and shell profiling enabled

Severity	Info
Status	Static analysis, not reproduced at runtime
Location	`packaging/android/AndroidManifest.xml:26, :28`

Description. `allowBackup="true"` includes dashboards, settings, trips and logs in backups. `profileable shell="true"` is set in release builds. Set both deliberately for release.

6 Areas examined without findings

Zip archive	Unsafe names rejected (absolute, `..`, `\`, `:`); count and size caps; declared size enforced; atomic writes
Binary dashboard reader	Bounds-checked cursor; string, count and depth caps; resource sizes checked
Expression engine	Nesting 64, compile-time stack bound, run-time index checks, guarded shifts and factorial
Value pipeline	Non-finite results set Invalid; bad expressions set Error
Channel table	Mutex-guarded pending writes; front buffer written by GUI latch only; no lost wake-up
Data engine threading	Links created and destroyed on the I/O thread; ordered shutdown and join; pool tasks post back with `QPointer`
Stream decoders	Frame, SLCAN, ELM327, DBC: lengths checked, buffers bounded
GPX and recording log	Size cap, coordinate range check, record lengths checked, truncated tail tolerated
Trip and track storage	`QSaveFile`, errors surfaced, replay isolated
Process execution	No `QProcess`, `system()` or `popen` in application source
Appliance shell service	Quoting, positional arguments to `runuser`, version validated, signature verified twice, downgrade refused
Secrets	None committed; signing keys held outside the repository

7 Test checklist

Area	Status	Notes
Linux build, warnings as errors	Passed	GCC 15.2, Qt 6.10.2 (Figure 1)
Unit test suites (74)	Passed	6.4 s, offscreen
Parser and core suites under ASan/UBSan (15)	Passed	No reports
Application start, built-in dashboard, simulator	Passed	Figure 2

Area	Status	Notes
Script loading, hostile text	Failed	SP-01; harness and application
CAN description, out-of-range offset	Failed	SP-02; harness and application
Qt CAN device open failure	Failed	SP-03; harness
Lap timing, point beyond 2 km	Failed	SP-05; harness
Lap timing, small track (control)	Passed	3/3 laps
Letter-mode expressions with functions	Failed	SP-10
Binary dashboard reader, crafted input	Failed	SP-11
Zip traversal and limits	Passed	Existing suite under ASan; static review
Editor and interactive UI	Not tested	No input driven beyond start-up
Android build and run	Not tested	Not attempted
Android USB serial reconnect	Not tested	Device and adapter required
Windows build and run	Not tested	No Windows host
Appliance image and USB update	Not tested	Scripts read only
Frame-time benchmark	Not tested	Not run
Vehicle, CAN, OBD2 hardware	Not tested	No hardware
Exit during import	Not tested	SP-08 by code reading
Silent adapter loss	Not tested	SP-09 by code reading

8 Limitations and release blockers

Limitations

- Linux desktop only built and run. Android, Windows and appliance findings (SP-04, SP-07, SP-14, SP-19) are static.
- Application run headless with software rendering and simulated data; start-up only, no interactive UI exercised. Rendering performance figures from this setup are not representative of target hardware.
- No hardware exercised. SP-03 triggered with a driver failing on this host, not with the vendor drivers on this path.
- Sampled review in limited time. QML, renderers, shaders, maps, media and voice reviewed lightly or not at all; no findings there is not evidence of no defects.
- No fuzzing; parsers exercised by existing tests and targeted harnesses only.

Release blockers

SP-01	Any shared dashboard can hang the application at every start.
SP-02	Memory corruption from a commonly shared file type.
SP-03	Crash on a routine hardware condition.
SP-05	Silent failure of lap timing on full-size circuits, if the feature is advertised.
SP-04	Android release; confirm on a device.

SP-06 to SP-09: next release. Low and Info: backlog.

Excluded candidates

Raised but not confirmed to report standard, not counted: script host destructor hang; watchdog interrupting the wrong script; voice listener stuck listening; unflushed recording tail; unknown palette keys dropped on save; 64-bit DBC encode cast; importer and document nesting mismatch.

Appendix A Severity definitions

Critical	Remote compromise, large-scale data loss, or a defect endangering vehicle operation, reachable without unusual conditions.
High	Crash, memory corruption, hang or security bypass reachable in normal use or from a commonly shared file; or silent failure of a core workflow.
Medium	Real impact requiring an uncommon condition, local attacker or timing; or a manual workaround exists.
Low	Limited impact, narrow trigger or secondary feature.
Info	Not a defect; hardening or configuration note.

Appendix B List of figures

1 Build and test record (console capture). 2
2 Application under test, built-in dashboard, simulated data. 2
3 SP-01. Harness output for the three script variants. 4
4 SP-01. Application problem banner reporting code executed during reading. 4
5 SP-01. Application display 8 s after start, looping script variant. 4
6 SP-02. Harness output, release build (top) and UBSan build (bottom). 5
7 SP-02. Engine speed gauge, built-in and hostile description. 5
8 SP-03. Debugger backtrace of the crash. 6
9 SP-05. Test geometry and 2000 m selection radius. 7
10 SP-05. Harness output for tracks A and B. 7
11 SP-10. Expression compiler output in both naming modes. 9
12 SP-11. Parse time against input size. Doubling the input multiplies the time by about four. 9
13 SP-11. Harness output, second run. 9