Race to Charge

Concurrency Bugs and State Integrity Violations
in EV Charging Infrastructure

Jaeyeong Lee
Independent Researcher

Seonhyeong Lee
SK Shieldus

4th USENIX Symposium on Vehicle Security and Privacy

VehicleSec '26

Why EVSE software is concurrency-sensitive

EVSE software couples protocol timing to contactors, power delivery, authorization, and billing state.

> 5M

public charging points worldwide

+1.3M in 2024

01 **Hardware-coupled state**

Contactor · CP · power transitions

02 **Independent protocol threads**

OCPP · ISO 15118 · MQTT/API

03 **One session state machine**

Authorization · billing · power

RESEARCH GAP

Most prior work focuses on protocol specifications and input handling; implementation-level shared-state timing remains underexamined.

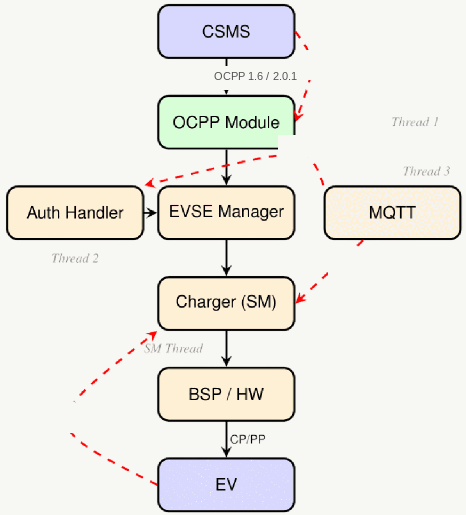
Adversarial timing of legitimate inputs

The attacker influences standard EVSE interfaces;
no code execution inside the charging process is assumed.

The attacker controls event timing —not message validity

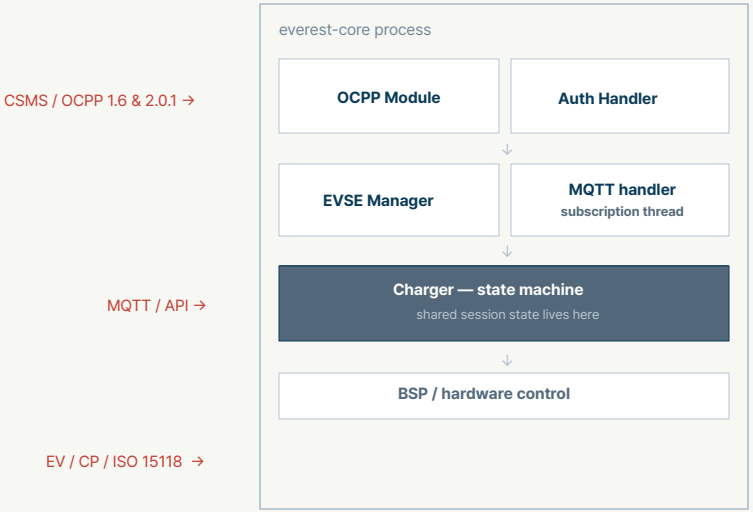
CSMS / OCPP	V1–V4
MQTT / API	V2, V3, V9-V10
EV / CP / ISO 15118	V1, V6–V8, V11
Startup timing	V5

IDs overlap when multiple channels are required.



Five execution contexts converge on shared state

We mapped five callback paths to seven shared fields, then audited and reproduced the vulnerable interleavings.



everest-core 2025.12.1 · THREE-PHASE METHOD

01 MAP THE ATTACK SURFACE
5 execution contexts × 7 shared fields

02 AUDIT EVERY ACCESS PATH
Read/write paths · lock coverage · atomicity · state validation

03 REPRODUCE AND VERIFY
Logic PoCs · TSan / ASan / UBSan · Raspberry Pi replay for V2

Six of seven mapped fields contain an explicitly unprotected access path

The paper's thread-to-shared-resource mapping shows where read/write paths cross protection boundaries.

Shared Resource	SM	OCPP	MQTT	Auth
shared_context	R/W P	–	W U	–
transaction_active	R/W P	R	–	R
authorized	R/W P	–	–	W U
event_queue	–	R/W P/U	–	–
evse_soc_map	–	R/W U	–	–
evse_evcc_id	–	R/W U	–	–
timeout_timer	–	–	–	W U

R read · W write · P protected · U unprotected
OCPP groups two callback contexts.

6 / 7

shared fields expose an unprotected path

event_queue

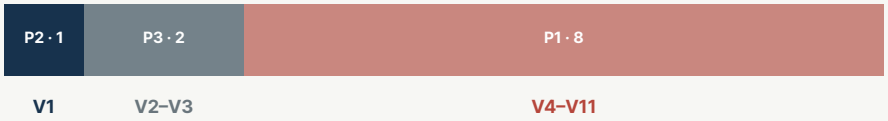
P/U in the same OCPP column reveals an inconsistent lock policy.

11 vulnerabilities and 11 public CVEs

Eight of the eleven share one primary root cause: inconsistent locking across module boundaries.

8 / 11

INCONSISTENT LOCKING



P2 Lack of atomicity

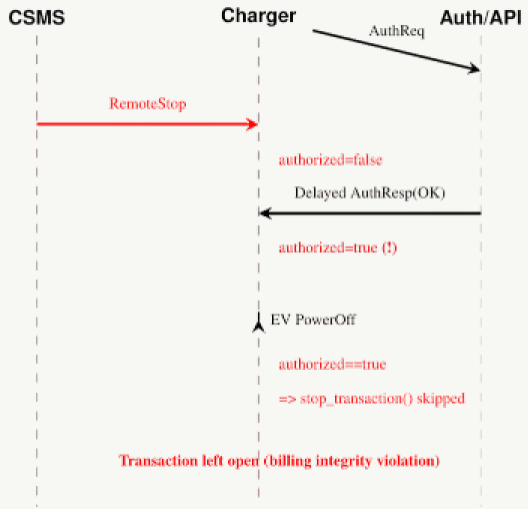
P3 Missing state validation

P1 Inconsistent locking

Shared-state ownership and locking must be specified across modules, not inferred independently inside each callback.

A delayed AuthResp defeats RemoteStop

V2 is a state-integrity failure: a valid response arrives in the wrong state.



Why the transaction remains open

1 **RemoteStop sets authorized = false.**

The charger enters StoppingCharging.

2 **A delayed AuthResp writes true.**

authorize() does not validate the current state.

3 **EV PowerOff observes true.**

stop_transaction() is skipped.

Result

Physical charging stops, but the OCPP transaction remains open.

The stop is accepted but never finalized

Unmodified everest-core 2025.12.1 · Raspberry Pi 4 · SIL EVSE modules

3 / 3

delay trials reproduced the integrity violation

Access required

CSMS + reachable MQTT/API

Not a CSMS-only exploit

DELAY	REMOTESTOP	OCPP ENDED	LOCAL FINISHED
30 ms	ACCEPTED	MISSING	MISSING
50 ms	ACCEPTED	MISSING	MISSING
70 ms	ACCEPTED	MISSING	MISSING

Accepted RemoteStop ≠ **TransactionEvent(Ended) / Transaction Finished**

The 11 findings span power, billing, authorization, and availability

Across the 11 findings, timing errors affect physical power, billing, authorization, and availability.

01

Unauthorized charging & billing integrity

V1–V2

Charging continues, or the transaction remains open after power-off.

02

Authorization bypass

V3

WithdrawAuth arrives before TransactionStarted.

03

Denial of service & state corruption

V4–V11

Crashes: V4, V5, V7, V11
Reliability risks: V6–V8
Internal state corruption: V9–V10

BOUNDARY CROSSED

protocol timing → hardware, billing, authorization, availability

Takeaways

Concurrency is part of the attack surface—not an implementation footnote.

01	**Concurrency flaws can affect physical power and billing integrity.**	V1–V3 demonstrate state-integrity failures without memory corruption.
02	**Per-module locks leave cross-module gaps.**	Eight of eleven flaws use inconsistent locking; event_queue is P/U.
03	**Scheduling must become a test input.**	Combine static candidate generation with schedule fuzzing across OCPP, MQTT/API, and EV/CP/ISO.

Scope: one open-source stack; V2 was validated with SIL modules, not production EVSE hardware.

ID	Component	Pattern	Impact	Verified	CVE
STATE-INTEGRITY VIOLATIONS					
V1	Charger	P2	Unauthorized charging continuation	L	CVE-2026-33015
V2	Charger	P3	Billing integrity violation	L	CVE-2026-33014
V3	AuthHandler	P3	Authorization bypass	L	CVE-2026-29044
SHARED-CONTAINER DATA RACES					
V4	OCPP 2.0.1	P1	Crash (DoS)	T	CVE-2026-26074
V5	OCPP 1.6	P1	Crash (DoS)	A, U	CVE-2026-26073
V6	OCPP 1.6	P1	Exception	A	CVE-2026-26072
V7	OCPP 2.0.1	P1	Memory corruption	A	CVE-2026-26071
V8	OCPP 2.0.1	P1	Exception	T	CVE-2026-26070
CROSS-THREAD RACES ON EXTERNAL INPUT					
V9	Charger	P1	State corruption	T, U	CVE-2026-33009
V10	Charger	P1	Undefined behavior	T, A, U	CVE-2026-27814
V11	AuthHandler	P1	Crash (DoS)	A	CVE-2026-27813

Pattern : P1 inconsistent locking · P2 lack of atomicity · P3 missing validation | Verified : T TSan · A ASan · U UBSan · L logic PoC