SPIRE + Regorus in OpenSOVD-CDA

Justin Harper (R&T)
Kenny Jarnagin (EBU Digital)
Tim Viola (Software Strategy)
Oliver Redgrove (Allison)
Naufal Shemshudeen (CCS Digital)
Carlton Bale (Eclipse Engagement)

Eclipse SDV Hackathon 2026 |
Project Overview

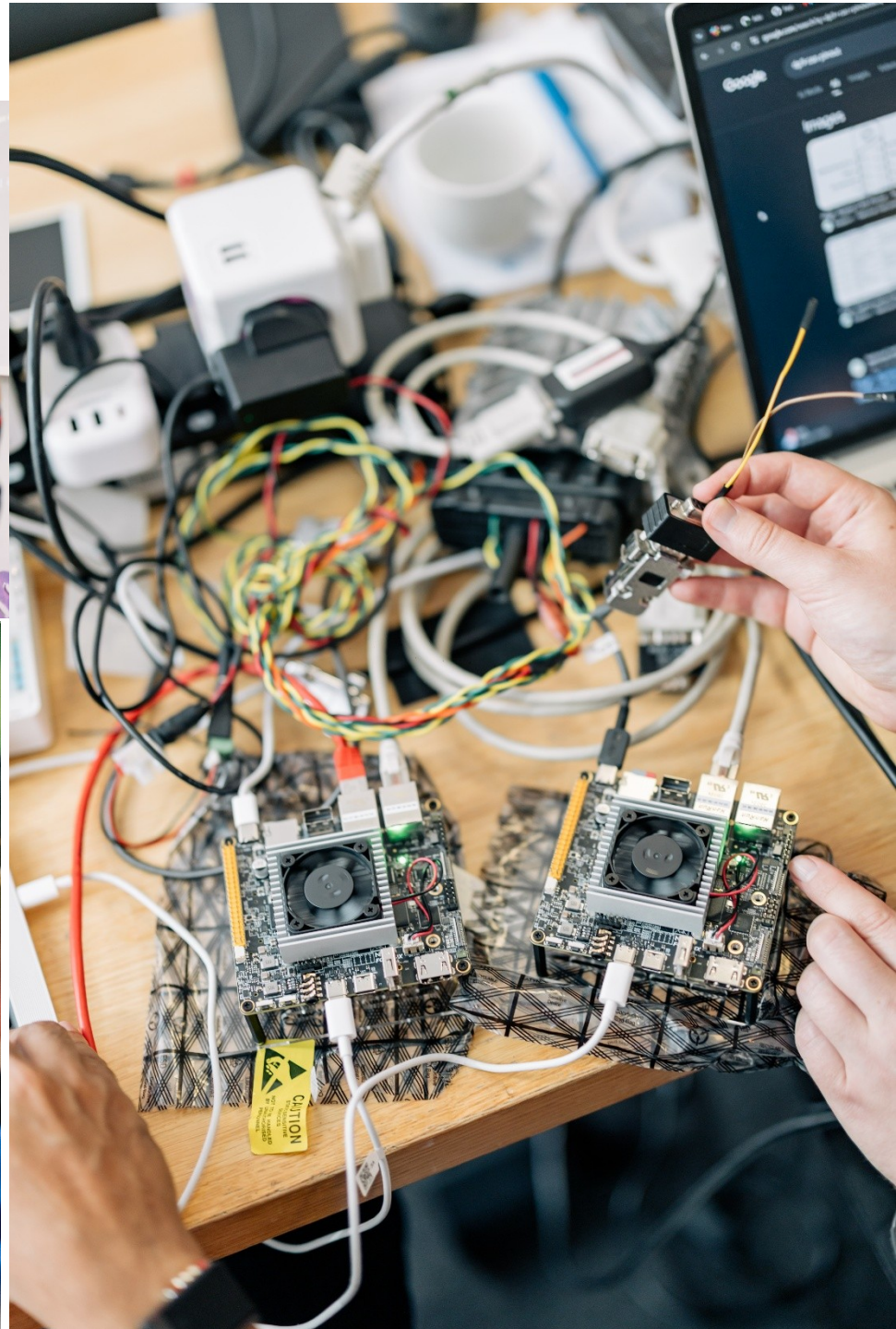
Team Name: Oxidized Powertrain Hack Force (OPHF)



Event Photos – Cummins Team







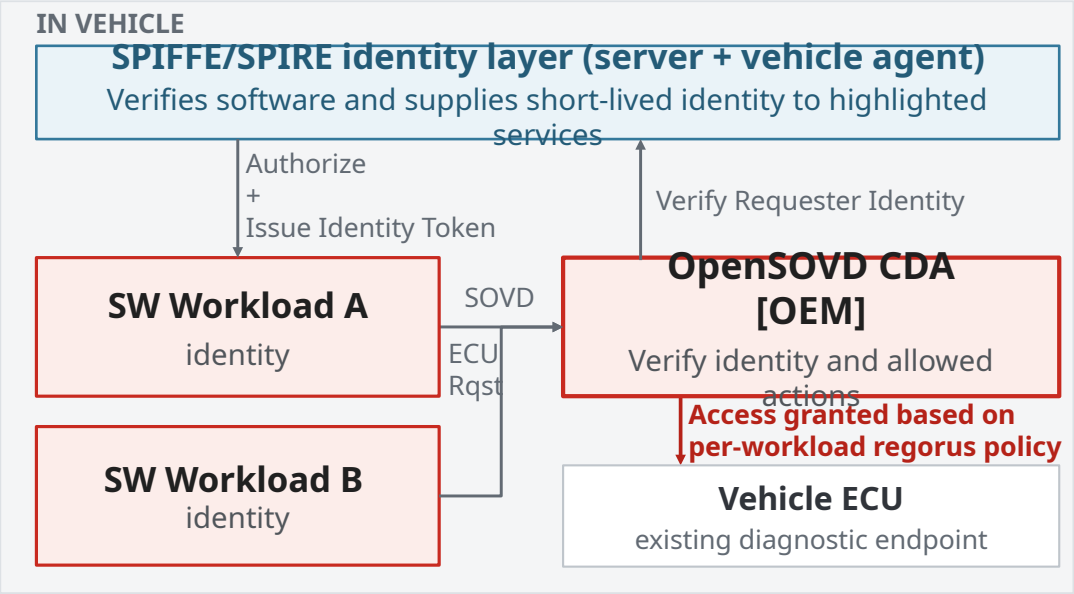


Cummins Challenge: Better In-Vehicle Security for SOVD CDA

In-Vehicle Authentication and Access Control using SPIRE+Regorus with OpenSOVD

Red: modified services
Blue: added identity layer

SIMPLIFIED COMMERCIAL SDV BLUEPRINT



CUSTOMER IMPACT

Authentication

[Is this software what it says it is?] SPIRE implicitly verifies running software is legitimate and issues workload identity [no long-lived secrets embedded in workloads like OAuth]

Authorization

[What is this software allowed to do?] Regorus checks permissions in a local policy for each workload identity before a command can proceed. [Flexible policy definition beyond what OAuth provides]

Connectivity Resilience

No round trip communication with remote systems (e.g. cloud), ensuring connectivity resilience. [No persistent cloud connection required like OAuth]

Improved in-vehicle authentication and authorization capabilities

A more flexible and manageable model of in-vehicle workload management for integrating software from mixed suppliers for OEMs

Cummins Internal Use Only

Why SPIFFE/SPIRE + Regorus instead of Oauth for workload authentication?

OAuth remains useful for delegated access and external APIs.
The in-vehicle gap is more basic: identify the running workload

OAuth-only service identity leaves vehicle questions

Who provisioned the bootstrap credential for each service?
How are many long-lived credentials managed when many services need individual identities?

- How is it rotated across vehicles?
- Does possessing the credential prove which workload is actually running?
- What happens when enterprise connectivity is unavailable?

SPIFFE/SPIRE + Regorus creates a local identity-and-policy plane

SPIRE Agent: implicitly attest the running workload from its runtime context (e.g. docker image signature), not from a long-lived static application secret embedded in the application
SPIFFE ID: assign stable workload identity such as spiffe://.../powertrain-mode-controller.
JWT-SVID: carry application identity through MQTT/uProtocol and local HTTP calls.
Regorus: evaluate explicit local policy: workload ID + requested operation = ALLOW or DENY.

Note: OAuth can use strong client authentication, including certificates. The point is not that OAuth is weak; it is that OAuth does not standardize workload attestation and per-workload credential issuance.

SPIRE creates the vehicle workload identity plane. Regorus turns that identity into explicit behavior policy.
OAuth stays whereever it is needed.

Cummins

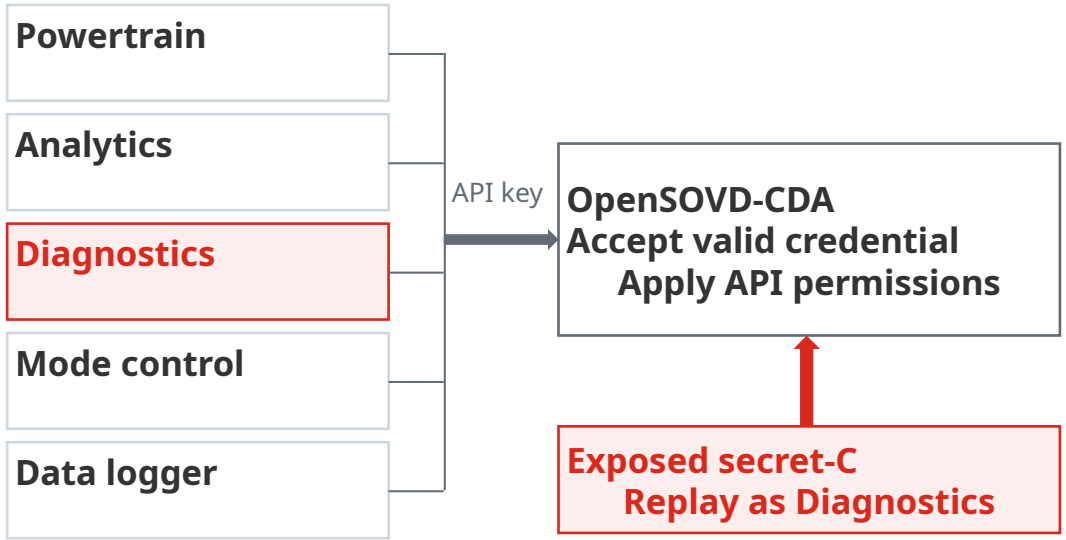
Cummins Internal Use Only

Embedded credentials can leak; Implicit identities can’t

Five workloads, one OpenSOVD-CDA: embedded secrets vs. runtime-attested identity

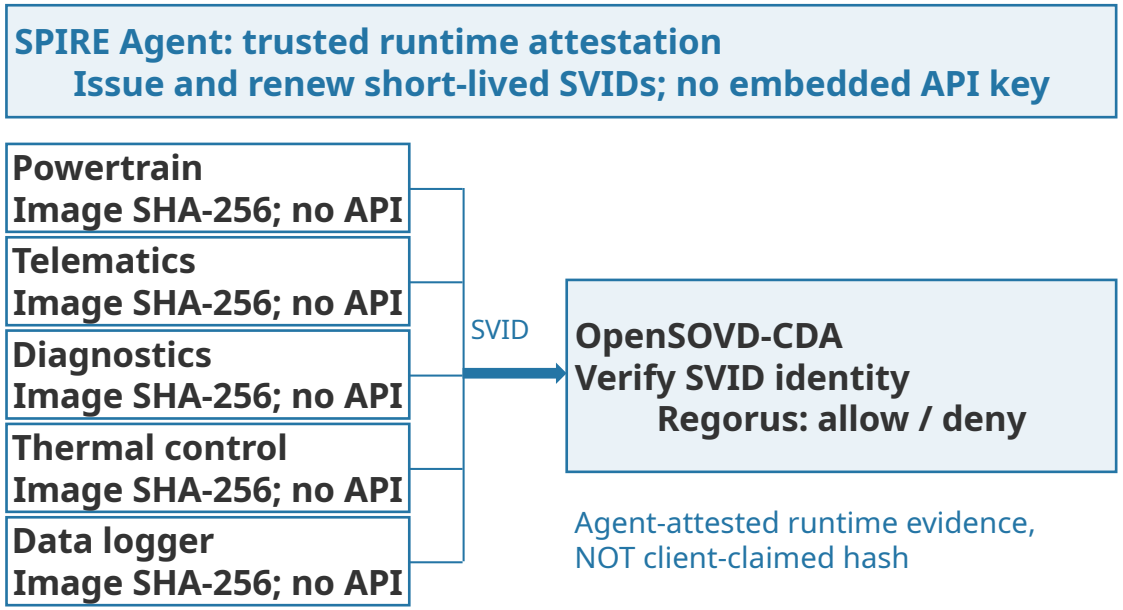
EMBEDDED API KEYS / CREDENTIALS

Each workload stores a reusable credential



Any one leaked key can be used to impersonate its workload.

SPIRE-ATTESTED WORKLOAD IDENTITY



Changed image → changed SHA-256 digest.

OEM manages trust and policy – avoid a large set of stored API secrets or other credentials

No manual per-workload secret tracking.

*Requires trusted agent/runtime and automated registration policy; a hash alone is not attestation.

Cummins Internal Use


Cummins