<epam>

Functional Safety for Xen Hypervisor

Artem Mygaiev, Xen Project / EPAM Systems

Cybersecurity Vehicle Forum, Munich 2026



EPAM Proprietary & Confidential.

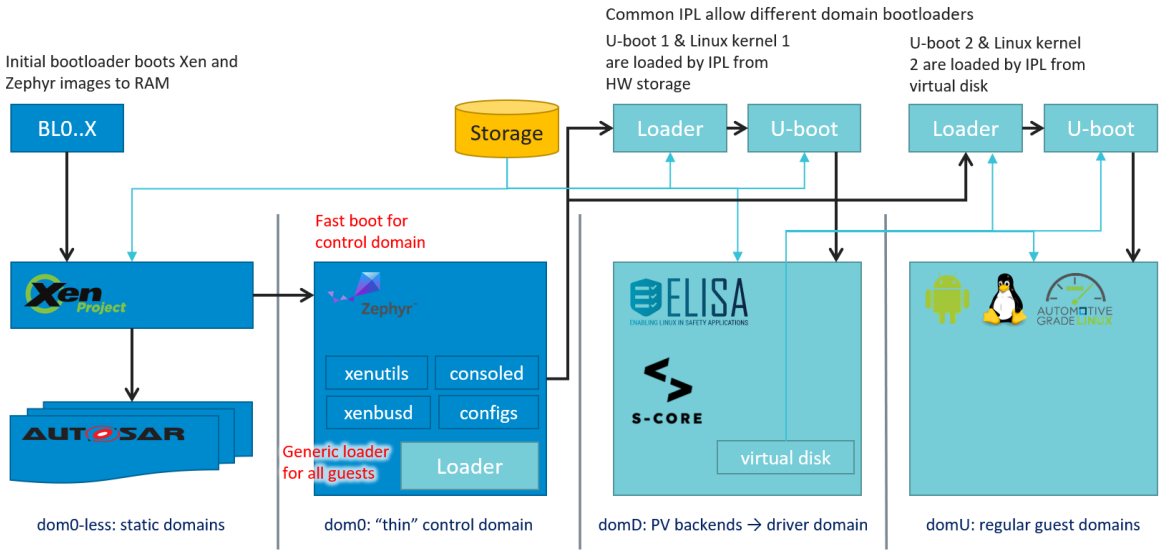
Why virtualization in automotive

Connected services, multimedia and AI increase software scale and integration complexity

Boundary	Architectural use	What it separates
OS containers	Modularity within a domain	Applications sharing an OS kernel
Virtual machines	Isolation between domains	Guest kernels, authority and resources
Hardware	Physical compute separation	Compute nodes with dedicated tasks

Isolation must cover shared CPU, memory, devices and trusted services

Mixed criticality on a shared compute



Connected services and infotainment

- Rich software and external interfaces
- Frequent software reconfiguration / updates

Safety-related vehicle functions

- Constrained authority and bounded response
- Validated failure and recovery behavior

Functional safety and FOSS

Safety lifecycle expectations	Open contribution model
Defined, competent owners	Contributors with varied backgrounds
Planned requirements and verification	Acceptance through technical review
Traceable, controlled changes	A codebase that keeps evolving

Common industry opinionated objections:

- Traditionally, Functional Safety require software to be developed from the start “with safety in mind”
- In FOSS community patches are usually accepted from anyone if they are good enough, participants are not trained or formally qualified
- Common pessimistic consensus in the industry that generic community-driven projects cannot bring products to functional safety compliance

Bridge the two with accountable roles and a maintained safety argument

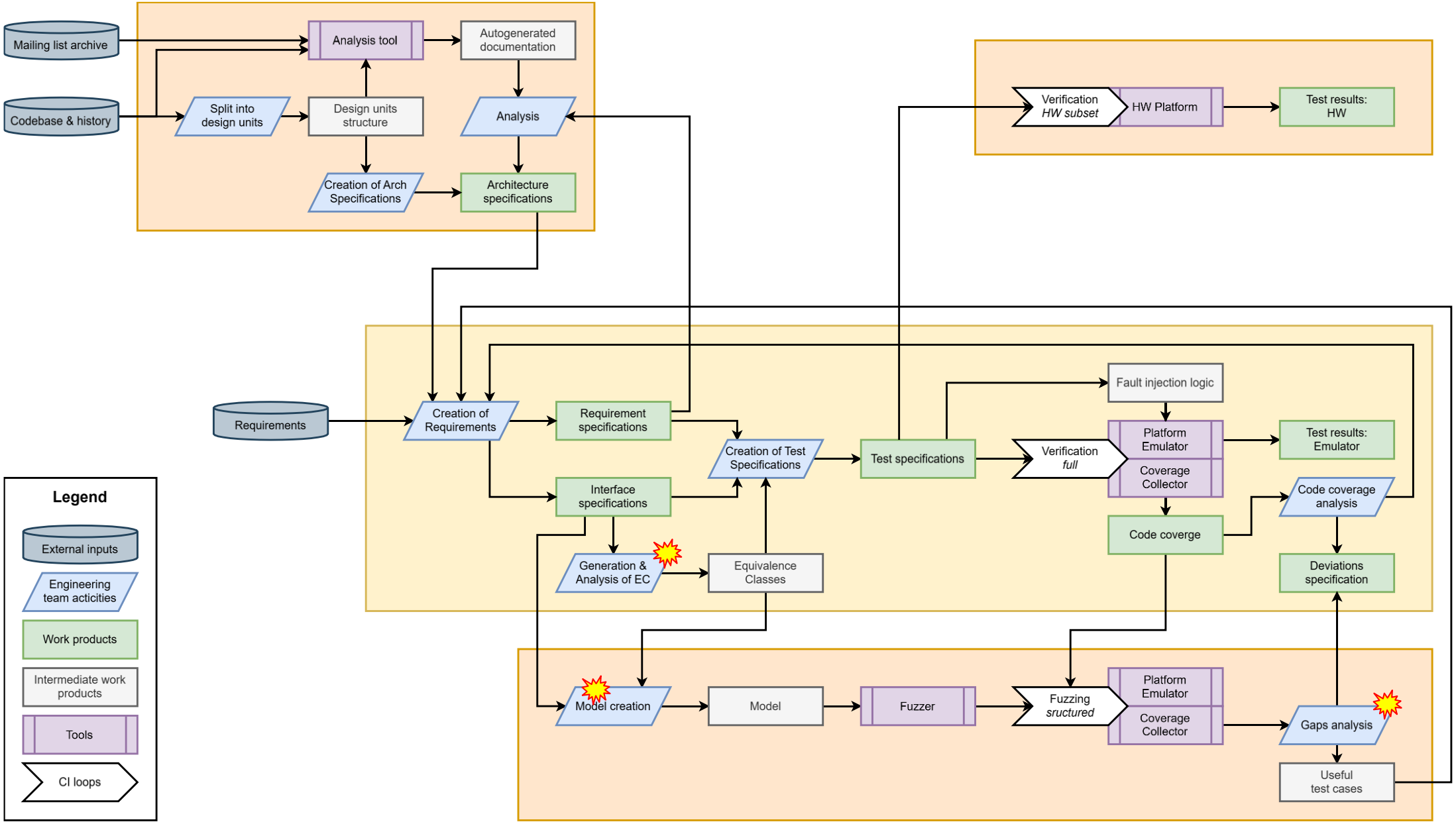
Security precedent: named ownership and coordinated vulnerability response

Building safety into upstream

Organize and train	**01**	FuSa SIG collaboration, maintainer training and MISRA C
Specify and analyze the baseline	**02**	Requirements, architecture, assumptions and DFA / DFMEA
Verify and close the gaps	**03**	Tool qualification, verification and MC/DC coverage
Maintain the common base	**04**	Shared engineering with program-specific goals and scope

Xen Project Launches Shared Functional Initiative for Safety-Critical Systems. Beyond maintaining source code, the committee is responsible for collaboratively developing and maintaining engineering artifacts intended to support organizations' downstream functional safety certification.

AMD epam RENESAS


Mailing list archive
Codebase & history
Analysis tool
Autogenerated documentation
Split into design units
Design units structure
Analysis
Creation of Arch Specifications
Architecture specifications
Verification HW subset
HW Platform
Test results: HW
Requirements
Creation of Requirements
Requirement specifications
Creation of Test Specifications
Test specifications
Verification full
Fault injection logic
Platform Emulator
Coverage Collector
Test results: Emulator
Interface specifications
Code coverage analysis
Code coverge
Generation & Analysis of EC
Equivalence Classes
Deviations specification
Model creation
Model
Fuzzer
Fuzzing sructured
Platform Emulator
Coverage Collector
Gaps analysis
Useful test cases
Legend
External inputs
Engineering team acticities
Work products
Intermediate work products
Tools
CI loops

Safety progress / defined baselines

Programs status reported at Xen Summit 2026

Program	Declared scope	Reported position
AMD	Minimal config ASIL D / SIL 3 Arm64-CR and AMD x86	Concept approved Final phase in progress
Renesas	Extended config ASIL B Arm64-CA	Concept approved Phase 1 in progress

Common, fully open codebase

Shared requirements and architecture specifications

Common tools – traceability, code analysis

FuSa processes on top of community

Shared work on verification approach and tooling

Shared work on missing features

Full alignment with community – no forks

SEooC: a reusable safety element with integration assumptions to validate

The product integrator validates hardware, firmware, guests and assumptions of use

Functional safety and cybersecurity

	Functional safety	Cybersecurity
Risk focus	Hazards from malfunctioning behavior	Threats involving compromise
Starting analysis	HARA, safety goals and ASIL	TARA and cybersecurity goals
Isolation role	Contain faults and interference	Constrain unauthorized actions
Evidence through change	Preserve the safety argument	Reassess threats and exposure

Both need explicit assumptions, verification and lifecycle ownership

Lifecycle cybersecurity assurance

ISO/SAE 21434:2021

Xen already has security engineering

Reporting and coordinated disclosure
CVEs, XSAs, patches and backports
Defined support boundaries

The next assurance objective

Vehicle-specific TARA and risk decisions
Requirements and validation evidence
Supplier and integrator responsibilities
Monitoring through the product lifecycle

Safety and cybersecurity engineering run together from the concept stage

But traditionally for the industry, safety gains more attention – this may change soon

Xen and trusted services assurance

Shared TEE services extend the assumptions a system integrator must justify

Xen integration responsibility

Guest identity and address translation

CPU and interrupt assignment

VM teardown and page ownership

Secure-service dependency

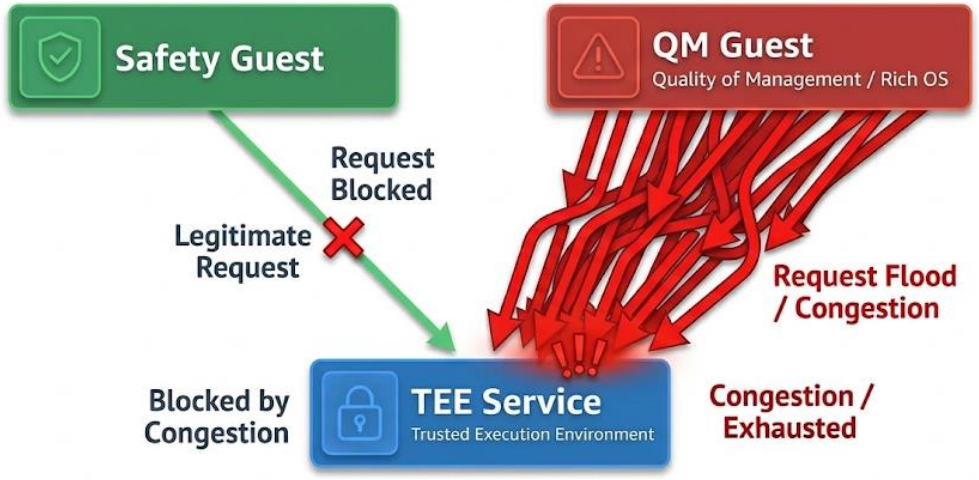
Caller attribution and access policy

Service blocking and shared-resource contention

Outstanding calls, device access and resource release

Example: a safety guest needs a TEE result while another guest floods the service

NOTE: OP-TEE and FF-A guest access are TECHNOLOGY PREVIEW in the Xen 4.22, but aiming for SUPPORTED



GlobalPlatform foundation

Existing GP mechanisms provide the starting point for deployment profiles.

Published foundation	Relevant coverage
TEE Architecture 1.3	Multiple TEEs, isolators, heterogeneous resources and trust
Core API 1.4	Caller/TEE path identities and remote-EE sessions
APIs, TMF and TEE PP	Memory, lifecycle, cancellation and secure failure
SESIP / automotive guidance	Assumptions, evaluation evidence and composition

The new work is a profile tying these existing provisions to particular mixed-criticality deployments and adding verifiable safety/timing semantics where justified

Safety support for mixed criticality

Two optional levels of support, selected for the system’s needs.

Coexistence	**Selected-service dependence**
Safety functions remain independent of TEE results	Selected TEE services contribute to a safety function
Bound interference and contain faults across shared dependencies	Also specify service correctness, completion and failure outcomes
Safety and non-safety workloads can share the platform	Mixed use of one TEE is possible where the guarantees hold

Security recovery must respect the system’s safety strategy

The profile states the supported configuration and the evidence needed for each claim

A focused proposal for TEE 2.0

Agree use cases	**01**	Coexistence and selected-service dependence
Derive product requirements	**02**	Trust, lifecycle, placement and bounded behavior
Map to existing GP clauses	**03**	Add normative bindings only where coverage is incomplete
Validate with reference platforms	**04**	Failure, overload and lifecycle scenarios with explicit evidence

Result: optional deployment profiles for safety-supported, mixed-criticality systems

Proposed joint work: GlobalPlatform, Xen, OP-TEE and automotive integrators

Thank you!

For more information, contact

Artem Mygaiev

Technology Solutions Director

artem_mygaiev@epam.com