Immutable Digital Twins for Built Heritage on a Permissioned Blockchain

DLT2026 – 8th Distributed Ledger Technology Workshop

Gavina Baralla · Marco Di Francesco · **Vadim Chilinciuc** · Roberto Tonelli · Gianni Fenu

netservice group
Guiding to Value

June 1–6, 2026

UNIVERSITÀ DEGLI STUDI DI CAGLIARI

Outline

Motivation and problem statement

Research Questions

System Architecture

Smart Contract Design

Implementation and evaluation

Discussion & Conclusions

The Problem

- Structural reports, BIM models, restoration records, IoT sensor streams — all must remain trustworthy for decades
- Traditional platforms rely on a central authority: audit trails can be modified or lost when systems change
- Institutions reorganize, software is decommissioned — historical records are at risk

Why Blockchain?

Append-only structure

Each record is cryptographically linked to its predecessors.

Tamper-evident

Retrospective modification is detectable without any trusted intermediary.

Decentralised

No central authority to be decommissioned or compromised over decades.

Three open gaps in existing blockchain-heritage architectures

MOTIVATION & PROBLEM STATEMENT

01

Asset Isolation

Existing proposals manage multiple assets through a shared contract; a flaw in one contract can compromise the entire collection.

02

Immutability Guarantees

Many architectures use upgradeable proxy patterns (UUPS, EIP-1967), allowing certification rules to change after deployment, weakening the evidentiary value of historical records.

03

GDPR Compliance

Heritage certification involves named professionals and institutional roles. Writing identifiers on an immutable ledger conflicts with the right to erasure.

Research Questions

THREE OPEN PROBLEMS TO SOLVE

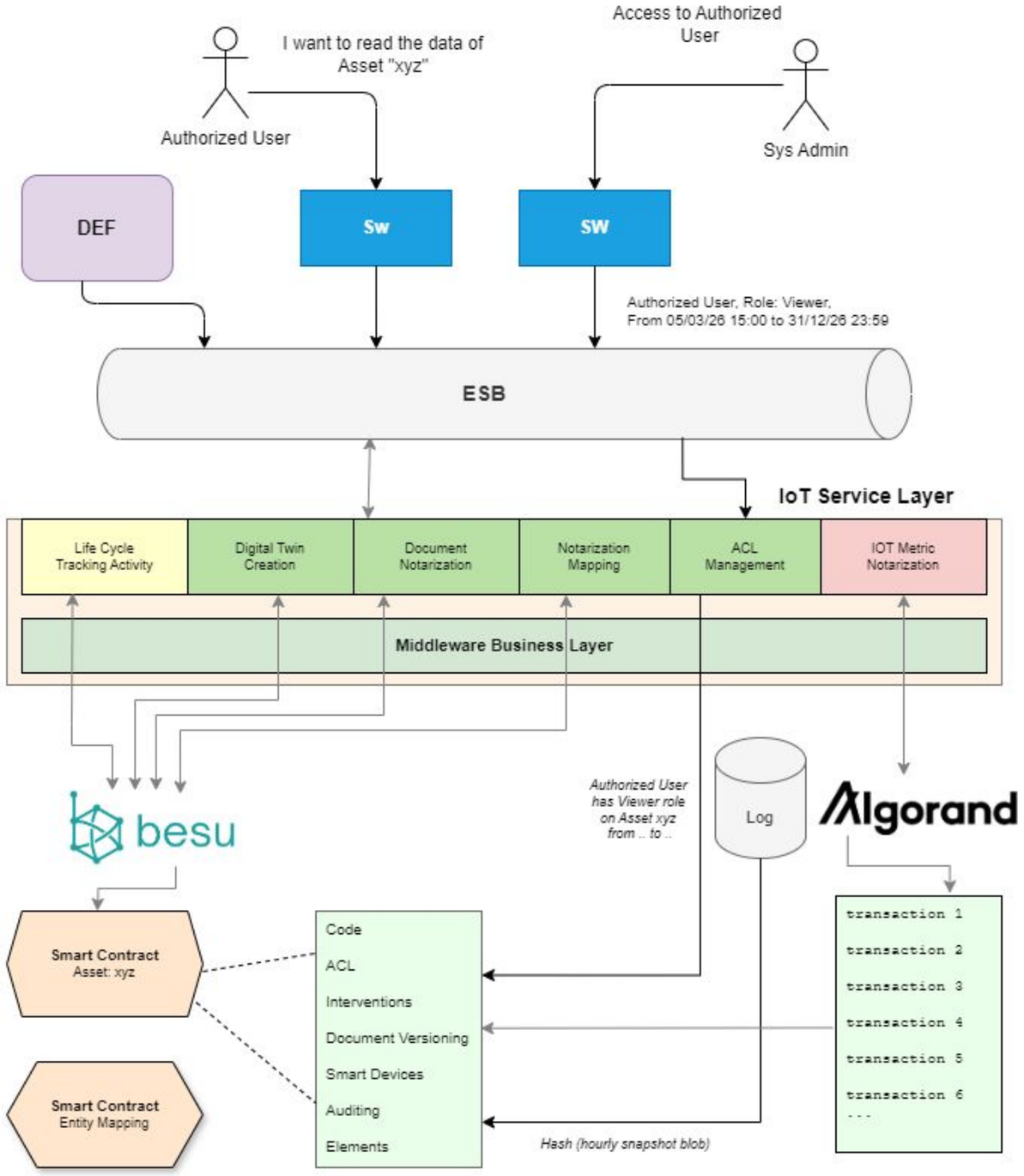
RQ1 Can per-asset contract isolation be achieved at scale without sacrificing the immutability of deployed logic?

RQ2 How can a blockchain architecture enforce strictly append-only, tamper-evident versioning across heterogeneous heritage artefacts?

RQ3 How can the system minimise on-chain exposure of personal identifiers while still supporting role-based certification?

System Architecture

FIVE-LAYER ARCHITECTURE



DEF (Heritage IS)

Manages BIM/IFC models, surveys, metadata. Generates per-asset cryptographic salts.

Enterprise Service Bus

Decouples internal systems from clients. Routes access control requests.

Middleware & REST API

Fastify + Ethers.js. Auth, custodial wallets, salted hash computation, certification.

Hyperledger Besu

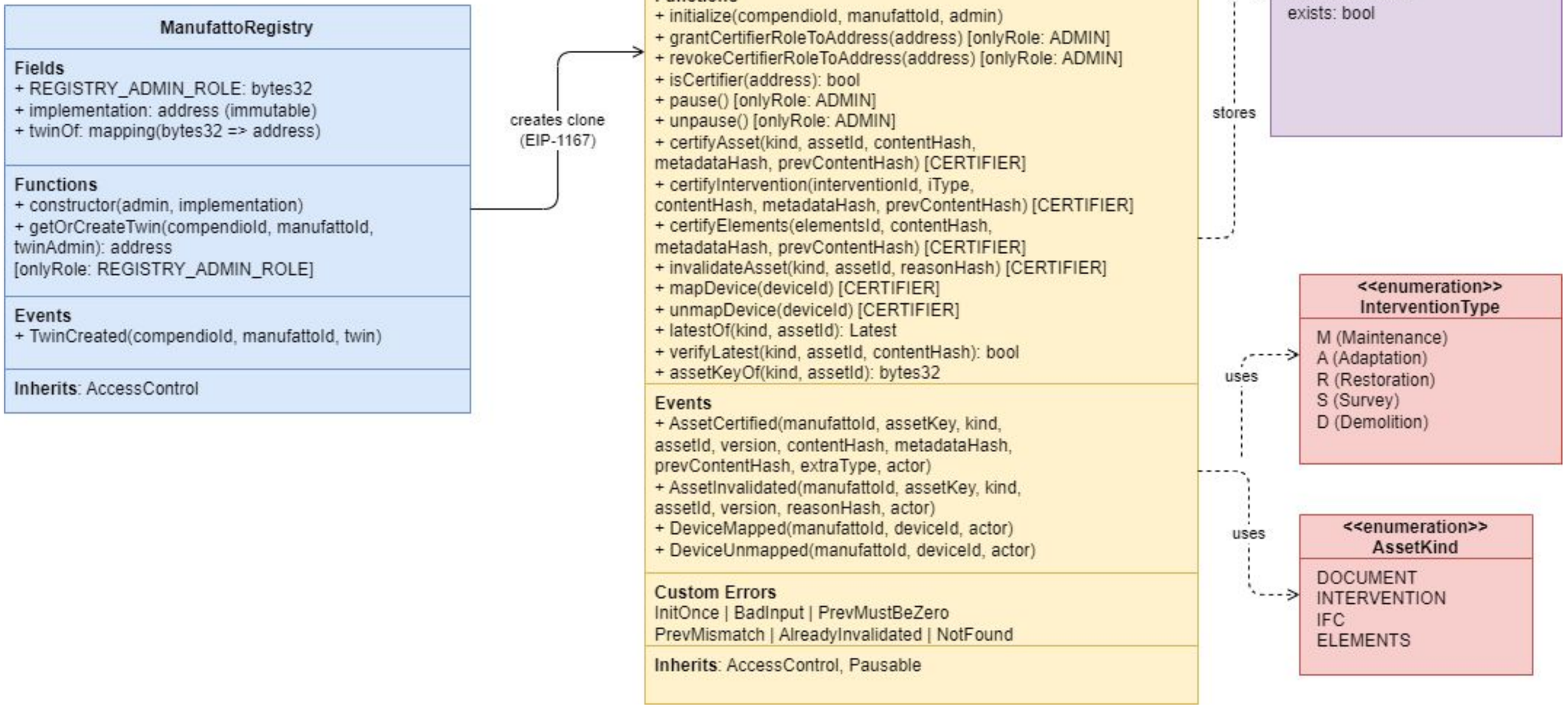
Permissioned network. Hosts smart contracts for the certification lifecycle.

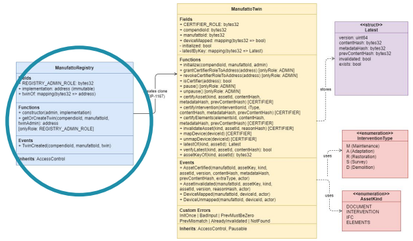
Algorand

Secondary chain for high-frequency IoT telemetry batch notarisation.

Two Contracts, One Responsibility Each

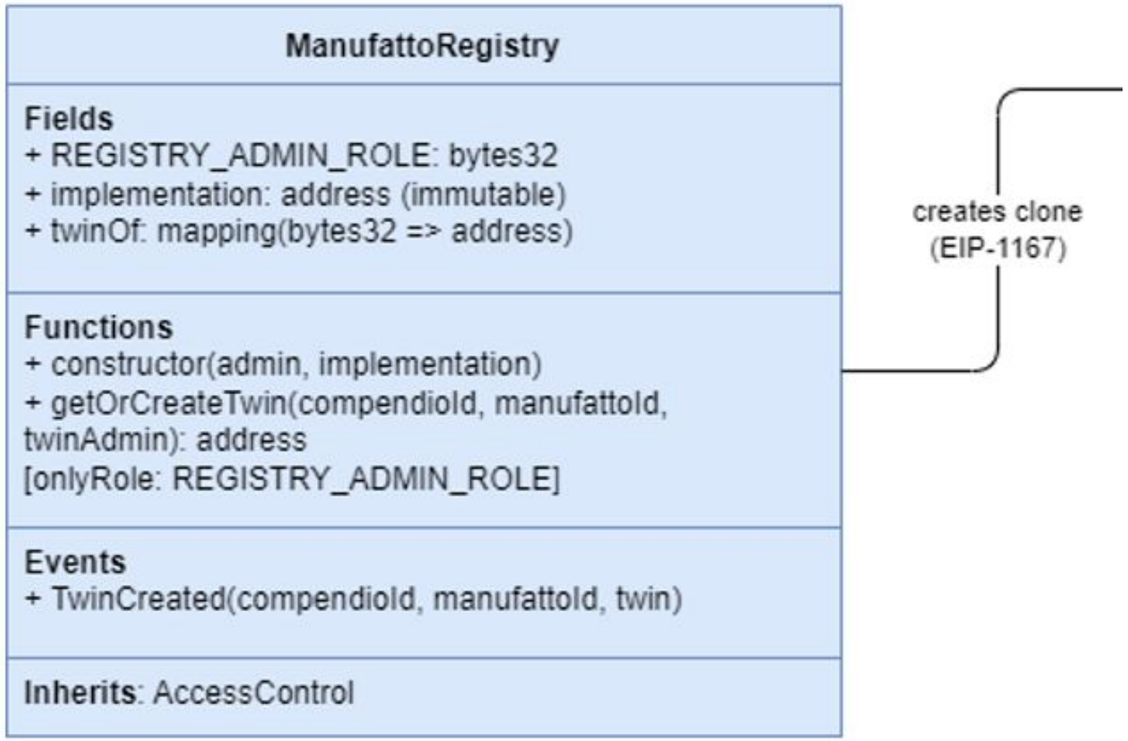
SMART CONTRACT DESIGN





Manufatto Registry Contract

SMART CONTRACT DESIGN

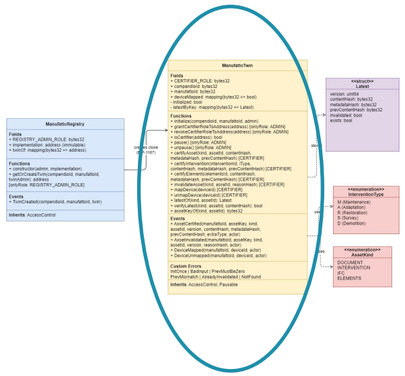


ManufattoRegistry

- Maintains mapping from asset identifiers to twin contract addresses.
- Deploys new EIP-1167 clones via getOrCreateTwin (protected by REGISTRY_ADMIN_ROLE, idempotent).
- Deploys 45-byte EIP-1167 clone per asset
- Implementation address stored as an immutable field — cannot be changed after deployment.

Manufatto Twin Contract

SMART CONTRACT DESIGN



ManufattoTwin

Fields
+ CERTIFIER_ROLE: bytes32
+ compendiumId: bytes32
+ manufattoId: bytes32
+ deviceMapped: mapping(bytes32 => bool)
- initialized: bool
- latestByKey: mapping(bytes32 => Latest)

Functions
+ initialize(compendiumId, manufattoId, admin)
+ grantCertifierRoleToAddress(address) [onlyRole: ADMIN]
+ revokeCertifierRoleToAddress(address) [onlyRole: ADMIN]
+ isCertifier(address): bool
+ pause() [onlyRole: ADMIN]
+ unpause() [onlyRole: ADMIN]
+ certifyAsset(kind, assetId, contentHash, metadataHash, prevContentHash) [CERTIFIER]
+ certifyIntervention(interventionId, iType, contentHash, metadataHash, prevContentHash) [CERTIFIER]
+ certifyElements(elementsId, contentHash, metadataHash, prevContentHash) [CERTIFIER]
+ invalidateAsset(kind, assetId, reasonHash) [CERTIFIER]
+ mapDevice(deviceId) [CERTIFIER]
+ unmapDevice(deviceId) [CERTIFIER]
+ latestOf(kind, assetId): Latest
+ verifyLatest(kind, assetId, contentHash): bool
+ assetKeyOf(kind, assetId): bytes32

Events
+ AssetCertified(manufattoId, assetKey, kind, assetId, version, contentHash, metadataHash, prevContentHash, extraType, actor)
+ AssetInvalidated(manufattoId, assetKey, kind, assetId, version, reasonHash, actor)
+ DeviceMapped(manufattoId, deviceId, actor)
+ DeviceUnmapped(manufattoId, deviceId, actor)

Custom Errors
InitOnce | BadInput | PrevMustBeZero
PrevMismatch | AlreadyInvalidated | NotFound

Inherits: AccessControl, Pausable

<<struct>>
Latest

version: uint64
contentHash: bytes32
metadataHash: bytes32
prevContentHash: bytes32
invalidated: bool
exists: bool

Manufatto Twin

- Each clone: 45 bytes of runtime bytecode, independent storage, fixed logic.
- Initialised exactly once (prevents re-initialisation attacks).
- Role management via OpenZeppelin AccessControl: DEFAULT_ADMIN_ROLE, CERTIFIER_ROLE.
- Supports pause/unpause for maintenance or incident response.

Storage Model

- latestByKey mapping
  Stores only the most recent state for each asset. This minimizes on-chain storage costs while maintaining instant access to current data.

- Versioned hash chain
  Links every state change to its predecessor. The full audit trail is preserved in block logs, reconstructable via events.

Asset Taxonomy (AssetKind)

- DOCUMENT · INTERVENTION · IFC · ELEMENTS
- InterventionType: M(aint.) · A(dapt.) · R(est.) · S(urvey) · D(emol.)

Composite Key & Latest Struct

- Each asset is identified by keccak256(manufattoId, kind, assetId).
- Latest stores: version, contentHash, metadataHash, prevContentHash, invalidated

Certification Functions

- certifyAsset · certifyIntervention · certifyElements — all call _certify which validates the hash chain before updating Latest

Append-Only Hash Chain

Version 1 — prevContentHash must be bytes32(0)

Version N — prevContentHash must match the latest certified contentHash

Mismatch → Transaction reverts with PrevMismatch at EVM level

Invalidated — invalidated flag set — no further versions can be certified

Result:
Any attempt to insert, overwrite, or skip a version is rejected at the EVM level.

Middleware Functional Areas

01 Auth & User Management

Custodial Ethereum wallet auto-assigned at registration. Keys managed by backend.

02 Pseudonymisation & Hashing

Salted hash computation on all identifiers + content before any on-chain write. No plaintext ever transmitted to chain.

03 Administrative Management

Twin contract creation, CERTIFIER_ROLE assignment, twin lifecycle management.

Privacy by Design

- **Pseudonymisation**

  H(identifier, salt) — on-chain observer cannot link record to real-world entity

- **Data Minimisation**

  Only salted hashes on-chain. No plaintext, no PII, no file content ever transmitted

- **Transient Permissions**

  Time-bounded access grants managed off-chain only — can be updated/deleted

- **Separation of Concerns**

  Durable CERTIFIER_ROLE on-chain vs transient access permissions off-chain

The Problem

High-frequency telemetry from structural health monitoring sensors.

The Solution

- Off-chain storage: Raw IoT data is stored in dedicated time-series databases.
- Aggregated proofs: Only cryptographic digests of data batches are written on-chain.
- Secondary chain: Optional use of Algorand for high-throughput notarisation.

Implementation

- Static mapping between Digital Twin and physical device IDs.
- Middleware handles batching and hashing of sensor streams.
- Verifiable link between certification records and physical monitoring.

Design Principle

Separate certification (slow/durable) from telemetry (fast/high-volume).

Gas Consumption Results

EVALUATION — DEPLOYMENT & GAS

508K

ManufattoRegistry
(0.8% block limit)

1.21M

ManufattoTwin impl.
(2% block limit)

181K avg

per asset clone
(min 27K idempotent)

~105–130K

per certification
transaction

Method	Min Gas	Max Gas	Avg Gas
getOrCreateTwin	27,556	185,538	181,268
certifyAsset	76,056	130,044	105,086
certifyElements / certifyIntervention	—	—	~130K
grantCertifierRoleToAddress	—	—	51,468
invalidateAsset	—	—	43,041
mapDevice / unmapDevice	—	—	55,268 / 33,330

Solidity 0.8.20 — Optimiser 200 runs - Zero gas price on permissioned Besu — figures represent computational cost only.

Contract	Statements	Branches	Functions	Lines
ManufattoRegistry	100%	83.33%	100%	100%
ManufattoTwin	100%	60.94%	100%	97.96%
Overall	100%	62.86%	100%	98.33%

Notes on Branch Coverage (62.86%)

- Single uncovered line: BadInput() revert guard in initialize() for zero-value inputs — prevented at application layer
- Remaining uncovered branches in inherited OpenZeppelin AccessControl and Pausable — not directly exercised by the test suite
- All branches from the contract's own logic (hash-chain validation, invalidation guards, role checks) are fully covered

The Inverted Trade-off

- Typical blockchain apps treat non-upgradeability as a constraint to mitigate via UUPS or Beacon proxies
- In heritage certification the trade-off is inverted: evidentiary value depends on the permanence of the rules that produced it, not only on data integrity
- Non-upgradeability provides this assurance by construction — at the cost of high pre-deployment verification burden
- Logic evolution requires new contract deployments — deliberate architectural choice, not an oversight

Limitations & Future Work

Key Management

Private keys in plaintext DB (prototype) — production requires HSM or dedicated KMS

Integration

DEF↔Middleware automated flow partially complete — full pipeline not yet implemented

Salt Governance

No formally enforced boundary for salt generation and storage across system components

Open Questions

Formal verification of hash-chain invariants; scalability to large heritage collections

RQ1

Per-asset isolation via EIP-1167 minimal proxy — each heritage asset has an independent, non-upgradeable contract at a fraction of full redeployment cost

RQ2

Strictly linear hash chain enforced at EVM level — insertion, overwriting or skipping of versions is rejected in the transaction itself

RQ3

All personal identifiers confined to the application layer, pseudonymised via salted hashing — no real-world identifier ever reaches the chain

Validated: 19-case test suite · 100% statement + function coverage · ~105–130K gas per certification tx

Thank You

Questions & Discussion

netservice group
Guiding to Value

vadim.chilinciuc@flosslab.com
marco.difrancesco@netservice.eu

UNIVERSITÀ DEGLI STUDI DI CAGLIARI

gavina.baralla@unica.it
roberto.tonelli@unica.it
gianni.fenu@unica.it