The Sovereign Jurisdiction Network

Raeez Lorgat

April 2026

Abstract. We specify the architecture of a decentralised network of proof-producing kernels — the zone nodes of the network — one per sovereign jurisdiction, connected by bilateral corridors that carry typed compliance state with pointwise composition semantics on the Applicable fragment and structured provenance on mixed-axis coordinates. The unifying purpose of the network is the *multi-jurisdictional clearing of contingent claims*: an entity that is harboured in several jurisdictions issues claims (equity, debt, sukuk, real-world-asset receivables, parametric insurance, intellectual-property royalties, structured derivatives, event-contingent contracts) whose admissibility composes across the harbour set, and the corridor protocol controls the depth of the reliance envelope under which such claims may be transferred, settled, and relied upon outside the issuance harbour.

The companion paper *The Multi-Harbored Institution* specifies the issuing entity object and the algebraic structure of composed compliance; the companion *Lex* paper specifies the dependently typed logic in which jurisdictional rules and the typed terms of each contingent claim are encoded; the companion *Intelligent Assets* paper specifies the typed-object surface and the universal-claims clearing layer on which claims issued under this network may be cleared, settled, priced, and bound to reliance. Each kernel is the sole writer to its own compliance-sensitive state, evaluates compliance against its jurisdiction's rules across a twenty-three-domain taxonomy, produces cryptographic proofs for every state change, and maintains an append-only audit trail. Corridors between kernels are parameterised by a re-evaluation domain set R, a partial domain-recognition map μ, and grade-recognition maps γ; we summarise (μ, γ) as a single recognition-depth parameter φ when the focus is the depth at which a receiving jurisdiction admits the issuing jurisdiction's evidence. The pair (R, φ) controls not only entity-level admissibility but the reliance envelope under which a contingent claim may be cleared, transferred, and settled across the corridor. The architecture has no network coordinator, no global ledger, and no central authority; cross-jurisdictional coordination proceeds through bilateral relationships alone. The construction sits outside the classical taxonomy of blockchain, federated data exchange, and message relay. We describe the write-path pipeline, the corridor protocol, the compliance passport, multi-harbor composition, delegated navigation of compliance surfaces, failure modes under a bounded-Byzantine adversary, and the formal verification obligations the construction incurs. We also describe a paired issuance-and-clearing interface in which programmable contingent claims originate under the authoritative kernel network and may clear on a universal-claims clearing layer through a bridge that preserves the compliance envelope and the corridor's recognition depth (R, φ). The institutional shorthand is exact: the multi-harboured institution mints and governs claims under this network; the universal-claims clearing layer clears, settles, prices, and binds reliance to them.

The current public evidence is delimited: the $N=1$ write-path property under four enumerated trust assumptions; the lattice laws of pointwise meet on the Applicable fragment; proof obligations for bilateral commit-safety under fail-stop, partition-heal terminal agreement, and accountable disagreement under bounded Byzantine behavior; and soundness obligations for passport verification. The artefact is a specification at the level of protocols, invariants, and proof obligations, not a claim that all target theorems are discharged.

1 Introduction

Institutions that operate across jurisdictions face a structural problem. Each jurisdiction maintains its own compliance requirements, regulators, audit expectations, and enforcement mechanisms. An entity incorporated in Singapore that opens a subsidiary in Dubai and conducts trade through a Pakistani special economic zone must satisfy three independent regulatory regimes simultaneously, with no standardised mechanism for one jurisdiction's compliance evaluation to inform another's.

The prevailing arrangement is manual reconciliation. Compliance officers in each jurisdiction independently evaluate the entity. Evidence gathered for one regulator is re-gathered for another. Legacy jurisdictional transfer restarts the evaluation from scratch. The switching cost is high enough that jurisdictions face muted competitive pressure on regulatory quality. The cause is structural: compliance evaluations are not portable, not composable, and not machine-verifiable.

The downstream consequence is that the financial instruments multi-jurisdictional entities issue inherit the same fragmentation. Standard valuation theory reduces every tradeable instrument to a probability-weighted discounted future cash flow—a *contingent claim*. Equity, debt, sukuk, real-world-asset receivables, parametric insurance, intellectual-property royalties, structured derivatives, and event-contingent contracts are all instances of the contingent-claim taxonomy; derivatives are claims constructed from claims (cf. Damodaran 2012; CFA Institute 2015; the companion paper *The Multi-Harbored Institution* §3 and §10). When an entity issuing a claim in jurisdiction A wishes that claim to be held, transferred, or settled by parties harboured in jurisdiction B, the absence of a portable, composable, machine-verifiable compliance state forces each receiving party to re-evaluate the issuer from scratch. The cost is paid in legal fees, settlement delay, and reduced liquidity; the structural cause is the same as for entity compliance.

This paper specifies a network architecture that addresses the structural problem at both the entity layer and the claim layer. The principal ideas are the following.

(1) **Proof-producing kernels.** Each jurisdiction deploys a kernel: a rule engine that is the sole writer to its own database, evaluates compliance across a fixed taxonomy of regulatory domains (twenty-three in the current working taxonomy, enumerated in Section 2.3), and produces cryptographic proofs for every state change. The kernel is sovereign; it answers to its jurisdiction's regulators alone.

(2) **Bilateral corridors.** Two kernels establish a corridor parameterised by (R, μ, γ), where R names the compliance domains the receiving kernel re-evaluates at the border, μ maps recognized source-jurisdiction domains to target-jurisdiction equivalents, and γ records any grade-recognition maps authorized by the corridor instrument. We summarise the recognition depth induced by (μ, γ) as φ and write the corridor pair (R, φ) when the focus is the depth at which the receiving jurisdiction admits the issuing jurisdiction's evidence into its own reliance class. Corridors are asymmetric and bilateral; each carries typed compliance state with pointwise composition on the Applicable fragment and structured provenance on mixed-axis coordinates. Crucially, (R, φ) controls not only entity-level admissibility but the *reliance envelope* under which a contingent claim issued by a multi-harboured entity in one jurisdiction may be cleared, transferred, and settled in another: shallow recognition

(large R, downgrading φ) yields a thin envelope and a thin reliance class; deep recognition (small R, identity-like φ) yields a thicker envelope and a richer reliance class. The corridor network bears the same relation to jurisdictional compliance as the Border Gateway Protocol (Rekhter, Li, and Hares 2006) bears to interdomain routing: a protocol for bilateral agreements between sovereign parties that produces a directed reachability graph without central coordination. Connectedness, when it occurs, is an adoption property of that graph rather than a theorem of the protocol.

(3) **Compliance passports.** Each entity accumulates a hash-chained, cryptographically signed record of every compliance evaluation. The passport travels with the entity and is verifiable by any receiving kernel without online availability of the issuing kernel, subject to ordinary trust in sovereign key provenance, accepted pack versions, revocation state, evaluator fidelity where foreign conclusions are consumed, and the cryptographic assumptions stated below.

(4) **Algebraic composition.** Each kernel produces a compliance tensor for an entity: a structured evaluation that maps each regulatory domain in the shared taxonomy to a pair consisting of a compliance grade on the three-chain NonCompliant $<$ Pending $<$ Compliant and an orthogonal applicability marker (`Applicable`, `NotApplicable`, or `Exempt`). When an entity operates in several jurisdictions simultaneously, composition on the Applicable fragment is the pointwise meet across all harbors; the mixed-axis cases where harbors disagree on applicability are preserved by a `MeetResult`-valued meet that carries the applicability provenance rather than collapsing it. On fixed applicable slices, the resulting structure admits a pointwise residual; operational planning uses the residual as an algebraic bound and a separate remediation operator to identify legally authorized improvement steps.

(5) **Delegated automated operation.** A delegated program authenticates through the kernel's standard write path, observes the multi-harbor compliance tensor, computes the remediation plan for the target slice, submits authorized operation envelopes for remediation actions, and observes the updated tensor. Concurrent operation across distinct entities reduces to independent invocations on disjoint hash chains; concurrent action on the same entity is serialised by the kernel's mutation firewall.

(6) **Paired issuance and universal-claims clearing.** Programmable contingent claims originate under the authoritative kernel network and may be admitted to a *universal-claims clearing layer* through a bridge that preserves the full compliance envelope and the corridor's recognition depth (R, φ). In the paired architecture studied here, that clearing layer is Moxie. The clearing layer consumes the composed Applicable-fragment tensor, the per-instrument envelope, and the corridor parameters as typed inputs; it provides matching, settlement, reliance binding, dispute, and recovery for any claim issued by an entity harboured in the network, parameterised per claim class by the authority, evidence, risk, dispute, and reporting structure that class requires. The protocol layer (this paper) carries the entity, the harbour set, the corridor, and the compliance passport; the clearing layer (the companion paper *Intelligent Assets*) carries the cleared claim's reliance envelope and execution.

In the resulting network, the compliance component of switching cost falls to the domains re-evaluated at the border plus the corridor's measured friction. Non-computational frictions remain. Within that

bound, multi-harbor institutions operate under a single, formally composed constraint surface, and the contingent claims they issue clear across jurisdictions through a substrate that consumes the same constraint surface as a typed input.

1.1 No network coordinator

The architecture has no network coordinator. No zone has authority over any other. There is no central network entity as a legal or technical object; only bilateral relationships exist between sovereign kernels. Each kernel is deployed and operated by its jurisdiction's zone operator, typically a government agency or licensed private operator authorised to run the jurisdiction's compliance infrastructure. Each corridor is negotiated bilaterally between two zone operators. The network topology emerges from these bilateral decisions; no central planning function intervenes. Were every corridor severed, each kernel would continue operating independently for its jurisdiction: it would lose portability and multi-harbor composition while retaining sovereignty and local compliance evaluation. The network adds connectivity; each kernel is self-sufficient.

1.2 Scope and limitations

This is a systems paper. It specifies an architecture, describes a construction, and reports on its implementation status. It does not present empirical evaluation of the network under production load. It does not claim to solve the political problem of regulatory coordination; it claims to reduce that problem to one of parameterisation (choosing R and μ per corridor), itself a bilateral negotiation between jurisdictions and not a technical constraint.

1.3 Terminology

The following terms are used throughout and defined here on first use:

- **Lex**: The typed logic for encoding jurisdictional compliance rules presented in the companion paper *Lex: A Logic for Jurisdictional Rules*. Lex combines defeasible reasoning, temporal stratification, authority-relative interpretation via tribunal modals, and typed discretion holes. Lex rules are what the compliance tensor evaluates.
- **Fiber**: An individual compliance rule evaluation against a specific statute. A fiber is the atomic unit of compliance assessment, one rule applied to one entity in one jurisdiction, producing a verdict. Fibers compose into the compliance tensor.
- **Pack**: A versioned, content-addressed bundle of jurisdictional rules. Distinct pack kinds bundle different rule categories (lawpacks for static legislation and regulation, regpacks for dynamic supervisory state, licensepacks for licensing rules, corridorpacks for bilateral corridor parameters, and so on). Con-

tent addressing means any change to a pack produces a new digest, making version drift detectable and enabling rule-asof evaluation.

- **Compliance tensor**: A function from the finite domain set to the tensor-value type. Each domain coordinate carries a compliance grade on the three-chain NonCompliant $<$ Pending $<$ Compliant together with an applicability marker drawn from {Applicable, NotApplicable, Exempt}. Applicability is a separate axis, not an intermediate point on the compliance chain. The full tensor-value structure is defined in the companion paper *The Algebra of Institutional Compliance*. One compliance tensor per (entity, jurisdiction, evaluation-time) triple.

1.4 Paper organization

Section 2 describes the kernel architecture. Section 3 specifies the write-path pipeline. Section 4 presents the compliance tensor and passport. Section 5 defines the corridor protocol, the corridor algebra, and the no-global-consensus claim under an explicit adversary model. Section 6 addresses multi-harbor composition with a worked example. Section 7 describes delegated automated operation on the compliance surface. Section 8 analyzes network topology and scaling. Section 9 discusses failure modes and security. Section 10 catalogs the formal verification obligations, distinguishing discharged from open. Section 11 states the open problems and scope. Section 12 positions the work against related systems. Section 13 discusses implications for jurisdictional competition. Section 14 describes the endgame.

2 The kernel

2.1 Architecture

Each economic zone deploys a single kernel. The kernel is a proof-producing rule engine + runtime engine — a proof-producing rule layer (proofs primary, verdicts derived) over a runtime execution engine that carries out the institutional work of the jurisdiction continuously, at machine speed, against the proved rules — compliant-by-default, legal-by-default, end-to-end, AI-native, at the scale of an economy. It is the sole authority for all compliance-sensitive state within its jurisdiction. It is not an operating system in the systems-kernel sense; it does not manage hardware resources or schedule processes. Following the established usage in the seL4-adjacent literature (Klein et al. 2009), we reserve the term "kernel" for components that hold a structural mediation property. Here that property is the $N = 1$ write-path discipline of Section 2.2: every state-changing operation in the jurisdiction passes through this single component, with no alternative write path to compliance-sensitive state. We say "kernel" in this restricted, structural sense, weaker than seL4's functional refinement but stronger than the conventional notion of a privileged service.

The kernel is a single process holding exclusive write credentials for the compliance-sensitive storage layer. Business-logic compute (entity formation, identity verification, treasury operations, cap-table management, governance workflows, document generation, credential issuance) runs outside the kernel as state-

less services. Those services read directly from the storage layer under read-only credentials and write exclusively through the kernel's internal write interface, authenticated with a shared internal token. Credential distribution, not documentation convention, enforces the $N = 1$ property.

2.2 The $N = 1$ structural property

The kernel's authority derives from a conjunctive structural property: it is the only process that can mutate compliance-sensitive state. We state the property precisely.

System model. Fix a single zone. Let P denote the set of processes running within the zone's operational perimeter and let $\text{kernel} \in P$ denote the distinguished kernel process. Let Σ denote the set of compliance-sensitive storage tables. Let $H : \{0,1\}^* \to \{0,1\}^{256}$ denote the content-addressing hash function, assumed collision-resistant against an adversary running in time polynomial in the security parameter. Signature schemes are assumed EUF-CMA.

Definition (kernel-mediated mutation). A mutation m on Σ is *kernel-mediated* iff

(K1) m's write credential is held by the kernel process and by no other process;

(K2) m passes through the `finalize_mutation` gate in the kernel's control flow;

(K3) m's effect on Σ uses only `INSERT` (no `UPDATE`, no `DELETE`);

(K4) m produces an event e linked into the per-entity hash chain by H.

Theorem 1 ($N = 1$ structural property). Assume the following trust assumptions hold:

(T1) *Credential isolation.* The credential-provisioning system issues storage-layer write credentials exclusively to the kernel process.

(T2) *Firewall coverage.* Every write route in the kernel's source tree is dominated, in its control-flow graph, by a call to `finalize_mutation`; the property is verified statically on every commit by the continuous integration pipeline.

(T3) *Schema-level DDL.* The storage schema exposes no `UPDATE` or `DELETE` privilege on Σ to any credential reachable from any process in P.

(T4) *Hash-chain integrity.* H is collision-resistant against the adversary.

Then every mutation of Σ observable by a corridor-participant verifier is kernel-mediated.

Proof sketch. Suppose, for contradiction, that some non-kernel-mediated mutation m of Σ is observable. By (K1) and (T1), no process other than kernel holds write credentials on Σ, so m must originate within kernel. By (T2) every kernel write route is dominated by `finalize_mutation`, so m satisfies (K2). By (T3) the schema admits only `INSERT` on Σ, so m satisfies (K3). Suppose m produces an event e that does not link into the per-entity hash chain. Either e collides with an event already in the chain, contradicting (T4); or e falls outside the chain, in which case any verifier holding the chain head detects m's absence and

refuses to attribute it. Hence m satisfies (K4) or is unobservable. In either case m is kernel-mediated, a contradiction. □

Remarks on the trust boundary. Theorem 1 is a *conditional* structural guarantee: each of (T1) through (T4) is a distinct trust assumption, and compromise of any one defeats the corresponding clause of kernel-mediation. (T1) fails under compromise of the credential-provisioning system; (T2) under supply-chain or CI compromise; (T3) under schema-level `ALTER` compromise by a privileged operator; (T4) under a collision-finding attack on H. These assumptions delineate the boundary of the guarantee.

The bar is weaker than full functional refinement against a formal model. Klein et al. (2009) prove that no execution of the seL4 C code reaches a state violating its abstract specification; Theorem 1 is a process-level invariant under a conjunction of operational assumptions, not a code-level invariant against arbitrary binary behaviour. The comparison to seL4 is orientational: our claim is a named structural invariant with enumerated trust assumptions, mechanised at the levels at which the architecture incurs obligations, strictly weaker than seL4-style refinement.

Mechanisms enforcing (T1) through (T4). Three complementary mechanisms carry the property:

(1) *Credential isolation.* Only the kernel process holds storage-layer connection credentials with write privileges. Business-logic services hold read-only credentials.

(2) *Mutation firewall.* Every state-changing handler in the kernel passes through a `finalize_mutation` gate that enforces audit journaling, sanctions fail-closed evaluation, and contract validation. A write handler that bypasses the gate is a defect. Continuous-integration gates check gate presence on every mutating route.

(3) *Append-only event store.* Every mutation produces a kernel event whose identifier is the content-addressed hash of event type, payload, and prior-event identifier. Events are linked per-entity into a hash chain following the Haber and Stornetta (1991) timestamping construction. The event store accepts only appends; `UPDATE` and `DELETE` are forbidden at the storage layer. The content-addressing hash H is SHA-256 throughout.

2.3 Compliance evaluation

The kernel evaluates compliance across a fixed, finite taxonomy of regulatory domains, shared across the kernels that participate in the corridor network. The current taxonomy is:

Domain	Scope
AML	Transaction monitoring, suspicious activity reporting
KYC	Identity verification, due diligence
Sanctions	OFAC, UN, EU sanctions list screening
Tax	Withholding, reporting, filing obligations
Securities	Issuance, trading, disclosure requirements
Corporate	Formation, dissolution, beneficial ownership
Custody	Asset safekeeping, segregation requirements
DataPrivacy	GDPR, PDPA, cross-border data transfer
Licensing	Business license validity, professional certifications
Banking	Reserve requirements, capital adequacy
Payments	Payment-service-provider licensing, instrument rules
Clearing	Central counterparty rules, netting, finality
Settlement	Delivery-versus-payment, settlement cycles
DigitalAssets	Instrument classification, exchange licensing
Employment	Labor contracts, social security, withholding
Immigration	Work permits, visa sponsorship, residency
IP	Patent, trademark, trade secret
ConsumerProtection	Disclosure, dispute resolution, warranties
Arbitration	Dispute resolution frameworks, enforcement
Trade	Import/export controls, customs, tariffs
Insurance	Solvency requirements, reinsurance
AntiBribery	FCPA, UK Bribery Act, UNCAC compliance
Sharia	Islamic-finance compliance

The taxonomy is pragmatic, not principled: it covers the regulatory surface area of institutional operations across the jurisdictions the protocol is designed to serve. The set is finite and fixed at composition time; extension follows a schema-evolution protocol (Section 8). The Sharia domain enters the meet on the same terms as any other: a conventional harbor reports `NotApplicable`, an Islamic harbor reports a substantive verdict, and the composition takes the meet of the two.

Each domain is evaluated through fibers: individual compliance rule evaluations against specific statutes. A fiber evaluates one Lex rule against one entity's state and produces a verdict. Fibers compose into the compliance tensor, where each domain's verdict is the meet of all fiber verdicts within that domain.

Each domain evaluates to a pair consisting of a *compliance grade* on a three-chain and a separate *applicability marker*.

The compliance grade is drawn from a three-chain:

- **Compliant.** The entity satisfies all requirements in this domain.
- **Pending.** Evaluation requires information not yet available.

- **NonCompliant.** The entity fails one or more requirements.

The ordering is `NonCompliant` $<$ `Pending` $<$ `Compliant`, with lower meaning more restrictive.

The applicability marker is drawn from {`Applicable`, `NotApplicable`, `Exempt`} and records whether the domain governs the entity in the jurisdiction at all: `Applicable` when the domain's rules apply and a grade is substantive, `NotApplicable` when the domain does not govern the entity's activity in that jurisdiction, and `Exempt` when the entity holds a signed policy artefact that removes the domain from scope. Applicability is a *separate axis*, not an intermediate point on the compliance chain.

Composition on the *Applicable fragment*, the sub-lattice where every coordinate is `Applicable`, is pointwise meet on the three-chain and yields the most restrictive grade across the jurisdictions in which the domain is substantively governed. The full tensor-value type, which carries the mixed-axis applicability information, is strictly richer. The compliance-tensor companion note states the mixed-axis impossibility result: no total Heyting algebra structure exists on the full tensor-value type that preserves the applicability provenance required by the audit trail. The production meet is therefore `MeetResult`-valued: on coordinates where both inputs are `Applicable`, it returns the three-chain meet; on mixed-axis coordinates, it returns a structured outcome that names which jurisdiction contributed which applicability status.

2.4 Lifecycle typestates

Lifecycle state machines in the kernel may be typestate-encoded: each lifecycle state is represented by a distinct type and only valid transitions are expressible. In host languages supporting phantom types, session types, or linear types, invalid transitions become static errors rather than runtime checks; in other host languages the same discipline reduces to runtime verification. The specification is language-agnostic; the enforcement level is implementation-dependent. The discipline applies uniformly to the lifecycles of entities (formation through dissolution), licences (application through grant and renewal), corridors (draft through active, with halted and suspended branches), migrations (the phased corridor-crossing protocol), and watchers (bonding through slashing or unbonding).

3 The write-path pipeline

3.1 Principal write path: full compliance

Every state-changing operation passes through a fixed pipeline. The principal path (end-user-originated writes) enforces the full compliance evaluation:

Step 1: Authentication. The caller presents a credential. The kernel extracts a uniform *caller identity* carrying role, entity scope, and delegated-program attribution. The attribution is not self-declared: it is bound at credential issuance time by the credential-provisioning system, making it cryptographically fixed

rather than advisory. Authentication uses constant-time comparison for bearer tokens. The same identity structure applies uniformly to human operators, API consumers, and delegated programs.

Step 2: Sanctions fail-closed gate. Before any compliance evaluation, the kernel screens the entity and its counterparties against sanctions lists. A license, humanitarian carveout, sovereign exemption, or shared-authority recognition is encoded before this point as rule-scope or applicability evidence inside the Sanctions evaluator. If the applicable Sanctions coordinate still returns `NonCompliant`, the operation is rejected. This is a legal requirement under the International Emergency Economic Powers Act (50 U.S.C. §1705) and its counterparts in other sovereign sanctions regimes.

Step 3: Pre-flight compliance evaluation. The kernel builds a compliance tensor for the entity in the target jurisdiction, evaluating every applicable domain via pluggable evaluators. Each evaluator invokes fibers and aggregates their results. The tensor $T(E, J) : \mathcal{D} \to \text{ComplianceState}$ maps each domain to one of the five states. Non-sanctions `NonCompliant` results surface as warnings without blocking the operation, permitting entities to be formed and then brought into compliance iteratively.

Step 4: Business-logic delegation. The primitive operation (entity formation, share issuance, payment instruction) proceeds through the business-logic service layer, invoked via a typed client that is the sole authorized gateway between the kernel and the outside.

Step 5: Operational evidence storage. The fact that the operation succeeded is itself domain evidence. The kernel maps operation types to evidence domains: entity formation produces Corporate evidence, captable changes produce Securities and Corporate evidence, identity verification produces KYC, AML, and DataPrivacy evidence. The mapping is fail-closed: unverified evidence produces no domain credit.

Step 6: Post-flight re-evaluation. The kernel re-evaluates the compliance tensor, this time incorporating the operational evidence from Step 5. Domains that were `Pending` in pre-flight may now have substantive verdicts. The two-phase evaluation design separates the sanctions gate and baseline from the evidence the operation produced.

Step 7: Verifiable-credential issuance. The kernel issues a signed W3C Verifiable Credential attesting to the post-flight compliance evaluation. The credential captures the tensor state, the evaluation timestamp, the issuing zone's decentralized identifier, and the tensor commitment digest. The credential is designed for interoperability with eIDAS cross-border trust services.

Step 8: Compliance passport update. The entity's compliance passport, a hash-chained record of all compliance evaluations, is extended with the new evaluation. The passport records the previous digest (as a hash-chain link), the current domain verdicts, the tensor commitment, and the verifiable-credential digest. The passport update is itself signed by the zone.

Step 9: Mutation journaling. The mutation is recorded in the append-only event store. The journal entry is content-addressed, hash-chained per entity, and carries the full observation context produced by the compliance evaluation.

3.2 Reduced path: internal service writes

Business-logic services write through an internal interface that enforces a reduced pipeline: authentication, sanctions preflight under the same fail-closed verdict semantics as the principal path, the persistence write within a single storage transaction, and mutation journaling. Internal writes do not run the full tensor evaluation, verifiable-credential issuance, or passport update. The design decision is that compliance evaluation happens once, at the point where user intent enters the kernel (through the operation engine); internal writes persist already-evaluated results.

This reduced path creates a structural obligation: every internal write must be attached, by the caller's typed client, to an evaluation context whose intent was covered on the principal path, or must pass full-tensor evaluation on its own terms. The attachment is carried on the request envelope as a cryptographic reference to the principal-path evaluation that authorised it. A write that carries neither a valid evaluation reference nor a full-tensor evaluation is rejected by the internal write interface as a scope defect. The discipline preserves the $N = 1$ property under the internal-write surface; a compromise of this discipline at deployment time reduces (T2) of Section 2.2 to a conditional guarantee over the reduced pipeline only.

3.3 Generic tabular writes

For persistence-layer writes that require no domain-specific validation, the kernel provides a generic tabular-write interface over a static allowlist of table, field, type, and nullability constraints. Writes outside the allowlist are rejected. Compound atomicity is provided by wrapping the writes in a single transaction. Idempotency keys deduplicate resubmission. The interface reduces the marginal cost of a new write operation to a configuration change, not a code change, while preserving the invariant that all writes funnel through audited paths.

3.4 The proof bundle

Every write through the kernel produces a proof bundle:

(1) **Mutation journal entry.** Content-addressed, hash-chained, append-only. Carries the operation type, resource identifiers, acting principal, timestamp, and compliance observation context.

(2) **Compliance observation context.** Domain outcomes (per-domain verdicts from fiber evaluations), evaluation duration, evidence types, and the tensor commitment digest. This is the raw material for the intelligence corpus: every compliance evaluation contributes to the system's understanding of jurisdictional requirements.

(3) **Verifiable credential.** Signed attestation of the post-flight compliance tensor in the W3C Verifiable Credentials Data Model. The subject captures entity, jurisdiction, overall status, per-domain results, tensor commitment, and hash-chain link.

(4) **Compliance passport update.** The entity's hash-chained compliance history, extended with the new evaluation. The passport is a sequence of snapshots, each linked to its predecessor by a content-addressed digest.

The proof bundle is the fundamental unit of portable compliance. When an entity crosses a corridor to a new jurisdiction, it carries its passport: a verifiable, tamper-evident history of every compliance evaluation the issuing jurisdiction performed.

4 The compliance tensor and passport

4.1 Tensor definition

A compliance tensor $T(E, J)$ for entity E in jurisdiction J is a function from the domain set $\mathcal{D}$ to a tensor-value type carrying a *compliance grade* on a three-chain and a separate *applicability marker*:

$$T(E, J) : \mathcal{D} \to \{\text{NonCompliant}, \text{Pending}, \text{Compliant}\} \times \{\text{Applicable}, \text{NotApplicable}, \text{Exempt}\}.$$

Applicability is separate from the grade: an `Exempt` coordinate is not a maximally-compliant grade in disguise, and a `NotApplicable` coordinate is not a middling-compliant grade in disguise; both carry provenance that the audit trail must preserve. The tensor is parameterised by a jurisdiction configuration specifying (i) the set of domains applicable in this jurisdiction, usually a proper subset or the full set $\mathcal{D}$, and (ii) for a synthetic zone composed from several jurisdictions, which source jurisdiction governs each domain. For a natural zone (single jurisdiction), all applicable domains are governed by that jurisdiction. For a synthetic zone, different domains may inherit from different source jurisdictions (Corporate from one, Tax from another, Securities from a third), with the per-domain attribution recorded in the configuration.

Each domain has a pluggable evaluator. The evaluator invokes fibers (individual Lex rule evaluations against specific statutes) and aggregates their verdicts via the fiber composition described in the companion Lex paper. The tensor supports both explicit state storage (pre-computed results loaded from evidence) and dynamic evaluation (evaluators invoked at tensor-evaluation time). There is one evaluator per domain.

4.2 Lattice structure

The compliance grade forms a bounded three-chain under the ordering NonCompliant $<$ Pending $<$ Compliant with lower meaning more restrictive. The meet on this chain is the minimum and yields the most restrictive grade. The three-chain is a finite distributive lattice and therefore a bounded distributive Heyting algebra in the per-domain factor; this is Qed-closed in `lex/formal/coq/Lex/TensorAlignment.v` as the theorem `grade_is_bounded_distributive_heyting`, which bundles thirteen lattice laws (idempotence, commutativity, and associativity of meet and join; absorption in both directions; distributivity in both directions; bounds with `grade_top` and `grade_bot`; and the Heyting identity $a \wedge (a \to b) =$

$a \wedge b$). The residuation statement $c \leq a \to b$ iff $a \wedge c \leq b$ is closed as the companion theorem `grade_residuation`.

The applicability marker {Applicable, NotApplicable, Exempt} is an orthogonal axis, not an intermediate point on the compliance chain: `NotApplicable` and `Exempt` are distinct applicability states with distinct provenance obligations, and collapsing them into a single total chain would destroy the audit information that distinguishes "not governed" from "substantively governed and cleared."

Composition on the *Applicable fragment*, the sub-lattice where every coordinate is `Applicable`, is pointwise meet on the three-chain. The full tensor-value type, which carries the mixed-axis applicability information, is strictly richer. The compliance-tensor companion note states the mixed-axis impossibility result: no total Heyting algebra structure exists on the full tensor-value type that preserves the applicability provenance. The production meet is therefore `MeetResult`-valued: on coordinates where both inputs are `Applicable`, it returns the three-chain meet; on mixed-axis coordinates, it returns a structured outcome that names which jurisdiction contributed which applicability status. This scope is the canonical compliance algebra: per-domain three-chain structure, Applicable-fragment meet-semilattice with pointwise residual on fixed applicability slices, and a `MeetResult`-valued meet on the full type.

The lattice structure underwrites multi-harbor composition (Section 6). The meet of two tensors is the pointwise `MeetResult`-valued meet across all domains, producing the most restrictive constraint surface consistent with the applicability provenance from both jurisdictions.

On fixed applicability slices, the structure also supports a residual operation. Given tensors T_{current} and T_{target}, the residual $T_{\text{current}} \Rightarrow T_{\text{target}}$ computes, per domain, the weakest sufficient grade that would make the implication valid inside the slice. The residual is an algebraic bound, not a legal remediation plan; a separate authorized remediation layer determines which evidence, filings, or discretion fills can realize the bound.

The lattice laws (associativity, commutativity, and idempotence of meet on the Applicable fragment; the Galois connection for the Heyting residual on fixed slices) are presented formally in the companion paper *The Algebra of Institutional Compliance*. The lattice algebra is load-bearing: composition, planning, and selective-disclosure operations derive from it.

The companion Lex formalisation at `lex/formal/coq/Lex/VerdictHeyting.v` mechanises the per-(entity, jurisdiction, domain) five-element verdict chain (`NonCompliant` $<$ `Pending` $<$ `NotApplicable` $<$ `Exempt` $<$ `Compliant`) as a bounded distributive Heyting algebra: thirteen lattice and Heyting laws closed by `Qed`, bundled into `verdict_is_heyting`. The `lex/formal/coq/Lex/TensorAlignment.v` file witnesses the relationship between that per-coordinate chain and the tensor-factor type described in this section. `lex_to_tensor` is the injective projection from the five-element verdict to the tensor-factor (`TF_App` g for grades on the three-chain, or `TF_NotApp` / `TF_Exempt` for the applicability singletons). `lex_to_tensor_injective`, `lex_meet_preserves_applicable_fragment`, and `lex_tensor_diagram_commutes_on_applicable` are Qed-closed and establish that on the Applicable fragment the two meets commute. `lex_meet_loses_provenance_on_mi` and `lex_meet_loses_provenance_exempt_vs_pending` are Qed-closed concrete counterwitnesses exhibiting pairs of tensor-factors that a chain-based total meet collapses and that the `MeetResult`-valued

meet distinguishes: these mechanise one direction of the mixed-axis impossibility result. The Lex five-chain and the tensor-factor algebra are therefore not in conflict: they are compositional-level views, meet-preserving in projection on their shared sub-lattice and separated by the mixed-axis result outside it.

4.3 Tensor commitments

A tensor evaluation produces a *tensor commitment*: a content-addressed digest of the canonical bytes of the evaluation result. This commitment binds the tensor state to a specific point in time and is included in verifiable credentials and passport entries. The commitment supports Merkle proofs for selective domain disclosure: a jurisdiction may verify a specific domain's state without seeing the full evaluation.

4.4 The compliance passport

The compliance passport is a hash-chained sequence of compliance evaluation snapshots. Each entry records:

- The entity the passport belongs to.
- The jurisdiction evaluated against.
- The aggregate compliance status.
- The per-domain compliance results.
- Summary statistics (passing and blocking counts).
- The tensor commitment digest.
- A hash-chain link to the previous passport state.
- The content-addressed digest of the current state.

The hash-chain link renders the passport tamper-evident: any modification to a historical entry breaks the chain. The construction follows Haber and Stornetta (1991) for linked timestamping, with each entry cryptographically bound to all its predecessors. The passport is signed by the issuing zone's signing key and is therefore verifiable by any party holding the zone's public key.

Passport as hash-chain and per-domain Merkle commitment. Formally, the passport is the pair $(\pi_{\text{chain}}, \pi_{\text{merkle}})$ where $\pi_{\text{chain}} = e_0, e_1, \ldots, e_N$ is a hash-chained sequence of signed evaluation entries with e_{i+1} binding $H(e_i)$, and π_{merkle} is, for each entry, a Merkle tree over the per-domain verdict records keyed by the shared domain taxonomy. The Merkle construction supports selective disclosure: a holder can reveal the verdict on a chosen subset of domains together with inclusion proofs, without revealing the remainder.

The passport has a default validity period with domain-specific time-to-live for domains whose regulatory cycles are shorter (sanctions screening being the salient example). The verifiable-credential format

follows the W3C Verifiable Credentials Data Model and is designed for interoperability with eIDAS 2.0 cross-border trust services.

4.5 Passport verification

A receiving jurisdiction verifies a foreign passport through the following procedure:

(1) **Signature verification.** Verify the signature under the issuing zone's verifying key. The receiving zone maintains a registry of known zone verifying keys indexed by key identifier and validity interval.

(2) **Hash-chain integrity.** Verify that each entry's link digest matches the preceding entry's content digest.

(3) **Tensor commitment verification.** Verify that the claimed per-domain results produce the claimed tensor commitment under the content-addressing hash (SHA-256).

(4) **Freshness.** Verify that the passport has not expired under its issuance timestamp and validity interval.

(5) **Revocation.** Verify that the issuing zone's verifying key was not revoked at the passport's issuance time, as recorded in the receiving zone's key registry with revocation timestamps.

(6) **Pack-version compatibility.** Verify that the passport's declared pack versions are within the receiving zone's corridor-parameter validity windows, ensuring that the μ translation and R policy apply under the pack versions the passport was issued under.

Theorem 7 (passport verification soundness). Under EUF-CMA signatures and collision-resistant content-addressing: a passport π verified by steps (1) through (6) is either (a) cryptographically attributable to the issuing zone's current non-revoked key under a current pack version, or (b) rejected with a specific failure code. □

Critically, verification does not require trusting the issuing zone's compliance evaluation. The receiving zone verifies the cryptographic integrity of the passport and then applies its own (R, μ) parameters to determine which domains to accept and which to re-evaluate locally. This is disclosed verification soundness: signed evaluation nodes, disclosed Merkle inclusions, non-revoked PCAuth or sovereign credentials, accepted pack windows, and replayable proof-bundle entries support the domains the receiver is allowed to inspect. It is not a privacy theorem over hidden domains. The passport provides evidence; the corridor parameters determine how that evidence is consumed.

5 The corridor protocol

5.1 Corridor definition

A corridor is a bilateral relationship between two kernels. It carries neither a shared ledger, nor a consensus mechanism, nor a message bus: it is a parameterised channel through which typed compliance state flows.

Formally, a corridor C is a tuple $(Z_A, Z_B, \tau, R, \mu, \gamma)$ where Z_A, Z_B identify the participating zones with their jurisdiction identifiers, τ is a corridor type classifier over zone metadata (cross-border, intra-federal, free-zone-to-host, or free-zone-to-free-zone), $R \subseteq \mathcal{D}$ is the re-evaluation set (the subset of compliance domains the receiving zone re-evaluates locally; domains in $\mathcal{D} \setminus R$ may be accepted via mutual recognition), $\mu : \mathcal{D} \rightharpoonup \mathcal{D}$ is a partial domain-recognition map translating source-jurisdiction domain names to target-jurisdiction equivalents, and γ is the family of grade-recognition maps authorized by the corridor instrument.

5.2 The (R, μ, γ) parameterisation

The triple (R, μ, γ) is the fundamental parameterisation of a corridor. It encodes the mutual recognition agreement between two jurisdictions in machine-executable form.

R: The re-evaluation set. When an entity's compliance passport arrives at the receiving zone, R determines which domains the receiving zone will re-evaluate locally rather than accepting from the passport. A smaller R means more mutual recognition (more domains accepted on faith); a larger R means more sovereignty (more domains re-evaluated locally).

The default constraint for general-purpose corridors is that Sanctions lies in R. A jurisdiction does not inherit a foreign sanctions evaluation as its own local clearance. A narrow subnetwork may encode Sanctions outside R only under a `SharedSanctionsAuthority` certificate: a signed legal artifact naming the common authority, the covered sanctions list or decision process, the bound jurisdictions, the validity window, the revocation procedure, and the corridor epochs to which the certificate applies. Characterizing those subnetworks and proving the certificate's governance treatment is a separate lawpack and corridor-governance obligation; absent that proof, the safe corridor rule is local sanctions re-evaluation.

μ: The domain-recognition map. Different jurisdictions may use different names for equivalent regulatory categories. μ translates between them where recognition is authorized. When μ is the identity function, both jurisdictions use the same domain taxonomy. When μ is non-trivial, the receiving zone maps foreign domain evaluations to local equivalents before deciding whether to accept them.

Recognition grades. The system supports four grades of per-domain recognition:

- **Full.** Foreign evaluation accepted without local re-evaluation; the domain is removed from R.
- **Partial.** Foreign evaluation accepted with a local review flag; the domain is removed from R but a review flag is carried forward.
- **Conditional.** Foreign evaluation accepted if a specified condition holds (e.g. "FATF member state", "bilateral treaty in force"). When the condition fails, the grade degrades to None.
- **None.** Full local re-evaluation required; the domain remains in R.

5.3 Corridor asymmetry

Corridors are asymmetric. Zone A's recognition of zone B's AML evaluation does not imply zone B's recognition of zone A's AML evaluation. Each direction carries its own (R, μ). This mirrors the structure of international regulatory arrangements: the EU's recognition of a third country's AML framework under the 4th Anti-Money Laundering Directive does not imply that country's recognition of the EU's framework.

5.3.1 Corridor algebra and its failure modes

Corridor composition asks: given a corridor C_{AB} from zone A to zone B with parameters (R_{AB}, μ_{AB}) and a corridor C_{BC} with parameters (R_{BC}, μ_{BC}), is there an induced composite C_{AC}? The question is load-bearing for multi-hop portability and for the consistency property: if an entity can route $A \to C$ directly or through B, the two paths ought to produce the same composed compliance state.

The composition rule. Define

$$R_{AC} \;=\; R_{AB} \cup \mu_{AB}^{-1}(R_{BC}), \qquad \mu_{AC} \;=\; \mu_{BC} \circ \mu_{AB}.$$

The composite re-evaluation set is the union under the first-hop preimage: any domain that either hop re-evaluates is re-evaluated in the composite. The composite domain mapping is the function composition of the two hop mappings.

Proposition 4 (associativity of route syntax). Let $\mathcal{C}$ be the category whose objects are triples $(\mathcal{D}, R, \mu)$ with $\mathcal{D}$ a finite domain set, $R \subseteq \mathcal{D}$, and $\mu : \mathcal{D} \rightharpoonup \mathcal{D}$ a partial function, and whose morphisms compose by

$$(R_{AC}, \mu_{AC}) \;=\; (R_{AB} \cup \mu_{AB}^{-1}(R_{BC}),\ \mu_{BC} \circ \mu_{AB}),$$

where $\mu_{AB}^{-1}(X) = \{d \in \mathrm{dom}(\mu_{AB}) \mid \mu_{AB}(d) \in X\}$. Then composition is associative.

Proof. Partial-function composition is associative. For the re-evaluation set, the two associations expand to $R_{AB} \cup \mu_{AB}^{-1}(R_{BC}) \cup (\mu_{BC} \circ \mu_{AB})^{-1}(R_{CD})$ and $R_{AB} \cup \mu_{AB}^{-1}(R_{BC} \cup \mu_{BC}^{-1}(R_{CD}))$ respectively. Apply $(\mu_{BC} \circ \mu_{AB})^{-1}(X) = \mu_{AB}^{-1}(\mu_{BC}^{-1}(X))$ and the distributivity $\mu_{AB}^{-1}(X \cup Y) = \mu_{AB}^{-1}(X) \cup \mu_{AB}^{-1}(Y)$ (valid for arbitrary functions, including partial ones, via the preimage construction over $\mathrm{dom}(\mu_{AB})$); the two expressions coincide. □

This proposition is only about the algebra of the syntactic route summary. It is not a theorem that arbitrary bilateral corridor instruments collapse to a single direct corridor with the same legal effect. Instrument clauses, pack-version windows, grade maps, and mixed-axis provenance require the route-coherence check below.

Invertibility of composition, a groupoid condition on the μ's, is a separate and strictly stronger property. It is not required for associativity; it is the additional condition that underwrites round-trippability ($A \to B \to A$ returning to identity on the shared domain sub-slice). The architecture does not assume groupoid structure.

Ledger obligation O-1 (route-coherence collapse). Let three corridors C_{AB}, C_{BC}, C_{AC} satisfy the cocycle equation $\mu_{AC} = \mu_{BC} \circ \mu_{AB}$ and $R_{AC} \supseteq R_{AB} \cup \mu_{AB}^{-1}(R_{BC})$ on the common vocabulary. The target theorem is that, after adding grade-map equations, pack-version compatibility, instrument-clause compatibility, and mixed-axis `MeetResult` provenance conditions, the direct route $A \to C$ produces the same carried state and fresh-evaluation obligations as the staged route $A \to B \to C$. The normalized equality checker over carried cells, fresh-evaluation domains, and instrument clauses is mechanized in `op/formal/coq/CorridorMonotone.v`, including soundness, completeness, false-obstruction, and decidability at the normalized layer. The open obligation is the semantic lift from concrete corridor data to that snapshot vocabulary, plus completeness for every named obstruction class. The target theorem is `normalize_route` soundness and obstruction completeness for the finite surfaces the protocol compares, not a general associativity theorem for all corridor instruments. The protocol may execute the staged route and surface non-coherence as `RouteCoherenceObstruction`; it does not silently choose a canonical collapse.

Failure modes. Corridor composition has three principal failure modes. Each is a genuine obstruction the architecture must recognise.

(1) **Route-coherence failure.** Write μ_{AC} for the direct $A \to C$ domain mapping and $\mu_{BC} \circ \mu_{AB}$ for the mapping composed through B. Let R_{AC} denote the direct re-evaluation mask, and $R_{AB} \cup \mu_{AB}^{-1}(R_{BC})$ the staged mask. If either equation disagrees on a domain, the two routes produce different carried states. The consistency equation is analogous to a cocycle law, but no sheaf-theoretic gluing theorem is claimed here. Disagreement is a genuine inconsistency between the bilateral agreements, not a pathology of the algebra; resolution requires renegotiation of at least one of the three corridor agreements. The architecture surfaces the failure as a typed `RouteCoherenceObstruction` value naming (i) the domains on which the two routes disagree and (ii) the corridor agreements whose renegotiation would close the gap; it does not silently select a winner.

(2) **Mutual-recognition asymmetry.** Suppose zone A accepts zone B's AML evaluation but zone B does not accept zone A's AML evaluation. Then the path $A \to B \to A$ does not return an entity to its starting state: the entity crosses $A \to B$ under full AML recognition and crosses back under required AML re-evaluation, and the return leg downgrades its AML verdict. This is not a pathology of the algebra; it reflects the structure of real international regulatory arrangements. The architecture preserves the asymmetry rather than smoothing it over. An entity planning a round trip must query both directions.

(3) **Pack-version skew.** Corridor parameters (R, μ) reference specific pack versions on either side. When one side updates its pack (tightens KYC rules in a new lawpack version, say), the corridor parameter may cease to be accurate for the new pack version. The protocol handles this conservatively: corridor parameters carry validity windows, and operations whose pack version falls outside the window require re-verification. Proceeding under stale parameters would admit attacks in which one side exploits the

lag between a pack update and corridor renegotiation to route operations under out-of-date recognition; the validity window forecloses this.

The three failure modes are distinct: cocycle failure is a disagreement between three bilateral agreements; mutual-recognition asymmetry is an intended feature of bilateral agreements; pack-version skew is temporal drift between agreement and ground truth. A uniform treatment would obscure the distinctions; the architecture diagnoses and resolves each on its own terms.

5.3.2 The no-global-consensus claim

The corridor network achieves cross-border safety without global consensus. The claim is an architectural decomposition, not an impossibility-evasion result.

System model. Let $\mathcal{V}$ denote the set of zone kernels, each a separate party. A corridor is an unordered pair $\{A, B\} \subseteq \mathcal{V}$ together with the corridor datum of Section 5. The network maintains two strata of state: (i) per-zone local state, governed by the zone's evaluation and event chain under Theorem 1, and (ii) bilateral receipt chains, each shared between exactly two zones. No state is shared among three or more zones simultaneously. We model cross-zone communication as point-to-point, authenticated, and eventually-delivered; the network is partially synchronous in the sense of Dwork, Lynch, and Stockmeyer (1988), with unbounded but eventually-bounded message delay after a Global Stabilization Time. Signature schemes are EUF-CMA.

Because no state is shared across three or more parties, the n-party atomic commitment problem addressed by Fischer, Lynch, and Paterson (1985) does not arise: every cross-zone operation reduces to a bilateral interaction between exactly two zones. The load-bearing protocol is the bilateral signed commitment (BSC) of Section 5.4, structurally simpler than Skeen-style n-party three-phase commit (Skeen, 1981) and than the coordinator-based Paxos Commit of Gray and Lamport (2006). Two-party atomic commitment does not require a coordinator-independent pre-commit state visible to $n \geq 3$ replicas; after finality, terminal recovery follows from bilateral signature durability, while pre-finality partitions remain in the bounded-blocking regime.

Fail-stop adversary. Under a fail-stop (crash-only) adversary for zone kernels, the BSC protocol achieves commit-safety: each kernel evaluates independently, signs its verdict, exchanges signatures, and transitions to a terminal state on the durable verdict pair. If a participant crashes after entering Verified, the surviving participant determines the outcome from the signed pair alone; no coordinator is required (Section 5.4).

Bounded-Byzantine adversary. Under a model in which a zone operator may deviate arbitrarily from the protocol (lie, fork, sign contradictory verdicts) subject to EUF-CMA unforgeability of signatures, the bilateral protocol does not by itself prevent damage during the window before a second conflicting verdict surfaces. Equivocation, the issuance of two conflicting signed verdicts under the same operation identifier and epoch, is the canonical Byzantine deviation. The architecture defends through three mechanisms:

(1) Signed verdicts are non-repudiable and content-addressed; equivocation is cryptographically detectable once both conflicting verdicts surface.

(2) A watcher layer (Section 9.4) observes receipt chains across corridors and surfaces conflicting verdicts by cross-checking.

(3) For high-value operations, the zero-knowledge lock proof of Section 9.6 binds the initiating zone's verdict to a nonce, epoch, and resource commitment, so equivocation produces two verifiable proofs under the same (zone-did, n, ϵ) with incompatible resource digests.

The conjunction yields *accountable safety* (Section 5.4): if safety is violated, a cryptographic certificate of blame is derivable. This is the accountable-safety regime studied by Haeberlen, Kouznetsov, and Druschel (2007, PeerReview) and sharpened for BFT by subsequent work (Kotla et al. 2007; Abraham et al. 2021). The defence is cryptographic accountability, not consensus.

What is not claimed. The protocol does not claim classical Byzantine fault tolerance. It does not prevent damage during the detection window before a second conflicting verdict surfaces. A zone operator in a strong Byzantine regime may temporarily route operations through a corrupted kernel until equivocation is detected, at which point the corridor-demotion and slashing mechanisms activate. The defence is temporal: detect, attribute, punish. Reputation dynamics, corridor partners adjusting R and recognition grades in response to observed behaviour, are operational rather than game-theoretic, and their steady-state characterisation is open (Section 11). We are explicit about the limit: the architecture removes the need for global consensus on entity compliance state while retaining local sovereignty; trust in zone operators remains a political property of the sovereign and not a computational property of the protocol.

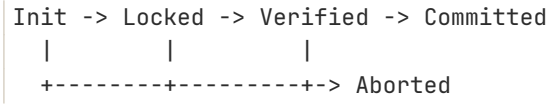
5.4 Bilateral signed commitment

Cross-zone operations (harbor addition or lawful harbor retirement, cross-border settlement, corridor-crossing trade) require coordination between two sovereign kernels. We name the load-bearing protocol the *bilateral signed commitment* (BSC). BSC is a target bilateral finality protocol rather than a closed claim of classical two-phase or three-phase atomic commitment: neither side is designated coordinator, and the safety theorem depends on the finality-certificate and recovery obligations below.

Protocol. Parties: zone kernels A and B. State:

$$s \in \{\text{Init}, \text{Locked}, \text{Verified}, \text{Committed}, \text{Aborted}\}.$$

Messages are signed envelopes $\langle m, \sigma_X \rangle$ under zone signing key k_X. An operation carries a unique identifier ω, a per-zone nonce n unique across (zone-did, ϵ), and an epoch ϵ.



Phase 1: Lock. The initiating zone locks the relevant resources (entity state, account balances, compliance passport). The lock is attested with a signature under the zone's signing key. For high-value operations, the lock is additionally accompanied by a Groth16 zero-knowledge proof (Groth, 2016) binding the initiating zone's resource commitment, nonce, epoch, corridor identifier, and zone decentralised identifier; the full relation and public-input schema appear in Section 9.6.

Phase 2: Sign verdict. Each zone independently evaluates compliance for the cross-zone operation and signs a verdict

$$v_X \;=\; \text{sign}_{k_X}\big(\omega, n, \epsilon, \text{tensor-state}_X, d_X\big)$$

containing:

- the operation identifier ω and type;
- the zone's compliance tensor state;
- the zone's decision $d_X \in \{\text{consent}, \text{reject}\}$;
- a signature over the verdict under the zone's signing key.

Verdicts are signed independently: neither zone observes the other's verdict before signing. This prevents one zone from conditioning its evaluation on the other's.

Phase 3: Exchange and terminate. Each side transmits v_X to the other. The transition Locked $\rightarrow$ Verified requires, at each side, durable possession of both v_A and v_B together with valid signatures under both zones' keys. We call this the *Verified-entry invariant*: entering Verified means both signed verdicts are durable at that side. Terminal transition additionally requires the finality certificate specified by ledger obligation O-2, or an abort certificate accepted by the timeout rules, to enter the corridor receipt chain.

From Verified plus the required terminal evidence, each side computes $d = \text{consent}(v_A) \wedge \text{consent}(v_B)$ on the certificate's verdict pair and transitions to Committed if d, otherwise to Aborted. Commit is final only when supported by receipt-chain finality evidence. Abort is idempotent; commit fails closed on repeated calls.

The BSC orchestrator is a pure state machine with no I/O: the caller drives transitions through `begin`, `lock`, `sign-verdict`, `exchange-verdicts`, `commit`, `abort`, and `timeout-check`. Invalid transitions produce errors. The certificate core is mechanized in `op/formal/coq/BSCInvariants.v`: `finality_certificate`, `abort_certificate`, and `terminal_evidence` are defined, and `finality_commit_iff_both_accept`, `terminal_evidence_functional`, and `recovery_from_terminal_evidence_is_pure` prove that accepted terminal evidence determines a unique terminal. The remaining object is an `AcceptedTerminalEvidence` receipt-chain witness tying that certificate to corridor id, (ω, ϵ), sequence or vector-clock position, endpoint signatures, key-epoch continuity, and inclusion proofs.

Invariants. Three invariants are preserved across every transition:

(I1) *Lock monotonicity.* Each side holds at most one active lock per (ω, ϵ) and releases it on transition to Committed, Aborted, or lock-timeout expiry.

(I2) *Verdict uniqueness.* Each side signs at most one verdict per (ω, ϵ). A second signature under the same key over the same (ω, ϵ) with a different payload is equivocation by that side.

(I3) *Verified-entry durability.* Entry to Verified at side X occurs only when both v_A and v_B are durable at X and valid under their respective keys.

Ledger obligation O-2a (BSC commit-safety under fail-stop). Assume a fail-stop (crash-only) adversary on either or both zones. If side X is in Committed, then side $\bar{X}$ is either in Committed (terminal agreement) or in a state from which only Committed is reachable on the terminal's recovery schedule.

Proof target. Durable local possession of both verdicts is insufficient: each side needs a guard excluding commit while the peer remains merely Locked and later times out. The release-gate proof must specify the finality certificate as a durable receipt-chain object and show that recovery is a pure function of accepted terminal evidence.

The certificate core is narrower and already mechanized in `op/formal/coq/BSCInvariants.v`: `finality_certificate`, `abort_certificate`, and `terminal_evidence` are defined; `finality_commit_iff_both_accept`, `terminal_evidence_functional`, and `recovery_from_terminal_evidenc` are Qed-closed. What remains open is binding those certificates to durable receipt-chain objects and the recovery finite-state machine.

Ledger obligation O-2b (BSC partition-heal terminal agreement). Assume the finality certificate required by ledger obligation O-2a is durable at both sides. For any partition schedule P opening after that point and any heal schedule that terminates, both sides reach the same terminal (Committed or Aborted) from the durable finality evidence, without relying on coordinator testimony.

Proof sketch target. Verdicts are content-addressed and signed. By (I3), either side's event store contains both verdicts linked into its corridor receipt chain at the moment Verified is entered. The commit decision is the pure function $d = \text{consent}(v_A) \wedge \text{consent}(v_B)$ of the pair. Both sides compute the same d on the same durable inputs; both transition to the same terminal. The partition-heal exchange is a receipt-chain integrity check, not a state merge. This is a target proof obligation until the finality-certificate dissemination and recovery model is closed in the public mechanization.

Remark on the precondition. If the partition opens during Locked, where one side has signed its verdict but the other has not received the signature, the protocol is blocking until heal. The block is bounded: the Locked-state validity window triggers unilateral lock release at expiry. Liveness is purchased through bounded locks, not through a coordinator-independent non-blocking property in the classical n-party sense. This matches the partial-synchrony analysis of Dwork, Lynch, and Stockmeyer (1988): termination under partition requires either synchrony bounds or explicit release mechanisms.

Ledger obligation O-3 (BSC accountable safety under bounded-Byzantine). Assume the signature scheme is EUF-CMA and that messages are content-addressed with a collision-resistant hash. If BSC reaches a state where side A is in Committed and side B is in Aborted (or the reverse), the target result

is either an accepted terminal-disagreement obstruction naming the missing finality precondition, or an equivocation certificate naming the zone that signed conflicting artefacts.

Proof target. Terminal disagreement alone does not imply equivocation; it may also reflect an underspecified finality condition, malformed imported evidence, or a timeout race. An equivocation certificate is accepted only when it contains two conflicting signed verdicts or terminal artefacts under one zone's key for the same (ω, ϵ) and incompatible resource commitment, with receipt-chain or watcher inclusion evidence. Slashing-enforceability is a distinct property requiring a bond custodian and an enforcement mechanism. Detection, attribution, slashing, corridor-demotion, and trust-level adjustment are distinct properties.

5.5 Foreign operation verification

When a zone completes an operation and sends a certificate to a partner zone, the receiving zone verifies the certificate. The verification:

(1) Checks the signature against the origin zone's known public key.

(2) Translates the foreign compliance state into the local domain space using μ and the applicable grade-recognition maps γ.

(3) **Re-evaluates the Sanctions domain locally.** A foreign sanctions verdict is evidence at most unless the corridor governance state carries an accepted `SharedSanctionsAuthority` certificate covering the domain, parties, validity window, legal basis, and revocation state. Legal carveouts are represented as Sanctions-rule scope or applicability evidence before evaluation, not as an override after a `NonCompliant` result.

(4) Computes an aggregate verdict incorporating the translated foreign state and the local re-evaluations for domains in R.

For general-purpose corridors, local sanctions re-evaluation is the default non-recognition constraint in the corridor protocol. Every other domain's treatment is configurable through (R, μ, γ). Sanctions is re-evaluated locally unless the corridor governance state carries an accepted `SharedSanctionsAuthority` certificate covering the domain, parties, validity window, legal basis, and revocation state.

5.6 Why this is not federation

Classical federation systems (X-Road, GAIA-X, various health data exchanges) move data between organizations. Corridors differ in three specific ways:

(1) **Typed compliance tensors, not opaque data.** Corridors carry compliance tensors: structured evaluations with defined domain semantics. X-Road carries data values (a name, an address, a KYC status). SWIFT carries payment instructions. A corridor message is not a data record; it is a compliance evaluation with algebraic structure.

(2) **Algebraic composition laws.** Compliance tensors support meet (greatest lower bound), join, and the Heyting residual. These are lattice operations with provable algebraic properties (associativity, commutativity, idempotence). When two tensors are composed, the result is the unique most-restrictive constraint satisfying both. Federation systems have no composition semantics; data received from a partner is consumed as-is or discarded.

(3) **Formally targeted composition.** The tensor algebra, the Lex type system, and the BSC protocol have formal specifications with machine-checkable obligations. Some narrow obligations are discharged; the route-coherence collapse, concrete BSC finality, accountable-disagreement proof, and full Lex metatheory remain open and are named in the companion ledger. The claim is that the composition is targeted at proof, not that every bit is proved.

The plumbing is familiar (bilateral channels, signed messages, mutual authentication). The content is new: typed tensors that compose point-by-point, with proof targets.

5.7 No transitive trust

Multi-hop corridors do not inherit trust transitively. If zone A has a corridor with zone B, and zone B has a corridor with zone C, an operation traversing $A \to B \to C$ is re-evaluated at every hop. Zone B does not relay zone A's compliance evaluation to zone C; zone B performs its own evaluation (incorporating zone A's passport via the A-B corridor parameters) and produces its own certificate, which zone C then verifies independently.

This is the correct behavior. Transitive trust in compliance would mean that zone A's regulatory standards propagate to zone C through zone B, even if zone C has no bilateral agreement with zone A. The real-world equivalent would be the EU accepting Pakistan's AML evaluation because Singapore accepted it, which no regulator would countenance.

Multi-hop composition evaluates transitive compliance across three-or-more-jurisdiction paths with per-domain evidence policies, re-evaluating sanctions at every hop unless every relevant hop is covered by a valid shared-authority certificate. When the three-jurisdiction cocycle fails (Section 5.3.1), the protocol reports the obstruction rather than silently selecting a path; the entity receives an explicit obstruction indicating which corridor agreements disagree on the relevant domain.

5.8 Graduated trust

Corridors carry a trust level in a finite set $L = \{\ell_0, \ell_1, \ldots, \ell_k\}$, with per-level operational-limit policies π_ℓ (bounded per-operation value, per-window volume, in-flight count, and settlement-window length). A corridor begins at ℓ_0 with conservative policy π_{ℓ_0} and is promoted one level on observed-settlement thresholds or demoted on observed-failure thresholds. Specific numeric thresholds in π are deployment-configured and negotiated bilaterally between zone operators; the architecture provides the mechanism (a typed trust-level variable carried on the corridor, policy tables keyed by level, and monotonic promotion/de-

motion rules), not the numbers. The mechanism ensures that new corridors operate under conservative constraints until they demonstrate reliability, while established corridors benefit from higher throughput and shorter settlement windows.

5.9 Corridor lifecycle

Corridors follow a typestate-encoded lifecycle: `Draft` → `PendingEndpointRatification` → `PendingRouteNotice` → `RatifiedPendingWindow` → `Active`, with `Halted`, `Suspended`, `Revoked`, `SupersededDraining`, `Retired`, and `DisputedOverlay` branches. A corridor in `Draft` cannot process operations. Endpoint ratification binds the sovereign authorities on both sides; route notice publishes the direct and staged routes whose coherence may be affected; the pending window gives counterparties time to halt before activation. A corridor in `Halted` state rejects all new operations while allowing in-flight operations to complete or time out. `Suspended` freezes all activity including in-flight operations. `SupersededDraining` keeps old receipts replayable while a new epoch accepts new operations. `DisputedOverlay` records an arbitration or watcher challenge without erasing the underlying state.

The governance state carried by the corridor is `(corridor_id, direction, epoch, pack_digest, state, trust_level, validity_window, endpoint_authorities, standing_set, watcher_set, dispute_forum, bond_custodian, route_impact_set, proof_status, gov_log_head, sanctions_authority_scope)`. The signed events are `Propose`, `EndpointRatify`, `RouteNotice`, `Activate`, `Tighten`, `Loosen`, `Halt`, `Suspend`, `Resume`, `Supersede`, `Drain`, `Retire`, `Dispute`, `Blame`, `Slash`, `SchemaEvolve`, and `SharedSanctionsAuthorityBind`. Watchers attest and surface blame; they do not amend corridor law. Slashing requires an accepted blame certificate plus bond-custodian authority. Tightening recognition is prospective and destination-sovereign; loosening recognition requires endpoint ratification, route-impact notice, and proof-status checks. Binding a shared sanctions authority is a loosening event and therefore requires endpoint ratification plus validity-window and revocation checks. These rules are a state-machine specification, not a completed theorem: the open obligations are governance authorization soundness, audit reconstruction, sovereignty preservation, sanctions-recognition safety, route-coherence publication, effective-window safety, emergency-transition safety, watcher/slashing soundness, schema-evolution coverage, and proof-status honesty.

6 Multi-harbor compliance composition

6.1 The composition problem

A multi-harbor entity operates in jurisdictions $J_1, J_2, \ldots, J_k$ simultaneously. It has a compliance tensor in each jurisdiction: $T_1, T_2, \ldots, T_k$. The entity must satisfy all jurisdictions simultaneously. What is its effective compliance constraint?

6.2 Pointwise meet

The answer is the pointwise meet across all tensors:

$$T_{\text{effective}} = T_1 \wedge T_2 \wedge \cdots \wedge T_k$$

where $\wedge$ is the meet operation on the compliance-state lattice, applied independently to each domain. For each domain d:

$$T_{\text{effective}}(d) = \text{meet}(T_1(d), T_2(d), \ldots, T_k(d))$$

This produces the most restrictive state across all harbors. If any jurisdiction says the entity is NonCompliant on AML, the effective AML state is NonCompliant, regardless of what the other jurisdictions say. If one jurisdiction says Pending and another says Compliant, the effective state is Pending.

The composition iterates over (zone, tensor-slice) pairs and applies the pointwise meet while tracking the source attribution per domain, recording which zone contributed each domain's most restrictive state. The source attribution is required by the residual operator (Section 6.3) to identify the binding harbor per domain.

6.3 The Heyting residual

Given the entity's current composed tensor T_{current} and a target tensor T_{target} (the compliance state the entity aims to achieve), the Heyting residual $T_{\text{current}} \Rightarrow T_{\text{target}}$ computes the compliance gap per domain:

- If $T_{\text{current}}(d) \geq T_{\text{target}}(d)$ the residual is `Compliant` (no improvement needed).
- If $T_{\text{current}}(d) < T_{\text{target}}(d)$ the residual encodes the specific improvement required.

The source attribution from the composition identifies the binding harbor per domain. If the entity is NonCompliant on Securities because of jurisdiction J_2's requirements, the residual identifies J_2 as the harbor where Securities compliance must be addressed.

6.4 Conjunctive consistency and composition obstructions

Zone composition requires that a composed (synthetic) zone's compliance constraints be at least as restrictive as the natural jurisdictions it draws from. The property is necessary: a synthetic zone that lifted constraints from its component jurisdictions would be a back-door for regulatory arbitrage; the composition would be a way to reduce the surface rather than a way to live within it.

A *composition obstruction* is any domain on which the synthetic zone's tensor would be less restrictive than a natural source jurisdiction's tensor. The check walks the pair (synthetic zone, source jurisdiction) across every domain and reports every obstruction. An obstruction means the zone composition is invalid as specified: either the synthetic zone must tighten its tensor on the offending domain (adopting the source's more restrictive state), or the source's constraint must be demonstrably inapplicable (for example, the source

jurisdiction does not govern the relevant activity in the synthetic zone's scope). A synthetic zone that cannot be constructed without composition obstructions is not a permissible composition; the architecture refuses to assemble it rather than silently producing a degraded constraint surface.

The check is a *conjunctive consistency check*: a pessimistic lattice join over sections detecting pointwise disagreement between the synthetic zone's tensor and each source jurisdiction's tensor. It is classically termed *descent* in categorical sheaf theory, but it is strictly weaker than full Grothendieck descent: the latter requires 1-cocycle agreement on triple overlaps between source covers. The architecture carries the cocycle check at the corridor-composition stratum (Section 5.3.1); the synthetic-zone assembly stratum still lacks that check. This gap is acknowledged; the companion paper on the algebra discusses the relationship in more formal detail.

6.5 Worked example

Consider an entity harbored in three jurisdictions, denoted J_1, J_2, J_3, and conducting a cross-border trade between J_1 and J_3. The entity must satisfy all three jurisdictions simultaneously.

Step 1: Per-harbor tensors. Each kernel evaluates the entity against its jurisdiction's rules, producing:

Each coordinate carries a (grade, applicability) pair; the grade slot is recorded as "_" when no substantive grade applies because the applicability marker is `NotApplicable` or `Exempt`.

```
T_1 (J_1):
  AML:         (Compliant, Applicable)    KYC:        (Compliant, Applicable)
  Sanctions:   (Compliant, Applicable)    Tax:        (Compliant, Applicable)
  Securities:  (Pending, Applicable)      Corporate:  (Compliant, Applicable)
  DataPrivacy: (Compliant, Applicable)    Banking:    (_, NotApplicable)
  Trade:       (Compliant, Applicable)

T_2 (J_2):
  AML:         (Compliant, Applicable)    KYC:        (Compliant, Applicable)
  Sanctions:   (Compliant, Applicable)    Tax:        (Compliant, Applicable)
  Securities:  (Compliant, Applicable)    Corporate:  (Compliant, Applicable)
  DataPrivacy: (Pending,   Applicable)    Banking:    (Compliant, Applicable)
  Trade:       (_, NotApplicable)

T_3 (J_3):
  AML:         (Compliant, Applicable)    KYC:        (Compliant, Applicable)
  Sanctions:   (Compliant, Applicable)    Tax:        (_, Exempt)
  Securities:  (_, NotApplicable)         Corporate:  (Compliant, Applicable)
  DataPrivacy: (Compliant, Applicable)    Banking:    (_, NotApplicable)
  Trade:       (Pending, Applicable)
```

Step 2: MeetResult-valued pointwise meet. Composition applies the production meet per domain. On Applicable coordinates both inputs share, the meet returns the three-chain greatest lower bound. On mixed-axis coordinates, the meet returns a structured outcome preserving the applicability provenance.

- **AML, KYC, Sanctions, Corporate, DataPrivacy:** every harbor reports `Applicable` with a substantive grade; the Applicable-fragment meet yields `Compliant` for AML, KYC, Sanctions, and Corporate, and `Pending` for DataPrivacy with binding harbor J_2.
- **Tax:** T_1 and T_2 report (`Compliant`, `Applicable`); T_3 reports (_, `Exempt`). Under the mixed-axis result this is a mixed-axis coordinate. The MeetResult records the Applicable-fragment meet restricted to $\{J_1, J_2\}$ (yielding `Compliant`) together with the mixed-axis flag that J_3 has exempted the entity from the Tax domain under a signed policy artefact. The operator does *not* output a synthetic `Exempt` or `Compliant` grade on the composed coordinate; it hands the applicability provenance to the rule layer.
- **Securities:** T_1 reports (`Pending`, `Applicable`); T_2 reports (`Compliant`, `Applicable`); T_3 reports (_, `NotApplicable`). Applicable-fragment meet on $\{J_1, J_2\}$ is `Pending` with binding harbor J_1; J_3's `NotApplicable` is preserved as applicability provenance.
- **Banking:** T_1 and T_3 report `NotApplicable`; T_2 reports (`Compliant`, `Applicable`). Applicable-fragment meet restricted to $\{J_2\}$ yields `Compliant` with `NotApplicable` applicability provenance from $\{J_1, J_3\}$.
- **Trade:** T_1 reports (`Compliant`, `Applicable`); T_2 reports `NotApplicable`; T_3 reports (`Pending`, `Applicable`). Applicable-fragment meet on $\{J_1, J_3\}$ is `Pending` with binding harbor J_3 and `NotApplicable` applicability provenance from J_2.

Step 3: Source attribution. The composition records which harbor binds each domain substantively and which jurisdictions contributed applicability provenance:

- Securities: bound by J_1 (Pending), with J_3 NotApplicable provenance
- DataPrivacy: bound by J_2 (Pending)
- Trade: bound by J_3 (Pending), with J_2 NotApplicable provenance
- Tax: Applicable-fragment meet on $\{J_1, J_2\}$ is `Compliant`; J_3 Exempt is preserved as a standing applicability condition

Step 4: Applicability reduction for planning. A delegated planner reduces the mixed-axis outcome to a planning projection on a fixed applicability slice. For each domain the planner fixes the applicability slice it is planning against, namely the set of harbors where the domain is `Applicable` for this planning cycle, and applies the pointwise residual and remediation operator on that slice. Coordinates with `Exempt` or `NotApplicable` markers are recorded in the plan as applicability conditions (for example, "J_3 Tax exemption must remain in force" as a standing assumption), not folded into the grade. Against the all-Compliant target:

- Securities: achieve `Compliant` in J_1 (complete the pending securities registration)
- DataPrivacy: achieve `Compliant` in J_2 (complete the outstanding data-protection impact assessment)

- Trade: achieve `Compliant` in J_3 (complete the pending customs documentation)

A delegated planner managing this entity receives the remediation plan as a structured object plus the applicability provenance. The plan names the three actions, the relevant jurisdictions, and the applicability conditions that must be preserved for the plan to remain valid. Execution proceeds through the kernel write path in the respective harbors. After each action, the affected kernel re-evaluates its tensor, corridor intelligence propagates the change, and the composed tensor is recomputed.

Step 5: Corridor crossings. The cross-border trade between J_1 and J_3 requires a corridor crossing. The corridor has parameters $R = \{\text{Sanctions}, \text{Trade}, \text{AML}\}$ and $\mu =$ identity (both endpoints use the shared domain taxonomy). J_1's passport for the entity crosses the corridor. J_3 re-evaluates Sanctions (mandatory), Trade, and AML locally; accepts the remaining domains via mutual recognition. The signed commitment protocol and its finality certificate coordinate the settlement.

6.6 Navigable compliance surfaces

The compliance tensor, Heyting residual, and source attribution together define a constraint surface that delegated planners can navigate. Given an entity's multi-harbor compliance state, a planner can:

(1) Compute the effective constraint surface via pointwise meet.

(2) Identify binding constraints per domain per harbor via source attribution.

(3) Compute the improvement prescription via the Heyting residual.

(4) Prioritise remediation actions by domain severity and harbor importance.

The planning layer consists of a policy engine, an action scheduler, and a deficiency planner, supported by an intelligence layer that consumes the compliance observation corpus. Every compliance evaluation feeds the system's understanding of jurisdictional requirements and can produce pack-update candidates with evidence and confidence estimates. The operational loop is described in Section 7.

7 Delegated automated operation

7.1 Caller authentication and identity

A delegated program authenticates to the kernel through the same mechanism as any other caller: a credential, verified by the kernel, from which a uniform caller identity is extracted bearing a role, an entity scope, and delegated-program attribution. When the attribution field is set, every mutation journaled through the mutation firewall is tagged accordingly in the audit trail. This is not a different write path. It is the same write path with provenance. Sanctions fail-closed evaluation applies identically. The tensor evaluation applies identically. The proof bundle is produced identically. A delegated program has no more

and no less authority than a human caller with the same role and scope. The distinction exists for audit attribution: regulators can query the event store for delegated-program mutations and review them separately.

7.2 The operation loop

A delegated planner managing an entity's compliance operates in a closed loop.

Observe. The planner reads the entity's current compliance tensor across all harbors where the entity operates. For a multi-harbor entity, this means reading $T(E, J_1), T(E, J_2), \ldots, T(E, J_k)$, one tensor per harbor. The planner also reads the entity's compliance passport (the full hash-chained history) and any pending operations.

Compose. The planner computes the effective tensor via pointwise meet: $T_{\text{eff}} = T_1 \wedge T_2 \wedge \cdots \wedge T_k$. This is a pure algebraic computation: no external call, no side effect. The planner thereby obtains the most restrictive constraint surface the entity faces across all jurisdictions.

Compute remediation. Given T_{eff} and a target tensor T_{target} (typically all-Compliant, or a specific compliance posture the entity wishes to achieve), the planner computes the remediation operator. The resulting plan identifies, for each domain, whether improvement is needed and which harbor is the binding constraint via source attribution.

Select action. A deficiency planner converts the residual into a prioritised sequence of typed remediation actions: submit a KYC document in harbor J_2, complete a securities registration in harbor J_1, file a tax return in harbor J_3. The selection is deterministic given the residual and the entity's state.

Execute. The planner submits each action through the kernel write path by supplying an operation envelope or a front-end request that compiles into one. The admitted operation is then subject to the same write-path discipline (sanctions fail-closed evaluation, pre-flight tensor evaluation, business-logic delegation, evidence storage, post-flight re-evaluation, verifiable-credential issuance, passport update, mutation journaling where the deployment has those stages wired) as any other caller. The planner has no backdoor. It uses the same admission boundary a human operator would use.

Observe again. After execution, the planner reads the updated tensor. A domain that was `NonCompliant` or `Pending` may now be `Compliant`. The effective tensor has changed. The remediation plan has changed. The loop repeats.

This loop is the fundamental unit of delegated automated compliance management. It is not a summary function. It is a typed proposal-and-admission loop through the kernel's write path, followed by observation of the composed result.

7.3 Multi-harbor interaction effects

When a delegated planner proposes an action in harbor J_1 and the kernel admits it, the effects can propagate to other harbors through two mechanisms:

Direct tensor change. The action in J_1 changes $T(E, J_1)$. Since T_{eff} is the pointwise meet of all harbor tensors, the effective tensor changes. If J_1 was the binding constraint for some domain d (i.e., $T(E, J_1)(d)$ was the most restrictive value), and the action improved that domain, then $T_{\text{eff}}(d)$ improves; this potentially changes the entity's effective compliance state in every harbor.

Corridor intelligence propagation. The kernel in J_1 emits a compliance observation into its observation corpus. Anonymized zone-level intelligence digests are exchanged bilaterally with corridor partners. If J_2 has a corridor with J_1, J_2's intelligence layer receives the updated observations. This can trigger re-evaluation of related fibers in J_2 because the observation is informative for J_2's own evaluators; J_1's evaluation does not bind J_2.

The corridor intelligence exchange is signed, replay-protected (monotonic sequence numbers per corridor), recipient-bound (preventing unintended forwarding), and supports multi-hop propagation with privacy-preserving aggregation (zone-level granularity, never entity-level). The sovereignty of each zone is preserved: intelligence digests are informative, not authoritative.

7.4 Scaling: N planners, M entities, K harbors

Consider the computational complexity when N delegated planners manage M entities across K harbors, with domain set of fixed size D.

Tensor composition. For each entity, composing K harbor tensors is $O(K \cdot D)$. With M entities, total composition cost is $O(M \cdot K \cdot D)$. This is a pure algebraic computation with no external I/O.

Remediation computation. Computing the remediation operator for one entity is $O(D)$. For M entities: $O(M \cdot D)$.

Action execution. Each admitted action is one write through the kernel's pipeline. The bottleneck is external I/O: sanctions screening (network call to sanctions lists), business-logic delegation, and storage-layer write. The kernel uses a set of background daemons with leader election and hash-based jitter to distribute work.

Cross-harbor propagation. An action in harbor J_1 triggers re-evaluation in at most $K - 1$ other harbors (one per corridor partner). Each re-evaluation is $O(D)$ (tensor recomputation) plus the cost of any fiber re-evaluations triggered by the new evidence. The total propagation cost per action is $O(K \cdot F)$, where F is the average number of fibers re-evaluated per harbor per incoming intelligence digest.

Per-cycle cost. One operation-loop cycle for one entity across K harbors: $O(K \cdot D)$ for composition + $O(D)$ for remediation + $O(1)$ for action execution (apart from I/O) + $O(K \cdot F)$ for propagation. The dominant cost is fiber re-evaluation.

Concurrency. Planners operating simultaneously on distinct entities are independent: each entity's compliance state is carried by its own hash chain, and the $N = 1$ write-path property (Theorem 1) ensures that writes to disjoint entities do not interact. Planners operating on the same entity are serialised by the kernel's mutation firewall under optimistic concurrency on the storage layer, with conflict detection through hash-chain verification. Lamport clock ordering (Lamport, 1978) of events within an entity's append-only event store yields a total order on mutations per entity, regardless of the number of planners attempting concurrent writes.

Theorem 8 (delegated-planner concurrency serialisability). For any finite set of concurrent delegated-planner actions $\{a_1, \ldots, a_n\}$ through the kernel's write path on entities $\{E_1, \ldots, E_m\}$, there exists a sequential schedule σ of the actions such that the resulting kernel state equals the state produced by executing σ serially.

Proof sketch. Actions on distinct entities commute because the $N = 1$ write-path property (Theorem 1) constrains each write to insert only into its entity's hash chain; distinct hash chains share no state. Actions on the same entity are ordered by the mutation firewall's optimistic-concurrency check: on hash-chain head mismatch, the losing action retries against the new head. The retry sequence defines a total order per entity. Combining the per-entity orders via any interleaving yields a valid sequential schedule σ. □

7.5 Simultaneous planners at scale

When many delegated planners operate simultaneously across the network, the following structural properties hold.

Independent observation. Each planner observes its own entity's tensor independently. Two planners managing two distinct entities in two distinct harbors do not interfere. They share no state except the corridor-intelligence digests, which are read-only summaries with no coordination requirements.

Convergent remediation. Several planners working on entities in the same harbor independently discover the same binding constraints, because Lex rules and fiber evaluations are deterministic. Their remediation actions follow similar patterns, generating a corpus of compliance observations that improves the intelligence layer for later evaluations. The compounding is indirect: admitted actions produce evidence that makes future evaluations more efficient.

No coordination protocol. Planners do not coordinate with each other. There is no planner-to-planner communication protocol, no shared planning state, no consensus mechanism. Each planner operates in its own loop, observing tensors and submitting actions through the kernel write path. Coordination is carried

by the kernel: the mutation firewall serialises writes to each entity via Section 7.4; the compliance tensor provides a consistent view; the corridor protocol propagates effects across harbors.

Graceful degradation. If a planner fails, its entity's compliance state is unchanged: no in-progress mutation can be left half-applied because the mutation-firewall gate is atomic. Another authorized planner can continue from the current tensor state. The compliance passport provides the full history; the tensor provides the current state; the remediation plan provides the prescription. No planner-specific state needs recovery.

8 Network topology and scaling

8.1 The network graph

Many kernels worldwide form a graph $G = (V, E)$, where V is the set of zones (each running its own kernel) and E is the set of established corridors. The graph is:

- **Sparse by construction.** Corridor establishment is a bilateral decision between two zone operators, involving negotiation of (R, μ) parameters that encode a mutual recognition agreement.
- **Disconnected components admitted.** The graph may contain isolated zones with no corridors, or disconnected components.
- **Directed.** Corridors are asymmetric; each direction has its own (R, μ). The graph is more precisely a directed graph in which each undirected corridor edge represents two directed edges.
- **Weighted.** Corridors carry metadata: trust level, operational limits, compliance domain coverage, fee structure. Routing algorithms (Dijkstra-weighted) find minimum-cost paths through the corridor graph.

8.2 Scaling properties

Adding a new zone. $O(k)$ in software, where k is the number of corridor partners the new zone wishes to establish. The kernel software is the same for every zone; the jurisdictional specificity comes from the compliance rules encoded in packs and Lex files, not from the kernel code. The marginal software cost of a new jurisdiction is a set of Lex rules and YAML operation definitions: data, not code.

The binding constraint on network growth is not software. Each corridor requires bilateral negotiation of (R, μ) between two sovereign regulators. Realistic parameters, benchmarked against existing bilateral regulatory instruments (bilateral investment treaties, double tax agreements, free trade agreements): 2 to 4 years elapsed time per corridor, on the order of $1M in legal and negotiation cost, with a 20 to 30% formation failure rate. Activation is a regulatory process with a non-zero probability of failure, not a technical deployment.

Adding a new compliance domain. Adding a new domain to the taxonomy requires every evaluator, every route, and every corridor parameter to address it. An implementation in a language with exhaustiveness checking surfaces any unhandled case at compile time rather than runtime. The cost of a new domain is proportional to the number of places that pattern-match on domains, not to the number of zones or corridors. The schema-evolution protocol prescribed by the architecture: add the domain variant; extend every evaluator; renegotiate every corridor's (R, μ) to reflect the new domain's treatment; and issue a new pack version carrying the extended taxonomy. Corridors operating under the old pack continue to function with the old taxonomy until their validity window forces renegotiation.

Adding a new harbor for an entity. One corridor crossing. The entity's compliance passport travels to the new harbor, the corridor's (R, μ) determines what is re-evaluated locally, and the entity begins accumulating compliance history in the new jurisdiction. The marginal cost of a new harbor is one corridor crossing, not a greenfield compliance engagement.

8.3 What compounds across zones

The observation corpus. Every compliance evaluation, in every zone, contributes to a growing body of structured compliance observations. The corpus records operation type, jurisdiction, domain outcomes, evidence types, and evaluation duration. This corpus compounds because each zone's evaluations are contextually informative for other zones operating under similar regulatory frameworks. A sanctions screening result in one jurisdiction informs the risk model for entities that also operate in another. A corporate-governance evaluation in one common-law free zone informs the template for similar evaluations in a peer common-law free zone.

Compliance passports. As entities accumulate compliance history across multiple zones, their passports become increasingly valuable; a well-attested entity with a long passport history is cheaper to evaluate than a new entity. The passport portability creates a genuine network effect: the more zones an entity has been evaluated in, the more evidence a new zone can draw from.

Operation definitions. The operation engine is driven by a declarative catalog of operation families keyed by jurisdiction. Each new jurisdiction definition benefits from the pattern established by existing definitions. The declarative operation model means jurisdictional coverage is a data problem, not a code problem.

Corridor intelligence. Corridor partners exchange anonymized zone-level intelligence digests, aggregated compliance observations that inform partner zones without exposing individual entity data. This exchange is signed, replay-protected, recipient-bound, and supports multi-hop propagation with privacy-preserving aggregation (zone-level granularity, never entity-level).

8.4 What does not compound

Sovereignty. Each kernel is sovereign. Zone A's compliance evaluation does not constrain zone B's evaluation. The network does not produce a "consensus compliance state": each zone evaluates independently. This is a feature, not a limitation: regulatory sovereignty is the property being preserved, not an obstacle being overcome.

Trust. Trust does not compound transitively. A corridor between A and B, and a corridor between B and C, does not create implicit trust between A and C. Every hop re-evaluates. This is by design (Section 5).

Regulatory authority. The network does not aggregate regulatory authority. A zone's regulator has authority over that zone's kernel and no other. The network provides information flow (compliance passports, corridor intelligence). Authority flow remains sovereign-local.

8.5 Hub-and-spoke topology and network value

The corridor topology is not uniformly distributed. Large financial centers (Delaware, Singapore, Switzerland, ADGM) serve as corridor hubs with many connections; smaller zones connect primarily through hubs. The corridor network is to jurisdictional compliance what BGP is to internet routing: a protocol for bilateral agreements between sovereign entities that produces a directed reachability graph without central coordination. Connected components emerge from adoption and negotiation capacity; they are not guaranteed by the protocol alone.

Network value concentrates at hub zones. A spoke that acquires a corridor to a hub gains composition access to every other zone already connected to that hub, at no additional corridor cost; a hub that adds another spoke gains marginal composition value proportional to the number of existing spokes. This is the mechanism behind the Odlyzko-Tilly (2005) $O(n \log n)$ regime: weakly superlinear network value, concentrated at hub zones, not Metcalfe n^2. The claim is an empirical hypothesis, not a theorem.

Two properties of the architecture bound the implications of this concentration:

(1) **No transitive trust.** A hub zone cannot "relay" compliance evaluations between spoke zones. Each corridor is bilateral. Hub status gives a zone more connections, not more authority.

(2) **Passport portability.** An entity's compliance passport is not tied to the hub zone. If a hub zone becomes unreliable or raises fees, entities can route through alternative paths (assuming corridors exist). The compliance history travels with the entity, not with the hub.

The concentration regime differs from Facebook-style winner-take-all network lock-in. Jurisdictional governance is differentiated and substitutable across regulatory cohorts; a superior second-mover zone can outcompete an inferior first-mover. Early entry captures composition value with the current hub set but does not prevent late-mover corridor formation. A clearinghouse gains power because it holds the asset records; in this network, the entity holds its own compliance passport, and the hub provides connectivity, not custody of compliance state.

9 Failure modes and security

9.1 Zone failure

A zone goes offline. All operations within that zone halt. Operations in other zones are unaffected. Cross-zone operations involving the failed zone enter a timeout state and eventually abort per the BSC timeout protocol.

This is sovereign governance, not a technical failure. If a jurisdiction's infrastructure goes down, operations in that jurisdiction stop, as they would if a country's banking system went offline. The network does not attempt to "work around" a failed zone by continuing operations on its behalf. That would violate the sovereignty property.

The BSC design target handles in-flight cross-zone operations through typed terminal evidence: if a zone fails to respond before finality, the operation remains bounded-blocking until the timeout rule produces an accepted abort certificate or the missing evidence arrives. Lock records are intended to ensure that resources are released on accepted abort. The recovery FSM is itself a proof obligation: a zone coming back online must resume from its last receipt-chain state and either terminate in agreement or return a typed obstruction.

9.1.1 Cross-kernel state reconciliation under partition

A partition of the corridor network, in which two kernels K_A and K_B cannot communicate for an interval $[t_0, t_1]$ while both continue to accept local operations, raises a distinct question from single-kernel crash recovery. During the partition, each kernel writes to its own append-only event store. Neither store is corrupted; each is the correct record of what its own jurisdiction's kernel observed and authorised. The reconciliation question is what happens to the corridor between them.

The architecture's answer is structural: there is no cross-kernel shared state to reconcile. Each kernel is sovereign over its own event store, its own compliance passports for entities harboured in its jurisdiction, and its own fiber evaluations under its own packs. A partition does not produce divergent copies of the same state; it produces independent progress on independent states. What does require reconciliation is the set of in-flight corridor operations whose BSC protocol was interrupted at t_0.

Ledger obligation O-2c/O-3 (partition-heal invariant preservation). Let ω be a corridor operation in state $s_X(\omega) \in \{\text{Init}, \text{Locked}, \text{Verified}, \text{Committed}, \text{Aborted}\}$ at side $X \in \{A, B\}$ at the moment a partition opens. After the partition heals, under the fail-stop or bounded-Byzantine adversary as specified, the recovery procedure must satisfy the following case analysis on $s_A(\omega) \times s_B(\omega)$:

- *Init at either side.* No lock was taken; no resources are reserved. The partition has no effect on ω. The initiator eventually times out and retries or abandons. Both sides agree on this outcome trivially.
- *Locked at X, any state at $\bar{X}$.* One side holds a lock; the other may or may not have received the lock notification. The lock carries a bounded validity window Δ_L. At expiry, the locking side releases unilaterally unless a valid finality certificate has already formed. On heal, the side that lost connectivity

reconciles its own view via the event-store exchange; its local lock, if held, times out on Δ_L. This is a bounded-block regime, not classical non-blocking recovery.

- *Verified at both sides with a shared finality certificate.* Both sides hold the certificate required by Section 5.4. On heal, both sides deterministically compute the same terminal from the same durable certificate, with no further coordination.
- *Verified at X, Locked at $\bar{X}$.* $\bar{X}$ has not received X's verdict (otherwise (I_3) would place $\bar{X}$ in Verified). On heal, $\bar{X}$ receives the verdict; if both verdicts are then valid, $\bar{X}$ can form the same finality certificate before its lock expires, and the operation advances to the shared-finality case. If $\bar{X}$'s lock window expired during partition, the result is a terminal-disagreement obstruction unless the trace also contains an equivocation certificate. The protocol must surface this obstruction; it must not infer Byzantine blame from the crossed state alone.
- *Committed or Aborted at either side.* Terminal. Each side's local event store records the terminal transition. Agreement is accepted only when both terminals are supported by the same finality certificate or by compatible timeout evidence. Otherwise the heal procedure returns a typed terminal-disagreement obstruction.

Proof target. Each case must be discharged against the BSC transition relation, the lock-expiry rule, the finality-certificate definition, and the accepted terminal-evidence verifier. The crossed-state case is the reason the certificate is load-bearing.

Reconciliation procedure. The reconciliation procedure on partition heal is therefore a metadata exchange, not a state merge. Each side publishes the head of its corridor-operation receipt chain. The other verifies the receipt chain's integrity against the hash-chain invariants and advances its view of the partner's terminal state for partition-time operations. No cross-kernel conflict resolution is required because no cross-kernel writes are attempted. Accepted recoveries must terminate in agreement; rejected recoveries must return a typed obstruction naming the missing finality, timeout, or equivocation evidence.

The one class of operation that is not auto-reconcilable is Byzantine equivocation: a malicious zone operator who, during the partition, issues conflicting signed verdicts to two partition halves. The bounded-Byzantine defence (Section 5.3.2) applies only under explicit observer assumptions: the conflicting signed artifacts must be published or otherwise imported into both sides' evidence surface, the watcher threshold must not be captured, and any slashing mechanism requires a bond custodian and accepted verifier. The partition temporally separates evidence gathering from attribution; non-repudiable signatures make attribution possible once the evidence is visible.

In sum, "no global consensus" in the partition setting is the architectural fact that the system has no cross-kernel shared state to coordinate over. It has bilateral receipt chains whose content-addressed structure makes independent reconciliation sufficient for ordinary corridor operations once terminal evidence is fixed. The BSC proof obligation is to formalise partition healing precisely enough that any accepted recovery terminates in agreement and every rejected recovery returns a typed obstruction with the evidence needed for governance.

9.2 Split-brain recovery after kernel restart

When a kernel restarts after a crash, it must recover to a consistent state. The append-only event store is the recovery mechanism: the kernel replays the event log from the last known-good state. Because the event log is append-only (no `UPDATE`, no `DELETE`), the event store is its own write-ahead log. The recovery procedure:

(1) Read the last committed event for each entity (the hash-chain head).

(2) Verify hash-chain integrity from the head backward to the last checkpoint.

(3) Resume the background daemons, each acquiring leader-election locks before processing. Leader election prevents split-brain: if a previous instance is still holding locks (stale connection), the new instance blocks until the lock timeout before acquiring leadership.

(4) In-flight BSC operations are recovered via the BSC state machine's recovery FSM, whose full public mechanization is a release gate. Operations in the `Locked` state that timed out during the crash are aborted only when an accepted timeout certificate exists. Operations in the `Verified` state require accepted terminal evidence carrying the finality certificate specified by ledger obligation O-2. Ambiguous states return typed obstructions or are resolved by the timeout protocol.

The critical invariant: no mutation is lost (append-only guarantees), and no mutation is applied twice (content-addressed event IDs with hash-chain linking make duplicates detectable).

9.3 Cascade failure when hub zone goes down

When a hub zone (one with many corridor connections) goes offline, the cascade effects are bounded:

Direct impact. All corridors involving the hub zone enter a degraded state. In-flight BSC operations involving the hub zone time out and abort. Entities whose only path to certain jurisdictions runs through the hub lose multi-harbor connectivity to those jurisdictions.

Bounded blast radius. Operations entirely within non-hub zones are unaffected. Corridors between non-hub zones are unaffected. The hub zone's failure does not propagate; it creates a connectivity gap, not a data corruption event.

No authority loss. Entities previously evaluated by the hub zone retain their compliance passports. The passports are self-contained (hash-chained, signed) and do not require the issuing zone to be online for verification. A receiving zone can verify a passport from a crashed hub zone using the hub's public key (which is cached in the receiving zone's key registry).

Recovery. When the hub zone comes back online, its corridors re-activate through the graduated trust mechanism. Corridors that were in Active state return to Active (not Probationary) because the bilateral relationship and settlement history are preserved in both zones' databases.

This is an availability attack on the hub zone's connectivity, not a safety violation. No compliance evaluations are falsified. No passports are corrupted. Operations are blocked, not corrupted.

9.4 The watcher layer

Corridor receipt chains are observed by an external watcher layer. A watcher is an independent party that subscribes to one or more corridors and produces signed attestations over receipt chain heads. The role is separated from the kernel deliberately: watchers do not write compliance state; they observe and attest. We specify the abstraction precisely.

Watcher protocol. A corridor $C = (Z_A, Z_B, \tau, R, \mu)$ admits a finite set $W(C)$ of watchers, each with a registered signing key. A watcher $w \in W(C)$ produces attestations of the form

$$\alpha_w \;=\; \text{sign}_{k_w}\big(\text{corridor-id}, h_{\text{head}}, t, \text{epoch}\big),$$

where h_{head} is the content-addressed digest of the corridor's receipt chain head as observed by w, t is the watcher's local timestamp, and epoch is the corridor epoch. Each watcher posts a bond at registration time, slashable on proven equivocation under Section 5.4.

Detection threshold. A fork is declared when the union of watchers' attestations over a single (corridor, epoch) pair contains two distinct digests $h_1 \neq h_2$ each backed by at least θ attestations from distinct watchers, where the threshold θ is part of the corridor parameters and is bounded below by $\lceil |W(C)|/3 \rceil + 1$ to defend against minority collusion. When two disjoint θ-witnessed views exist, the bilateral evidence suffices to invoke the slashing mechanism on the equivocating zone via Section 5.4.

Three-level ordering for tiebreak. When attestations agree on the digest but disagree on the sequence, fork detection uses three-level ordering: timestamp (primary, with a 5-minute maximum clock skew tolerance); watcher-attestation count (secondary, more attestations preferred); lexicographic digest (tertiary, deterministic tiebreak). This is a tiebreaker rule for ordering rather than a fork resolution rule.

Watcher collusion. If a strict majority of watchers in $W(C)$ collude with a malicious zone operator, fork detection at threshold θ fails. The mitigation is structural diversity: watchers are operated by independent parties with misaligned incentives, such as the zone's regulator, the corridor partner, and independent auditors. The bonding mechanism creates an economic disincentive proportional to the bond size, but it does not prevent collusion in adversarial models stronger than EUF-CMA. This residual risk is a fundamental limitation of observer-based detection schemes and is named here rather than absorbed into the protocol's safety claim.

9.5 Corridor denial-of-service

An attacker who can flood a corridor's communication channel can block all cross-zone operations without compromising safety. The corridor's BSC operations time out and abort. No false compliance evaluations are created. No passports are corrupted. The attack is on availability, not integrity.

Properties preserved under DoS. Sanctions fail-closed evaluation remains enforced (each zone evaluates independently). Hash-chain integrity remains intact (no events are lost or modified). Compliance passports remain valid and verifiable. Local operations within each zone continue unaffected.

Properties degraded under DoS. Cross-zone operations are blocked. Corridor intelligence digest exchange is interrupted. Entities cannot cross corridors or establish new multi-harbor positions.

Mitigation. Rate limiting on corridor endpoints (per-source IP and per-corridor). Graduated trust limits (Probationary corridors have low throughput caps, limiting the blast radius of a DoS on new corridors). Corridor suspension (the zone operator can suspend a corridor under active attack, preserving in-flight operations while blocking new ones).

9.6 Malicious zone operator

A zone operator who controls the kernel could issue false compliance evaluations: marking a sanctioned entity as Compliant, for example. The architecture provides graduated defences.

Graduated trust. New corridors start at Probationary with conservative limits. A malicious zone cannot immediately route high-value operations.

Passport verification. The receiving zone cryptographically verifies the passport's integrity but applies its own (R, μ, γ) to determine what to accept. With a sufficiently large R (re-evaluating many domains locally), the receiving zone is largely immune to false evaluations.

Local sanctions re-evaluation. For general-purpose corridors, Sanctions is re-evaluated locally (Section 5). A malicious zone cannot route sanctioned entities through such corridors merely by falsifying its own sanctions status; shared-authority subnetworks require their own explicit premises.

ZK lock proofs. For high-value corridor operations, the lock phase is accompanied by a zero-knowledge argument. We commit to Groth16 (Groth, 2016) over a BLS12-381 pairing-friendly curve, whose soundness rests on the knowledge-of-exponent (KoE) and q-power discrete-logarithm assumptions standard for that scheme. The relation proved is

$$\mathcal{R}_{\text{lock}} \;=\; \Big\{ \big((r, c, z, n, \epsilon, I); (E, P, F_A)\big) : r = H(E, P, F_A) \;\wedge\; \text{verify}_{k_A}(v) = 1 \Big\},$$

where $v = \text{sign}_{k_A}(\omega, n, \epsilon, F_A)$, c is the corridor identifier, $z = \text{zone-did}_A$, and I is the validity interval. The public inputs are

$$\mathbf{x} = (r,\ c,\ z,\ n,\ \epsilon,\ I);$$

the witness is $\mathbf{w} = (E, P, F_A)$. Thus the proof certifies that the prover knows an entity state, jurisdictional policies, and fiber evaluations whose content-addressed digest is the public resource commitment, and that the prover's consent verdict verifies under its own key. Only the initiating zone's fiber commitment appears; the receiving zone's fiber hashes are neither required nor present as public input (neither zone sees the other's verdict before signing; Section 5.4).

Non-equivocation property. Nonce uniqueness per (zone-did, ϵ) is enforced at the credential layer: the credential-provisioning system issues each zone a fresh nonce space per epoch and refuses to countersign a lock proof whose nonce has already been consumed. Under this discipline, a zone that issues two lock proofs (π_1, π_2) sharing (z, n, ϵ) but carrying different resource digests $r_1 \neq r_2$ has thereby signed incompatible commitments under its own key.

Open proposition schema (equivocation extraction). Assume a specified lock-proof relation with public inputs $(z, n, \epsilon, \omega, r)$, knowledge soundness for that relation, EUF-CMA signatures, nonce-credential uniqueness, and an accepted blame verifier. The intended theorem is that two verifying lock proofs sharing (z, n, ϵ, ω) but carrying incompatible resource commitments either yield conflicting signed artefacts under the key bound to z or return a typed malformed-witness obstruction. This remains open until the circuit relation, witness format, nonce credential discipline, and blame-verifier rules are fixed and proved.

The paper specifies Groth16 as the reference scheme. A PLONKish implementation with transparent setup (FS-ROM over a polynomial commitment scheme) is an admissible substitution; the relation and public-input schema are unchanged, but the soundness assumption shifts from KoE to PCS-binding plus the random-oracle heuristic. Choice of scheme is a deployment decision; the protocol and its accountability properties are scheme-parametric.

Trust revocation. A corridor can be demoted or suspended. Three consecutive failures trigger automatic demotion. A zone operator or regulator can manually suspend a corridor, halting all cross-zone operations while allowing in-flight operations to complete.

9.7 Corridor compromise

If a corridor's communication channel is compromised (man-in-the-middle), the attacker could intercept or modify compliance state in transit.

Hybrid post-quantum signatures. The architecture specifies a hybrid signature scheme combining a classical elliptic-curve signature (e.g., Ed25519) with lattice-based and hash-based post-quantum signatures (e.g., the ML-DSA and SLH-DSA families standardised in NIST's FIPS 204 and FIPS 205), following the conjunctive-verification pattern analysed by Bindel et al. (2019). The hybrid scheme protects corridor

communications against both classical and quantum adversaries: the signature verifies only if every component verifies, so compromising any one component does not forge the composite. An implementation may activate the post-quantum components behind feature flags during the migration period; the architectural commitment is that the verification interface always accepts the hybrid envelope.

Key rotation with dual-key window. Zone keys rotate through a key-rotation interface. During rotation, both the old and new keys are valid for a specified window, ensuring that in-flight corridor operations signed with the old key remain verifiable. The zone maintains a registry of all currently-valid verifying keys, indexed by key identifier and validity interval.

Content addressing. All corridor messages are content-addressed: the message identifier is the content-addressed digest of the message's canonical bytes. Modification of any field changes the digest, making tampering detectable.

9.8 Receipt chain integrity

Corridor operations produce receipt chains: append-only sequences of receipts backed by a Merkle Mountain Range (MMR; Todd, 2012; Bünz et al., 2020) for efficient inclusion proofs. The receipt chain enforces:

- **Hash-chain continuity.** Each receipt's predecessor root equals the previous receipt's final state root.
- **Content addressing.** Each receipt's next-state root is the SHA-256 digest of the canonical byte encoding of the payload.
- **MMR commitment.** The chain's aggregate root is the MMR commitment over all next-state roots.

Fork detection uses signed watcher attestations with timestamp bounds, a 5-minute maximum clock skew tolerance, and three-level ordering: timestamp (primary), watcher attestation count (secondary), lexicographic digest tiebreaker (tertiary).

9.9 PII protection

The architecture specifies field-level encryption at rest for personally identifiable information, with authenticated encryption (e.g., AES-256-GCM) applied below the storage-layer interface. Log-level PII scrubbing in the request-handling middleware redacts sensitive fields (email, phone, national-identity numbers) before log emission. Outbound HTTP clients disable redirect policies to prevent server-side-request-forgery attacks. These are defense-in-depth measures above the cryptographic guarantees of the proof bundle itself.

10 Formal verification obligations

10.1 What is targeted for proof

The architecture names a set of formal obligations, some of which are discharged and some of which remain open. Honesty about the distinction is itself a contribution: compliance infrastructure whose formal targets are unclear cannot be audited, and compliance infrastructure whose claims exceed its proofs is a failure mode worth foreclosing.

State-machine properties (TLA+). The kernel's mutation-firewall invariants, the BSC corridor-commit protocol, the leader-election protocol for background daemons, the authentication state machine, the compliance-layer ordering, the Smart-Asset-VM execution model, the key-rotation protocol, and emergency halt semantics are targeted for TLA+ specification and bounded model checking. Until the public `.tla` artifacts and run parameters are cited, this paper treats those checks as specification obligations and scaffolding evidence, not as closed confirmation of safety or liveness.

Lattice algebra (Lean). The compliance-tensor lattice properties (associativity, commutativity, idempotence of meet and join; bottom absorption; top identity; the Galois connection for the Heyting residual; pointwise-meet correctness; domain-array identity and absorption) are stated as proof obligations with partial mechanized evidence. They target the algebraic obligations from which multi-harbor composition and the residual-based planning algorithm derive.

Type-system soundness (Coq). The Lex type system is mechanised in Coq. The mechanisation establishes decidability of admissibility, the full typing rules for the admissible core, and the substitution/shift commutation lemmas at arbitrary ambient depth (`lex/formal/coq/Lex/DeBruijn.v`, `subst_subst_ws_at_depth` and `shift_subst_commute_ws_at_depth`) that the admissible fragment requires. Progress is closed conditionally on the confluence property (`lex/formal/coq/Lex/Typing.v`, `progress` Qed-closed taking a `confluence_property` premise). Confluence itself has a parallel-reduction scaffold (`Lex/Confluence.v` with `par_refl`, `step_implies_par`, `par_implies_steps` all Qed-closed), with the Tait-Martin-Löf diamond as the residual induction. Preservation is partially closed: the seven irreducible-constructor cases (`Lex/Preservation.v`) and four binder-case cases (`Lex/Typing.v`, `preservation_case_step_defeasible_*`, `preservation_case_step_match_*`) land as Qed. The substitution-preserves-typing metatheorem is stated as a named Prop obligation whose full closure requires the same Tait-Martin-Löf induction. Strong normalisation is open. The paper claims only what is Qed-closed on trunk and names the residual obligations with their concrete blockers rather than overstating closure.

Bilateral signed commitment (proof target). The BSC protocol's commit-safety obligation (Section 5.4), partition-heal terminal-agreement obligation (Section 5.4), partition-heal invariant-preservation obligation (Section 9.1.1), accountable-safety obligation under bounded-Byzantine behavior (Section 5.4), and recovery finite-state-machine obligation remain proof targets. Current evidence is the state-machine speci-

fication, the payload-abstract session theorem, the BSC kappa invariants, and the certificate-core terminal-evidence determinism lemmas. Durable receipt-chain agreement, concrete payload instantiation, recovery FSM correctness, and Byzantine certificate extraction remain release gates. Accountable safety depends on the non-repudiability of signed verdicts (EUF-CMA), collision-resistant content addressing, explicit finality evidence, and watcher or receipt-chain inclusion assumptions.

Cryptographic soundness. Merkle inclusion-proof soundness and completeness under collision-resistant hashing are standard proof obligations. This paper does not claim closed public mechanization unless a specific public lemma is cited. The equivocation-extraction schema of Section 9.6 remains open until its circuit relation, witness format, nonce discipline, and blame verifier are fixed.

10.2 What is explicitly not proved

The paper is explicit about limits that neither the current proof set nor any plausible extension of it can remove:

Evaluator fidelity. The domain evaluators implement jurisdictional compliance rules in Lex. The rules are type-checked and the type system's soundness is partially mechanized, but whether a specific AML evaluator correctly encodes Pakistan's AML requirements is a question of fidelity between Lex code and statutory text, validated by jurisdiction experts, not by a proof assistant.

Operational equivalence. Refinement checking relates the abstract specification to an implementation-level model (route-to-model conformance). This is conformance checking, not full functional equivalence: it verifies that the implementation follows the specification's state machine, not that every line of implementation code is correct.

Cryptographic hardness. The system relies on standard cryptographic assumptions: the hardness of discrete logarithm on a named elliptic curve, collision resistance of the content-addressing hash, authenticated encryption of the field-encryption primitive, and, for ZK lock proofs, knowledge-of-exponent on a pairing-friendly curve. These assumptions are cited, not proved.

Byzantine adversary for zone operators. As discussed in Section 5.3.2, the protocol is safety-preserving under fail-stop assumptions and accountability-preserving under bounded-Byzantine assumptions (where signatures are unforgeable). It does not promise classical Byzantine fault tolerance. The defense against a Byzantine zone operator is reputation and cryptographic accountability, not consensus.

10.3 The formal-gate discipline

The architecture commits to treating formal proofs as CI artifacts. Gates for model-checking, refinement conformance, dynamic trace witnessing, operation-kernel legality, callback-contract validation, pass-

port hash-chain integrity, and proof-count regression should run on every commit. A release attestation digest-binds the gate reports. If any gate fails, the release is rejected.

This is an architectural commitment, not a guarantee that every gate is currently green. Formal proof mechanization is a moving target; the open obligations in the preceding subsection reflect current gaps, and honest accounting is required. The gate discipline ensures that new gaps are visible (a new admitted theorem in Coq raises a regression alarm) rather than silently accumulating.

11 Open problems and scope

The architecture specified in this paper admits several classes of open problem. These are the problems the construction makes tractable rather than the problems the construction solves.

Corridor parameter negotiation. The (R, μ) parameterisation is machine-executable. The negotiation process between jurisdictions, by which R and μ are agreed upon, is a regulatory and diplomatic process rather than a technical one. The architecture provides the machinery; the policy decisions are human.

Route-coherence resolution. When three bilateral corridor agreements fail the direct-versus-staged route equations (Section 5.3.1), at least one agreement must be renegotiated. Whether a general protocol for minimum-change renegotiation exists, and whether such renegotiation converges when multiple route-coherence failures interact, are open questions. The failure itself is detectable; the resolution is open.

Formal proof completeness. Preservation for the full Lex type system, unconditional progress, strong normalization beyond the flat affine fragment, confluence through the parallel-reduction diamond, and the `step_match_ctor_fire` shift-compatibility repair are open metatheoretic obligations. Their mechanization is in progress; this paper does not claim them as proved.

Byzantine zone-operator models. Section 5.3.2 describes the safety properties under fail-stop and bounded-Byzantine assumptions. A precise characterisation of the reputation dynamics in the Byzantine regime (under which corridor-partner update rules does equivocation provably converge to isolation of the malicious operator) is open work. The defence-in-depth response given in Section 9 is operational rather than game-theoretic.

Privacy-preserving compliance composition. Zero-knowledge proofs over the full compliance tensor would allow an entity to demonstrate that its composed compliance state satisfies the receiving jurisdiction's requirements without revealing the constituent evaluations. The admissible fragment's finitary character suggests that compilation is feasible, but the decision procedures that back fiber evaluation must be encoded as circuits. This is open engineering and, in some subcases, open cryptographic research.

Corridor network governance. Corridors are bilateral, but multi-hop corridors create externalities that bilateral negotiation may not handle well. Section 5.9 specifies the target governance state machine: endpoint ratification, route-impact notice, validity windows, watcher registries, dispute forums, bond

custodians, epoching, drain states, emergency halt/suspension, shared-sanctions-authority binding, and proof-status tracking. The open work is the theorem suite: authorization soundness, audit reconstruction, sovereignty preservation, sanctions-recognition safety, route-coherence publication, effective-window safety, watcher/slashing soundness, schema-evolution coverage, and proof-status honesty.

Non-computational switching costs. The architecture reduces the compliance component of jurisdictional switching cost. Physical assets, local employees, banking relationships, contractual obligations, and reputational capital create friction that no algebraic operation can eliminate. The competitive dynamics (Section 13) depend on the relative weight of these costs, which is an empirical question.

The contribution of this paper is the specification of an architecture in which these problems are well-posed. The problems themselves are the research agenda the architecture creates.

12 Related work

12.1 Blockchain systems

Bitcoin. Nakamoto (2008) introduced decentralized consensus through proof of work. The key insight is that mutually distrusting parties can agree on a transaction ordering without a central authority. The sovereign jurisdiction network does not solve this problem because it does not need to: within a jurisdiction, the kernel is the trusted authority (trusted by that jurisdiction's regulator). Between jurisdictions, the corridor protocol provides bilateral agreement without global consensus.

Ethereum. Buterin (2014) and Wood (2014) extend the blockchain model with a Turing-complete smart contract platform. The Ethereum yellow paper specifies a global state machine where every node executes every transaction. This global execution model is antithetical to jurisdictional sovereignty: a compliance evaluation under Pakistani law should not be executed (or validated) by a node operating under Swiss law. The sovereign jurisdiction network inverts this: execution is local, verification is bilateral, and there is no global state.

Cosmos IBC. Kwon and Buchman (2016) specify an inter-blockchain communication protocol that relays packets between sovereign chains. IBC is the closest existing analogue to corridors. The distinction is algebraic. IBC packets carry application-specific payloads whose interpretation is bilateral per-application (ICS-20 for fungible tokens, ICS-721 for non-fungible tokens, and so on); the protocol is parametric over the payload schema and does not define cross-chain composition semantics over the payload content. Corridors carry compliance tensors typed against the shared domain taxonomy of Section 2.3, with specified lattice operations (meet, join, Heyting residual) and a specified verdict-lattice structure. The receiving zone composes the payload with its own compliance state under the well-defined lattice operations. The typing is algebraic.

12.2 Federation systems

X-Road. The Estonian Information System Authority's X-Road is the data exchange layer underlying Estonia's digital government. X-Road enables secure data exchange between organisations, for example a citizen's identity data flows from the population registry to a bank. X-Road exchanges data; corridors compose compliance constraints. The distinction is algebraic: X-Road moves values (a name, an address, a KYC status), while corridors move typed tensors that participate in lattice operations. X-Road's "information system can query another information system" is necessary and insufficient for portable compliance. The compliance state must be composable as well as transferable.

SWIFT. The Society for Worldwide Interbank Financial Telecommunication provides the messaging infrastructure for international financial transactions. SWIFT messages (MT/MX, `pacs.008`) carry payment instructions between banks. The sovereign jurisdiction network includes a SWIFT `pacs.008` adapter for settlement on traditional rails. SWIFT is a messaging standard; the kernel is a compliance evaluation engine. They are complementary, not competing.

eIDAS. European Parliament Regulation (EU) No 910/2014 establishes a framework for electronic identification and trust services across EU member states, enabling cross-border credential portability. The kernel's W3C Verifiable Credentials are designed for interoperability with eIDAS trust services. The key distinction: eIDAS provides an identity and authentication framework within a pre-existing legal harmonization (the EU single market). The sovereign jurisdiction network operates between jurisdictions with no pre-existing harmonisation. The (R, μ) parameterisation encodes whatever degree of mutual recognition the bilateral parties negotiate, from full (eIDAS-like) to zero.

12.3 Distributed trust management

PolicyMaker, KeyNote, SPKI/SDSI. Blaze, Feigenbaum, and Lacy (1996) introduce decentralised trust management, in which policy is expressed declaratively and authorisation is the answer to a proof-of-compliance query. KeyNote (Blaze et al. 1999), SPKI/SDSI (Ellison et al. 1999), and XACML (OASIS 2005) extend the approach. The corridor (R, μ) tuple plays the corresponding role in the compliance setting: a declarative policy on which domains the receiving zone accepts and how it names them. The distinction is in the object being authorised. Decentralised trust management authorises an action under a policy; corridors carry typed compliance tensors whose composition is algebraic and whose outcome is a composed lattice element rather than a Boolean authorisation. The trust-management literature's mechanism is "policy compliance" as a proof-checker primitive; the corridor mechanism is pointwise lattice composition with residual.

12.4 Accountable distributed systems

PeerReview. Haeberlen, Kouznetsov, and Druschel (2007) introduce practical accountability for distributed systems: a protocol in which every participant's actions are witnessed and audited through a tamper-evident log, yielding cryptographic evidence of misbehaviour. The corridor protocol's accountable-safety property (Section 5.4) is in the same family: equivocation by a zone operator produces a pair of conflicting signed artefacts verifiable by a polynomial-time blame function. The distinction is in scope. PeerReview audits a general distributed application; the BSC protocol is a target bilateral finality protocol whose accountable-safety obligation reduces, in one case, to non-equivocation over (ω, ϵ) under one zone's key. The two works share the intuition that non-repudiable logs plus audit yield cryptographic accountability without classical Byzantine fault tolerance.

12.5 Routing systems

BGP. Rekhter, Li, and Hares (2006) specify the inter-domain routing protocol of the internet. Each autonomous system makes independent routing decisions; BGP propagates reachability information between autonomous systems. The corridor network bears the same relation to jurisdictional compliance as BGP bears to interdomain routing: a protocol for bilateral agreements between sovereign parties, producing a directed reachability graph without central coordination. Each node (zone) is sovereign, connections are bilateral, and the topology emerges from bilateral agreements rather than central planning. BGP's analogue of the (R, μ) tuple is RPSL, the Routing Policy Specification Language (Alaettinoglu et al., RFC 2622): a declarative language for import/export policy between autonomous systems. RPSL expresses which routes an autonomous system accepts from and announces to each peer; (R, μ) expresses which compliance domains the receiving zone re-evaluates and how it translates domain names across jurisdictions. The distinction is in what is routed: BGP propagates reachability of IP prefixes, while corridors propagate compliance evaluations composed under the lattice operations of Section 4. BGP's path selection uses AS-path length and policy; corridor routing uses compliance-tensor compatibility (Jaccard similarity over domain coverage) and cost (Dijkstra-weighted).

12.6 CBDC cross-border systems

mBridge and Project Dunbar. The BIS Innovation Hub's mBridge (2022) and Project Dunbar (BIS, Reserve Bank of Australia, Bank Negara Malaysia, Monetary Authority of Singapore, South African Reserve Bank, 2022) explore multi-CBDC cross-border payment platforms. These projects address the settlement layer, namely how central bank digital currencies move between jurisdictions. The sovereign jurisdiction network addresses the compliance layer, namely how compliance state is evaluated, composed, and verified across jurisdictions. The two are complementary: mBridge/Dunbar provide the payment rails, while the kernel network provides the compliance proofs that authorize movement on those rails. The kernel's

SWIFT `pacs.008` adapter and other payment-rail adapters are designed to integrate with exactly these kinds of settlement infrastructure.

12.7 Verified systems

seL4. Klein et al. (2009) is the first formally verified general-purpose operating system microkernel. seL4 proves functional correctness: the C implementation refines the abstract Haskell specification, which refines the Isabelle/HOL formal model. The sovereign kernel network's formal verification ambition is narrower: the obligations named in Section 10 target specific architectural properties (tensor algebra, type-system soundness, protocol safety) rather than full functional correctness of the implementation. seL4 sets a standard to which this work asymptotes in the specific subsystems it covers; the comparison is orientational, not equal.

12.8 Regulatory frameworks

Basel Accords. The Basel Committee on Banking Supervision establishes international standards for banking regulation. Basel III's capital adequacy, stress testing, and leverage ratio requirements are the kind of compliance rules that the kernel's Banking and Clearing domain evaluators encode. The Basel framework operates through "national implementation": each country implements Basel requirements in its own regulatory framework. The sovereign kernel network mirrors this: each kernel encodes its jurisdiction's implementation of international standards, and corridors parameterize how one jurisdiction's implementation relates to another's.

EU passporting. European Parliament Directive 2014/65/EU allows financial firms authorized in one EU member state to operate in other member states without additional authorization. EU passporting is the real-world precedent for the corridor concept. The distinction: EU passporting is a legal arrangement backed by a common regulatory framework (the single market); corridors are a technical protocol backed by pointwise composition semantics. Corridors can encode EU-style full mutual recognition ($R = \{\text{Sanctions}\}$, $\mu = \text{identity}$) or much more conservative arrangements (large R, non-trivial μ). The system is parametric over the degree of mutual recognition.

12.9 Timestamping and ordering

Haber and Stornetta. Haber and Stornetta (1991) introduced hash-chain timestamping: linking documents into a chain where each document's hash includes its predecessor's hash, creating a tamper-evident sequence. The kernel's append-only event store and compliance passport both use this construction. Each kernel event's identifier is the content-addressed digest of event type, payload, and prior-event identifier, directly implementing the Haber-Stornetta chain. The compliance passport extends this to compliance evaluations: each passport entry is hash-linked to its predecessor, creating a verifiable compliance history.

Lamport clocks. Lamport (1978) establishes a partial ordering on events in distributed systems without requiring synchronized physical clocks. The kernel event store provides a total order within each entity's event chain (hash-chain linking) and a partial order across entities (no cross-entity ordering is required or claimed). For cross-zone operations, the BSC protocol targets agreement on the commit point without requiring global clock synchronisation; each zone timestamps with its own clock, and the protocol's safety properties remain proof obligations under the configured timeout, finality, and recovery assumptions.

13 Jurisdictional competition

13.1 The structural argument

When the cost of moving compliance state between jurisdictions falls materially, jurisdictions may face sharper competitive pressure on regulatory quality under the adoption and switching assumptions named below. Tiebout (1956) argued that mobile consumers "vote with their feet" among local governments, and competition between jurisdictions leads to efficient provision of local public goods under strong assumptions. The Tiebout model makes three assumptions that do not hold in practice: perfect mobility (moving is costless), perfect information (consumers know what each jurisdiction offers), and no externalities (one jurisdiction's policies do not affect others).

The sovereign jurisdiction network addresses two of these three limitations directly:

Information through passports. Tiebout assumed perfect information about jurisdictional quality. In practice, comparing regulatory frameworks requires expensive human analysis. The compliance passport provides a machine-verifiable record of a jurisdiction's signed and attested compliance evaluations. An entity considering a new harbor can inspect the compliance passports of entities already operating there. The shared domain-indexed tensor provides a standard representation that makes cross-jurisdictional comparison tractable. This reduces the information asymmetry that shields low-quality jurisdictions from competitive pressure.

Mobility through corridors. Tiebout assumed costless mobility. In practice, the compliance switching cost, re-evaluation from scratch in the new jurisdiction, is a major barrier to jurisdictional mobility. Corridors reduce this cost to the re-evaluation of domains in R, plus measured corridor friction and any corridor-formation bottleneck that remains. The compliance passport travels with the entity; the history is not abandoned at the border.

Externalities remain. Tiebout assumed no externalities. The sovereign jurisdiction network does not address the externality problem by itself: jurisdictions that compete on regulatory laxity ("races to the bottom") can impose costs on other jurisdictions through regulatory arbitrage. The architecture makes regulatory arbitrage more visible (compliance passports reveal the evaluating jurisdiction's standards) and faster (corridors enable rapid movement), but it does not prevent a jurisdiction from setting low standards.

The cross-harbor meet prevents laxity from relaxing an in-scope stricter constraint on cross-harbor operations; it does not bind purely local operations or eliminate externalities. Whether increased mobility leads to laxity-seeking, quality-seeking, pooling, or segmented equilibria is a parameter-regime question that depends on entity preferences, local-operation shares, floor strength, inspection intensity, non-computational switching costs, and corridor governance.

13.2 The competition mechanism

When an entity can carry its compliance passport from jurisdiction J_1 to jurisdiction J_2 and have its evaluation history cryptographically verified and composed point-by-point, the switching cost drops to the cost of re-evaluating the domains in R (the corridor's re-evaluation set), plus measured corridor friction. For corridors with small R and low friction, this cost can be materially lower than a full restart, but it is not zero.

Jurisdictions that impose unnecessarily high compliance burdens face a sharper exit signal from entities able and willing to use corridors. Jurisdictions that produce high-quality compliance evaluations may attract entities when partner zones respect those evaluations, corridor friction is low enough, passport signals are informative enough, inspection is strong enough to deter local-only arbitrage, and non-computational preferences do not dominate. The architecture creates a measurable competitive channel; the equilibrium effect remains open. The theorem target is a finite two-stage game over corridor formation and entity mobility, with mixed-equilibrium existence, exit-signal monotonicity, meet-floor, visibility, and parameter-regime classification proved under stated assumptions rather than inferred from the architecture.

13.3 What this does not solve

The mechanism requires jurisdictions to deploy kernels and establish corridors. It does not compel them to do so. A jurisdiction that refuses to participate faces no competitive pressure from the network; its entities cannot benefit from portable compliance. The system creates incentives for participation but does not mandate it.

The system also does not solve regulatory capture, corruption, or incompetence within a jurisdiction. A corrupt regulator controlling a kernel can issue false compliance evaluations. The graduated trust model and sanctions override provide defense in depth (Section 9), but they are mitigations, not solutions. The system makes the consequences of bad regulation more visible (entities leave) but does not prevent bad regulation from occurring.

14 The endgame

At maturity, this network may consist of hundreds of sovereign kernels connected by thousands of bilateral corridors. Delegated planners can manage institutional compliance across multiple harbors simulta-

neously. The marginal cost of a new jurisdiction is a set of Lex rules and YAML definitions: data, not code. The marginal cost of a new harbor is one corridor crossing. An institution operating in 20 jurisdictions simultaneously is normal, not exceptional.

In this steady state, thousands of delegated planners operate the operation loop described in Section 7: observing tensors, computing remediation plans, submitting actions through the kernel write path, observing the updated tensor, and iterating. Each planner manages one or more entities across one or more harbors. The planners do not coordinate with each other. They do not need to. The kernel provides the coordination surface, the compliance tensor, and the write path provides the execution mechanism.

The observation corpus, the accumulated record of every compliance evaluation performed by every kernel, grows with every evaluation in every zone. Corridor intelligence propagates anonymized insights between partner zones. The rule and operation corpus expands to cover new jurisdictions, new regulatory domains, and new operation types. Each addition is a data contribution, not a code change. The fixed domain taxonomy provides the shared coordinate system; the rules and operation definitions provide the jurisdiction-specific content.

There is no network coordinator. No zone has authority over any other. There is no central network entity; only bilateral relationships between sovereign kernels. The topology emerges from these bilateral decisions. Some zones become hubs through many corridor partnerships; some remain spokes; some are isolated. The connectivity pattern mirrors jurisdictional reality: zones that have regulatory relationships establish corridors; zones that do not, do not.

The competitive dynamics described in Section 13 can operate continuously once adoption, corridor coverage, and non-computational switching conditions hold. The compliance passport makes evaluation history visible. The corridor lowers the compliance component of switching cost. Whether that produces quality competition, bottom-seeking competition, or segmented equilibria is an open political-economy question rather than an architectural theorem.

What this endgame requires is deployment: kernels standing up in real jurisdictions, corridor parameters negotiated bilaterally, entities enrolled under real regulatory authority. The architecture specified in this paper is the precondition for that deployment, not its substitute.

References

Basel Committee on Banking Supervision. *Basel III: A Global Regulatory Framework for More Resilient Banks and Banking Systems.* Bank for International Settlements, 2011.

Bindel, N., Brendel, J., Fischlin, M., Goncalves, B., and Stebila, D. (2019). "Hybrid Key Encapsulation Mechanisms and Authenticated Key Exchange." In *Post-Quantum Cryptography (PQCrypto)*, LNCS 11505, Springer.

BIS Innovation Hub, Hong Kong Monetary Authority, Bank of Thailand, Digital Currency Institute of the People's Bank of China, Central Bank of the United Arab Emirates. *Project mBridge: Connecting Economies Through CBDC.* 2022.

BIS, Reserve Bank of Australia, Bank Negara Malaysia, Monetary Authority of Singapore, South African Reserve Bank. *Project Dunbar: International Settlements Using Multi-CBDCs.* 2022.

Alaettinoglu, C., Villamizar, C., Gerich, E., Kessens, D., Meyer, D., Bates, T., Karrenberg, D., and Terpstra, M. (1999). "Routing Policy Specification Language (RPSL)." RFC 2622, Internet Engineering Task Force.

Blaze, M., Feigenbaum, J., and Lacy, J. (1996). "Decentralized Trust Management." In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*.

Blaze, M., Feigenbaum, J., Ioannidis, J., and Keromytis, A. D. (1999). "The KeyNote Trust-Management System, Version 2." RFC 2704, Internet Engineering Task Force.

Bünz, B., Kiffer, L., Luu, L., and Zamani, M. (2020). "FlyClient: Super-Light Clients for Cryptocurrencies." In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*.

Buterin, V. (2014). *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform.* White paper.

Čech, E. (1932). "Théorie générale de l'homologie dans un espace quelconque." *Fundamenta Mathematicae*, 19, 149-183.

Dwork, C., Lynch, N. A., and Stockmeyer, L. (1988). "Consensus in the Presence of Partial Synchrony." *Journal of the ACM*, 35(2), 288-323.

Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., and Ylonen, T. (1999). "SPKI Certificate Theory." RFC 2693, Internet Engineering Task Force.

Estonian Information System Authority. *X-Road: Data Exchange Layer.* `https://x-road.global/`.

European Parliament. Regulation (EU) No 910/2014 on Electronic Identification and Trust Services for Electronic Transactions in the Internal Market (eIDAS). Official Journal of the European Union, 2014.

European Parliament. Directive 2014/65/EU on Markets in Financial Instruments (MiFID II). Official Journal of the European Union, 2014.

Fischer, M. J., Lynch, N. A., and Paterson, M. S. (1985). "Impossibility of Distributed Consensus with One Faulty Process." *Journal of the ACM*, 32(2), 374-382.

Gray, J. and Lamport, L. (2006). "Consensus on Transaction Commit." *ACM Transactions on Database Systems*, 31(1), 133-160.

Grothendieck, A. (1960-61). *Revêtements Étales et Groupe Fondamental (SGA 1).* Séminaire de Géométrie Algébrique du Bois-Marie, IHÉS.

Groth, J. (2016). "On the Size of Pairing-Based Non-Interactive Arguments." In *EUROCRYPT*, LNCS 9666, Springer.

Haber, S. and Stornetta, W. S. (1991). "How to Time-Stamp a Digital Document." *Journal of Cryptology*, 3(2), 99-111.

Haeberlen, A., Kouznetsov, P., and Druschel, P. (2007). "PeerReview: Practical Accountability for Distributed Systems." In *Proceedings of the ACM SIGOPS Symposium on Operating Systems Principles (SOSP)*.

Klein, G., Elphinstone, K., Heiser, G., et al. (2009). "seL4: Formal Verification of an OS Kernel." In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles (SOSP)*.

Kwon, J. and Buchman, E. (2016). *Cosmos: A Network of Distributed Ledgers.* White paper.

Lamport, L. (1978). "Time, Clocks, and the Ordering of Events in a Distributed System." *Communications of the ACM*, 21(7), 558-565.

Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System.*

OASIS. *eXtensible Access Control Markup Language (XACML), Version 2.0.* OASIS Standard, 2005.

Odlyzko, A. and Tilly, B. (2005). *A Refutation of Metcalfe's Law and a Better Estimate for the Value of Networks and Network Interconnections.* Preprint, University of Minnesota.

Rekhter, Y., Li, T., and Hares, S. (2006). "A Border Gateway Protocol 4 (BGP-4)." RFC 4271, Internet Engineering Task Force.

Skeen, D. (1981). "Nonblocking Commit Protocols." In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 133-142.

Society for Worldwide Interbank Financial Telecommunication. *SWIFT Standards.* `https://www.swift.com/standards`.

Tiebout, C. (1956). "A Pure Theory of Local Expenditures." *Journal of Political Economy*, 64(5), 416-424.

Todd, P. (2012). *Merkle Mountain Ranges.* OpenTimestamps documentation.

Wood, G. (2014). *Ethereum: A Secure Decentralised Generalised Transaction Ledger.* Yellow paper.