Rings: An Algebraic Structured Peer-to-Peer Network for Decentralized Identity and Resource Ownership

Ryan J. Kung

***Abstract*—Rings is an implemented algebraic structured peer-to-peer network for decentralized identity, resource ownership, transport, storage, privacy, and application protocols. Application servers, DNS authorities, and relay providers in conventional Internet deployments frequently become trusted bottlenecks for naming, routing, and user state. Rings instead models identities and resources as algebraic objects over a CorrectChord circular overlay $\mathbb{Z}/2^{160}\mathbb{Z}$, pairing browser WebAssembly and native runtimes over WebRTC/SCTP while keeping all cryptographic operations on carriers disjoint from the routing identifier ring. This paper unifies the core algebraic design with six production subsystems implemented in Rings 1.0: range-proved finger convergence, non-transitive WebRTC Chord routing and single-hop bridge delegation, conflict-free replicated data types (CRDTs) with HLC removal stability and Merkle data provenance, crash-durable local stores and age-gated replicated relay custody, cleanest-first slot accounting across shared node tables, and Sphinx/AEZ-v5 onion Kleisli pipelines bridging counterparty-signed service receipts to DRanking 2.0. We evaluate the Rust implementation empirically on real hardware and summarize exhaustive Stateright/BFS verification across $9,496,521$ reachable finite states (9.50×10^6) and 54 refuted safety mutants. Core object:** $\mathsf{Rings} = (R, \mathcal{R}_N, \mathbf{Set}/R, \Omega_N, \mathsf{T}_A^{\mathcal{E}}, \Pi, \mathbb{G}, \mathcal{X})$ **with** $R = \mathbb{Z}/2^{160}\mathbb{Z}$**,** $\Omega_N(X, H) = (X, \mathrm{owner}_N \circ H)$**, and** $\mathsf{CryptoCarrier} \cap R = \varnothing$**.**

***Index Terms*—Structured peer-to-peer networks, Chord, WebRTC, non-transitive routing, CRDTs, crash consistency, Sphinx onion routing, AEZ v5, formal verification, Stateright.**

I. Introduction

Centralized application platforms simplify deployment, but they also concentrate identity, naming, storage, and traffic control at service operators. Cloud and platform centralization therefore becomes a privacy, availability, and autonomy risk for users who need direct communication and durable control over their data [1]–[4]. The Rings project is motivated by this observation: a sovereign network should let a participant own an identity, establish a session, route to another identity or resource, and run application protocols without turning a single application server into the root of authority.

Rings is implemented as a network, but this paper is organized around the algebraic structure that makes the implementation coherent. It states the ideal carrier, ownership, transition, durability, admission, and privacy laws that the Rings 1.0 implementation realizes. The older intuition was that Decentralized Identifiers (DIDs) directly form a finite ring suitable for all routing and cryptographic operations. The refined model is more precise: Chord identifiers form an additive circular carrier for placement, interval order, replica selection, and finger targets; threshold schemes, signatures, encryption, wide-block onion encipherment, and zero-knowledge protocols use their own fields or groups.

The resulting design has three practical consequences. First, DIDs and virtual DIDs (VIDs) share the same 160-bit placement surface, so identities, mailboxes, service descriptors, chunks, and application rendezvous points can be routed by one owner function. Second, Rings builds on Chord and CorrectChord rather than replacing the overlay literature: successor lists, multi-predecessor relations, join, rectify, stabilize, range-proved finger convergence, and passive 1-hop gap-bridge delegation are the maintenance substrate on which Rings semantics are layered [1], [5]–[7]. Third, state transitions return typed effects instead of directly performing network IO, which allows the overlay, storage, durability, load-accounting, and onion-pipeline state machines to be reasoned about purely, model-checked with Stateright [8], and then interpreted by browser or native runtimes.

II. Problem Statement and Design Goals

Rings addresses a narrow but recurring gap between existing decentralized systems. Kademlia-derived systems provide scalable lookup, and contemporary frameworks such as Veilid [9] add Rust/WebAssembly node runtimes and private safety routes over Kademlia, yet the XOR metric does not by itself give Rings the circular ownership law needed for successor lists, replicated ownership, and interval-based placement. Relay networks and mixnets improve metadata resistance, but they are not structured P2P ownership systems. Blockchains provide global ordering, but embedding consensus in every message or storage operation would turn Rings into a ledger rather than an algebraic structured peer-to-peer substrate.

The problem is therefore to specify a network object

$$\mathsf{Rings}$$

that simultaneously supports:

1) self-certifying identity and delegated sessions;
2) structured lookup, range-proved finger convergence, and non-transitive WebRTC routing;
3) browser-native and native transport and crash-durable storage surfaces;

Ryan J. Kung is with the Rings Network project (e-mail: ryankung@ieee.org; ORCID: 0009-0000-3026-1111). Manuscript created February 8, 2023; consolidated and empirically evaluated for Rings 1.0 in November 2026. Source code and formal-verification artifacts: https://github.com/RingsNetwork/rings.

4) CRDT storage with tombstone stability, per-writer cryptographic provenance, and replicated relay custody;
5) Sybil-resistant slot accounting, credit reclaim, and bounded load under free identities;
6) Sphinx/AEZ-v5 onion Kleisli pipelines and counterparty-signed service receipts.

This paper makes the following contributions. It presents Rings as an algebraic structured peer-to-peer network, modeling the identifier space as the additive cyclic group $R = \mathbb{Z}/2^{160}\mathbb{Z}$ and lifting resource placement into $\mathbf{Set}/R$. It specifies range-proved finger convergence and bounded multi-predecessor non-transitive Chord routing over restricted WebRTC graphs (Section IV). It establishes the removal-stability law, HLC data provenance, factored carrier eviction, and signed lease heartbeats for replicated CRDT storage (Section V). It defines the crash-meaning authority lattice $C \times M$, presence markers, per-platform crash durability, group commit, link-local replay windows, and age-gated replicated relay custody (Section VI). It proves fair slot accounting across five shared node tables, credit reclaim, and single-charge chunk reassembly under free identities (Section VII). It replaces legacy relay framing with client-sealed Sphinx/AEZ-v5 onion Kleisli pipelines (Section VIII), bridges 1.0 service receipts into the companion DRanking 2.0 protocol [10] (Section IX), and presents empirical micro- and macro-benchmarks together with exhaustive Stateright formal-verification metrics across all Rings 1.0 subsystems (Section X).

III. Algebraic System Model, Threat Model, and Effects

A. Carrier, System Scope, and Threat Model

Assumption 1 (System scope)**.**

$$\begin{aligned}
\mathsf{fault} &= \mathsf{fail\text{-}stop}\ (\textit{overlay}) \mid \mathsf{crash}\ (\textit{local}),\\
\mathsf{auth}(m,\sigma,k) &= V(k,\sigma,m),\\
\mathsf{net} &= \mathsf{async},\\
\mathsf{clock} &= |\,\mathrm{clock}_p - \mathrm{clock}_q\,| \le \sigma,\\
\mathsf{io} &= I_X : A_X^* \to \mathsf{IO},\\
\mathsf{consensus} &\notin \mathsf{Overlay},\\
\mathsf{order} &= \mathsf{SidecarWitness},\\
\mathsf{crypto_ops} &\cap \mathsf{route_ops} = \varnothing.
\end{aligned}$$

Clock tolerance σ bounds skew across synchronized honest nodes.

Unified Threat Model: Rings 1.0 operates under four explicit adversarial and fault assumptions across its layers while deliberately scoping out global traffic analysis without cover or majority ring capture:

1) *Link and Network Adversary:* A passive observer monitors arbitrary underlay links and WebRTC NAT paths, while an active adversary delays, duplicates, reorders, drops, or replays valid signed control/onion messages on any link it controls. Asynchronous replay and credit exhaustion are defeated at every node boundary by durable/link-local replay windows, generation-tagged credit reclaim, and sliced-Bloom filters.
2) *Colluding Onion Relays:* Up to $H-1$ relays on an H-hop client loop may be honest-but-curious or actively tag, shrink, or relabel packets without breaking payload confidentiality on any segment whose endpoints are honest. For sender–destination unlinkability, header/payload cryptographic unlinkability on the entry segment requires at least one honest relay in $\{r_{0,1}, r_{0,2}\}$ separating $\diamond$ from h_1, while resistance to deterministic inter-hop timing correlation across $r_{0,2}$ additionally assumes non-collusion between guard $g = r_{0,1}$ and symbol hop h_1 unless cover pacing masks per-hop release delays $L(f_i)$; Sphinx header MACs, uniform 205 B layers, and AEZ v5 wide-block encryption (Section VIII) prevent bitwise correlation and detect single-attempt tagging with probability $\ge 1 - 2^{-128}$.
3) *Byzantine Writers and Sybil Rotation:* Since minting a DID has zero cost ($\mathsf{cost}(\mathsf{did}) = 0$), an adversary may forge unsigned messages, spawn arbitrarily many DIDs, rotate identities rapidly, or inject stale DHT deltas. Unauthenticated mutations and skewed HLC timestamps are rejected by `DataProvenance`; memory/table exhaustion is bounded by cleanest-first connection slot floors (Section VII); and Sybil endorsement inflation is bounded by dual-signed `ServiceReceipts` in DRanking 2.0 (Section IX).
4) *Local Storage Crashes:* Native file systems and browser IndexedDB backends may crash at arbitrary points between write, rename, and barrier sync; `RecordAuthority` ($C \times M$) enforces platform-native barriers (`F_FULLFSYNC`, `MOVEFILE_WRITE_THROUGH`, `fsync`, or strict IndexedDB) and witnessed presence markers (Section VI).
5) *Out of Scope:* Rings 1.0 does not prevent end-to-end timing correlation by a global passive adversary observing both a client's guard link and its world exit when real-time interactive latency prohibits heavy cover padding, does not defend against $> 50\%$ Sybil capture of the global overlay without DRanking 2.0 admission filtering, and trusts local client execution hosts/kernels.

Definition 1 (Identifier carrier)**.**

$$\begin{aligned}
M &= 2^{160},\\
R &= \mathbb{Z}/M\mathbb{Z},\\
\delta_a(x) &= (x-a) \bmod M,\\
(a,b]_R &= \{x \in R : 0 < \delta_a(x) \le \delta_a(b)\},\\
\mathsf{RouteOps}_R &= \{0, +, -, \delta_a, (_,_]_R\},\\
\mathsf{Mult}_R &\notin \mathsf{RouteOps}_R.
\end{aligned}$$

Note that $(a,a]_R = \varnothing$ since $\delta_a(a) = 0$, so self-ownership is handled prior to interval tests.

Definition 2 (Live topology)**.**

$$\begin{aligned}
N &\subset R, \quad |N| \ge 2,\\
\mathrm{owner}_N(k) &= \arg\min_{u \in N} \delta_k(u),\\
\mathrm{pred}_N(n) &= \arg\min_{p \in N \setminus \{n\}} \delta_p(n)\\
&= \arg\max_{p \in N \setminus \{n\}} \delta_n(p),\\
\mathrm{Succ}_N^q(n) &= \mathrm{take}_q\big(\mathrm{sort}_{\delta_n}(N \setminus \{n\})\big).
\end{aligned}$$

Each live node maps to a unique identifier in $N \subset R$ (hence $\mathrm{owner}_N(k)$ and $\mathrm{pred}_N(n)$ have unique minimizers); bootstrap singletons ($|N| = 1$) resolve $\mathrm{owner}_N(k) = n$ trivially.

Assumption 2 (Correct-Chord stable base)**.**

$$\begin{aligned} r &= |\operatorname{Succ}_N^r(n)|, \\ \mathsf{Steady}(N) &\Rightarrow |N| \geq r+1, \\ |N| < r+1 &\Rightarrow \mathsf{Init}(N), \\ \mathsf{Maintain} &= \{\mathsf{join}, \mathsf{rectify}\} \\ &\quad \cup \{\mathsf{stabilize}\}_{CorrectChord}. \end{aligned}$$

This is the CorrectChord stable base of Zave [6], [11].

Definition 3 (Affine replica set)**.**

$$\begin{aligned} \rho &\in \mathbb{N}_{>0}, \\ \operatorname{rot}_i^\rho(k) &= k + \lfloor M \cdot i/\rho \rfloor, \\ \operatorname{Rep}_N^\rho(k) &= \{\operatorname{owner}_N(\operatorname{rot}_i^\rho(k)) : 0 \leq i < \rho\}. \end{aligned}$$

Storage redundancy uses affine rotations of the entry key ($\rho = 6$ by default for data topics). Successor lists remain the CorrectChord safety root for topology maintenance.

Definition 4 (Rings overlay object)**.**

$$\mathcal{R}_N = (R, N, \operatorname{owner}_N, \operatorname{pred}_N, \operatorname{Succ}_N^r, \operatorname{Rep}_N^\rho, F, E),$$

$$\begin{aligned} F &\subseteq N \times \mathbb{N} \times N, \\ E &= \textstyle\prod_{v \in VID} E_v, \\ \mathsf{SafetyRoot}(\mathcal{R}_N) &= (\operatorname{pred}_N, \operatorname{Succ}_N^r), \\ \mathsf{PerfWitness}(\mathcal{R}_N) &= F. \end{aligned}$$

Proposition 1 (Rotation equivariance)**.**

$$\begin{aligned} t \in R, \quad N + t = \{n + t : n \in N\} \Rightarrow \\ \operatorname{owner}_{N+t}(k+t) = \operatorname{owner}_N(k) + t. \end{aligned}$$

Proof.

$$\begin{aligned} \forall n \in N. \quad \delta_{k+t}(n+t) &= ((n+t) - (k+t)) \bmod M \\ &= \delta_k(n). \end{aligned}$$

□

Constraint 1 (Carrier separation)**.**

$$\begin{aligned} \mathsf{PlacementOps} &= \{0, +, -, \delta_a, (_, _]_R\} \\ &\quad \cup \{\operatorname{owner}_N, \operatorname{Rep}_N^\rho\}, \\ \mathsf{CryptoOps} &= \{\times, ^{-1}, \cdot_{scalar}, \mathsf{interp}\} \\ &\quad \cup \{\mathsf{pair}, \mathsf{subgroup}, \mathsf{AEZ}\}, \\ \mathsf{PlacementOps} &\cap \mathsf{CryptoOps} = \varnothing, \\ \mathsf{CryptoCarrier} &\in \{\mathbb{F}_q, \mathbb{G}, \mathbb{Z}_q, \{0,1\}^*\}. \end{aligned}$$

Table I summarizes how Rings 1.0 assigns each protocol object to its disjoint algebraic carrier and cryptographic domain separator.

B. Architecture as Interfaces

Rings 1.0 factors its architecture into six composable interfaces (Figure 1), separating identity authentication, transport signaling, overlay ownership, and capability-confined execution.

Definition 5 (Layer tuple)**.**

$$\mathcal{A} = (I, T, O, X, P, L)$$

Table I
CARRIER BOUNDARIES AND DOMAIN TAGS IN RINGS 1.0

Object	Carrier or law
Chord IDs	additive cyclic group $R = \mathbb{Z}/2^{160}\mathbb{Z}$
DID / Session	verifier over `rings-delegation-v1`
VID placement	$H_\nu : X_\nu \to R$ then owner map Ω_N
Storage merge	factored CRDT semilattice $\mathsf{E}(x) \sqcup \mathsf{R}(x)$
Data provenance	Merkle batch `rings-core:data-entry:mutation`
Lease descriptors	`rings-node:{online-node,service,mail}-descriptor`, `rings-turn:service-descriptor`
Store authority	crash $\times$ meaning lattice $C \times M$
Load accounting	slot floor f_T + fair pool $\mathcal{P}_T$ over $\mathcal{R}$ slots
Onion pipeline	Kleisli category $\mathbf{Kl}(M)$ with Sphinx + AEZ v5
Service receipts	`rings-core:service-receipt:provisional` $\to$ DRanking `rings-service-receipt-v1`
Effects	writer monad $\mathsf{T}_A^{\mathcal{E}}$ over action lists

$$\begin{array}{c|c} I & (DID, \Pi, \text{Session}) \\ T & (C, A_T, \mathsf{offer}, \mathsf{answer}, \mathsf{send}, \mathsf{recv}) \\ O & \mathcal{R}_N \\ X & (S_X, A_X, \mathsf{cap}, I_X) \\ P & \{\mathcal{P}_\nu\}_{\nu \in Namespace} \\ L & \mathsf{App} \circ P \end{array}$$

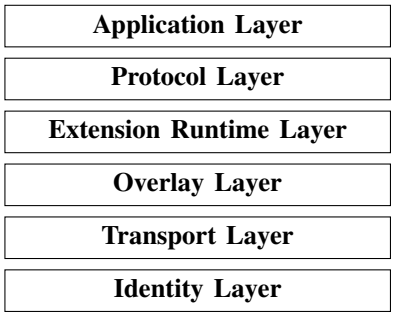

Figure 1. Layers of Rings Network

Definition 6 (Identity proof)**.**

$$\Pi = (K, \Sigma, V, D, \mathsf{enc}, \mathsf{ttl})$$

$$\begin{aligned} V &: K \times \Sigma \times B^* \to \{\top, \bot\}, \\ D &: K \to R, \\ \mathsf{enc} &: \Sigma \to B^*, \\ \mathsf{ttl} &: \Sigma \to Time. \end{aligned}$$

Invariant 1 (Session authority)**.**

$$d \in X_D, \qquad s \in K, \qquad D(k_d) = H_D(d).$$

$$\begin{aligned} \operatorname{Session}(d, s, e) \Rightarrow V\big(k_d, \sigma_{d \to s}, \\ \texttt{rings-delegation-v1}\|d\|s\|e\big) \\ = \top \wedge now < e, \end{aligned}$$

checked at each authenticated hop via $\mathsf{HopCheck}(d, s, e) \equiv \operatorname{Session}(d, s, e)$.

Definition 7 (Transport interface)**.**

$$T = (C, \mathsf{offer}, \mathsf{answer}, \mathsf{send}, \mathsf{recv})$$

$$A_T = \{\mathsf{Offer}, \mathsf{Answer}, \mathsf{Ice}, \mathsf{Open}, \mathsf{Send}, \mathsf{Recv}, \mathsf{Close}\}.$$

$$I_T : A_T^* \to \mathsf{IO}_{WebRTC} \cup \mathsf{IO}_{Native}.$$

Figures 2 and 3 contrast standard multi-round-trip trickle-ICE negotiation against Rings' 1-RTT overlay SDP/ICE signaling exchange (which front-loads candidate gathering into one Offer/Answer DHT exchange prior to direct-path ICE connectivity checks and DTLS/SCTP setup).

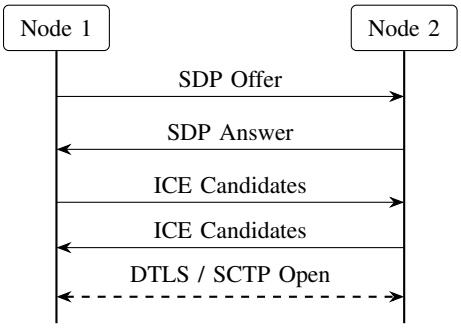

Figure 2. Standard trickle-ICE / SDP signaling exchange

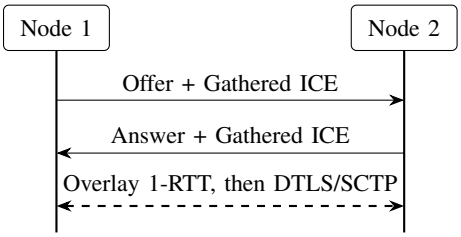

Figure 3. Rings single round-trip overlay SDP/ICE signaling exchange

Definition 8 (Runtime interpreter)**.**

$$\mathcal{X} = (S_X, A_X, \mathsf{cap}, I_X)$$

$$\begin{aligned} \mathsf{cap} &\subseteq A_X, \\ I_X &: A_X^* \to \mathsf{IO}, \\ I_X(\epsilon) &= \mathsf{Noop}, \\ I_X(\alpha \cdot \beta) &= I_X(\alpha); I_X(\beta). \end{aligned}$$

Figure 4 illustrates how the browser host FFI enforces capability check cap *between network/timer events and pure core steps.*

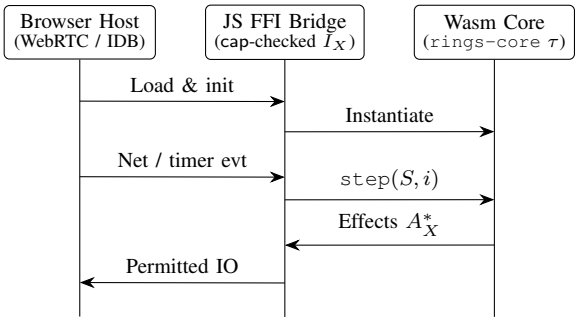

Figure 4. Wasm runtime confinement and capability-checked effect boundary

Obligation 1 (Runtime confinement)**.**

$$\begin{aligned} \tau &: S \times I \to \mathcal{E} + (S \times O \times A_X^*), \\ \mathrm{set}(\pi_A(\tau(s,i))) &\subseteq \mathsf{cap}, \\ a \notin \mathsf{cap} &\Rightarrow I_X(a) = \mathsf{Reject}(a). \end{aligned}$$

Capability checks apply atomically per step(S, i) *transition on the host FFI boundary (aborting state commit and rejecting host IO if any emitted action falls outside* cap*).*

C. Category and Effect Monad

We summarize namespace placement and pure effect emission using lightweight slice-category and writer-monad notation as a uniform interface vocabulary for the pure transition reducers verified in Section X, rather than relying on deep categorical machinery in later protocol proofs.

Definition 9 (Namespace object)**.**

$$\begin{aligned} \nu &\in \{D, V, M, H, U, Q, W\}, \\ H_\nu &: X_\nu \to R, \\ (X_\nu, H_\nu) &\in \mathbf{Set}/R, \end{aligned}$$

indexing DID, VID, Mailbox, Hidden Service, User/Relay, Queue/Quota, and World-Exit namespaces.

Definition 10 (Placement functor)**.**

$$\begin{aligned} \Omega_N &: \mathbf{Set}/R \to \mathbf{Set}/N, \\ (X, H) &\mapsto (X, \mathrm{owner}_N \circ H). \end{aligned}$$

$$P_\nu^N = \mathrm{owner}_N \circ H_\nu : X_\nu \to N.$$

Proposition 2 (Placement functoriality)**.**

$$f : X_\nu \to X_\mu, \qquad H_\mu \circ f = H_\nu \implies P_\mu^N \circ f = P_\nu^N.$$

Proof. $\mathrm{owner}_N \circ H_\mu \circ f = \mathrm{owner}_N \circ H_\nu$. □

Definition 11 (Writer effect monad)**.**

$$\begin{aligned} A^* &= \mathsf{FreeMonoid}(A), \\ \mathsf{T}_A(X) &= X \times A^*, \\ \eta(x) &= (x, \epsilon), \\ (x, \alpha) \gg\!= f &= \mathsf{let}\ f(x) = (y, \beta)\ \mathsf{in}\ (y, \alpha \cdot \beta), \\ f^\star(x, \alpha) &= (x, \alpha) \gg\!= f. \end{aligned}$$

Proposition 3 (Effect associativity)**.**

$$(h^\star \circ g^\star) \circ f^\star = h^\star \circ (g^\star \circ f^\star).$$

Proof. $((\alpha \cdot \beta) \cdot \gamma) = (\alpha \cdot (\beta \cdot \gamma))$. □

Definition 12 (Typed failure stack)**.**

$$\mathsf{T}_A^{\mathcal{E}}(X) = \mathcal{E} + (X \times A^*).$$

$$\begin{aligned} \pi_A(e \in \mathcal{E}) &= \epsilon, \\ \pi_A(x, \alpha) &= \alpha, \\ \mathsf{RecoverableReport} &\in A. \end{aligned}$$

Here $\mathcal{E}$ captures hard transition aborts (discarding emitted actions via $\pi_A(e) = \epsilon$), whereas recoverable failures emit RecoverableReport $\in A$ *along the normal $(X \times A^*)$ branch. For $(\sigma_S \times \sigma_O) : S_\nu \times O_\nu \to S_\mu \times O_\mu$, $\mathsf{T}_{\sigma_A}^{\sigma_\mathcal{E}}(g) = \sigma_\mathcal{E} + (g \times \sigma_A^*)$.*

Definition 13 (Protocol object and morphism)**.**

$$\mathcal{P}_\nu = (X_\nu, S_\nu, I_\nu, O_\nu, A_\nu, \mathcal{E}_\nu, H_\nu, \tau_\nu),$$
$$\tau_\nu : S_\nu \times I_\nu \to \mathsf{T}^{\mathcal{E}_\nu}_{A_\nu}(S_\nu \times O_\nu).$$

A protocol morphism $\Sigma : \mathcal{P}_\nu \to \mathcal{P}_\mu = (\sigma_X, \sigma_S, \sigma_I, \sigma_O, \sigma_\mathcal{E}, \sigma^*_A)$ *satisfies* $H_\mu \circ \sigma_X = H_\nu$ *and* $\mathsf{T}^{\sigma_\mathcal{E}}_{\sigma_A}(\sigma_S \times \sigma_O) \circ \tau_\nu = \tau_\mu \circ (\sigma_S \times \sigma_I)$.

IV. Chord Implementation — PeerRing and Overlay Routing

A. Base Chord and CorrectChord Maintenance

$$\begin{aligned} \mathsf{Base} &= \mathsf{Chord}, \\ \mathsf{Maintenance} &= \mathsf{CorrectChord}, \\ \mathsf{Object} &= \mathcal{R}_N, \\ \mathsf{Witnesses} &= \{\text{Algorithms } 1,\ 2,\ 3\}. \end{aligned}$$

This section fixes Chord as the base lookup protocol and CorrectChord as the ring-safety maintenance law [1], [6] (Figures 5 and 6), then extends the pure overlay state machine (`PeerRing`) with range-proved finger convergence and non-transitive WebRTC routing.

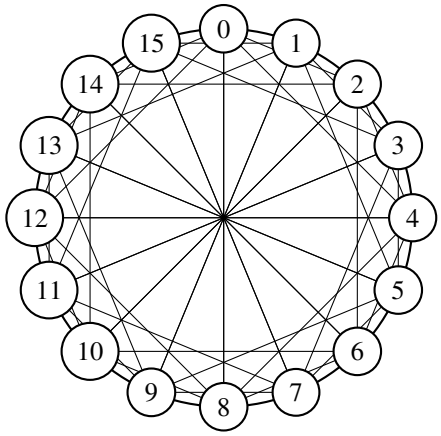

Figure 5. Chord identifier ring and finger chords

Definition 14 (Local node state and finger targets)**.**

$$S_n = (p_n, L_n, F_n, E_n), \qquad \mathrm{target}_i(n) = n + 2^i,$$
$$\mathrm{finger}_i(n) = \mathrm{owner}_N(\mathrm{target}_i(n)),$$

where $p_n \in N \cup \{\bot\}$, $L_n \in N^{\le r}$, $F_n : [0, L) \to N \cup \{\bot\}$ *(with* $F_n(i) \in \{\bot, \mathrm{finger}_i(n)\}$ *upon convergence), and* $E_n \subseteq E$.

Invariant 2 (Ring fixpoint and lookup owner preservation)**.**

$$\mathrm{Inv}_{ring}(N, S) \equiv \forall n \in N.\ p_n = \mathrm{pred}_N(n) \\ \wedge L_n = \mathrm{Succ}^r_N(n),$$

$$\mathsf{Steady}(N) \wedge \mathrm{Inv}_{ring}(N, S) \Rightarrow \mathrm{Lookup}(S, k) \Downarrow \mathrm{owner}_N(k).$$

Remark 1 (Static-Ring vs. Bounded-Churn Lookup Scope)**.** *Invariant 2 guarantees exact owner resolution and* $O(\log |V|)$*-hop lookup [1], [5], [6] on a steady, finger-converged ring* $\mathsf{Steady}(N)$*. Under transient churn or before finger convergence completes, some fingers remain sparse or stale; payload delivery then preserves loop-free correctness within* $\le 2|V|$ *hops via bridge-delegated delivery (Theorem 1).*

Algorithm 1 DHT Join Transitions

function BeginJoin($n, known$)
  $pred[n] \leftarrow none$; $a \leftarrow FindSuccessor(n.did)$
  return $Remote(known, a)$
function InstallSuccessor(n, s)
  $succ[n] \leftarrow take(r, [s])$
  return $Remote(s, QuerySuccList(s))$
function InstallSuccList($n, s, list$)
  $succ[n] \leftarrow take(r, s :: list)$
  if $head(succ[n]) \neq none$ **then**
    $h \leftarrow head(succ[n])$
    return $Multi(Notify(h, n.did), SyncStorage(h))$
  else
    return $Noop$

Algorithm 2 DHT Lookup Algorithm

function Lookup(key, n)
  if $key = n.did$ **then**
    return $Local(n)$
  $s \leftarrow head(succ[n])$
  if $s = none$ **then**
    return $Repair(QuerySuccList(n))$
  if $key \in (n.did, s]$ **then**
    return $Local(s)$ ▷ Delegated by Law 5 (B2) on bridged gaps
  $next \leftarrow closest_preceding_node(n, key)$ ▷ Searches $F_n \cup L_n$
  if $next = none$ **then**
    return $Repair(QuerySuccList(s))$
  else
    return $Remote(next, FindSuccessor(key))$

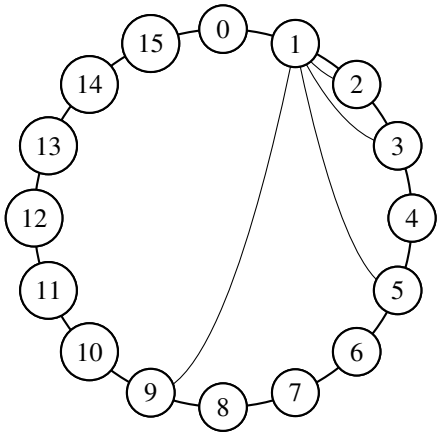

Figure 6. Chord exponential finger intervals from node 1

Algorithm 3 DHT Stabilization

```
1: procedure STABILIZE(n, topo)
2:     actions ← []; c ← topo.pred; s ← head(succ[n])
3:     if c ≠ none and c ∈ (n.did, s) then
4:         succ[n] ← insert(succ[n], c)
5:         actions ← actions :: FindSuccessor(c)
6:     succ[n] ← take(r, merge(succ[n], topo.succ))
7:     if head(succ[n]) ≠ none then
8:         h ← head(succ[n])
9:         actions ← actions :: Notify(h, n.did)
10:    return Multi(actions)
```

B. Range-Proved Finger Convergence

Classical Chord refreshes one random finger slot per stabilization period, requiring $\Theta(m)$ periods ($m = 160$) to converge and failing to distinguish unverified neighbor hints from proved entries [5]. Because Rings nodes span browser and WAN peers whose asynchronous NAT-traversing WebRTC handshakes are orders of magnitude costlier than local memory transitions, `rings-core` (`dht::finger::convergence`) specifies finger maintenance as a pure transition $\text{step} : (S, \text{Event}) \to (S, \text{Actions})$ built on the Chord Range Lemma and four operational disciplines.

Lemma 1 (Chord Range Lemma [1])**.** *With half-open interval $[x, y)_R = \{z \in R : 0 \le \delta_x(z) < \delta_x(y)\}$ ($[x, x)_R = \varnothing$), for any nonempty live membership view $V \subseteq R$ and target $x \in R$,*

$$\text{succ}_V(x) = y \iff y \in V \ \wedge\ [x, y)_R \cap V = \varnothing.$$

Specifically, for node $n \in V$ and slot $i \in [0, L)$ ($L \le 160$) with target $t_i = n + 2^i$, let $s = \text{succ}_V(t_i)$ and $d = \delta_n(s)$. If $s = n$, then $\text{succ}_V(t_j) = n$ for all $j \in [i, L)$; otherwise $2^i \le d$ and

$$\forall j \ge i : \quad 2^j \le d \implies \text{succ}_V(t_j) = s.$$

Proof. By definition, $\text{succ}_V(t_i) = s$ implies $s \in V$ and $[t_i, s)_R \cap V = \varnothing$. If $s = n$, every t_j with $i \le j < L$ lies in $[t_i, n)_R$ (since $2^i \le 2^j < 2^{160}$), so $[t_j, n)_R \cap V = \varnothing$ and $\text{succ}_V(t_j) = n$. Otherwise $2^i \le d$, and for every $j \ge i$ with $2^j \le d$ we have $t_j \in [t_i, s]_R$, so $[t_j, s)_R \subseteq [t_i, s)_R$ is empty in V and $\text{succ}_V(t_j) = s$. □

Definition 15 (Range proof)**.** *A lookup for slot i answered by s yields a range proof $\pi = (i, \text{id}, e_\pi, s, \text{end})$, where* id *is the lookup UUID [12], e_π is the evidence epoch at issue, and*

$$\text{end}(i, s) = \begin{cases} L - 1, & s = n, \\ \min\big(L - 1, \lfloor \log_2 \delta_n(s) \rfloor\big), & 2^i \le \delta_n(s), \\ \mathsf{Invalid}, & \textit{otherwise.} \end{cases}$$

Proof π covers the contiguous slot interval $[i, \text{end}]$.

Corollary 1 (Lookups per pass)**.** *If the true successor values partition the L slots into m_{runs} maximal runs of equal successor, a complete convergence pass requires at most $m_{\text{runs}} \le \min(L, |V|)$ routed lookups; under uniform identifier placement, $\mathbb{E}[m_{\text{runs}}] = O(\log |V|)$.*

Proof. Each answered lookup for slot i yields $\pi = (i, \text{id}, e_\pi, s, \text{end})$ covering the maximal run $[i, \text{end}]$ of identical successor s by Lemma 1, advancing cursor $c \leftarrow \text{end} + 1$ past the entire run. Because there are at most L slots and each distinct successor begins at most one maximal run before wrapping past n, $m_{\text{runs}} \le \min(L, |V|)$. Under uniform random placement of $|V|$ DIDs on $R = \mathbb{Z}/2^{160}\mathbb{Z}$, expected inter-node spacing is $2^{160}/|V|$, so slots with $2^j \lesssim 2^{160}/|V|$ share $\text{head}(n)$ while only the top $O(\log |V|)$ slots span distinct sub-intervals, giving $\mathbb{E}[m_{\text{runs}}] = O(\log |V|)$. □

Lemma 2 (Local Range Confirmation)**.** *Let $h = \text{head}(n)$ be the first clockwise successor of n. When n receives an authenticated stabilization report from h echoing n's active stabilization token and stating $\text{pred}(h) = n$, then*

$$\forall j : \quad 2^j \le \delta_n(h) \implies \text{succ}_V(t_j) = h.$$

Applying local confirmation sets $\mathsf{ver}_j \leftarrow \top$ for all $j \le \min(L - 1, \lfloor \log_2 \delta_n(h) \rfloor)$ with zero routed network lookups; if an active attempt's slot lies in that range, it is superseded via PROGRESS$(\min(L - 1, \lfloor \log_2 \delta_n(h) \rfloor))$.

Proof. Because h is n's live successor head and h itself attests on n's fresh stabilization correlation token that $\text{pred}(h) = n$, the half-open arc $(n, h)_R$ contains no member of V observable between n and h. For every finger index $j \ge 0$ with $2^j \le \delta_n(h)$, the finger target $t_j = n + 2^j$ lies in $(n, h]_R$, so $[t_j, h)_R \subseteq (n, h)_R$ is empty in V and $\text{succ}_V(t_j) = h$ by Lemma 1. Setting $\mathsf{ver}_j \leftarrow \top$ for $0 \le j \le \min(L - 1, \lfloor \log_2 \delta_n(h) \rfloor)$ therefore proves the prefix interval using only the existing stabilization RPC (PROGRESS avoids issuing any `FindSuccessor` lookup for those slots). □

Definition 16 (Monotone Evidence Epochs)**.** *Each node n maintains an evidence state $(e_n, (\mathsf{chg}_j, \mathsf{ver}_j)_{j<L})$, where $e_n \in \mathbb{N}$ is the node's monotone topology epoch [13], $\mathsf{chg}_j \in \mathbb{N}$ is the epoch at which hint hint_j last changed, and $\mathsf{ver}_j \in \{\bot, \top\}$ records whether slot j holds a range-proved peer.*

Law 1 (Hint change, slot-wise admission, and discontinuity)**.**

1) Hint change: *For any unverified topology/hint transition $H \to H'$ with changed slots $C = \{j : H_j \ne H'_j\} \ne \varnothing$, advance $e \leftarrow e + 1$ and set $(\mathsf{chg}_j, \mathsf{ver}_j) \leftarrow (e, \bot)$ for all $j \in C$.*
2) Slot-wise admission: *Proof $\pi = (i, \text{id}, e_\pi, s, \text{end})$ admits slot set $A(\pi) = \{j \in [i, \text{end}] : \mathsf{chg}_j \le e_\pi\}$. If $A(\pi) = \varnothing$, π is rejected as* Stale*; otherwise, for every $j \in A(\pi)$, $\mathsf{hint}_j \leftarrow (s = n\ ?\ \bot : s)$ and $\mathsf{ver}_j \leftarrow \top$.*
3) Discontinuity: *Losing the last successor increments $e \leftarrow e + 1$, clears $\mathsf{ver}_j \leftarrow \bot$ for all j, and returns the attempt owner to* Idle*.*

Definition 17 (Single-Owner Attempt State)**.** *At most one finger lookup attempt α exists at node n:*

$$\alpha \in \mathsf{Idle} \ \mid\ \mathsf{AwR}(u, p, t) \ \mid\ \mathsf{AwA}(u, p, y, e, t),$$

where $\mathsf{AwR}(u, p, t) \equiv \mathsf{AwR}((u, p), e_\pi, \tau_r)$ awaits a lookup report for slot p and request UUID u issued at evidence epoch e_π before monotonic deadline $t = \tau_r = \text{issued} + T_r$

($T_r = 10\,\mathrm{s}$), and $\mathsf{AwA}(u,p,y,e,t) \equiv \mathsf{AwA}(\pi,\tau_a)$ *awaits WebRTC transport admission of proved peer y at epoch e before lease deadline $t = \tau_a = \text{deferred} + T_a$ ($T_h = 180\,\mathrm{s}$, $T_a = T_h + 30\,\mathrm{s} = 210\,\mathrm{s}$).*

Law 2 (Attempt consumption and outcomes)**.** *A report (u',p',y') consumes α iff either $\alpha = \mathsf{AwR}(u,p,t) \wedge (u',p') = (u,p)$ or $\alpha = \mathsf{AwA}(u,p,y,e,t) \wedge (u',p',y') = (u,p,y)$; all other reports are dropped as* Stale *without mutating state or failure streak $k \in [0,255]$. With cursor $c \in \mathbb{N}$ and retry cooldown deadline $\nu \in \mathbb{N}_\perp$ (an attempt may start from* Idle *only when $\nu = \perp$ or $\mathrm{now} \geq \nu$), consuming reports and timeout expirations ($\mathrm{now} \geq \tau_r$ or $\mathrm{now} \geq \tau_a$) transition (α, c, k, ν) via [14]:*

$$\begin{aligned}
\textsc{Fail}(p):&\ (\mathsf{Idle},\ \mathrm{run}(p)+1,\ k+1,\ t+f(k+1)),\\
\textsc{Retire}(\mathrm{end}):&\ (\mathsf{Idle},\ \mathrm{end}+1,\ k+1,\ t+f(k+1)),\\
\textsc{Progress}(\mathrm{end}):&\ (\mathsf{Idle},\ \mathrm{end}+1,\ 0,\ \perp),
\end{aligned}$$

where $\textsc{Fail}(p)$ *handles expired or empty lookup attempts (*AwR*),* $\textsc{Retire}(\mathrm{end})$ *handles admission refusal/expiry (*AwA*),* $\textsc{Progress}(\mathrm{end})$ *records admitted range confirmation,* $\mathrm{run}(p) = \max\{j \in [p,L) : \forall l \in [p,j],\ \mathsf{hint}_l = \mathsf{hint}_p\}$ *when* $\mathsf{hint}_p \neq \perp$ *and* $\mathrm{run}(p) = p$ *when* $\mathsf{hint}_p = \perp$*, and* $f(k) = \min(2^{\min(k,6)}\,\mathrm{s}, 60\,\mathrm{s})$ *with minimum issue spacing* $T_{\min} = 1\,\mathrm{s}$*. (Under Law 1(3), losing the last successor clears* $\alpha \leftarrow \mathsf{Idle}$ *and preserves monotonic backoff pacing until a head is restored.)*

Definition 18 (Cyclic Slot Selection (`next_fix_slot`))**.** *With $h = \mathrm{head}(n)$ and first routable slot $r = \lfloor \log_2 \delta_n(h) \rfloor + 1$ ($r = 0$ if $h = \perp$), let $c \in \mathbb{N}$ be the cyclic cursor, $A = \{j \geq \max(c,r) : \neg\mathsf{ver}_j\}$, and $B = \{j \in [r, \max(c,r)) : \neg\mathsf{ver}_j\}$. The selector is*

$$\mathrm{next_fix_slot}(c) = \mathrm{sel}(c) = \begin{cases} \min A, & A \neq \varnothing, \\ \min B, & A = \varnothing \wedge B \neq \varnothing, \\ \perp, & \text{otherwise.} \end{cases}$$

Proposition 4 (Starvation freedom)**.** *Let $\mathcal{U}$ be the set of maximal contiguous populated equal-hint runs and individual unpopulated ($\perp$) or unretired slots among routable entries remaining unverified across attempts. Between any two selections in $\mathcal{U}$, every other element of $\mathcal{U}$ is selected at least once, and periodic revalidation reopens proved ranges whenever* $\alpha = \mathsf{Idle} \wedge (\neg\exists j \geq r : \neg\mathsf{ver}_j \ \vee\ k > 0)$.

Proof. Every completed or failed attempt at slot j advances c strictly past j ($\mathrm{end}+1 > j$ or $\mathrm{run}(j)+1 > j$). Because $\mathrm{sel}(c)$ scans cyclically from c, it visits U in strict cyclic index order, preventing a single unreachable NAT peer from monopolizing the single attempt owner. □

Law 3 (Report Claims (`NotifyPredecessor` / `ReportFloor`))**.** *Stabilization and successor-list synchronization reports are correlated by two-phase claim tokens* $\mathsf{tok} = (h, \mathrm{id}, \phi)$ *with* $\phi \in \{\mathsf{Req}, \mathsf{Proc}\}$*. A report claims its token atomically by moving* $\mathsf{Req} \to \mathsf{Proc}$*, dials at most $K+1$ distinct candidates for stabilization (and at most K for successor sync) re-checking $\phi = \mathsf{Proc}$ before each dial, and clears* tok *on completion or on any* SuccessorHeadChanged *transition (`ReportFloor`).*

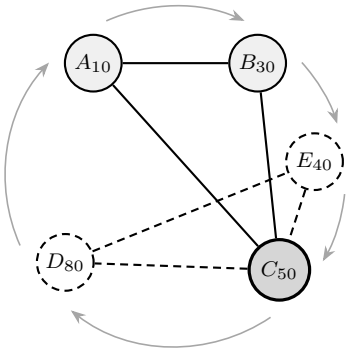

Figure 7. Interleaved ring $A(10) \to B(30) \to E(40) \to C(50) \to D(80)$ over two non-transitive WebRTC cliques $V_1 = \{A,B,C\}$ (solid) and $V_2 = \{C,D,E\}$ (dashed) sharing bridge node $C(50)$.

C. Non-Transitive WebRTC Overlay and Bridge-Delegated Delivery

In browser and native WebRTC deployments, symmetric NATs, enterprise firewalls, and disjoint LANs violate underlay transitivity [7], [15]: if node a links to b and b links to c, a may still fail ICE negotiation with c. Figure 7 illustrates two full-mesh WebRTC cliques $V_1 = \{A,B,C\}$ and $V_2 = \{C,D,E\}$ joined at an articulation (bridge) node $C(50)$ with interleaved ring order $A(10) \prec B(30) \prec E(40) \prec C(50) \prec D(80)$.

Rather than injecting unstructured neighborhood gossip or virtual fingers [16], [17] (which would violate the direct-link invariant $\mathrm{Referenced}(S_n) \subseteq \mathrm{Admitted}(n)$), Rings resolves non-transitive WebRTC routing with zero extra control frames via four mechanisms:

Definition 19 (Extended Topology Carrier and Multi-Predecessors)**.** *Each node $n \in V$ maintains*

$$\begin{aligned} S_n = \big(&n,\ \mathtt{succ}_n,\ \mathtt{pred}_n,\ \mathrm{Preds}_K(n),\\ &\mathtt{fingers}_n,\ \mathtt{bridged}_n,\ \Phi_n,\ \Pi_n\big), \end{aligned}$$

where $\mathrm{Preds}_K(n) = \langle p_1, \ldots, p_m \rangle$ *($m \leq K$, supporting capacity $K = 4$) is ordered by descending clockwise distance $\delta_n(p_1) > \cdots > \delta_n(p_m)$ (ascending counter-clockwise distance $\delta_{p_i}(n)$), with primary predecessor $\mathtt{pred}_n = p_1$, Φ_n denoting the finger-convergence evidence/attempt state (Definitions 16–18), and Π_n denoting the report-claim and dial-backoff state (Laws 3–4). All referenced peers satisfy*

$$\begin{aligned} \mathrm{Referenced}(S_n) = \big(&\mathrm{set}(\mathtt{succ}_n) \cup \mathrm{set}(\mathrm{Preds}_K(n))\\ &\cup \mathrm{ran}(\mathtt{fingers}_n)\big) \setminus \{n\}\\ \subseteq\ &\mathrm{Admitted}(n). \end{aligned}$$

Law 4 (Multi-Predecessor Rectification and Candidate Dial Backoff)**.**

1) Empty-confirmed commit: *When $r = \mathrm{head}(n)$ remains routable ($\mathrm{Routable}(n,r)$), n commits $\mathtt{Stabilize}(r)$ and emits $\mathtt{Notify}(r)$ even if confirmation filtering leaves `confirmed` empty.*

2) Candidate dial backoff (`DialBackoff`): *Failed or expired candidate handshakes enter a bounded negative cache* $\mathcal{B}_n$ *(*$B_{\max} = 64$*) with exponential cooldown* $\beta(f) = \min(\beta_0 \cdot 2^{\min(f-1,4)}, \beta_{\max})$ *(base* $\beta_0 = 5\,\text{s}$*, cap* $\beta_{\max} = 60\,\text{s}$*, configurable to* 30–300 s*), suppressing redundant SDP offers while clearing* $\mathcal{B}_n(c)$ *immediately on explicit connects or admission.*
3) Supersede-on-Closer, Retain-on-Sub-Ring (`rectify_preds`): *Given geometrically admissible non-self candidate* $c \in R \setminus \{n\}$ *(*c *passes peer identity verification and is reachable or actively notified) and primary predecessor* p_1*: if* $p_1 = \bot$ *or* $\delta_n(c) > \delta_n(p_1)$ *(*c *is strictly closer to* n*),* $\text{Preds}_K(n) \leftarrow \langle c \rangle$*; if* $\delta_n(c) \le \delta_n(p_1)$ *(notification from a farther sub-clique predecessor), insert* c *into* $\text{Preds}_K(n)$ *in descending* $\delta_n(\cdot)$ *order, deduplicate, and truncate to* K*.*

Lemma 3 (Articulation Retention and Sub-Clique Local Finger Proof)**.** *On any transitive ring, once* $\text{pred}_V(n)$ *notifies* n *and stale prior entries expire or depart,* $\text{Preds}_K(n)$ *collapses to the singleton global predecessor* $[\text{pred}_V(n)]$*. At an articulation node* C *joining* $m \le K$ *internally stabilized cliques* $V_1, \ldots, V_m$*,* $\text{Preds}_K(C)$ *retains all* m *sub-clique predecessors* $p_i = \text{pred}_{V_i}(C)$ *within two stabilization rounds (round 1 installs* p_1 *at the head, and round 2 retains all farther* p_i *without interim eviction), protecting every bridge link* (p_i, C) *and allowing each* p_i *to filter* `reported.preds` *to* `confirmed.pred` $= \text{Some}(p_i)$ *and prove its local sub-clique finger range* $[0, \min(L-1, \lfloor \log_2 \delta_{p_i}(C) \rfloor)]$ *over* V_i *via Lemma 2.*

Proof. In a transitive ring, the true global predecessor $\text{pred}_V(n)$ has strictly maximal clockwise distance $\delta_n(\text{pred}_V(n))$ among all $p \neq n$; once $\text{pred}_V(n)$ stabilizes and sends `Notify`, Law 4(3) satisfies $\delta_n(\text{pred}_V(n)) > \delta_n(p_1)$ for any prior p_1 and collapses $\text{Preds}_K(n)$ to $[\text{pred}_V(n)]$ (with unreachable stale predecessors removed by transport eviction). At an articulation bridge C linking $m \le K$ internally stabilized cliques $V_1, \ldots, V_m$, let $p_1 = \text{pred}_{V_{\text{near}}}(C)$ denote the global primary predecessor of C (closest counter-clockwise across $\bigcup_i V_i$). Each sub-clique predecessor $p_i = \text{pred}_{V_i}(C)$ satisfies $\delta_C(p_i) \le \delta_C(p_1)$, so once p_1 occupies index 0 in round 1, every other p_i's `Notify` falls into the sub-ring retention branch of `rectify_preds` rather than superseding p_1. Since there are $m \le K$ distinct sub-clique predecessors and `rectify_preds` retains the K largest $\delta_C(\cdot)$ values, all $\{p_1, \ldots, p_m\}$ persist inside $\text{Preds}_K(C) \subseteq \text{Referenced}(S_C)$ within two stabilization rounds, pinning their WebRTC links and enabling each p_i to filter `reported.preds` for itself and apply Lemma 2 over its routable sub-clique V_i on $[0, \min(L-1, \lfloor \log_2 \delta_{p_i}(C) \rfloor)]$ (resolving cross-clique keys in $(p_i, p_1]_R$ via Law 5). □

Law 5 (Passive 1-Hop Gap-Bridge Hints and Bridge-Delegated Delivery)**.** *Whenever* n *processes an authenticated* `TopoInfo` *report from* $r \in$ `succ`$_n$*, the unconfirmed peers* $U(n, r) = (\text{set}(\texttt{reported.preds}) \cup \text{set}(\texttt{reported.succ})) \setminus \{p : \rho_n(p)\}$*—where* $\rho_n(p) \equiv (p \in \text{Referenced}(S_n) \cup \{n\})$*—update the zero-TTL partial map* `bridged`$_n$: Did $\rightharpoonup$ Did *(*$|\texttt{bridged}_n| \le 2K^2$*, with* $p \mapsto r$ *iff* $r \in \text{Referenced}(S_n) \wedge p \notin \text{Referenced}(S_n)$*). Payload delivery carries* $\sigma \in \text{RouteStage} = \text{aim}(\mathsf{Did}_\perp) \times \mathsf{bridged}(\mathbb{B})$ *(fields* $(\texttt{via}, \texttt{handed_off})$ *with* $\text{aim}(\sigma, T) = \texttt{via}.\text{unwrap_or}(T)$*; here* `via` $= \text{Some}(b)$ *is fixed only at origination by a predecessor-less node via* `reply_via` *for* `FindSuccessorReport` *replies and is delivered directly once* b *is reached without mid-route mutation):*

- Rule B1 (Bridged Delivery on Local Minimum in `delivery_step`): *Node* n *first delivers directly if* $\text{Linked}_n(T)$ *(or to* $a = \text{aim}(\sigma, T)$ *if* $a \neq T \wedge \text{Linked}_n(a)$*), and otherwise runs greedy clockwise forwarding (*`route_toward`*) over linked referenced peers* $\text{Referenced}(S_n) \cap \text{Linked}(n)$ *along* $(n, a]_R$*. Only when greedy hits a local minimum (*$\text{Handoff}(h)$*) with* $\neg\sigma.\texttt{handed_off}$*, if* $\texttt{bridged}_n(a) = \text{Some}(b)$ *satisfies* $b \neq n \wedge \text{Linked}_n(b) \wedge \neg\,\text{Linked}_n(a)$*,* n *delegates to* (b, σ) *when* $\delta_n(b) < \delta_n(a)$ *and to* $(b, (\sigma.\texttt{via}, \top))$ *otherwise; if no bridged hint applies, it falls back to* $(h, (\sigma.\texttt{via}, \top))$*. When* $\text{Handoff}(h)$ *occurs with* $\sigma.\texttt{handed_off} = \top$*,* `delivery_step` *returns* $\bot$ *(terminating the route acyclically).*
- Rule B2 (Gap-Bridge Lookup Delegation in `lookup_successor`): *When DHT key resolution* `find_successor`*(*k*) would otherwise terminate with local head* $h \notin \{k, n\}$ *(*$k \neq n$*) across an unlinked gap, if* $\exists (p \mapsto b) \in \texttt{bridged}_n$ *with* $b \in \text{Referenced}(S_n)$*,* $p \notin \text{Referenced}(S_n)$*,* $\text{Linked}_n(b)$*, and* $\delta_n(k) \le \delta_n(p) < \delta_n(h)$*, node* n *delegates* `FindSuccessorSend`*(*k*) via the bridge* b^* *minimizing* $\delta_n(p)$*.*

Theorem 1 (Bounded Single-Hop Bridge Delegation and Route Safety)**.** *Without source-routing hop lists on the wire,* delivery_step *over* $\text{RouteStage} = \text{aim}(\mathsf{Did}_\perp) \times \mathsf{bridged}(\mathbb{B})$ *preserves all four route-safety invariants on any underlay graph* $G = (V, L)$*: (1)* `Linked` *(*$\{n_k, n_{k+1}\} \in L$*), (2)* `NeverOvershoots` *(*$\delta_{n_k}(n_{k+1}) < \delta_{n_k}(\text{aim})$ *on every non-crossing hop), (3)* `SingleCrossing` *and* `Acyclic` *(the boolean crossing bit transitions* $\bot \to \top$ *at most once, visiting no state* (n_k, σ_k) *twice and bounding every route to* $\le 2|V|$ *hops), and (4) all-pairs cross-clique delivery in* ≤ 3 *hops (*2 *hops from a clique local minimum or bridge-adjacent peer,* 3 *hops when greedy advances through a closer clique mate first) between two cliques* V_1, V_2 *sharing bridge* C *whenever* $(V_1 \cup V_2) \setminus \{C\} \subseteq \texttt{succ}_C \cup \text{Preds}_K(C)$ *(supporting up to* $s+m$ *total non-bridge peers under minimal* m*-clique notifications, or up to* $s + K = 2K = 8$*, i.e.* $C_{\text{sub}} \le 4$*, when* $\text{Preds}_K(C)$ *holds* K *entries; larger or multi-bridge cliques establish direct WebRTC links seamlessly via Rings' DHT-discovered stateless cryptographic UDP/TCP TURN and multi-hop* `onion::udp` *Onion TURN in* `rings-turn-service`*).*

Proof. Because $0 < \delta_{n_k}(n_{k+1}) < \delta_{n_k}(a) \iff \delta_{n_{k+1}}(a) = \delta_{n_k}(a) - \delta_{n_k}(n_{k+1}) < \delta_{n_k}(a)$ on $R = \mathbb{Z}/2^{160}\mathbb{Z}$, Invariant (2) implies $\delta_{n_k}(\text{aim}(\sigma, T)) \in \mathbb{N}$ strictly decreases across every non-crossing greedy hop within each fixed stage σ, visiting at most $|V| - 1$ nodes. At any local minimum 'Handoff(h)', $\text{Referenced}(S_n) \cap \text{Linked}(n)$ contains no node in $(n, a]_R$, so any linked bridge b satisfies $\delta_n(b) > \delta_n(a)$ and both the wrapping branch of Rule B1 and fallback handoff flip `handed_off` : $\bot \to \top$ (while σ.`via` never mutates mid-

route, changing to $\perp$ only upon reaching $a = \text{via}$ where b delivers directly to T); since 'Handoff' with `handed_off` $= \top$ returns $\perp$, at most one ring-wrap or gap-bridge crossing occurs per route, yielding $\leq 2|V|$ hops total. In a two-clique graph $V_1 \cup V_2$ sharing bridge C, whenever C's succ_C and $\text{Preds}_K(C)$ cover $(V_1 \cup V_2) \setminus \{C\}$ (up to $s+2=6$ combined non-bridge peers under minimal two-predecessor stabilization via Lemma 3, or up to $s+K=8$ peers when $\text{Preds}_K(C)$ holds $K=4$ entries), every $u \in V_i \setminus \{C\}$ learns $\texttt{bridged}_u(v) = \text{Some}(C)$ for $v \in V_j \setminus \{C\}$ from C's `TopoInfo`. Within full-mesh clique V_i, greedy forwarding takes at most one intra-clique hop $u \to w$ to the local minimum relative to v (0 hops if u itself is the local minimum or C is greedy-closest in $(u, v]_R$), whereupon Rule B1 delegates $w \to C$ and C delivers directly $C \to v$ in at most 3 hops total. □

V. CRDT Storage and Removal Stability

A. DID, VID, and Replicated Entry Algebra

Definition 20 (DID and VID namespaces)**.**

$$\begin{aligned} \mathcal{D} &= (X_D, H_D, \Pi, \text{Session}), \\ H_D &: X_D \to R, \\ \mathcal{V} &= (X_V, H_V, \text{Auth}_V), \\ H_V &: X_V \to R, \\ \text{PrivateKey}(H_V(x)) &= \perp, \\ \text{Auth}_V(op, x, \sigma, caps) &\in \{\top, \perp\}. \end{aligned}$$

Definition 21 (Biased circular order)**.**

$$A \leq_C B \iff \delta_C(A) \leq \delta_C(B),$$

decided relative to pivot C via Algorithm 4.

Algorithm 4 Biased Circular Comparison

```
function BiasedCompare(A, B, C)
    a ← δ_C(A); b ← δ_C(B)
    if a < b then return A <_C B
    else if a > b then return B <_C A
    else return A = B
```

Definition 22 (Replicated entry algebra and placement invariant)**.**

$$\begin{aligned} \mathcal{E}_v^{join} &= (E_v, \sqsubseteq, \sqcup, \perp), \\ \mathcal{E}_v^{final} &= (E_v, \mathsf{conflict}_v, \mathsf{finalize}_v), \end{aligned}$$

where $(E_v, \sqsubseteq, \sqcup, \perp)$ is a join-semilattice [18] ($x \sqcup y = y \sqcup x$, $(x \sqcup y) \sqcup z = x \sqcup (y \sqcup z)$, $x \sqcup x = x$, $x \sqcup \perp = x$), and placement satisfies

$$\text{Inv}_{place} \equiv \forall \nu, x.\ \text{replicas}(H_\nu(x)) \subseteq \text{Rep}_N^\rho(H_\nu(x)).$$

Definition 23 (Mailbox and hidden service descriptor)**.**

$$\begin{aligned} mbox(d, e) &= H_M(\texttt{rings.mailbox} \| d \| e), \\ entry_M &= (\eta, exp, proof_s, dst, cipher), \\ h &= (\nu, \rho, Intro, Relay, Cap, K, exp, rank), \\ VID(h) &= H_H(h), \qquad \mathsf{Lookup}(h) = P_H^N(h). \end{aligned}$$

B. Carrier State, HLC Timestamps, and Cryptographic Data Provenance

Definition 24 (Data Carrier and HLC Timestamps)**.** *A Rings data-topic carrier (`dht::entry`) is a 4-tuple*

$$x = (V, D, \rho, \beta),$$

where $V : \text{Bytes} \times R \rightharpoonup \text{Dot}$ maps live elements and their writer DIDs (`dots` plus multi-writer `co_dots`) to addition dots, $D : \text{Digest} \times R \rightharpoonup \text{Dot}$ maps Keccak-256 element digests $h(v)$ and writer DIDs $o \in R$ to per-author removal dots (tombstones; unscoped $D : \text{Digest} \rightharpoonup \text{Dot}$ is used only on single-recipient `RelayMessage` carriers), $\rho : R \rightharpoonup \text{Ver}$ maps each writer DID to its reset floor register (`register` plus `co_registers`), and $\beta \in \mathbb{N}_\perp$ is the carrier liveness bound (joining by $\max$ with $\perp < t$). Each version or dot d carries a Hybrid Logical Clock (HLC) timestamp [19]

$$\tau_{\text{HLC}}(d) = (w, c, o),$$

ordered lexicographically by wall-clock time $w = \tau(d)$, logical counter $c \in \mathbb{N}$, and writer/origin DID $o \in R$. On `EntryKind::Data`, a removal $((e, o), r) \in D$ covers *an addition $((v, o'), d) \in V$ iff $o = o' \wedge h(v) = e \wedge d \leq r$, and reset floor $\rho(o)$ covers only additions/removals authored by o with $d \leq \rho(o)$ (preserving strict non-interference across distinct writers $o \neq o'$). A carrier x is* normalized *iff no addition in V is covered by D or ρ, and no removal in D is covered by ρ. The delta-state join $x \sqcup y$ [20], [21] is pointwise* $\max$ *followed by normalization.*

Law 6 (Per-Writer Cryptographic Provenance (`DataProvenance`))**.** *Every element addition (`Append` / `Reset`, i.e., `Extend`) and removal tombstone (`Retract`, i.e., `Tombstone`) carries a per-writer cryptographic provenance proof $\pi_{\text{prov}} \in \mathsf{DataProvenance}$ signed over the canonical domain separator* `rings-core : data-entry : mutation` *and a Merkle batch claim* $(\mathsf{did}, \mathsf{kind}, \mathsf{ver}, \text{leaf_count}, r_{\text{merkle}})$*:*

$$\begin{aligned} \mathsf{Verify}\big(k_o, \sigma_o, &\texttt{rings-core} : \texttt{data-entry} : \texttt{mutation} \\ &\|\mathsf{op}\| k_{\text{topic}} \| r_{\text{merkle}} \| (w, c, o)\big) = \top, \end{aligned}$$

where $\mathsf{op} \in \{\mathsf{Reset}, \mathsf{Append}, \mathsf{Retract}\}$, $k_{\text{topic}} \in R$ is the placement key, r_{merkle} commits to the batch of element digests $e = h(v)$, and $o = D(k_o)$ matches the HLC origin DID. Unauthenticated deltas or wall-clock timestamps exceeding local clock $t + \sigma$ are rejected at admission.

C. Removal Stability Horizon, Factored Carrier, and Tombstone Bounds

Definition 25 (Stability Horizon and Retirement)**.** *For data topics (`EntryKind::Data`), writes request default retention* `DEFAULT_TTL_MS` $= 600\,\text{s}$ *up to element horizon $H =$* `MAX_TTL_MS` $= 6\,000\,\text{s}$ *($6 \cdot 10^6$ ms) under tight clock-skew tolerance $\sigma =$* `TS_OFFSET_TOLERANCE_MS` $= 3\,\text{s}$ *(whereas offline relay-inbox mail permits* `DEFAULT_RELAY_INBOX_TTL_MS` $= 86\,400\,\text{s}$ *up to* `MAX_RELAY_INBOX_TTL_MS` $= 7\,\text{d}$ *bounded by signed hold instant). A removal (e, r) (or floor ρ) is* stable *at clock t iff*

$$t \geq \tau(r) + H + \sigma.$$

The retirement projection $\mathrm{retire}_t(x)$ *drops additions with* $t \geq \tau(d) + H$ *and removals/registers with* $t \geq \tau(r) + H + \sigma$. *Because the horizon* H *is uniform and* $d \leq r \implies \tau(d) \leq \tau(r)$, retire_t *never leaves a live addition covered by a retired removal and is a join-semilattice homomorphism satisfying* $\mathrm{retire}_t \circ \mathrm{retire}_s = \mathrm{retire}_{\max(s,t)}$ *(distinct from the carrier-level expiry predicate* $\mathrm{retired_at}(x, t)$*, which empties surviving additions once* $t \geq \beta$ *if* $\beta \neq \bot$*).*

Lemma 4 (Stable removes cover only retired adds)**.** *If* (e, r) *is stable at the clock* t_p *of some node* p*, then every addition* (v, d) *covered by* (e, r) *is already retired at the clock* t_q *of every node* q *at the same instant (*$t_q \geq t_p - \sigma \geq \tau(r) + H \geq \tau(d) + H$*).*

Proof. Since removal (e, r) covers addition (v, d), Definition 24 gives $d \leq r$ in lexicographic HLC order, hence $\tau(d) \leq \tau(r)$. By clock-skew admissibility, any two synchronized honest clocks t_p, t_q at the same physical instant differ by at most σ, so $t_q \geq t_p - \sigma$. Combining stability at p ($t_p \geq \tau(r) + H + \sigma$) yields $t_q \geq \tau(r) + H \geq \tau(d) + H$, so retire_{t_q} removes (v, d) on every read, join, or sync at q. □

Law 7 (Remove survival and Factored Carrier)**.** *An unstable remove or register leaves a holder's replica only by (i)* retire_t *at stability, (ii) an ack-gated range handoff whose receiver holds it, or (iii) a platform loss event—never by storage eviction, cap, or truncation. To enforce this, every normalized carrier* x *factors into an element side and a remove side:*

$$x = \mathsf{E}(x) \sqcup \mathsf{R}(x),$$
$$\mathsf{E}(x) = (V, \varnothing, \bot, \beta), \qquad \mathsf{R}(x) = (\varnothing, D, \rho, 0).$$

$\mathsf{E}(x)$ *is stored in the data store at authority* $(\mathsf{Cch}, \mathsf{Reb})$*, whereas* $\mathsf{R}(x)$ *is persisted in the dedicated* removal store *(*`dht-removals`*) at* $(\mathsf{Dur}, \mathsf{Ret})$ *(*`RecordAuthority::Retained`*). Byte-budget eviction maps* $x \mapsto \mathsf{R}(x)$ *(dropping only* $\mathsf{E}(x)$*),* $\mathrm{put}(k, x)$ *flushes* $\mathsf{R}(x)$ *before writing* $\mathsf{E}(x)$*, and reads project* $\mathrm{get}_t(k) = \mathrm{retire}_t(\mathsf{E}_k \sqcup \mathsf{R}_k)$.

Theorem 2 (No Resurrection)**.** *Under Law 7, if holder* p *holds* (e, r) *at* t_0 *and experiences no loss event before* $\tau(r) + H + \sigma$*, then across any sequence of joins, evictions, and crashes, no read at* p *after* t_0 *ever serves an addition covered by* (e, r).

Proof. Before stability ($t < \tau(r) + H + \sigma$), Law 7 keeps $D_p(e) \geq r$ in the durable removal store across evictions and crashes, so $\mathrm{get}_t(k) = \mathrm{retire}_t(\mathsf{E}_k \sqcup \mathsf{R}_k)$ normalizes away any covered addition. At and after stability, Lemma 4 guarantees that every covered addition has $t \geq \tau(d) + H$ and is stripped by retire_t. □

Remark 2 (Clock-Skew Boundary for Browser Nodes)**.** *Lemma 4 and Theorem 2 rely on bounded physical clock skew* $|t_p - t_q| \leq \sigma = 3\,\mathrm{s}$ *across synchronized honest holders. When a browser device wakes from sleep with wall-clock drift exceeding* σ*, Law 6 temporarily rejects skewed writes until OS/NTP time resynchronizes, prioritizing strict no-resurrection safety over skewed liveness. Without read fencing, a pure wall-clock projection* $t_q \mapsto \mathrm{retire}_{t_q}(c[k])$ *at a waking node* q *whose wall clock lags a synchronized peer* p *by more than* σ *(*$t_q < t_p - \sigma$*) would fail to retire and transiently surface an unretired addition* (v, d) *whose covering tombstone* (e, r) *has already reached stability and been garbage-collected at* p*. Fencing local reads against a combined monotonic and Byzantine-safe admitted-HLC lower bound (*$\tau - \sigma \leq t_{\mathrm{admit}}$ *under Law 6) prevents backward wall-clock jumps and awake clock stalls from resurrecting retired additions; if OS/browser suspension pauses both wall and monotonic clocks prior to NTP resynchronization, no single untrusted peer can safely advance* q*'s read floor past* t_q *without risking premature-retirement DoS, so read fencing complements rather than replaces OS/NTP clock resynchronization.*

Law 8 (Tombstone Bound (`Storage Tombstone Full`))**.** *Each data carrier is bounded by* $T_{\max} = D_{\max} =$ `ENTRY_DATA_MAX_LEN` $= 1024$ *removes. Any peer delta* δ *with* $|D_\delta| > T_{\max}$ *is inadmissible, and any operation or join whose retired result* $y = \mathrm{retire}_t(x \sqcup \delta)$ *has* $|D_y| > T_{\max}$ *is refused with* $\mathsf{Err}(\mathsf{Exhausted})$ *(*`Storage Tombstone Full`*), leaving the stored carrier unchanged. Every admitted carrier encodes in at most* $W = w_0 + D_{\max}(P_{\max} + \lambda + w_D) + T_{\max} w_T + w_\rho = 33\,731\,696\,B$ *(*$w_0 = 47$*,* $P_{\max} = 32\,768$*,* $\lambda = 3$*,* $w_D = 69$*,* $w_T = 101$*,* $w_\rho = 65\,B$*;* $33.73\,MB \leq \frac{3}{4}$`TRANSPORT_MAX_SIZE` $= 45\,MB \leq 60\,MB$*), and any admissible sync delta simulates a sequence of* $\leq 2|D_\delta| + |V_\delta| + 1$ *writer operations.*

D. Fetch Cache Linearization and Signed Lease Heartbeats

Law 9 (Fetch-Cache Pure-Read Linearization)**.** *The local fetch cache* $c : K \rightharpoonup X$ *is a* $(\mathsf{Vol}, \mathsf{Reb})$ *mirror whose mutations (*$\mathrm{put}, \mathrm{obs}, \mathrm{evict}$*) execute atomically under the cache lock while reads are pure projections* $t \mapsto \mathrm{retire}_t(c[k])$ *that perform zero store writes or deletions. Consequently, every cache transition linearizes [22] at its store put and every read at its store get, eliminating concurrent read-retire lost-update races.*

Law 10 (Signed Lease Heartbeats (`LeaseHeartbeat`))**.** *To eliminate the* 120 *tombstones per hour per service (*$3\,600\,\mathrm{s}/I$*) and* $O(N_{\mathrm{reg}} \cdot \mathrm{churn})$ *read-modify-write overhead generated when registrants overwrite timestamped descriptors every* $I = 30\,\mathrm{s}$*, a registrant signs a fixed lease descriptor* d *under the unified* `SignedDescriptor` *domain family (*`rings-node:{online-node,service,mail}-descriptor` *and* `rings-turn:service-descriptor`*) binding its identity, process epoch, capabilities, freshness window* $T = 90\,\mathrm{s}$*, and lease expiration* $\ell = t + \Lambda$ *(*$\Lambda = 10\,\mathrm{min}$*). Each heartbeat at clock* t *emits only* $\mathrm{Extend}([d])$ *(writing zero tombstones and performing zero registry reads), rotating* $d \mapsto d'$ *only when* $t + 2I \geq \ell$ *or capabilities change. Readers at clock* t *accept the verified descriptor of greatest* ℓ *satisfying* $\ell \leq t + \Lambda + \sigma$*,* $T \leq \Lambda$*,* $t \leq \ell$*, and* $t \leq \tau^*(d) + T + \sigma$.

VI. Durable Local Stores and Replicated Custody

A. Authority Lattice, Presence Markers, and Platform Crash Durability

A Rings node persists replay windows, onion entry guards, removal records, and relay mail across native file systems and browser IndexedDB backends. We unify their presence

Table II
AUTHORITY ASSIGNMENTS ACROSS RINGS 1.0 LOCAL STORES

Local Store	Auth. (c, m)	Discharged Obligations
DHT Data Store	(Cch, Reb)	Disposable / rebuildable
Fetch Cache / Measure	(Vol, Reb)	Pure-read mirror
Onion Entry Guards	(Dur, Reb)	A, D (`DurableCache`)
CRDT Removal Store	(Dur, Ret)	A, D, N (`Retained`)
Relay Custody Carriers	(Dur, Ret)	A, D, N (r-repl. on Dur°)
Transaction Replay	(Dur, Aut)	A, D, N, W, F (`Authoritative`)

and crash-survival guarantees via a product authority lattice (`RecordAuthority`) [23].

Definition 26 (Crash-Meaning Authority Lattice $C \times M$)**.** *The crash chain $C = \mathsf{Vol} \sqsubset \mathsf{Cch} \sqsubset \mathsf{Dur}$ and meaning chain $M = \mathsf{Reb} \sqsubset \mathsf{Ret} \sqsubset \mathsf{Aut}$ form the componentwise lattice $C \times M$ with monotone obligation map $O(c, m) = O(c) \cup O(m)$:*

$$O(\mathsf{Vol}) = O(\mathsf{Cch}) = \varnothing,$$
$$O(\mathsf{Dur}) = \{\textit{Atomicity (A)}, \textit{Survival (D)}\},$$
$$O(\mathsf{Reb}) = \varnothing, \qquad O(\mathsf{Ret}) = \{\textit{No-Eviction (N)}\},$$
$$O(\mathsf{Aut}) = \{N, \textit{Witnessed-Presence (W)}, \textit{Fail-Closed (F)}\}.$$

By the Admissibility Law, a store may open at (c, m) only if $c \sqsupseteq \mathrm{req}(m)$, where $\mathrm{req}(\mathsf{Reb}) = \mathsf{Vol}$ and $\mathrm{req}(\mathsf{Ret}) = \mathrm{req}(\mathsf{Aut}) = \mathsf{Dur}$. Table II details the authority levels assigned across Rings 1.0 local stores.

Law 11 (Presence Markers (`.p`) and Per-Platform Crash Durability)**.**

1) Presence markers (`.p`): *In an authoritative native store, each record $n = \mathrm{hex}(\mathrm{SHA1}(k))$ (used strictly as a fixed-width local filesystem key over canonical internal storage keys; SHA-256 is a drop-in substitute if untrusted external keys are exposed) has an empty witness marker n.p created and directory-synced only after n.tmp $\to n$ is flushed and directory-synced. On deletion, removing n.p is followed by a parent-directory sync barrier before unlinking n and syncing the directory again, preserving $P \subseteq R$ across arbitrary OS crash points. At restore (`open`), unmarked data records $n \in R \setminus P$ are repaired in place by re-creating n.p, whereas any $n \in P \setminus R$ is classified as $\mathsf{Missing}(n)$ and fails stream n closed (discharging W for every partial loss λ_{part}, whereas total directory loss λ_{tot} leaves $P = R = \varnothing$ and serves as the documented operator reset).*
2) macOS / iOS: *File data and directory entries are flushed via* `fcntl(F_FULLFSYNC)` *[24].*
3) Windows: *Temporary file data flushes via* `FlushFileBuffers` *and directory mutations execute via* `MoveFileExW` *using the* `MOVEFILE_REPLACE_EXISTING` *and* `MOVEFILE_WRITE_THROUGH` *flags [25].*
4) Linux / Unix: *Data files flush via* `fsync` *followed by parent directory* `fsync` *after atomic rename [23].*
5) Browsers (IndexedDB): *Read-write transactions at* Dur *open with explicit $\{\textit{durability} : \text{“}\texttt{strict}\text{”}\}$ (forcing OS buffer flush / SQLite* `EXTRA` *or full WAL checkpoint across Chromium, Gecko, and WebKit [26]–[30]) combined with* `navigator.storage.persist()` *[31]. Because browsers delete origin storage whole ($P = \varnothing$) and ephemeral contexts can only claim durability (Dur°), node-local stores open at (Dur°, m) with session end as λ_{tot}, while custody of other nodes' mail requires $r \geq 2$ replicated holders.*

B. Group Commit and Link-Local Volatile Replay Windows

Law 12 (Group Commit (`put*`))**.** *Replay transitions $t_1 < t_2 < \ldots$ execute in lock order on speculative tables $\hat{T}$ with undo log entries u_i and coalesce into batches B_j whose distinct write-set W_j is committed by a single $\mathrm{put}^*(W_j)$ (with concurrent readers isolated from uncommitted $\hat{T}$ until commit): on native targets, all n.tmp files are pushed before a single drive-cache barrier (*`F_FULLFSYNC`*), renamed, and directory-synced once, after which* `Authoritative` *batches run* `witness_batch` *to create missing n.p markers and perform a final directory sync; on IndexedDB, all puts share one* `strict` *read-write transaction. Callers are released only after $\mathrm{put}^*(W_j) = \mathsf{Ok}$ (latency $\leq 2\phi$); on* Err*, $\hat{T}$ rolls back via u_i to the last durable state T.*

Law 13 (Link-Local Volatile Replay Windows ($\mathcal{W}_{s,g}$))**.** *Direct neighbor control traffic (`Join`|`Sync`|`Leave`|`Send`) and application namespaces registered in D_r as delegating replay (such as Sphinx onion cells, which enforce at-most-once admission in their own Bloom filter R_i and process epoch e_i) are classified by $\mathrm{ll}(t, c)$ on authenticated connection c of live generation g and admitted against an in-memory per-generation window $\mathcal{W}_{s,g}$ instead of the durable store $\mathcal{W}_s$. This eliminates durable disk flushes on the onion/control hot path while preserving per-generation duplicate/fork detection and cross-restart replay immunity.*

C. Receiver Custody Set and Age-Gated Replicated Relay Custody

Definition 27 (Receiver Custody Set ($\mathcal{K}_s$))**.** *To prevent relay mail admitted into $\mathcal{W}_s$ from being lost if the node restarts before the application consumes it from the volatile inbox, the receiver record of stream s co-persists $(\mathcal{W}_s, \mathcal{K}_s)$ atomically in a single $(\mathsf{Dur}, \mathsf{Aut})$ record, where $\mathcal{K}_s \subseteq \mathbb{N} \times \mathrm{Digest} \times \mathbb{N}$ holds admitted but untaken deliveries (q, h, x) with spent horizon $x = ts + ttl + H_I + 2\epsilon$ (for clock-skew safety margin $\epsilon > 0$ and $|\bigcup_s \mathcal{K}_s| \leq$* `RELAY_INBOX_MAX_LEN` $= 64$*).*

Law 14 (Custody Lifecycle and Age-Gated Replicated Relay Chain)**.**

1) Custody drain and release: *On drain after put^*, an element (s, q, h) already in $\mathcal{K}_s$ is requeued without re-checking $\mathcal{W}_s$; a new element commits $(\mathcal{W}'_s, \mathcal{K}_s \cup \{(q, h, x)\})$ before queueing; and the carrier tombstone/Ack is issued only after the application takes the message and commits $(\mathcal{W}_s, \mathcal{K}_s \setminus \{(q, h, x)\})$. By information-theoretic indistinguishability, no bounded-window receiver can safely admit un-custodied mail that has fallen behind live traffic ($q < \mathrm{rmin}(\mathcal{W}_s)$, where $\mathrm{rmin}(\mathcal{W}_s)$ is the lowest sequence tracked by $\mathcal{W}_s$).*
2) Age-gated replicated custody ($\mathcal{M}_r(k)$): *Because browser holders operate at claimed durability Dur° and experience high churn [32], a node advertises relay custody only after surviving an online age gate $A_{\min}$*

(`DEFAULT_RELAY_AGE_GATE_MS` $= 2\,\text{min}$ *floored at* ≥ 4 *stabilization periods, and* $A_{\min} = 30\,\text{min}$ *in production relay profiles). Each mail key* k *is replicated across a chain* $\mathcal{M}_r(k)$ *of the first* r *aggated holders clockwise from* $\text{owner}_N(k)$ *(*$r = 4$ *to* `DEFAULT_RELAY_INBOX_REPLICAS` $= 8$*).*

3) Chain replication and digest repair: *Holders forward head-to-tail along* $\mathcal{M}_r(k)$ *(digest first, payload on miss) and acknowledge tail-to-head only after all* $q(k) \leq r$ *members persist the element. Adjacent chain members probe liveness every* $P_c = 2.5\,\text{s}$ *(suspect after* $\tau_c = 7.5\,\text{s}$*) and repair via bundled carrier digest-set comparison on membership change or periodic pass, restoring* r *copies within* $D \leq T_d + r\delta$ *(where* $T_d = P_c + \tau_c = 10\,\text{s}$ *is worst-case silent failure detection and* δ *is per-hop repair transfer time).*

VII. Fair Admission, Slot Accounting, and Load Bounds

A. Cleanest-First Slot Roster and Fair Slot Accounting

Because a Rings DID is a free public-key digest, any node-wide table keyed solely by origin DID can be exhausted by one adversary minting thousands of identities. Rings anchors resource accounting to the authenticated transport connection slot.

Definition 28 (Cleanest-First Slot Roster (`SlotRoster`))**.** *A node's connection registry holds at most*

$$\mathcal{R}(s) = 2(F + s + 1)$$

active neighbour edges, where $F = 160$ *is the finger table size and* s *is the successor capacity (*$\mathcal{R}(4) = 330$ *at* $s = 4$*;* $\mathcal{R}(3) = 328$ *at* $s = 3$*). The* `SlotRoster` *indexes slots* $\kappa \in \{0, \ldots, \mathcal{R}(s) - 1\}$*, tracking per-slot metadata* $(epoch, strikes, last_seen_ms)$ *and live floor utilization across shared tables* T*:*

$$w_t(\kappa) = \max_T \frac{\min\big(\text{use}^t_{T,\kappa},\, f_T\big)}{f_T}.$$

When a new edge e *is admitted, it is assigned the* cleanest *free slot* $\kappa(e) = \arg\min_{\kappa \in F_t} w_t(\kappa)$ *(ties broken by least index), and charges in table* T *remain bound to* κ *until they expire by* T*'s own horizon law.*

Law 15 (Slot Accounting)**.** *Every shared table* T *is parameterized by a per-slot guaranteed floor* f_T*, a shared overflow pool* $\mathcal{P}_T$*, a local share* L_T *for principal* λ*, and dedicated shares* S_d *for* $d \in D_T$*, yielding the hard node-wide bound*

$$B_T(s) = \mathcal{R}(s)\, f_T + \mathcal{P}_T + L_T + \sum_{d \in D_T} S_d.$$

With slot usage use_κ*, pool overflow* $o_\kappa = \max(0, \text{use}_\kappa - f_T)$*, and active claimant set* $\mathcal{A}(t) = \{\kappa : o_\kappa > 0 \text{ or } \kappa \text{ has a queued/pending overflow request}\}$ *of size* $n(\mathcal{A}(t)) = |\mathcal{A}(t)| \geq 1$ *(waiting claimants persist in* $\mathcal{A}(t)$ *until served or withdrawn), an item introduced by slot* κ *is admitted iff either:*

1) $\text{use}_\kappa < f_T$ *(within the slot's guaranteed floor), or*
2) $\sum_{\kappa'} o_{\kappa'} < \mathcal{P}_T$ *and* $o_\kappa < \lceil \mathcal{P}_T / n(\mathcal{A}(t)) \rceil$ *(within the slot's fair pool claim).*

Theorem 3 (Multi-Table Slot Isolation)**.** *Across all five shared node tables—(1) Application Inbox / Mailbox Ledger (*S_μ*-bounded), (2) Relay Store (*S_μ*-bounded), (3) CRDT Removal Store (*$f = 128$*,* $\mathcal{P} = 16\,384$*,* $L = 7\,168$*,* $S_{\text{custody}} = 16\,384$*, yielding* $B_{\text{rem}}(4) = 82\,176$ *items at default* $s = 4$*;* $81\,920$ *at* $s = 3$*), (4) Transaction Replay Table (*$B_{\text{rep}}(4) = 15\,680 \leq 16\,384$*) and per-lane Origin Quota (*$f = 1$*,* $\mathcal{P} = 2\,048$*,* $L = 1\,056$*,* $S_{\text{inbox}} = 640$*, yielding* $B_{\text{orig}}(4) = 4\,074 \leq 4\,096$*), and (5) Chunk Reassembly Buffer O—Law 15 and Definition 28 guarantee:*

1) $|T| \leq B_T(s)$ *in all reachable states;*
2) *any edge holding slot* κ *with* $\text{use}_\kappa < f_T$ *is never refused, regardless of how many origins, edges, or slots other parties control;*
3) *an over-floor slot is guaranteed at least* $f_T + \lfloor \mathcal{P}_T / \mathcal{R}(s) \rfloor$ *items as prior claimants expire;*
4) *under cleanest-first assignment (*`SlotRoster`*), an adversary rotating Sybil connections cannot degrade a newcomer's floor with a single departure-and-rejoin (unlike least-free-index selection), multiplying the single-attacker floor-pollution threshold by* $\mathcal{R}(s) - c$ *(*$300\times$ *at* $\mathcal{R}(4) = 330, c = 30$*) distinct admissions within one horizon* h *(while sustained high-rate Sybil flooding past* $\mathcal{R}(s) - c$ *relies on connection rate limiting or DRanking 2.0 admission filtering; Section IX).*

Proof. Items (1)–(3) follow because floor checks read use_κ alone, departures never erase slot charges, and with $n \leq \mathcal{R}(s)$ active claimants, any slot below $\lfloor \mathcal{P}_T / \mathcal{R}(s) \rfloor$ satisfies $o_\kappa < \lceil \mathcal{P}_T / n \rceil$ and can refill once draining items drop overflow below $\mathcal{P}_T$. Item (4) holds because a free slot with $w_t(\kappa) = 0$ is always selected before any partially charged slot κ' with $w_t(\kappa') > 0$. □

B. Replay Stream Eviction and Credit Reclaim

Law 16 (Replay Stream Spent-Horizon Eviction)**.** *A receiver replay stream* s *with maximum admitted transaction expiry* $x_s = \max(ts + ttl)$ *is* spent *at clock* t *once* $t \geq x_s + \iota + \sigma$ *(for default control/storage/E2E traffic, the spent horizon is* $H_{spent} = 600\,\text{s}$*; for offline-capable application streams,* $\iota = H_I + \sigma$ *where* $H_I = 7\,\text{d}$*). Because transactions past* $x_s + \iota + \sigma$ *are rejected by envelope TTL and relay hold-horizon checks prior to replay lookup, spent streams holding no custody* $\mathcal{K}_s$ *are lazily evicted without reopening replay. With* $f = 32$*,* $\mathcal{P} = 3\,328$*,* $L = 1\,024$*, and inbox share* $S_\mu = 768$*, no honest slot can be permanently locked out.*

Law 17 (Credit Reclaim (`Credit/Shrink/Returned`) and Hard Frame Bound)**.** *Each edge has four lanes* ℓ *(one priority control lane with floor* $g_{\text{prio}} = W = 16$*, and three data lanes with floor* $g_\ell = 1$ *and cap* $W = 16$*). To bound total in-flight and buffered frames node-wide without dropping frames, credit frames carry a lane generation* g *tracking receiver-consumed frames* R_{rx} *and sender-committed frames* c_{tx}*:*

- $\mathsf{Credit}(A, g)$*: receiver grants limit* A *in generation* g*;*

- $\mathsf{Shrink}(L', g)$*: receiver reclaims unused pooled credit on an idle* ($T_{\text{idle}} = 1\,\text{s}$) *or over-share lane down to* $L' = \min(R_{\text{rx}} + g_\ell, A)$*;*
- $\mathsf{Returned}(B, g)$*: sender commits at most* $B = \max(c_{\text{tx}}, L')$ *frames and ignores older generations.*

With frame pool $\mathcal{P}_F = 256$ *frames (*$16\,\text{MiB}$*), where* $c(s) = \mathcal{R}(s)(W+3)$ *sums per-slot guaranteed floors across the four lanes and* $W_{shrink} = \mathcal{P}_F$ *reserves reclaimed pooled credits until* $\mathsf{Returned}(B, g)$ *clears, total node frame liability satisfies the hard bound*

$$\sum_{e,\ell} \text{liab}(e, \ell) \;\le\; B_F(s) = c(s) + W_{shrink} = \mathcal{R}(s)(W+3) + \mathcal{P}_F$$

($6\,488$ *frames at* $s = 3$*;* $6\,526$ *at* $s = 4$*), with zero honest frame drops and guaranteed per-edge pooled throughput* $\ge \min(3(W-1), \lfloor \mathcal{P}_F / n \rfloor)$.

C. Single-Charge Chunk Reassembly and Per-Category Barrier

Law 18 (Single-Charge Chunk Reassembly and Credit-Time Rate Floor)**.** *A chunked message* $[0, n)$ *of chunk size* d_0 *reserves its full footprint* $m = 2nd_0$ once *on its first chunk in both the mailbox ledger and the open-message table* O *(*$f_O = 256\,\text{KiB}$*,* $\mathcal{P}_O = 128\,\text{MiB}$*,* $B_O(s) = \mathcal{R}(s) f_O + \mathcal{P}_O$*, i.e.* $210.5\,\text{MiB} \le 256\,\text{MiB}$ *at* $s = 4$*), waiting in* `ResourceOrderedQueue` *order (*$O((g + k + r) \log n)$ *per event, where* g *is granted requests,* k *displaced waiters moved to another blocker, and* r *named resources) and moving into dispatch at completion without re-charging. To prevent slowloris reservation pinning without penalizing receiver-throttled links, the receiver tracks* credit-time $\hat{\tau}$ *for the active head-of-line message on lane* ℓ*, which advances over* $[t_k, t_{k+1})$ *by* $\min(t_{k+1} - t_k,\, c_k d_0 / r_{\text{floor}})$ *only while the sender holds* $c_k = A - N > 0$ *unused credits. Messages must maintain a streaming rate floor (*$r_{\text{floor}} \in [r_{\min}, 10\,\text{B/ms}]$ *with* $r_{\min} = 1\,024\,\text{B/s}$*) with grace window* $G = 15\,\text{s}$ *(up to* $(W+1)d_0 / r_{\text{floor}} + \Delta$*, where* Δ *bounds link/RTT slack), expiring attributably if received bytes fall below* $b < r_{\text{floor}} \hat{\tau} - W d_0$.

Law 19 (Per-Category Reassembly Barrier (`Control | Interactive | Bulk`))**.** *Inbound dispatch maintains independent ordering barriers per traffic category* $\kappa \in \{\mathsf{Control}, \mathsf{Interactive}, \mathsf{Bulk}\}$ *(mapped to protocol lanes* $\{ctl, sto, e2e, app\}$*): a frame* f *of category* κ *waits only behind unfinished reassembly events* $u \in U$ *with* $seq(u) < seq(f) \wedge \kappa(u) = \kappa$*. This guarantees exact sequence order within every category (including chunked control frames) while preventing bulk data reassembly from blocking control or interactive lanes.*

VIII. Privacy Layer — Cryptographic Carriers and Sphinx Onion Kleisli Pipeline

A. Separate Cryptographic Carriers

Definition 29 (End-to-end carrier, signed frame, and secret sharing)**.** *End-to-end encryption (ElGamal* $(g^k, m \cdot h^k)$ *on prime-order group* $\mathbb{G} \neq R$*) and Shamir threshold secret sharing over* $\mathbb{F}_q[X]$ *[33] operate strictly off the routing carrier:*

$$\begin{aligned} f &= (sid, seq, last, cipher, pk_s, \sigma_s), \\ \text{FrameValid}(f) &\iff V(pk_s, \sigma_s, m_f) = \top \\ &\quad \wedge \text{Session}(D(pk_s), sid, e), \\ P \in \mathbb{F}_q[X], &\quad sh_i = (i, P(i)) \in \mathbb{F}_q^2, \\ \text{reconstruct}(\{sh_i\}) &= P(0). \end{aligned}$$

Here m_f *is the canonical serialization of* $(sid, seq, last, cipher)$*, and payloads exceeding link* d_0 *are sliced deterministically into* $total = \lceil |data| / d_0 \rceil$ *indexed chunks carried inside authenticated frames before reassembly.*

Invariant 3 (Chunk reassembly)**.**

$$\begin{aligned} \text{complete}(m) \iff \forall i < total(m).\, \exists! c_i \\ \wedge \sum_i |c_i| \le B_m \wedge now < ttl(m). \end{aligned}$$

B. Operation Signature and Kleisli Pipeline Composition

Rings 1.0 replaces legacy relay framing with the **Sphinx Onion Kleisli Pipeline** (`rings-node::onion`), generalizing onion circuits from simple exit tunnels to client-sealed loops over a static operation signature Σ composed in the Kleisli category $\mathbf{Kl}(M)$ [34]–[40].

Definition 30 (Operation Signature Σ and Partial Σ-Algebras)**.** *The static shape signature is*

$$\Sigma = \{\mathsf{relay}\} \cup \Sigma_W \cup \Sigma_{\mathsf{invoke}},$$

containing $1 + 3 + 120 = 124$ *rows, where* $\Sigma_W = \{\mathsf{tcp}, \mathsf{https}, \mathsf{udp}\}$ *is the set of world-facing exit stream symbols,* $\Sigma_{\mathsf{invoke}} = \{\mathsf{invoke}\langle \text{In}, \text{Out}, L\rangle : \text{In} \in \{0\} \cup \mathcal{W},\, \text{Out} \in \mathcal{W},\, L \in \mathcal{L}\}$ *has fixed output width* $W(f) = \text{Out}$ *across discrete width classes* $\mathcal{W} = \{32, 64, 256, 1024, 16384\}\,\text{B}$ *and latency classes* $\mathcal{L} = \{10, 100, 500, 2000\}\,\text{ms}$*, and open invoke backends* $\omega \in \Omega$ *(*$|\omega| \le 32$*) include:*

`dht.find_successor`, `dht.notify`, `dht.sync`, `storage.lookup`, `storage.store`, `service.lookup`, `service.register`, `direct.message`.

Each node n *publishes signed registrations* $\text{Reg}_n \subseteq \{\mathsf{relay}\} \cup \Sigma_W \cup (\Sigma_{\mathsf{invoke}} \times \Omega)$ *(via* `rings-node:onion-{exit,invoke}-descriptor`*) carrying its 128-bit process epoch* e_n *and quota* $\text{quota}(n, f, \omega)$*, acting as a partial* Σ*-algebra with generic effect [38], [41]*

$$[\![f]\!]_n : \text{Args}_f \times \text{In}_f \to M\, \text{Out}_f.$$

Definition 31 (Kleisli Pipeline Composition (Laws L1, L2))**.** *A pipeline* $P = (f_1, \bar{a}_1) \ggg \cdots \ggg (f_n, \bar{a}_n)$ *is a well-formed morphism of the free category* $\mathcal{F}(G_\Sigma)$ *iff* $1 \le n \le n_{\max} = 4$*,* $\text{Out}_{f_k} = \text{In}_{f_{k+1}}$ *for all* $k < n$*, intermediate symbols satisfy* $\mathsf{Intermediate} \in \text{pos}(f_k)$*, and world-facing stream symbols (*$\mathsf{tcp}, \mathsf{https}, \mathsf{udp}$*) appear only at* $n = 1$*. Placed on nodes* $h_1, \ldots, h_n$*,* P *denotes the Kleisli composite in* $\mathbf{Kl}(M)$*:*

$$[\![P]\!](v_0) = \big([\![f_1]\!]_{h_1}(\bar{a}_1, -) \ggg \cdots \ggg [\![f_n]\!]_{h_n}(\bar{a}_n, -)\big)(v_0),$$

satisfying ***Law L1 (Identity)****:* $[\![\mathsf{relay}]\!] = \eta$ *and* $\mathrm{Loop}(\mathsf{relay} \ggg P) \overset{d}{=} \mathrm{Loop}(P)$ *via normal-form* nf *reduction, and* ***Law L2 (Associativity)****:* $(f \ggg g) \ggg h = f \ggg (g \ggg h)$.

C. Client-Sealed Loops, Uniform Sphinx Header, and AEZ v5 Carry

Definition 32 (Client-Sealed Loop and Hall-Filter Draw Order (Law L7))**.** *For client $\diamond$, entry guard g, and $n \in [1,4]$ symbol hops $h_1, \ldots, h_n$, each segment $k \in [0,n]$ inserts $s = 2$ relays $r_{k,1}, r_{k,2}$ (the minimal s such that no node is adjacent to both $\diamond$ and a symbol hop):*

$$\begin{aligned}\mathrm{Loop}(P):\ \diamond \to g = r_{0,1} \to r_{0,2} \to h_1 \to r_{1,1} \to r_{1,2} \\ \to \cdots \to h_n \to r_{n,1} \to r_{n,2} = g \to \diamond,\end{aligned}$$

*with total hops $H(n,s) = (s+1)n + s \le \hat{H} = 14$ ($H \in \{5, 8, 11, 14\}$) and $H-1$ distinct peer DIDs (**Law L7: Guard Closure**). Route selection draws $h_1, \ldots, h_n$ first, then guard g, then remaining relays $r_{k,j}$ using a bipartite-matching **Hall filter** $\mathcal{F}(\pi, p)$ (retaining candidates at position p for partial placement π only if extending π admits a complete system of distinct mutually routable representatives across the remaining $\{1, \ldots, H\}$ role slots) so that if the first draw succeeds, no subsequent draw can fail, and exit marginals over the Hall-admissible exit candidate set N_{h_1} equal exact quality weights $q(d)/q(N_{h_1})$.*

Definition 33 (2870-Byte Sphinx Header and 205-Byte Uniform Per-Hop Layer (Laws L3, L4))**.** *Every hop $i \in [1,H]$ across all roles (`HopRole::Transit | Terminal | GuardReturn`) peels an identical 205-byte layer*

$$\begin{aligned}\lambda_i &= (f_i, \bar{a}_i, \mathrm{next}_i, e_i, x, \nu_i, \kappa_i, \gamma_{i+1}),\\ \ell &= 1 + 64 + 20 + 16 + 8 + 16 + 64 + 16 = 205\,\mathrm{B},\end{aligned}$$

*from a fixed-length Sphinx header [34] $\chi_i = \alpha_i \| \beta_i \| \gamma_i$ with blinded SEC1 group element $|\alpha_i| = 33\,\mathrm{B}$, padded routing block $|\beta_i| = \hat{H}\ell = 14 \times 205 = 2870\,\mathrm{B}$, and MAC $|\gamma_i| = 16\,\mathrm{B}$ ($|\chi_i| = 2919\,\mathrm{B}$), where $x = \lceil t_{\mathrm{build}}/Q \rceil Q + X_0$ ($Q = 30\,\mathrm{s}$, $X_0 = 120\,\mathrm{s}$, $V = 150\,\mathrm{s}$) and $\kappa_i = (\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \in \{0,1\}^{256} \times \{0,1\}^{256}$ partitions the 64-byte carry field into a 32-byte inbound carry seed σ_{in} and a 32-byte outbound segment seed σ_{out} (filled with uniform dummy bytes at pure transit relays). Each loop selects a cell bucket class $b \in [16\,\mathrm{KiB}, 12\,\mathrm{MiB}]$ with uniform carry slot width $C_b = b - 2919\,\mathrm{B}$ ($C_{16\,\mathrm{KiB}} = 13\,465\,\mathrm{B}$, 82% payload efficiency, **Law L3**) and deterministic release time $t_i^{\mathrm{dep}} = t_i^{\mathrm{arr}} + L(f_i)$ (**Law L4**).*

Definition 34 (AEZ v5 Wide-Block Body Carry (`OnionBodyCipher`))**.** *For each segment k, the client draws a 32-byte seed σ_k (delivered as σ_{out} to symbol producer $p_k = h_k$ for intermediate segments $k \ge 1$) and derives 48-byte AEZ v5 keys [36], [42] $k_{r_{k,j}}$ and k_{c_k} via HKDF-SHA256. Producer p_k applies one authenticated wide-block encipherment $\mathrm{E}^{\tau}_{k_{c_k}}$ ($\tau = 16$ zero-byte authenticator) over canonical serialization $\mathrm{enc}(v_k)$ followed by $s = 2$ length-preserving wide-block encipherments $\mathrm{E}^{0}_{k_{r_{k,j}}}$ ($\tau = 0$):*

$$\begin{aligned}y_{k,0} &= \mathrm{E}^{0}_{k_{r_{k,1}}} \circ \mathrm{E}^{0}_{k_{r_{k,2}}} \circ \mathrm{E}^{16}_{k_{c_k}}\big(\mathrm{pad}(\mathrm{enc}(v_k))\big),\\ y_{k,j} &= \mathrm{D}^{0}_{k_{r_{k,j}}}(y_{k,j-1}), \qquad b_k = \mathrm{D}^{16}_{k_{c_k}}(y_{k,2}).\end{aligned}$$

D. SURB Batching, Cover Loops, Replay Admission, and Security Laws

Law 20 (SURB Batching, Constant-Rate Cover, and Sliced-Bloom Replay (Law L9))**.**

1) SURB batching: *A Single-Use Reply Block for world-facing session ς is the 2 979-byte ($20 + 2\,919 + 32 + 8 = 2\,979\,\mathrm{B}$) client-sealed suffix $\upsilon = (\mathrm{next}, \chi_\upsilon, \sigma_\upsilon, x_\upsilon)$ [43], [44]. After deducting the 16 B AEZ authenticator ($C_0 = C_{16\,\mathrm{KiB}} - 16 = 13\,449\,\mathrm{B}$) and 6 B credit-frame overhead, a 16 KiB credit loop leaves 13 443 B for SURBs—batching $k = \lfloor 13\,443/2979 \rfloor = 4$ SURBs plus its own suffix ($k+1 = 5$ reply credits per forward loop, $Q_{\max} = 256$)—whereas a 7 B-framed stream data cell carries up to $C_0 - 7 = 13\,442\,\mathrm{B}$ of application payload.*
2) Dormant-by-default constant-rate cover loops: *Outbound onion lanes open in dormant mode (`open_dormant`, emitting zero unsolicited cover cells on non-participating DHT links) and wake automatically when real onion cells arrive or depart. Across active bursts and their bounded dwell window $[t, t+D)$ ($D = Q = 30\,\mathrm{s}$), links emit at constant rate $r = \rho B/V \approx 98.3\,\mathrm{units/s}$ ($\rho = 0.9$, substituting cover cells when no real cell is queued), parking at zero idle overhead outside active dwell windows by default (`OnionIdleFloor::OFF`, with optional configurable background cover up to $r_{\mathrm{idle}} = 1\,\mathrm{unit/s}$ via `OnionIdleFloor::DEFAULT`). This yields an analytical single-link paced emission ceiling of $98.3 \times 13\,442\,\mathrm{B} \approx 1.32\,\mathrm{MB/s}$ at $b = 16\,\mathrm{KiB}$ (sustaining $1.045 \pm 0.003\,\mathrm{MB/s}$ end-to-end across a 5-node live WebRTC loopback, above the $1.00\,\mathrm{MB/s}$ CI throughput floor; Section X).*
3) `SlotRoster`-backed sliced-Bloom replay admission (Law L9): *Every received cell is charged $u(b) = b/16\,\mathrm{KiB}$ units against the sender DID's ledger $\Lambda_i(d)$ ($|\operatorname{dom}\Lambda_i| \le 2\mathcal{R}$, per-sender budget $B = 16\,384$ units, global budget $G = 64B$ over $V/Q = 5$ quanta)* before *SEC1 decoding and ECDH. Valid tags ν_i are recorded in expiry-quantized sliced Bloom filters [45], [46] $R_i[x]$, structured as a scalable chain of 76 960 B blocks ($K = 26$ slices of $M = 23\,680$ bits, sealing $B = 16\,384$ tags per block up to 64 global blocks) queried via process-secret-keyed* $\mathrm{Keccak\text{-}256}(\kappa \| \nu_i \| j)$ *probes (per-block false-positive rate $\le 2^{-26}$, union FP $\le 2^{-20}$, max memory $\approx 4.70\,\mathrm{MiB}$ at 64 blocks), ensuring every AEZ key is used at most once across restarts and clock rollbacks.*

Theorem 4 (Onion Pipeline Security Laws (Laws L5, L5′, L6, L8))**.** *Under the Gap-DDH assumption over secp256k1 in the Random Oracle Model (modeling HKDF as a random oracle against active header-peeling queries [34]), together with ChaCha20 PRG, HMAC-SHA256 EUF-CMA, and AEZ v5 strong PRP / robust AEAD assumptions [36], [47], let $\pi_i(P)$ denote hop i's local interface projection (role, adjacent links, symbol f_i, arguments $\bar{a}_i$, expiry x, and cell class b), π_i° its position-erased projection (identical to $\pi_i(P)$ with hop index i omitted), $\sigma(P)$ the visible loop class, and D_i the cumulative deterministic delay from client build time t_{build} to arrival $t_i^{\mathrm{arr}} = t_{\mathrm{build}} + D_i$ modulo Q. Every loop built by Definitions 32–34 satisfies:*

1) ***Law L5 (Noninterference):*** $\pi_i(P) = \pi_i(P') \wedge \sigma(P) = \sigma(P') \implies \mathrm{view}_i(P) \approx_c \mathrm{view}_i(P')$*: no hop learns another hop's arguments* $\bar{a}_j$*, invoke backend name* ω_j*, or segment value* $v_k \neq \hat{v}_i$ *[48].*
2) ***Law L5′ (Position Hiding):*** *For loops* P, P' *of arbitrary lengths* $H, H' \leq \hat{H} = 14$*, a hop's distinguishing advantage between positions* i *and* i' *with matching local projection* π_i° *under uniform build-phase quantization modulo* Q *is bounded by*

$$\begin{aligned}\mathrm{Adv}_i \ &\leq \hat{H}\big(\epsilon_{\text{Gap-DDH}} + \epsilon_{\mathrm{PRG}} + \epsilon_{\mathrm{MAC}}\big) \\ &\quad + \min\big(1,\, |D_i - D'_{i'}|/Q\big).\end{aligned}$$

3) ***Law L6 (Client Path Authority):*** next_i *is sealed in* λ_i *by* $\diamond$ *independently of runtime values* v_k*, admitting the Arrow composition fragment (⋙) while excluding promiscuous monadic bind (≫=) [40].*
4) ***Law L8 (Byte Unlinkability and Tagging Resistance):*** *Consecutive edges carry independent uniform ciphertexts against external link observers and pure transit relays (*$\chi_i \neq \chi_{i+1}$*,* $y_{i-1} \neq y_i$*; on intermediate segments* $k \geq 1$*, a corrupted symbol producer* $p_k = h_k$ *holds segment seed* σ_{out} *and can correlate ciphertexts on segment* k *only if colluding with a downstream link observer), class relabeling fails* $\gamma_i = \mathrm{MAC}_{K_i}(b \| \beta_i)$*, and any modification* $\tilde{y} \neq y_{k,j}$ *is rejected by consumer* $\mathrm{D}^{16}_{k_{c_k}}$ *with probability* $\geq 1 - q \cdot 2^{-128} - \epsilon_{\mathrm{AEZ}}$ *across* q *active forgery queries (*$\geq 1 - 2^{-128} - \epsilon_{\mathrm{AEZ}}$ *per single attempt) [47].*

Proof Sketch. We outline the hybrid sequence across an H-hop loop ($H \leq \hat{H} = 14$), adapting the Sphinx and AEZ reductions of Danezis–Goldberg [34], Hoang et al. [36], and Degabriele–Stam [47]: *Game* $G_0 \to G_1$ *(Gap-DDH Blinding & Random-Oracle Key Schedule):* Under Gap-DDH over secp256k1 and random-oracle HKDF, even an active adversary submitting chosen group elements α to honest header-peeling relays cannot distinguish an uncorrupted hop j's ephemeral shared secret and derived keys $(k_j^{\mathrm{enc}}, k_j^{\mathrm{mac}}, k_j^{\mathrm{blind}})$ from uniform independent keys except with advantage $\hat{H}\epsilon_{\text{Gap-DDH}}$, ensuring fresh unlinkable $\alpha_j \neq \alpha_{j+1}$ across re-blinded segments. *Game* $G_1 \to G_2$ *(PRG Pad & Fixed-Width Position Hiding):* Replacing ChaCha20 keystream expansions over $|\beta| = \hat{H}\ell = 2\,870\,\mathrm{B}$ at each uncorrupted hop incurs advantage at most $\hat{H}\epsilon_{\mathrm{PRG}}$ over $\hat{H}$ peeling steps. Because all three `HopRole` variants pack identical 205-byte slots shifted and padded to constant width $2\,870\,\mathrm{B}$, the decrypted header tail after hop i is uniform noise for both H and H', isolating only the local arrival-to-expiry offset $(x - t_i^{\mathrm{arr}})$ under uniform build phase $t_{\mathrm{build}} \bmod Q$ (absent external inter-hop timing observation of deterministic releases $L(f_i)$): with $t_i^{\mathrm{arr}} = t_{\mathrm{build}} + D_i$, this offset at positions (i, D_i) and $(i', D'_{i'})$ has statistical distance at most $\min(1, |D_i - D'_{i'}|/Q)$, establishing Law L5′. *Game* $G_2 \to G_3$ *(Header MAC & Path Authority):* Under EUF-CMA security of HMAC-SHA256 ($\hat{H}\epsilon_{\mathrm{MAC}}$), no adversary without k_j^{mac} can forge $\gamma_j = \mathrm{MAC}_{k_j^{\mathrm{mac}}}(b \| \beta_j)$ or substitute another hop class b or routing descriptor λ_j; thus next_j and $\bar{a}_j$ remain fixed by client $\diamond$ independently of intermediate payload values v_k (Law L6), while hops $j \neq i$ see only uniform XORed/encrypted slots for $\pi_i(P) = \pi_i(P')$ (Law L5). *Game* $G_3 \to G_4$ *(Wide-Block Body Carry):* For any segment k, the producer wraps fixed-width padded plaintext $\mathrm{pad}(\mathrm{enc}(v_k))$ through $\mathrm{E}^{16}_{k_{c_k}}$ and $s = 2$ passes of $\mathrm{E}^0_{k_{r_{k,j}}}$, with keys derived from fresh nonce-protected seed σ_k and replay protection enforced once via $R_i[x]$ (Law 20). By AEZ v5 strong PRP security ($\tau = 0$) at relays and robust AEAD security ($\tau = 16$) at the consumer [36], [47], intermediate ciphertexts $y_{k,j}$ are pseudorandom permutations over $\{0,1\}^{8C_b}$ (yielding segment-k byte unlinkability $y_{i-1} \neq y_i$ against external observers and transit relays that lack σ_k), and any bit-flip, truncation, or tagging $\tilde{y} \neq y_{k,j}$ anywhere on the segment causes consumer $\mathrm{D}^{16}_{k_{c_k}}$ to fail its 0^{128} authenticator check with advantage at most $q \cdot 2^{-128} + \epsilon_{\mathrm{AEZ}}$ across q forgery queries (Law L8). □

IX. Decentralized Trust and Ranking — DRanking Bridge

A. Sidecar Witness Ordering

Definition 35 (Sidecar witness and partial order)**.**

$$\begin{aligned}w &= (\mathit{domain}, \mathit{root}, \mathit{height}, \mathit{time}, \mathit{finality}), \\ \mathrm{Witness}(msg, w) &\iff \mathrm{Include}(H(msg), w.\mathit{root}) = \top \\ &\quad \wedge \mathrm{Final}(w) = \top, \\ a \prec b &\iff \mathit{domain}(w_a) = \mathit{domain}(w_b) \\ &\quad \wedge \mathit{time}(w_a) < \mathit{time}(w_b), \\ \chi &: \mathit{Domain}_a \times \mathit{Domain}_b \to \mathit{Order},\end{aligned}$$

where $\mathrm{Include}(H(msg), w.\mathit{root})$ *checks a canonical inclusion proof against a finalized sidecar state root and* χ *provides a deterministic cross-domain ordering comparator reconciling witnesses from distinct domains.*

B. Counterparty-Signed Service Receipts and the DRanking 2.0 Bridge

In an open network where identity creation is free ($\mathsf{cost}(\mathsf{did}) = 0$), unsigned local behavior counters or naive median/trimmed-mean votes over unverified peer reports are vulnerable to Sybil amplification [49]. Rings 1.0 therefore establishes a cryptographic evidence primitive on the wire—the counterparty-signed **Service Receipt** (`ServiceReceipt`)—which bridges directly into the companion **DRanking 2.0** verifiable distributed ranking and admission protocol [10].

Definition 36 (Rings 1.0 Service Receipt (`ServiceReceipt`))**.** *A service receipt is a dual-authenticated tuple*

$$\rho = \big\langle \mathsf{ver},\, \mathsf{srv},\, a,\, b,\, k,\, n,\, d_{\mathrm{req}},\, d_{\mathrm{rsp}},\, \sigma_a,\, \sigma_b \big\rangle,$$

implemented on live probe transcripts with distinct provider/beneficiary tags (`rings-core:service-receipt:provisional:{provider, beneficiary}`*) and bridging into DRanking 2.0 under the finalized epoch separator* `rings-service-receipt-v1\0`*, where:*

- $\mathsf{srv} \in \{\mathsf{relay}, \mathsf{store}, \mathsf{probe}\}$ *names the metered service kind;*
- $a \in R$ *is the service provider DID and* $b \in R$ $(b \neq a)$ *is the served beneficiary DID;*

- $k \in \mathbb{N}$ *is the epoch index and* n *is a fresh per-epoch nonce;*
- $d_{\mathrm{req}}, d_{\mathrm{rsp}}$ *are the cryptographic digests of the exact signed request and completion transactions; and*
- σ_a, σ_b *verify under the distinct account keys of* a *and* b *over* `rings-service-receipt-v1\0‖ver‖srv‖` $a\|b\|k\|n\|d_{\mathrm{req}}\|d_{\mathrm{rsp}}$.

Proposition 5 (DRanking 2.0 Bridge Morphism [10])**.** *Because forging* σ_b *for an honest beneficiary* $b \notin \mathsf{Adv}$ *is cryptographically infeasible, manufacturable receipts are restricted to adversarial self-dealing (*$b \in \mathsf{Adv}$*). In the companion DRanking 2.0 protocol [10], distinct-nonce receipts* (k, n) *for pair* (a, b) *fold into a cap-budgeted receipt matrix* $W_k[b, a] \le w_{\max}$ *with normalized stochastic flow* $C_k[b, a] = W_k[b, a] / \max(w_{\max}, W_k[b, \cdot]\mathbf{1})$*. Assuming honest seed support (*$\mathrm{supp}(s) \subseteq H$*), uniformly bounded attenuation (*$\sup_k \alpha_k \le \alpha < 1$*), and initial adversarial weight* $T_0(\mathsf{Adv}) \le \frac{\alpha g}{1-\alpha}$*, the seed-anchored attenuated PageRank/EigenTrust [50], [51] epoch evolution preserves*

$$t_{k+1} = \alpha_k \, C_k^{\mathsf{T}} t_k + (1 - \alpha_k)\, s,$$

$$T_k(\mathsf{Adv}) = \sum_{v \in \mathsf{Adv}} t_k(v) \;\le\; \frac{\alpha g}{1 - \alpha}$$

for all $k \ge 0$*, where* $g = \sup_k \sum_{b \in H, a \in \mathsf{Adv}} C_k[b, a] t_k(b)$ *is the honest endorsement leakage. Soulbound rank* $\hat{t}_k(v)$ *then weights VRF committee sortition and is folded across epochs via Nova-family Incrementally Verifiable Computation (IVC) [52] so browser nodes verify global admission state in* $O(1)$ *time.*

C. Security Predicates and Controls

Table III maps each primary threat class in Rings 1.0 to its failure predicate and cross-layer enforcement mechanism.

Table III
SECURITY PREDICATES AND CONTROLS ACROSS RINGS LAYERS

Attack	Predicate	Rings 1.0 + 2.0 Control
Sybil	many DIDs, one actor	`SlotRoster` cleanest-first floors + fair pool (Section VII); DRanking receipts [10]
Replay	stale valid σ	durable $(\mathcal{W}_s, \mathcal{K}_s)$, link-local $\mathcal{W}_{s,g}$, onion sliced-Bloom $R_i[x]$
MITM	wrong key for DID	DID proof, `rings-delegation-v1`, `rings-core:data-entry:mutation`
DoS	unbounded resource use	5-table slot accounting, credit reclaim Shrink, credit-time $\hat{\tau}$ floor
Tagging	onion byte correlation	Sphinx 2870 B header + AEZ v5 wide-block carry ($\tau = 16$)

Obligation 2 (Replay rejection)**.**

$$\begin{aligned} \mathrm{accept}(s, \eta, t) &\Rightarrow \eta \notin Seen_s \wedge t < exp(s), \\ Seen'_s &= Seen_s \cup \{\eta\}, \\ \mathsf{emitDelivery} &< Seen'_s. \end{aligned}$$

X. EVALUATIONS AND FORMAL VERIFICATION

A. Experimental Setup and Reproducibility

All empirical benchmarks and formal-verification explorers in this section were executed directly against the shipped Rust transition reducers and networking crates of Rings 1.0 (workspace version `1.0.0-pre.8`, commit `6b5a3434`; source and reproduction commands in `papers/README.md` at https://github.com/RingsNetwork/rings) on an Apple M1 Max workstation (10 ARM64 cores, 64 GiB RAM, Darwin 27.0.0) using `rustc 1.97.0` and hardware ARMv8 AES/POLYVAL instructions enabled via `.cargo/config.toml` (`-cfg aes_armv8 -cfg polyval_armv8`). Pure overlay/CRDT/storage benchmarks and all 16 Stateright/BFS model-checking reporters ran under `-profile ci` (`opt-level=3`, release assertion semantics preserved), whereas AEZ v5 cipher microbenchmarks (`rings-aez`), multi-flow gateway harnesses (`rings-gateway`), and Sphinx/Bloom/live 5-node WebRTC loopback tests (`rings-node`) ran sequentially (`-test-threads=1`) under full `-release` (`opt-level=3`, fat LTO).

B. Stateright State Relation and Quantitative Formal Verification

Definition 37 (State relation and Stateright exploration)**.**

$$\frac{\begin{array}{c} C = (S, Q), \\ Q = a :: Q_0, \quad I_A(a) = i, \quad \tau(S, i) = (S', O, \beta) \end{array}}{\begin{array}{c} (S, Q) \to (S', Q_0 \cdot \beta) \\ \mathsf{Explore} = \mathsf{TLAStyle}(\to), \end{array}},$$

$$\begin{aligned} \mathrm{Impl}_A(s, i) &\preceq I_A^{\sharp}(\tau(s, i)), \\ \pi_{state}(\mathrm{Impl}_A) &= \pi_{state}(I_A^{\sharp} \circ \tau). \end{aligned}$$

The exploration relation follows TLA+ style and Stateright finite-state model checking [8], [53] directly over the pure transition functions of `rings-core`*,* `rings-transport`*,* `rings-node`*, and* `rings-turn-service`*.*

Across all four model-checked workspace crates, exhaustive BFS state-space exploration verified **9,496,521 reachable finite states** (plus 102,400 seeded Horizon simulation transitions and 1 931 official AEZ v5 known-answer test vectors [42]) and refuted **54 deliberately injected safety mutants** via replayable counterexample traces (Table IV). Critically, all 16 suites directly execute the shipped production Rust state types and transition reducers (`PeerRing`, `topology::step`, `EntryCrdt`, `RecordAuthority`, `AdmissionLedger`, `CreditWindow`, `IceGatherState`, `OnionReorder`, `TurnAllocationCore`, `OnionUdpPinholeTable`), abstracting only the asynchronous network scheduler, frame drop/duplication, and storage crash prefixes. In `core::dht_stateright`, Stage 2 verifies `Always` safety and `Sometimes` reachable convergence while intentionally recording a negative bounded-convergence counterexample (`no-conv CE: 519,886 states`), reproducing Zave's impossibility result [6] that asynchronous Chord lacks

worst-case bounded-step convergence under unconstrained adversarial scheduling. Table V details the six Stage-6 Rejoin BFS configurations covering simultaneous link flaps, suffix restarts, and churn chains across 2 474 679 states up to depth 60, while every Rejoin mutant triggers a minimal counterexample within 10–788 states (9.41 ms total).

C. Overlay Convergence and Non-Transitive WebRTC Routing

Figure 8 evaluates range-proved finger convergence across uniformly distributed random 160-bit DID rings of size $N \in \{16, 32, 64, 128, 256, 512, 1024\}$ (30 seeded trials per size; 15 at $N = 1024$). By Lemma 2, stabilization with the immediate clockwise head automatically confirms an average of 156.20 slots at $N = 16$ (still 149.40 slots at $N = 1024$) with zero lookups. Across all remaining outer slots, Rings converges cleanly in **3.13–9.20** routed lookups on average (p95 = 5–10, worst-case 12 at $N = 1024$), yielding a 51.06× **to** 17.39× **speedup** over a classical 160-slot Chord refresh scan while preserving exact sparse-finger table equality and $\approx \frac{1}{2}\log_2 N$ multi-hop routing (1.70–4.59 mean hops across 500 random source-destination pairs per size, max 8).

Table VI tests non-transitive WebRTC routing across 50 random interleaved two-clique overlays per size ($C_{\text{sub}} \in \{2, 3, 4, 8, 16\}$ non-bridge peers per clique, $|V| = 2C_{\text{sub}} + 1$), seeding `Notify` transitions at the bridge to exercise the full $2K = 8$ `TopoInfo` capacity regime (whereas passive minimal two-predecessor notifications without extra peer claims cover $s + 2 = 6$ non-bridge peers, i.e. $C_{\text{sub}} \leq 3$). Within the full $2K$ bridge capacity ($C_{\text{sub}} \leq K = 4$, matching Theorem 1(4)), Rings bridge-delegated delivery achieves **100.00% cross-clique delivery** in 2.32–2.61 mean hops (max = 3) with zero loops. By contrast, a Greedy-Only Chord router strands 64.50%–81.25% of traffic at local minima, while an Unbounded Handoff router (without the single-crossing guard) enters routing loops on 29.00%–59.75% of pairs. Beyond capacity ($C_{\text{sub}} = 8, 16 > K = 4$), passive 1-hop `TopoInfo` hints cover at most $2K = 8$ articulation peers without gossip while keeping loops strictly at 0.00% (vs. 74.22%–83.57% loops for Unbounded Handoff).

D. CRDT Carrier Operations, Provenance, and LeaseHeartbeats

Table VII reports real microbenchmarks for CRDT normalization/retirement (`Entry::join` and `Entry::retired_at`), Merkle batch `DataProvenance` signing and verification (`rings-core:data-entry:mutation`, where provenance payload bytes cover the full signed CRDT carrier serialization and verification checks one ECDSA signature plus all B individual $O(B \log B)$ Merkle leaf inclusion proofs), and a 1-hour (3 600 s) registry simulation ($I = 30$ s, $T = 90$ s, $\Lambda = 600$ s). Single-batch ECDSA provenance verifies in 135.18 µs (495 B wire proof), whereas a 64-element batch amortizes to 1.77 µs **sign** and 12.16 µs **verify per element** (20.17 KiB). Over one hour with $N_{\text{reg}} = 100$ active registrants, `LeaseHeartbeat` cuts legacy Overwrite churn from 12 000 tombstones and 12 100 reads down to 0 **tombstones and** 0 **reads** while reducing total registry wire bytes by **46.6%** (8 922.17 → 4 762.01 KiB).

E. Durable Local Store and Group Commit (`put*`)

Figure 9 measures real APFS storage durability (`FileStorage` issuing macOS `fcntl(F_FULLFSYNC)` drive-cache barriers). Unsynced `Disposable::put` completes in 0.213 ms (4 692.4 rec/s), whereas strict per-record durability costs 23.398 ms for `DurableCache::put` and 35.386 ms (28.3 rec/s) for `Authoritative::put` with `.p` witness markers. Group-commit `Authoritative::put_batch` amortizes a single drive barrier across $B \in \{1, 4, 16, 64, 128\}$ records, cutting amortized commit cost from 34.609 ms/rec ($B = 1$) to 0.601 **ms/rec (1 663.5 rec/s)** at $B = 128$—58.86× faster than standalone `Authoritative::put` (35.386 ms/rec) and 57.59× over `put_batch` at $B = 1$.

F. Sybil-Resistant Slot Accounting and Admission Latency

Figure 10 demonstrates Theorem 3(4) at default roster capacity $\mathcal{R}(4) = 330$ with 30 honest occupied slots (300 free slots, floor $f_T = 32$). Under a baseline `LeastFree` allocator, a single adversary rotating through just $k = 1$ departure-and-rejoin reuses the lowest free slot and immediately drops the next honest newcomer's clean floor survival to **0.0%**. Rings' cleanest-first `SlotRoster` isolates newcomers at **100.0% clean-floor survival** across all $k = 1 \ldots 299$ attacker rotations—raising the pollution budget 300× above `LeastFree` before every free slot is touched ($k \geq 300$), while high-rate Sybil floods across a full horizon are mitigated alongside DRanking 2.0 admission filtering (Section IX). Table VIII further confirms that `SlotAccounting` enforces per-slot bounds in 10.8–11.3 ns/op for floor admission and 118.3–188.0 ns/op for charge/discharge across $|T| \in \{100, 1\,000, 5\,000\}$ live items.

G. Sphinx/AEZ v5 Onion Pipeline, Sliced-Bloom, and Gateway Throughput

Table IX summarizes native `-release` measurements for the Sphinx onion pipeline, AEZ v5 wide-block body cipher, sliced-Bloom replay admission, and live 5-node WebRTC loopback. A relay processes a full 16 KiB cell (parse, ECDH, re-blind, PRG/MAC peel, $C_{16\,\text{KiB}}$ AEZ decipher, and re-encode) in 108.0 µs (9 257 **cells/s)**, dominated by secp256k1 ECDH (42.5 µs) and blinding (32.7 µs) (82.0 µs constituent crypto sum vs. 94.6 µs header peel after +12.6 µs SEC1 point parse/encode and copy overhead, plus +5.5 µs packet framing on top of 7.9 µs KDF/AEZ for whole-cell relay processing), whereas client construction scales linearly from 451.4 µs header (489.1 µs cell) at $n = 1$ ($H = 5$) to 1 265.1 µs (1 294.6 µs cell) at $n = 4$ ($H = 14$). Using ARMv8 hardware AES/POLYVAL, AEZ v5 sustains 1 348.1–1 666.0 **MiB/s** across all carry classes (16 KiB–1 MiB), outperforming standard `ChaCha20-Poly1305` (379.5–395.9 MiB/s) by ≈ **4.2**× while simultaneously guaranteeing wide-block unlinkability at $\tau = 0$ and $\geq 1 - 2^{-128}$ untaggability at $\tau = 16$.

Across 1–64 allocated blocks (16 384–1 048 576 tags, 0.07–4.70 MiB), the sliced-Bloom replay filter inserts in 1.06–1.55 µs, queries in 1.05–2.03 µs, and exhibits only 0, 0, 0,

Table IV
QUANTITATIVE STATERIGHT / BFS FORMAL VERIFICATION AND MUTANT REFUTATION ACROSS RINGS 1.0 CRATES (`-PROFILE CI`)

Crate / Harness	Scope / Configurations	States	Mutants	Time (ms)	Verified Invariants & Bounds
`core::dht_stateright`	Stages 1–4 (Notify N=3, 4; Disc.; CRDT SEC; ABA)	618,155	—	19,359.27	CorrectChord, SEC merge, ABA safety (no-conv CE: 519,886 states)
`core::finger_retry`	Stage 5 resource-law explorer (depth 9)	134,371	—	839.98	Single owner, monotone epochs, $\leq K+1$ stab fan-out (479,544 trans.)
`core::rejoin_model`	Stage 6 (6 overlays; Table V)	2,474,679	4 / 4	53,091.65	Inert retired gens, unavailable-head swap, repair on vacate (CE: 10–788)
`core::relay_custody`	Stage 7 storage-less run	192	—	37.32	Single-use query token, 2,496 transitions, ack-gated leave handoff
`core::repair_model`	Virtual (1c) + Handoff (1c, 2c crashes)	21,105	4 / 4	880.39	Replicated mail handoff safety, r-member convergence
`core::horizon_model`	512 seeds × 200 interleaved steps	(102,400 trans.)	7 / 7	5,042.84	Theorem 2 (No Resurrection) under $t \geq \tau(r) + H + \sigma$
`core::durability`	4 auth. modes × `put`/`rem`/`batch`	53 pfx (80 out.)	—	0.015	Lattice $C \times M$, `.p` presence witness, `put`* undo rollback
`core::resource_order`	Unbounded (49,542) + Capped-2 (58,506)	108,048	7 / 7	1,482.72	Bounded priority fair scheduling across control/interactive/bulk
`transport::credit`	24 configs (dup, honest, windows)	5,916,645	5 / 5	18,973.67	Generational `Credit`/`Shrink`/`Returned` zero-drop hard frame bound
`transport::ice_gather`	3 ICE tiers (Host, Reflexive, Relay)	330	5 / 5	0.92	1-RTT SDP/ICE completion & tier-monotonic candidate freeze
`transport::turn_tcp`	Native WebRTC TURN/TCP bridge	16	7 / 7	1.09	Framed TURN-TCP relay admission & close-order safety
`node::onion_admission`	Replay/unit ledger BFS (depth 3)	3,616	3 / 3	92.12	Law L9 pre-ECDH unit charge, K=26 Bloom & epoch-reset safety
`node::exit_and_lanes`	ExitQuota (2,540) + OutboxLanes (10,559)	13,099	2 / 2	123.97	Per-link exit budget isolation & outbox lane ordering
`node::session_order`	Reorder (16,481) + SurbPool (2,540)	19,021	2 / 2	737.42	Exact forward/reply sequence delivery & least-expiry SURB drain
`node::lease_readiness`	LeaseFresh (179,574) + ReadinessLatch (4,897)	184,471	2 / 2	460.41	Admissible `SignedDescriptor` bounds & startup readiness latch
`turn_service::bridge`	TURN-Onion UDP bridge BFS (depth 6)	2,773	6 / 6	10.97	UDP session binding, allocation expiry, & byte-cap safety
Total / Aggregate	**16 model suites + 1,931 AEZ v5 KATs**	**9,496,521**	**54 / 54**	**101,134.76**	**Zero invariant violations across production reducers**

Note: Total States (9,496,521) sums the 14 finite-state BFS state counts (excluding `horizon_model`'s 102,400 seeded simulation transitions and `durability`'s 53 crash prefixes with 80 outcomes); BFS models instantiate scaled-down finite bounds ($N \in \{3, 4\}$, small credit/queue windows) directly over the shipped production Rust reducers.

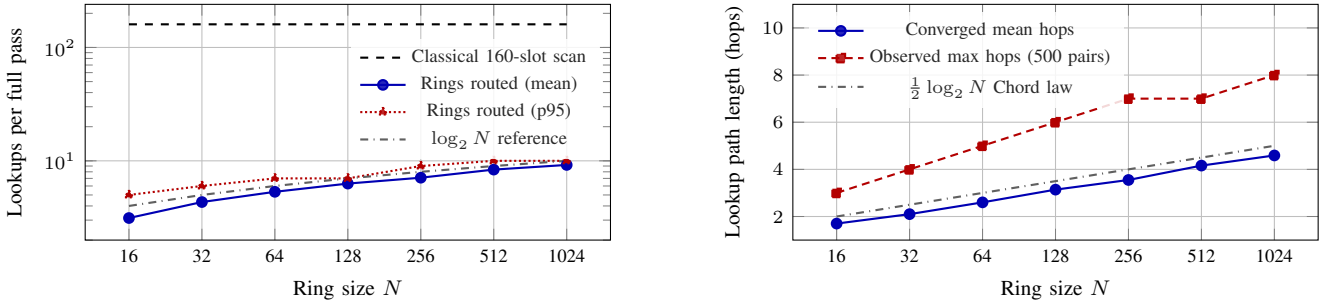

Figure 8. Finger convergence overhead and converged lookup hops across $N \in [16, 1024]$ random DIDs (`bench_paper_finger_convergence_scaling`). Left: Rings range proofs cut routed lookups by 51.06× ($N = 16$) to 17.39× ($N = 1024$). Right: Converged sparse-finger delivery tracks the theoretical $\frac{1}{2}\log_2 N$ Chord curve.

Table V
STAGE-6 REJOIN BFS BREAKDOWN ACROSS SIX TOPOLOGY REGIMES (`TEST_REJOIN_MODEL`)

Configuration	States	Depth	Premise (Unstable)	Time (ms)
`departure`	4,862	21	—	51.33
`replacement`	109,840	35	91,312 (25,472)	2,821.23
`truncation`	237,328	36	219,664 (174,048)	5,972.63
`restart`	542,589	47	464,848 (408,404)	9,860.47
`chain`	545,796	60	545,796 (531,172)	12,694.39
`lossy_flap`	1,034,264	42	890,762 (653,760)	21,682.19
4 Rejoin Mutants	Refuted at 788, 43, 10, and 21 states			**9.41**

Table VI
NON-TRANSITIVE TWO-CLIQUE ROUTING COMPARISON (50 RANDOM OVERLAYS/SIZE; K=4)

C_{sub}	$\lvert V\rvert$	Rts	Rings Bridge Del.%	Rings Bridge Hops	Rings Bridge Max	Greedy Del.%	Unbnd. Hnd. Del.%	Unbnd. Hnd. Loop%
2	5	400	**100.00**	2.32	3	35.50	71.00	29.00
3	7	900	**100.00**	2.51	3	24.00	50.33	49.67
4	9	1,600	**100.00**	2.61	3	18.75	40.25	59.75
8	17	6,400	**47.03**	2.83	4	13.45	25.78	74.22
16	33	25,600	**26.13**	3.11	6	10.10	16.43	83.57

and 2 empirical false positives over 10^6 fresh probes across the $1, 4, 16$, and 64-block configurations (2.0×10^{-6} empirical rate at 64 blocks vs. analytical 64-block union bound $9.24 \times 10^{-7} \leq 2^{-20}$). Over a live 5-node WebRTC/SCTP

Table VII
CRDT LATTICE OPERATIONS, MERKLE PROVENANCE, AND 1-HOUR LEASEHEARTBEAT SAVINGS

Part A: CRDT `join` / `retire` & Provenance (`rings-core`)			
Operation	n/B	Lat. (μs)	Size / Config
CRDT `join` / `retire`	16	17.7 / 0.34	50% ovl, 25% tomb
CRDT `join` / `retire`	64	113.7 / 0.84	Normalized join
CRDT `join` / `retire`	256	431.9 / 3.59	Factored retire
CRDT `join` / `retire`	1024	2,173.7 / 10.2	Max carrier elements
`DataProv.` sign / vfy	1	53.7 / 135.2	495 B proof
`DataProv.` sign / vfy	16	68.7 / 189.9	4, 320 B Merkle
`DataProv.` sign / vfy	64	113.4 / 778.4	20, 653 B Merkle

Part B: 1-Hour Registry (I=30 s, Λ=600 s, **rot**=540 s)			
N_{reg}	Leg. Tmb / Rd	HB Tmb / Rd	Wire KiB (Leg. → HB)
10	1,200 / 1,210	**0 / 0**	892.2 → **476.2** (-46.6%)
50	6,000 / 6,050	**0 / 0**	4,461.1 → **2,381.0** (-46.6%)
100	12,000 / 12,100	**0 / 0**	8,922.2 → **4,762.0** (-46.6%)

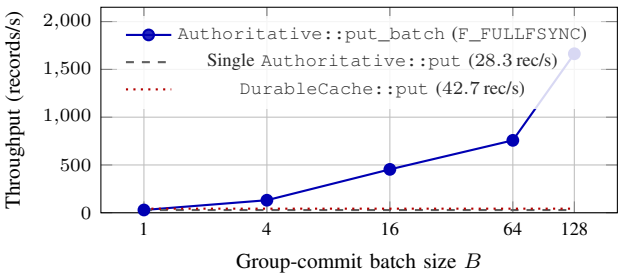

Figure 9. APFS `F_FULLFSYNC` durability throughput across group-commit batch sizes $B \in [1, 128]$ (`FileStorage`). Coalescing amortizes `F_FULLFSYNC` from 35.386 ms/rec (28.3 rec/s) to 0.601 ms/rec (1 663.5 rec/s, 58.86×).

loopback circuit, 8 MiB TCP downloads average 1.045 ± 0.003 **MB/s** (8.03 ± 0.03 s, 3 runs) at 16 KiB and 0.744 ± 0.047 **MB/s** (11.32 ± 0.75 s, 6 runs) at 64 KiB, where each cell consumes 4× the unit admission budget under constant link pacing and incurs multi-chunk SCTP hysteresis. Finally, `rings-gateway` scales cleanly from 64 to 1 024 concurrent multiplexed flows (Table X), delivering 10.68–14.69 MiB/s with sub-millisecond first-flow setup (0.029–0.113 ms) and 8.03–87.88 MB peak RSS.

H. Analytical and Operational Complexity Bounds

Table XI summarizes the closed-form analytical and operational bounds verified across the Rings 1.0 implementation.

I. Limitations and Threats to Validity

Four methodological boundaries qualify the empirical and verification results reported in this section:

- *Single-host local execution environment:* Throughput and durability benchmarks ran on one Apple M1 Max workstation using in-memory overlay simulations, local APFS `F_FULLFSYNC` barriers, and a 5-node loopback WebRTC/SCTP harness; geo-distributed WAN latency, asymmetric packet loss, hairpinned NAT traversal, and multi-browser Wasm/IndexedDB variance are deferred to multi-host testbed evaluation.

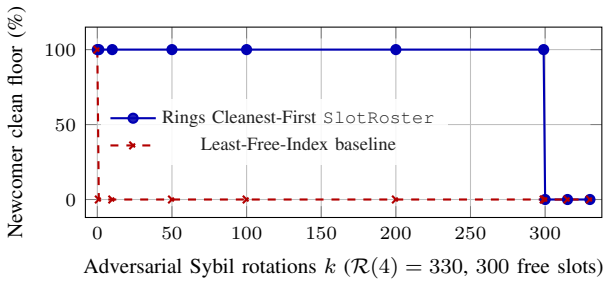

Figure 10. Newcomer clean-floor survival under Sybil slot rotation (`bench_paper_slot_roster_and_accounting`).

Table VIII
SLOTACCOUNTING MICROBENCHMARK LATENCY

Items $\lvert T\rvert$	Admit (ns/op)	Chg/Dischg (ns/op)	`floor_use` (μs)
100	11.3	118.3	2.04
1,000	10.8	163.2	5.95
5,000	11.0	188.0	5.87

- *Static-ring scaling vs. continuous live churn:* Figure 8 quantifies range-proved convergence on fixed N-node rings up to $N = 1\,024$; under sustained high churn, overlapping epoch invalidations require range revalidation (Section IV-B) alongside the bridge delegation fallback (Theorem 1).
- *Non-transitive clique capacity:* Passive 1-hop `TopoInfo` hints guarantee 100% cross-clique delivery without loops for $C_{\text{sub}} \leq K = 4$, remaining strictly loop-free (0% loops) beyond $2K = 8$ articulation peers (Table VI); larger Symmetric-NAT or UDP-blocked cliques form direct WebRTC edges without centralized servers via Rings' DHT-discovered stateless cryptographic UDP/TCP TURN (`rings-turn`) and split-trust multi-hop `onion::udp` Onion TURN (`rings-turn-service`).
- *Finite-state model-checking bounds:* Stateright/BFS verification exhaustively checks bounded configurations and finite queue depths (Table IV), whereas unbounded majority Sybil capture and economic equilibrium fall outside Rings 1.0 and rely on the companion DRanking 2.0 admission layer (Section IX).

XI. RELATED WORK

Structured Overlays and Non-Transitive NAT Routing. Distributed hash tables give Rings its closest technical ancestry. IPFS and libp2p adopt Kademlia-derived XOR routing [2], [4], [54], while Veilid [9] and I2P unify XOR DHTs with encrypted safety/private routes or garlic tunnels across desktop and browser/Wasm runtimes; all excel at content-addressed or tunneled lookup but lack Chord's clockwise interval owner law required for affine replica sets, age-gated chain custody, and contiguous successor-list maintenance. Rings adopts Chord's circular geometry and range lemma [1], [5], builds directly on Zave's CorrectChord safety substrate [6], [11], applies Bamboo-style exponential timeout backoff [14] to stabilize slow WebRTC peers, and solves non-transitive NAT/firewall cliques [7], [15] without unstructured gossip or virtual fin-

Table IX
SPHINX/AEZ V5 ONION PIPELINE, CHACHA20-POLY1305 COMPARISON, SLICED-BLOOM REPLAY, AND 5-NODE WEBRTC LOOPBACK (`-RELEASE`)

Relay Peel ($C_{16\text{KiB}}$)		**AEZ v5 vs. ChaCha20-Poly1305 (MiB/s) & Build**						**Sliced-Bloom** (10^6 **probes**)			
Op	μs	Class (Carry B)	AEZ τ=16 E/D	AEZ τ=0 E/D	ChaCha E/D	H	Build (μs)	Blks (Tags)	MiB	Ins/Qry (ns)	FP
ECDH $d \cdot \alpha$	42.5	16 KiB (13,465)	1348.1 / 1441.3	1529.8 / 1612.0	395.9 / 390.4	5	451.4 / 489.1	1 (16k)	0.07	1058.6 / 1049.6	0
Blind $z \cdot \alpha$	32.7	64 KiB (62,617)	1651.8 / 1658.3	1665.6 / 1666.0	393.1 / 393.0	8	731.0 / 750.1	4 (65k)	0.29	1081.8 / 1082.9	0
HKDF sched.	0.7	256 KiB (259,225)	1650.7 / 1584.6	1651.0 / 1655.5	390.7 / 379.5	11	997.1 / 1027.0	16 (262k)	1.17	1177.8 / 1288.8	0
PRG ChaCha	4.7	1 MiB (1,045,657)	1614.7 / 1601.7	1632.5 / 1631.2	386.0 / 387.3	14	1265.1 / 1294.6	64 (1.05M)	4.70	1548.4 / 2029.4	2
MAC HMAC	1.4	**Header Peel (real):** 94.6 μs (10, 568/s); **Whole Cell:** 108.0 μs (9, 257/s); **Build:** hdr / full cell ($n = 1..4$)									
KDF + AEZ	7.9	**5-Node WebRTC 8 MiB Download:** b=16 KiB: $\mathbf{1.045 \pm 0.003}$ MB/s (8.03 s, 3 runs); b=64 KiB: 0.744 ± 0.047 MB/s (11.32 s, 6 runs)									

Table X
GATEWAY CONCURRENT FLOW SCALING (`RINGS-GATEWAY`, 1024 B/FLOW)

Flows	First Setup (ms)	Transfer (ms)	App. MiB/s	Peak RSS (MB)
64	0.113	4.43	14.106	8.03
256	0.051	18.18	13.751	24.07
512	0.029	34.05	14.686	45.38
1,024	0.032	93.66	10.677	87.88

Table XI
ANALYTICAL AND OPERATIONAL BOUNDS OF RINGS 1.0

Metric / Subsystem Quantity	**Rings 1.0 Bound**
Routed finger lookups per full pass	$\le m_{\text{runs}} \le \min(L, \lvert V\rvert)$, $\mathbb{E}[m_{\text{runs}}] = O(\log \lvert V\rvert)$
Local finger range	0 routed lookups (stabilization proof)
$[0, \min(L-1, \lfloor \log_2 \delta_n(h) \rfloor)]$	
Report connection fan-out (stab / sync)	$\le K + 1$ / $\le K$ per claimed report
Non-transitive bridge-delegated route	$\le 2\lvert V\rvert$ hops worst-case; ≤ 3 hops ($C_{\text{sub}} \le K$)
CRDT carrier tombstones / wire size	$T_{\max} = 1024$; $W = 33.73\,\text{MB} \le 60\,\text{MB}$
Registry heartbeat tombstones / reads	0 tombstones, 0 reads (`Extend` lease)
Group commit `put*` flushes / latency	1 drive barrier per batch; 0.601 ms/rec (B=128)
Replicated relay chain repair delay	$D \le T_d + r\delta$ ($T_d \le 10$ s silent, 0 s graceful)
Connection slot roster capacity $\mathcal{R}(s)$	$2(F + s + 1) = 330$ ($s = 4$) or 328 ($s = 3$)
Node-wide frame liability $B_F(s)$	$\mathcal{R}(s)(W + 3) + \mathcal{P}_F = 6\,526$ ($s = 4$); $6\,488$ ($s = 3$)
Open chunk reassembly buffer $B_O(4)$	$330 \times 256\,\text{KiB} + 128\,\text{MiB} = 210.5\,\text{MiB} \le 256\,\text{MiB}$
Onion loop hops $H(n, s)$ / header $\lvert\chi\rvert$	$H = (s + 1)n + s \le 14$; $\lvert\chi\rvert = 2919$ B ($\ell = 205$ B)
Onion replay Bloom filter FP / memory	$\le 2^{-20}$ ($K = 26, M = 23\,680$); 4.70 MiB (64 blks)
Onion 16 KiB SURB batch / rate	$k = 4$ (5 credits/loop); 1.32 MB/s link ceiling (1.045 MB/s live)

gers [16], [17] via bounded multi-predecessors and single-crossing bridge delegation.

Relay Mixnets, Wide-Block Ciphers, and Typed Pipelines. Existing anonymous networks decouple routing from structured ownership: Tor provides low-latency telescoping circuits without fixed packet padding [3], [47], whereas Mixminion, Loopix, and Nym introduce Sphinx packets, SURBs, and Poisson/constant cover loops over dedicated mix directories [34], [35], [43], [44], [55]. Rings unifies Sphinx fixed-length headers, client-sealed Loopix loops, and AEZ v5 wide-block encryption [36], [42], [56] into its DHT as an Arrow/Kleisli effect pipeline [37]–[41], [57] whose hops discover Σ-algebraic service descriptors via the ring itself.

Conflict-Free Storage, Crash Consistency, and Browser Runtimes. On storage and durability, Rings extends state-based and delta-state CRDTs [18], [20], [21] and Hybrid Logical Clocks [13], [19] with factored removal separation, per-writer Merkle provenance, linearizable pure-read caches [22], and platform-verified crash barriers [23]–[26] tailored to asymmetric browser churn [32], [58], [59]. Table XII contrasts Rings 1.0 across all six architectural dimensions.

XII. CONCLUSION

Rings is an algebraic structured peer-to-peer network: structured because ownership, range-proved finger convergence, non-transitive WebRTC bridge routing, affine CRDT storage replication, and age-gated relay custody are defined over CorrectChord's circular overlay, and algebraic because each layer preserves an explicit carrier, authority lattice, slot-accounting law, or Kleisli morphism rather than hiding semantics inside ad hoc network procedures.

$$\begin{aligned}
\mathsf{Rings} &= (R, \mathcal{R}_N, \mathbf{Set}/R, \Omega_N, \mathsf{T}_A^{\mathcal{E}}, \Pi, \mathbb{G}, \mathcal{X}).\\
\mathsf{Overlay} &= \mathsf{CorrectChord} + \mathsf{RangeFinger} + \mathsf{BridgeRoute},\\
\mathsf{Storage} &= \mathsf{E}(x) \sqcup \mathsf{R}(x),\ \tau_{\text{HLC}} = (w, c, o),\ T_{\max} = 1024,\\
\mathsf{Durability} &= (c, m) \in C \times M,\ \text{put}^*,\ (\mathcal{W}_s, \mathcal{K}_s),\\
\mathsf{Admission} &= \mathsf{SlotRoster}(\mathcal{R}),\ f_T + \lfloor \mathcal{P}_T / \mathcal{R}(s) \rfloor,\\
\mathsf{Privacy} &= \mathbf{KI}(M)\ \text{Sphinx/AEZ-v5}\ (\Sigma,\ H \le 14),\\
\mathsf{Trust} &= \mathsf{ServiceReceipt} \to \mathsf{DRanking\ 2.0}.
\end{aligned}$$

REFERENCES

[1] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '01)*. ACM, 2001, pp. 149–160.

[2] P. Maymounkov and D. Mazières, "Kademlia: A peer-to-peer information system based on the XOR metric," in *Peer-to-Peer Systems*, ser. Lecture Notes in Computer Science, vol. 2429. Springer, 2002, pp. 53–65.

[3] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation onion router," in *13th USENIX Security Symposium (USENIX Security 04)*. San Diego, CA: USENIX Association, Aug. 2004, https://www.usenix.org/conference/13th-usenix-security-symposium/tor-second-generation-onion-router.

[4] J. Benet, "IPFS - content addressed, versioned, P2P file system," arXiv:1407.3561 [cs.NI], 2014, https://arxiv.org/abs/1407.3561.

[5] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for internet applications," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 17–32, 2003.

Table XII
ARCHITECTURAL COMPARISON OF RINGS 1.0 AGAINST DEPLOYED P2P OVERLAYS AND ANONYMITY NETWORKS [1]–[4], [6], [9], [17], [54], [55]

System	Routing / NAT Model	Storage / Durability	Admission / DoS	Onion Privacy	Verification
Kademlia / libp2p	XOR buckets; relay fallback	Content-addressed / ad hoc	Peer scoring / rate limits	None (external TLS/Noise)	Empirical / property tests
Veilid [9] / I2P	XOR DHT; safety/garlic tunnels	Signed DHT records / leases	Route limits / hashcash	Multi-hop safety/private routes	Unit / network integration tests
Chord / RELOAD	Circular R; virtual fingers	Successor replication	Cert-bound / static	None	TLA+ / Alloy (Zave)
Tor / Loopix / Nym	Directory mix/circuit paths	None (relay-only)	Guards / staking	Sphinx / SURBs / cover	Game-based crypto proofs
Rings 1.0	CorrectChord + range fingers + bridge	Factored CRDT + HLC + $C \times M$	Cleanest-first $\mathcal{R}$ + 5-table floors	Kleisli Sphinx + AEZ v5 loops	9.50M Stateright states + 54 mutants

[6] P. Zave, "Reasoning about identifier spaces: How to make Chord correct," *IEEE Transactions on Software Engineering*, vol. 43, no. 12, pp. 1144–1156, 2017, https://arxiv.org/abs/1610.01140.

[7] M. J. Freedman, K. Lakshminarayanan, S. Rhea, and I. Stoica, "Non-transitive connectivity and DHTs," in *Proceedings of the 2nd Conference on Real, Large Distributed Systems (WORLDS '05)*. USENIX Association, 2005, pp. 55–60.

[8] J. Nadal, "Stateright: Model checking for implementing distributed systems," https://www.stateright.rs/, 2026, accessed 2026-07-04.

[9] C. Rioux and K. Bowden, "The internals of Veilid, a new decentralized application framework," DEF CON 31 Technical Release & Documentation, https://veilid.com/, 2023, source repository: https://gitlab.com/veilid/veilid; accessed 2026-11-25.

[10] R. J. Kung, "The DRanking protocol: Verifiable distributed ranking and sybil-bounded admission for structured peer-to-peer networks," Rings Network Technical Report (Companion Specification), 2026, https://rings.rs/papers/dranking/.

[11] P. Zave, "How to make Chord correct (using a stable base)," arXiv:1502.06461 [cs.DC], 2015, https://arxiv.org/abs/1502.06461.

[12] K. R. Davis, B. Peabody, and P. J. Leach, "Universally unique identifiers (UUIDs)," Internet Engineering Task Force, RFC 9562, 2024.

[13] C. J. Fidge, "Timestamps in message-passing systems that preserve the partial ordering," in *Proceedings of the 11th Australian Computer Science Conference*, 1988, pp. 56–66.

[14] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz, "Handling churn in a DHT," in *Proceedings of the 2004 USENIX Annual Technical Conference*. USENIX Association, 2004, pp. 127–140.

[15] B. Ford, P. Srisuresh, and D. Kegel, "Peer-to-peer communication across network address translators," in *Proceedings of the 2005 USENIX Annual Technical Conference*. USENIX Association, 2005, pp. 179–192.

[16] D. A. Bryan, B. B. Lowekamp, and C. Jennings, "SOSIMPLE: A serverless, standards-based, P2P SIP communication system," in *Proceedings of the First International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications (AAA-IDEA '05)*. IEEE, 2005, pp. 42–49.

[17] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne, "Resource location and discovery (RELOAD) base protocol," Internet Engineering Task Force, RFC 6940, 2014.

[18] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski, "Conflict-free replicated data types," in *Stabilization, Safety, and Security of Distributed Systems*, ser. Lecture Notes in Computer Science, vol. 6976. Springer, 2011, pp. 386–400.

[19] S. S. Kulkarni, M. Demirbas, D. Madeppa, B. Avva, and M. Leone, "Logical physical clocks and consistent snapshots in globally distributed databases," in *Principles of Distributed Systems (OPODIS 2014)*, ser. Lecture Notes in Computer Science, vol. 8878. Springer, 2014, pp. 17–32.

[20] P. S. Almeida, A. Shoker, and C. Baquero, "Delta state replicated data types," *Journal of Parallel and Distributed Computing*, vol. 111, pp. 162–173, 2018.

[21] N. Preguiça, "Conflict-free replicated data types: An overview," arXiv:1806.10254 [cs.DC], 2018, https://arxiv.org/abs/1806.10254.

[22] M. P. Herlihy and J. M. Wing, "Linearizability: A correctness condition for concurrent objects," *ACM Transactions on Programming Languages and Systems*, vol. 12, no. 3, pp. 463–492, 1990.

[23] T. S. Pillai, V. Chidambaram, R. Alagappan, S. Al-Kiswany, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau, "All file systems are not created equal: On the complexity of crafting crash-consistent applications," in *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI '14)*. USENIX Association, 2014, pp. 433–448.

[24] Apple, "`fsync(2)` and `fcntl(2):F_FULLFSYNC`," macOS manual pages, 2026, accessed 2026-10-23.

[25] Microsoft, "`MoveFileExW` function (`winbase.h`): `MOVEFILE_WRITE_THROUGH`," https://learn.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-movefileexw, 2026, accessed 2026-10-23.

[26] W3C Web Applications Working Group, "Indexed database API 3.0," W3C Editor's Draft, https://w3c.github.io/IndexedDB/, 2026, section "Transaction durability"; accessed 2026-10-23.

[27] Chrome for Developers, "A change to the default durability mode in IndexedDB," https://developer.chrome.com/blog/indexeddb-durability-mode-now-defaults-to-relaxed, 2024, default changes from strict to relaxed in Chrome 121; accessed 2026-10-23.

[28] Mozilla, "IndexedDB parent actors (`ActorsParent.cpp`)," https://github.com/mozilla-firefox/firefox/blob/main/dom/indexedDB/ActorsParent.cpp, 2026, `BeginWriteTransaction` and `GetDefaultSynchronousMode`; accessed 2026-10-23.

[29] WebKit, "IndexedDB SQLite backing store (`SQLiteIDBBackingStore.cpp`)," https://github.com/WebKit/WebKit/blob/main/Source/WebCore/Modules/indexeddb/server/SQLiteIDBBackingStore.cpp, 2026, `commitTransaction`; accessed 2026-10-23.

[30] SQLite, "Write-ahead logging, and PRAGMA synchronous," https://www.sqlite.org/wal.html, 2026, accessed 2026-10-23.

[31] MDN Web Docs, "Storage quotas and eviction criteria," https://developer.mozilla.org/en-US/docs/Web/API/Storage_API/Storage_quotas_and_eviction_criteria, 2026, accessed 2026-10-23.

[32] C. Blake and R. Rodrigues, "High availability, scalable storage, dynamic peer networks: Pick two," in *Proceedings of the 9th Workshop on Hot Topics in Operating Systems (HotOS IX)*, 2003, pp. 1–6.

[33] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[34] G. Danezis and I. Goldberg, "Sphinx: A compact and provably secure mix format," in *Proceedings of the 30th IEEE Symposium on Security and Privacy*. IEEE, 2009, pp. 269–282.

[35] A. M. Piotrowska, J. Hayes, T. Elahi, S. Meiser, and G. Danezis, "The Loopix anonymity system," in *Proceedings of the 26th USENIX Security Symposium*. USENIX Association, 2017, pp. 1199–1216.

[36] V. T. Hoang, T. Krovetz, and P. Rogaway, "Robust authenticated-encryption: AEZ and the problem that it solves," in *Advances in Cryptology – EUROCRYPT 2015*, ser. Lecture Notes in Computer Science, vol. 9056. Springer, 2015, pp. 15–44.

[37] E. Moggi, "Notions of computation and monads," *Information and Computation*, vol. 93, no. 1, pp. 55–92, 1991.

[38] G. Plotkin and J. Power, "Algebraic operations and generic effects," *Applied Categorical Structures*, vol. 11, no. 1, pp. 69–94, 2003.

[39] J. Hughes, "Generalising monads to arrows," *Science of Computer Programming*, vol. 37, no. 1–3, pp. 67–111, 2000.

[40] S. Lindley, P. Wadler, and J. Yallop, "Idioms are oblivious, arrows are meticulous, monads are promiscuous," *Electronic Notes in Theoretical Computer Science*, vol. 229, no. 5, pp. 97–117, 2011.

[41] G. Plotkin and M. Pretnar, "Handlers of algebraic effects," in *Programming Languages and Systems (ESOP 2009)*, ser. Lecture Notes in Computer Science, vol. 5502. Springer, 2009, pp. 80–94.

[42] V. T. Hoang, T. Krovetz, and P. Rogaway, "AEZ v5: Authenticated encryption by enciphering," CAESAR competition submission, third round, 2017.

[43] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[44] G. Danezis, R. Dingledine, and N. Mathewson, "Mixminion: Design of a type III anonymous remailer protocol," in *Proceedings of the 2003 IEEE Symposium on Security and Privacy*. IEEE, 2003, pp. 2–15.

[45] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[46] P. S. Almeida, C. Baquero, N. Preguiça, and D. Hutchison, "Scalable Bloom filters," *Information Processing Letters*, vol. 101, no. 6, pp. 255–261, 2007.

[47] J. P. Degabriele and M. Stam, "Untagging Tor: A formal treatment of onion encryption," in *Advances in Cryptology – EUROCRYPT 2018*, ser. Lecture Notes in Computer Science, vol. 10822. Springer, 2018, pp. 259–293.

[48] J. A. Goguen and J. Meseguer, "Security policies and security models," in *Proceedings of the 1982 IEEE Symposium on Security and Privacy*. IEEE, 1982, pp. 11–20.

[49] J. R. Douceur, "The sybil attack," in *Peer-to-Peer Systems, First International Workshop (IPTPS 2002)*, ser. Lecture Notes in Computer Science, vol. 2429. Springer, 2002, pp. 251–260.

[50] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab, Tech. Rep. 1999-66, 1999, http://ilpubs.stanford.edu:8090/422/.

[51] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust algorithm for reputation management in P2P networks," in *Proceedings of the 12th International Conference on World Wide Web (WWW 2003)*. ACM, 2003, pp. 640–651.

[52] A. Kothapalli, S. Setty, and I. Tzialla, "Nova: Recursive zero-knowledge arguments from folding schemes," in *Advances in Cryptology – CRYPTO 2022*, ser. Lecture Notes in Computer Science, vol. 13510. Springer, 2022, pp. 359–388.

[53] L. Lamport, *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers*. Addison-Wesley, 2002.

[54] Protocol Labs, "libp2p: A modular network stack," 2026, https://libp2p.io/.

[55] C. Diaz, H. Halpin, and A. Kiayias, "The Nym network: The next generation of privacy infrastructure," Whitepaper, 2021, https://nym.com/nym-whitepaper.pdf.

[56] R. Anderson and E. Biham, "Two practical and provably secure block ciphers: BEAR and LION," in *Fast Software Encryption (FSE 1996)*, ser. Lecture Notes in Computer Science, vol. 1039. Springer, 1996, pp. 113–120.

[57] A. Mokhov, G. Lukyanov, S. Marlow, and J. Dimino, "Selective applicative functors," *Proceedings of the ACM on Programming Languages*, vol. 3, no. ICFP, pp. 90:1–90:29, 2019.

[58] H. Alvestrand, "Overview: Real-time protocols for browser-based applications," RFC 8825, 2021, https://www.rfc-editor.org/rfc/rfc8825.

[59] WebAssembly Working Group, "WebAssembly core specification," W3C Recommendation, 2019, https://www.w3.org/TR/2019/REC-wasm-core-1-20191205/.