Structure of Verification Integration

This document maps the formal theorems to specific sections of the SHIFT paper, providing detailed content outlines for each.

1. Introduction (Page 1)

Goal: Frame SHIFT's design choices (supporting Writes/Simple, rejecting Atomics) as theoretical necessities, not implementation gaps.

- **Content Block 1: The Core Question**
  - **Insert Location:** Replace paragraph starting "However, we identify a fundamental constraint…"
  - **Draft Text:** "However, a fundamental question arises: what are the theoretical limits of transparent cross-NIC failover? We prove that SHIFT achieves precisely what is possible under the transparency constraint…"
  - **Key Concepts:** Mention the three barriers: Indistinguishability (ACK vs Packet loss), Non-idempotency (Atomics/Queue effects), and the Consensus Barrier (CN=1).
- **Content Block 2: Contribution List**
  - **Insert Location:** New Item (2) in the contribution list.
  - **Draft Text:** "We formally prove that SHIFT's coverage is optimal: transparent failover for atomic and uncoordinated two-sided operations is impossible due to the consensus hierarchy barrier. Supporting these requires receiver-side coordination (SHIFT's handshake) or persistent metadata. All proofs are mechanically verified in Rocq 9.0."

2. Section 3.1: Insight: Boundary of Cross-NIC Fault Tolerance (Page 4)

Goal: Provide the "Sender's Dilemma" intuition, formalized by the Indistinguishability Theorem, leading to the classification of operations.

- **Subsection 3.1.1: The Transparency Barrier**
  - **Context (Why Transparency?):** For One-Sided RDMA (WRITE/READ), the receiver CPU is bypassed ("silent"). Thus, any failover logic **must** be transparent—residing entirely on the sender—because we cannot execute code on the receiver to track state or send application-level ACKs.
  - **Paragraph 1: Intuition (The Dilemma).**
    - Start with: "When a network anomaly causes a timeout, the sender faces two indistinguishable scenarios…"
    - Explain: Scenario A (Packet Loss -> Retry Needed) vs. Scenario B (ACK Loss -> Retry Dangerous).
  - **Paragraph 2: Formalization (Theorem 1).**
    - State: "**Theorem 1 (Indistinguishability).** For any transparent overlay, there exist executions with identical sender observations but opposite correctness requirements."
    - Detail: Construct traces T_1 (Packet Loss) and T_2 (ACK Loss) where $\text{SenderView}(T_1) = \text{SenderView}(T_2)$.
  - **Paragraph 3: Implication (The Consensus Barrier).**
    - State: "Resolving this ambiguity is equivalent to solving Consensus."
    - Argument: Since transparent mechanisms can only **read** remote state (CN=1), they cannot solve the 2-Consensus problem required to agree on "Did it commit?". Thus, ambiguity is inherent.
- **Subsection 3.1.2: The Optimality of SHIFT**
  - **Table:** Insert the "Class Partition" table (Idempotent Writes | Atomics | Two-Sided).
  - **Analysis:**
    - **Writes:** Supported. Safe by definition/protocol (idempotent overwrite).
    - **Atomics:** Rejected. Unsafe because FADD/CAS are non-idempotent (Thm 2). Retry corrupts state.
    - **Two-Sided:** Handshake Required. Since Two-Sided RDMA involves the receiver CPU, we can relax the transparency constraint **internally**. The handshake synchronizes queue state, mitigating the "Queue Sliding" impossibility (Thm 2b).

3. Section 3.2: Design Challenges (Page 5)

Goal: justify specific SHIFT mechanisms using the theorems proved in 3.1.

- **Challenge 1: Passive Switching (Writes & Atomics)**
  - **Mechanism:** "Best-Effort Retry" for Writes.
  - **Justification:** Since we proved we **cannot** know if the packet failed (Thm 1), we must pick a default. Retry is the only choice that preserves liveness. For Writes, this is safe (idempotent).
  - **Mechanism:** "Error/Abort" for Atomics.
  - **Justification:** Since we cannot know if it failed, and retry is provably unsafe (Thm 2), the **only** correct transparent action is to abort and notify the application.
- **Challenge 2: Active Switching (Two-Sided Ops)**
  - **Mechanism:** The 3-Way Handshake.
  - **Problem (Queue Sliding):** Explain that blind retry of SEND/RECV causes queue misalignment. Even if the data is safe, the **consumption** of the Receive Queue element is non-idempotent.
  - **Theorem Reference: Theorem 2 (Queue Sliding Case).** "Retrying a SEND operation consumes an additional Receive WQE, desynchronizing the message-to-buffer mapping."
  - **Solution:** The handshake explicitly resynchronizes sequence numbers and queue indices between sender/receiver SHIFTLib instances. This breaks transparency **internally** to preserve it for the application.

4. Appendix C: Formal Verification (New Section)

Goal: The rigorous "Territory" behind the "Map" in Section 3.

- **C.1 System Model**
  - Define $State = (Memory, ReceiveQueue)$.
  - Define `Transparency` as a function of local history only.
- **C.2 Indistinguishability (Proof of Thm 1)**
  - Present the formal trace construction:
    - T_1: `[Send(W), PacketLost, Timeout]`
    - T_2: `[Send(W), Execute(W), AckLost, Timeout]`
  - Show $\text{SenderView}(T_1) = \text{SenderView}(T_2)$.
- **C.3 Non-Idempotency (Proof of Thm 2)**
  - **Case A: Atomics.** Show FADD double-count and CAS ABA problem.
  - **Case B: Queue Sliding.** Formalize the queue consumption.
    - Lemma: $length(Q_{\text{after\_retry}}) = length(Q_{\text{initial}}) - 2$.
    - Result: Message M_2 lands in buffer B_3 instead of B_2.
- **C.4 Consensus Barrier (Proof of Thm 3)**
  - Reduce "Failover Decision" to "2-Process Consensus".
  - Cite Herlihy: Read-Only ops (transparency) have CN=1.
  - Conclusion: Transparency implies impossibility of perfect failover for non-idempotent ops.

Summary of Dependencies

- `paper_section_proof.typ` provides the exact text for Section 1 and 3.1.
- `theory_appendix.typ` provides the definitions and proof sketches for Appendix C.