



The Classification Problem

Justin H. Kuiper, CISSP

2026-05-01



Non Sequitur Publishing

THE CLASSIFICATION PROBLEM

The Implications of Edge Degraded Ops — P8 of 11

Author:	Justin H. Kuiper, CISSP	ORCID:	0009-0008-7099-3286
Publisher:	Non Sequitur Publishing		nonsequitur.tech
Version:			v0.1-seed
Date:			2026
DOI:	Pending — Zenodo depositX		

COPYRIGHT

© 2026 Justin H. Kuiper, CISSP. Published by Non Sequitur Publishing.

This is a v0.1-seed preprint. All rights reserved. No part of this document may be reproduced, distributed, or transmitted in any form without written permission from the author, except for brief quotations in scholarly reviews or non-commercial educational use with full attribution.

Do not cite this version without confirming whether a revised, DOI-stamped edition has been issued at nonsequitur.tech.

ABSTRACT

Classification in enterprise information systems is a labeling problem: a human or system attaches a sensitivity marker to data at creation, and access control policy enforces that marker at retrieval. This model fails at the tactical edge under DDIL conditions not because it was improperly implemented, but because the model's foundational assumptions — that classification decisions are made at a single authoritative point, that labels remain stable across the data lifecycle, that enforcement happens at access time in a connected environment — do not hold when data is created on one disconnected node, transmitted over an intermittent link via DTN custody transfer, merged with a conflicting copy via CRDT reconciliation, routed by an AI Supervisor operating on a locally-cached policy, and eventually validated against a shore-authoritative classification baseline that the node has not seen for seventy-two hours.

This paper argues that classification at the tactical edge is not a labeling problem — it is a state-coherence problem. The classification of a data object is not an attribute painted onto it at origin; it is a property that participates in the distributed-system primitives of the substrate: replication, conflict resolution, custody transfer, AI-mediated routing, and governance



enforcement. When that property is treated as a label rather than as distributed state, three compounding failure modes emerge: classification drift, derivative sensitivity collapse, and enforcement against a stale classification baseline. Together, these failure modes produce the condition the HGC³AE² framework terms confident misalignment applied at the classification layer — a system operating in apparent compliance with its classification rules while enforcing a picture of data sensitivity that no longer corresponds to the operational ground truth.¹

The paper develops a substrate-grounded classification architecture in which the write-ahead log carries classification lineage as a first-class write entry, CRDT merge rules include classification merge semantics, DTN bundle custody transfers carry cryptographically-signed classification tags, and the AI Supervisor reasons over a classification-aware operational model trained on cataloged reclassification history. The §6 Governor Application grounds this architecture in the HGC³AE² framework: classification routing policy belongs to the H half; classification lineage cataloging and curriculum-building belong to the C³ half; classification tag authentication and fail-closed CDS enforcement belong to the AE² half.

§1 — OPERATIONAL PICTURE: CLASSIFICATION UNDER DDIL CONSTRAINT

To understand the classification problem at the tactical edge, begin with an operational scenario that the enterprise classification literature was not designed to address.

A DDG is operating in a contested maritime environment with intermittent SATCOM access and degraded UHF tactical links. Over a seventy-two-hour period, its node has accumulated state from a mix of sources: sensor fusion outputs generated locally by the ship's own systems, targeting data received via an STN bundle from an E-2D Hawkeye before the link degraded, ISR imagery thumbnails that arrived via MUOS before the satellite passed below the horizon, and logistics records synchronized during a twelve-minute SATCOM window forty-eight hours ago. Each of these objects was classified at origin by the originating system — CONFIDENTIAL//REL TO USA, SECRET//NOFORN, TOP SECRET//SI//NOFORN, UNCLASSIFIED//FOUO — and each carried its classification tag when it arrived on the ship's node. The node's WAL records every write. The CRDT layer has merged several objects where the same operational picture was updated by multiple sources before the links degraded.

The ship is now approaching a mission window. The AI Supervisor must make routing decisions: which state to push to the ship's battle management system, which state to queue for transmission over the next link window, which state to withhold pending human review. The CDS on the node is enforcing boundary crossings. But three things have changed since the classification tags on those objects were first set.

First, the targeting data from the E-2D was subsequently updated by a FLASH message that arrived via HF before the ship entered the degraded window. The FLASH message carried a RELEASABILITY change: the data had been upgraded to TS//HCS-P. The FLASH message arrived on a different node in the ship's network — the combat direction center — but not on the primary tactical node before its link degraded. The tactical node's WAL contains the original custody entry; it does not contain the reclassification event. The classification tag on the targeting data, as far as the tactical node knows, is SECRET//NOFORN. The classification tag as known to anyone who received the FLASH is TS//HCS-P.

¹ Justin H. Kuiper, CISSP, "Mitigating Confident Misalignment in Agentic Systems: The HGC³AE² Framework," Non Sequitur Publishing, 2026, v1.0-preprint, §2 (confident misalignment as the dominant failure mode of agentic systems operating on stale context).



Second, the ISR imagery thumbnails were processed by the ship's onboard AI inference system, which produced a derived object: a fused track combining imagery-derived geolocation with the targeting data. Under derivative classification rules, this fused object inherits the highest classification of its source materials. If the targeting data is SECRET//NOFORN, the fused track is SECRET//NOFORN. If the targeting data has been reclassified to TS//HCS-P, the fused track must also carry TS//HCS-P — but the AI system that created it did not know about the reclassification, so it created a SECRET//NOFORN object from what it thought were SECRET//NOFORN sources.

Third, the logistics records, originally UNCLASSIFIED//FOUO, were stored in the WAL alongside operational planning data. The CRDT layer, during a local merge cycle, produced a compound object that combines a logistics query result with a mission planning parameter. Neither object, taken alone, is above FOUO. But the combination — which logistics route, on which date, supporting which mission objective — may constitute an aggregation that exceeds the classification of its individual components. The system does not have a mechanism to detect this aggregation-driven sensitivity uplift.

The AI Supervisor is now attempting to route bundles across the CDS boundary. The CDS is enforcing classification tags. The tags are wrong in at least two of these three cases. The enforcement is, in the formal sense of the word, compliant — the CDS is doing what it was told to do — but it is enforcing a fiction. This is the classification problem at the tactical edge.

It is worth being precise about why this scenario is not simply a failure of data freshness or synchronization. Those are the enterprise answers to this scenario: improve synchronization, reduce the window of divergence, re-query before acting. At the tactical edge, synchronization is not a tunable parameter. Connectivity is not reliably available. The window of divergence is not a defect to be engineered away — it is an operational condition that the architecture must treat as a first-class reality. A classification architecture that can only maintain integrity when fully connected is not a classification architecture for DDIL operations. It is a classification architecture that has been deployed in an environment where its assumptions do not hold.²

The classification problem, then, is this: in a substrate where connectivity is intermittent, where state is replicated across nodes with different operational pictures, where objects are derived and merged by automated processes, and where the AI Supervisor makes routing decisions at operational tempo, the classification integrity of data objects cannot be maintained by labeling at creation and enforcing at access. It requires classification to be a first-class property of the substrate's distributed state model — tracked in the WAL, incorporated into CRDT merge semantics, carried through DTN custody chains, and governed by the H half before enforcement by the AE² half.

The remainder of this paper develops the architecture for that model.

§2 — STRUCTURAL FAILURES: WHY CLASSIFICATION BREAKS UNDER DEGRADATION

The operational scenario in §1 is not an edge case. It is the structural consequence of deploying a classification architecture whose assumptions — single point of origin, stable labels, connected enforcement — are violated by DDIL conditions. This section characterizes the

² Kevin Fall, “A Delay-Tolerant Network Architecture for Challenged Internets,” *Proceedings of ACM SIGCOMM 2003*, 27–34, <https://doi.org/10.1145/863955.863960> (DDIL as structural condition, not transient fault).



failure modes precisely, so that the substrate primitives in §4 can be mapped onto them with specificity.

2.1 — Classification Drift

Classification drift occurs when the classification state of an object, as known to one node, diverges from the classification state of that same object as known to another node or as would be determined by the authoritative classification authority. Drift can be upward (the node believes an object is less sensitive than it actually is) or downward (the node believes an object is more sensitive than it currently is). Upward drift is the dangerous case: a node acting on an object it believes is SECRET when that object has been upgraded to TS//HCS-P may route it, transmit it, or share it in ways that violate the TS//HCS-P handling requirements.

Classification drift is structurally inevitable in a DDIL environment. Reclassification events — whether originating from the original classifier, from a derivative classifier upstream, or from a declassification or upgrade authority — are themselves messages that must traverse the same challenged links as any other data. In a connected environment, reclassification messages arrive promptly and nodes update their local classification state. In a DDIL environment, reclassification messages may be delayed by minutes, hours, or days. During the delay window, any node that holds the affected object operates on a stale classification label.³

The severity of drift is determined by two parameters: the magnitude of the classification change (FOUO → SECRET carries different consequences than SECRET → SECRET//NOFORN) and the duration of the divergence window. Enterprise classification systems do not design for extended divergence windows because enterprise environments do not operate with extended connectivity gaps. The tactical substrate must design for drift as a structural steady state, not a transient fault condition.

A second driver of classification drift is the WAL's custody record. When an object arrives on a node via DTN bundle transfer, the WAL records the custody event with the classification tag that the bundle carried. If the bundle was prepared before a reclassification event at the source node, the WAL entry carries the pre-reclassification tag. The WAL's record of that object is now divergent from the authoritative classification, but the WAL has no mechanism to update historical entries — that would violate the WAL's append-only guarantee, which is precisely the property that makes it a trustworthy ground truth. The resolution is not to make the WAL mutable, but to make classification state a separate, CRDT-managed property that can be updated through the normal append-append reconciliation process.⁴

2.2 — Derivative Sensitivity Collapse

Derivative classification rules require that a derived object inherit the classification of its most sensitive source material. This is a well-established principle in the IC's classification

3 Dorothy E. Denning, "A Lattice Model of Secure Information Flow," *Communications of the ACM* 19, no. 5 (1976): 236–243, <https://doi.org/10.1145/360051.360056> (the lattice model requires consistent classification ordering; drift violates the lattice's transitivity property).

4 C. Mohan, Don Haderle, Bruce Lindsay, Hamid Pirahesh, and Peter Schwarz, "ARIES: A Transaction Recovery Method Supporting Fine-Granularity Locking and Partial Rollbacks Using Write-Ahead Logging," *ACM Transactions on Database Systems* 17, no. 1 (1992): 94–162, <https://doi.org/10.1145/128765.128770> (WAL append-only property as ground truth; the resolution requires treating classification state as a separate log stream, not a mutation of existing entries).



system.⁵ The principle is tractable in a connected environment where the derivation provenance can be traced and the current classification of all source materials can be verified at derivation time. At the tactical edge, two conditions undermine it.

The first is provenance opacity. An object arrives on the tactical node via a multi-hop DTN custody chain. Its classification tag was set by the originating node. Between the origin and the current node, the object may have transited intermediary nodes that added contextual annotations, stripped metadata for bandwidth reduction, or merged the object with locally-available state. Each of these operations may have altered the object's effective sensitivity without altering its classification tag. By the time the object arrives on the tactical node, its derivation provenance may be partially or wholly opaque.

The second is the time-of-derivation problem. When the ship's AI inference system creates a fused track from multiple source objects, it records the classification of those sources at the time of derivation. If any source is subsequently reclassified, the derived object's classification tag is no longer accurate — but there is no automated mechanism to propagate the source reclassification to the derived object. In a connected environment with a central classification authority, derived object classification can be maintained by tracking derivation relationships centrally and triggering re-evaluation when sources change. In a disconnected environment with no central authority available, this mechanism is unavailable.⁶

The result is derivative sensitivity collapse: derived objects gradually lose their connection to their accurate classification, because the source classification lineage cannot be maintained under DDIL conditions. Derived objects created by the AI Supervisor — fused tracks, summary reports, planning products — are particularly vulnerable because the Supervisor may create many derived objects at high tempo, each with a complex derivation provenance that the substrate has not been designed to track.

2.3 — Enforcement Against a Stale Classification Baseline

Even when individual object classification tags are correct, enforcement may fail if the enforcement mechanism operates against a classification baseline that does not reflect current policy. Classification policy is not static. Control markings change, compartment caveat structures are revised, releasability designations are updated. In a connected environment, the CDS receives policy updates promptly. In a DDIL environment, the CDS may be operating against a classification policy baseline that is days or weeks old.

The consequence is enforcement against a stale baseline. The CDS is doing exactly what it was configured to do — enforcing the policy it was given — but the policy it was given no longer

⁵ Andrei Sabelfeld and Andrew C. Myers, "Language-Based Information-Flow Security," *IEEE Journal on Selected Areas in Communications* 21, no. 1 (2003): 5–19, <https://doi.org/10.1109/JSAC.2002.806121> (information flow security requires that derived values not exceed the security level of their inputs without explicit downgrading authority — the converse, that derived values inherit the maximum classification of their inputs, is the derivative classification principle).

⁶ Andrew C. Myers and Barbara Liskov, "A Decentralized Model for Information Flow Control," *Proceedings of SOSP 1997*, 129–142, <https://doi.org/10.1145/268998.266669> (decentralized flow tracking in environments where central authority is unavailable — the model developed here applies their principals-and-policies framework to the DDIL derivation provenance problem).



corresponds to the current authoritative policy. Objects that are now releasable may be withheld; objects that have had their releasability revoked may be shared.⁷

This failure mode is structurally analogous to the problem identified in the HGC³AE² framework as context integrity failure applied to the enforcement layer.⁸ The CDS's enforcement context — its classification policy baseline — has become stale in exactly the way that any context becomes stale in a DDIL environment. The solution is the same one the framework prescribes for context integrity failure: treat the classification policy baseline as dynamic state that must be managed, replicated, and reconciled through the substrate's state-management primitives, not as a configuration artifact that was correctly set at deployment time and assumed to remain valid indefinitely.

2.4 — Aggregation-Driven Sensitivity Uplift

The aggregation problem in classification is well-known in the database security literature.⁹ Individual data elements, each properly classified at a low level, may in combination produce an aggregation whose effective sensitivity exceeds the classification of any individual element. Database systems address this through aggregation rules and query result classification — but these solutions presuppose a centrally-managed database with a schema-aware classification engine.

At the tactical edge, the aggregation problem is compounded by the CRDT merge operation. When the CRDT layer merges two objects from different nodes, it applies a conflict resolution function that produces a new merged object. This merged object inherits the classification tags of its inputs according to the highest-water-mark rule (take the most restrictive classification of all inputs). But highest-water-mark does not capture aggregation: two FOUO objects merged by the CRDT layer produce a FOUO merged object, even if the combination of their contents creates an aggregation that should be classified SECRET.

No automated classification architecture can fully solve the aggregation problem: detecting that a combination of individually FOUO elements reveals SECRET-level information requires domain knowledge and judgment that the substrate cannot encode.¹⁰ What the substrate can do is create the conditions under which the aggregation problem can be detected and surfaced to human governance. This is the role the §6 governance application will specify.

7 Ravi S. Sandhu, "Lattice-Based Access Control Models," *IEEE Computer* 26, no. 11 (1993): 9–19, <https://doi.org/10.1109/2.241422> (access control policy is a lattice-ordered set of permissions; enforcement against a stale policy enforces an outdated lattice, which may have incorrect ordering relative to current policy).

8 Kuiper, "HGC³AE² Framework," §2 (context integrity failure: the system acts on a contextual world that no longer corresponds to the actual one — classification policy staleness is context integrity failure applied to the enforcement layer).

9 Elisa Bertino and Ravi Sandhu, "Database Security — Concepts, Approaches, and Challenges," *IEEE Transactions on Dependable and Secure Computing* 2, no. 1 (2005): 2–19, <https://doi.org/10.1109/TDSC.2005.9> (aggregation problem defined: individually-innocuous data elements may in combination exceed their individual classification levels).

10 Ravi S. Sandhu and Pierangela Samarati, "Access Control: Principles and Practice," *IEEE Communications Magazine* 32, no. 9 (1994): 40–48, <https://doi.org/10.1109/35.312842> (the aggregation problem requires semantic knowledge beyond what access control mechanisms can mechanically supply; human judgment is required at aggregation boundaries).



The four failure modes compound in practice. A single connectivity gap can trigger all four simultaneously: reclassification drift creates stale labels on existing objects; new derivations during the gap collapse the lineage; the CDS enforces against a stale policy baseline; and CRDT merges during the gap create aggregations the substrate never reviewed. The table below maps each failure mode to its trigger condition, operational consequence, and the substrate mechanism that addresses it.

Failure Mode

Trigger Condition

Operational Consequence

Substrate Response

Classification Drift

Reclassification event not yet received by node

Node enforces stale classification labels — routes or withholds at wrong level

WAL classification state writes + CRDT reconciliation at reconnect

Derivative Sensitivity Collapse

Source object reclassified after derived object was created

Derived object carries pre-reclassification classification



WAL provenance entries + automated re-evaluation trigger on source reclassification
Stale Baseline Enforcement
Policy update not yet received by CDS
CDS enforces outdated classification policy — may permit or block incorrectly
Classification policy as CRDT-managed distributed state; policy replicated via DTN
Aggregation-Driven Sensitivity Uplift
CRDT merge combines low-classification objects into sensitive aggregate
Merged object carries incorrect (too-low) classification
CRDT aggregation rule set flags merged object; CDS holds at boundary for human review

§3 — WHERE ENTERPRISE PATTERNS FAIL

Enterprise classification architectures are designed for connected, hierarchical environments. Four enterprise patterns fail structurally at the tactical edge, and understanding exactly why they fail is necessary for designing the substrate primitives in §4.

3.1 — Static Labeling at Origin

The dominant enterprise classification pattern assigns a classification label to data at creation. The label is metadata attached to the object. Access control policy at query or retrieval time evaluates the label. This is the model that underpins virtually all document management systems, email classification, and file-level access control in connected enterprise environments.

Static labeling fails at the tactical edge because the label, once assigned, is not a living property of the object — it is an annotation that may become stale while the object continues to circulate. When the object is transmitted via DTN custody transfer, the bundle carries the label. When the object is stored in the WAL, the entry carries the label at the time of write. When the object is replicated via CRDT merge, the resulting object carries the label from the merge resolution function. None of these operations has visibility into whether the label has been changed by the authoritative classifier since the object was last synchronized. The label that circulates with the object is a point-in-time snapshot of a property that may have changed.¹¹

This is not a failure of the labeling mechanism — the label itself is a well-specified data structure. It is a failure of the labeling model's assumption that the label-to-object relationship is stable over the object's lifecycle. At the tactical edge, the lifecycle of an object spans connectivity windows, custody transfers, and CRDT merges that may span days. The label-to-object relationship is not stable over that lifecycle.

3.2 — Network-Level Classification Enforcement

A second enterprise pattern trusts the classification of the network segment rather than the object. In a properly-configured MLS network, nodes and links are accredited to a specific

¹¹ Joseph A. Goguen and José Meseguer, "Security Policies and Security Models," *Proceedings of the IEEE Symposium on Security and Privacy 1982*, 11–20, <https://doi.org/10.1109/SP.1982.10014> (noninterference and the requirement for stable security policies — static labeling assumes policy stability that DDIL operation cannot guarantee).



classification level, and data is presumed safe to transit because the network itself is appropriately accredited. Cross-domain solutions are positioned at boundaries between accreditation levels, but within a single-level network, classification enforcement is handled by admission to the network rather than by object-level tagging.

Network-level enforcement fails at the tactical edge because the tactical substrate is not a single-level network. The ship's tactical node connects to a variety of sources and sinks via links of varying accreditation levels: SIPR for SECRET, JWICS for TS//SCI, unclassified link for logistics. DTN custody chains may traverse heterogeneous link types between hops. A bundle that begins at a TS//SCI source and traverses an intermediate hop at a lower accreditation level is not automatically protected by network-level enforcement — the bundle's classification tag must be enforced at each hop, not by the network layer but by the object-level AE² mechanism.¹²

The deeper failure is that network-level enforcement assumes a known, stable network topology with fixed accreditation boundaries. The tactical substrate's topology changes as ships move, links degrade, and new custody hops become available. Enforcement that depends on topology stability cannot survive topology change.

3.3 — Synchronous Access Control at Query Time

Most enterprise access control architectures enforce classification at the point of access — when a user queries for data, the access control system evaluates the user's clearance against the data's label and permits or denies the query. This model assumes that enforcement happens at a well-defined, synchronous point and that the authoritative classification state is available at that point.

At the tactical edge, the AI Supervisor does not make queries in the enterprise sense. It makes routing decisions and action decisions at operational tempo, and it does so by reasoning over the local state it holds, which is the most recent state the WAL contains. There is no synchronous query-evaluate-deny cycle; there is a continuous stream of inferences over locally-held state. Access control enforcement that requires a synchronous query to a central authority fails entirely when the central authority is unreachable for seventy-two hours.¹³

This failure mode is directly named in the HGC³AE² framework's treatment of silent degradation: a system that enforces correctly when connected but silently ceases to enforce when disconnected is not compliant — it is compliant by coincidence during its connected windows and noncompliant during its degraded windows, without surfacing the degradation to

12 Scott Burleigh, Adrian Hooke, Leigh Torgerson, Kevin Fall, Vint Cerf, Bob Durst, Keith Scott, and Howard Weiss, "Delay-Tolerant Networking: An Approach to Interplanetary Internet," *IEEE Communications Magazine* 41, no. 6 (2003): 128–136, <https://doi.org/10.1109/MCOM.2003.1204759> (DTN custody transfer traverses heterogeneous link types; network-level accreditation cannot cover a custody chain spanning multiple link types with different accreditation levels).

13 Denning, "A Lattice Model," 240 (the lattice model's enforcement at access time presupposes an authority that can evaluate the requester's clearance against the data's label; the authority must be available at access time for enforcement to occur).



the human operator.¹⁴ Enterprise synchronous access control enforces compliantly when connected. At the degraded edge, it enforces against stale state, or it does not enforce at all.

3.4 — Manual Reclassification and Human-in-the-Loop Review

Enterprise classification systems depend on human review at classification decision points: the original classifier, the derivative classifier, the declassification authority, and the FOIA coordinator are all human roles exercising human judgment. This is not an accident. The sensitivity of classification decisions — which affect national security, operational security, and individual privacy — requires human accountability.¹⁵

At the tactical edge, the requirement for human judgment is unchanged. What changes is the operational tempo. During a seventy-two-hour connectivity gap, the ship may produce thousands of data objects through AI inference, sensor fusion, and automated reporting. Human review of each object at creation and at each subsequent state change is not operationally feasible. An architecture that requires human review for each classification decision will either paralyze operations or be bypassed in favor of automated decisions without governance. Neither outcome is acceptable.

The resolution is not to eliminate human judgment but to structure its exercise appropriately. Human governance sets the classification policy, the derivation rules, the aggregation detection thresholds, and the escalation conditions. The substrate executes that policy at operational tempo. Human review occurs at governance checkpoints — mission transitions, reconnection events, and aggregation alerts — not at every classification operation. This is the H half's role in the classification architecture, which §6.1 will specify.

§4 — THE SUBSTRATE PRIMITIVE: CLASSIFICATION AS DISTRIBUTED STATE

The substrate primitives that address the classification problem are not classification-specific inventions. They are applications of the existing substrate — DTN, WAL, CRDTs, and the AI Supervisor — to the specific requirements of classification integrity under DDIL constraint. The design work is in specifying exactly how each primitive must be applied, what contracts it must honor, and what it cannot do without human governance.

4.1 — Classification Lineage in the Write-Ahead Log

The WAL is the ground truth of what happened on the node: what was written, in what order, by what source, at what time. For classification integrity, the WAL must be extended to treat classification state as a first-class write entry rather than as metadata attached to other entries.

This distinction matters precisely. In a conventional logging model, a data write carries classification metadata as an annotation: the WAL entry says “object X was written at time T with classification C.” The classification metadata is a property of the entry, not a write in its own

¹⁴ Kuiper, “HGC³AE² Framework,” §2 (silent degradation: the system fails by continuing rather than by stopping, maintaining the appearance of correct operation while enforcing against stale state).

¹⁵ Kuiper, “HGC³AE² Framework,” §3 (human authority in HGC³AE² is not ornamental — the human defines truth boundaries and escalation conditions under which system outputs can be considered legitimate).



right. In the model proposed here, classification state has its own WAL entry type: a classification state write records “the classification of object X was set to C at time T by authority A.” This separates the lifecycle of the data object from the lifecycle of its classification state. The data object can be updated (producing new WAL entries) without affecting its classification lineage. The classification state can be updated (by a reclassification event) without requiring the data object itself to be rewritten.

The consequence for reclassification is significant. When a reclassification message arrives from the shore-authoritative node after a link window reopens, it arrives as a classification state write event. The WAL records it in order. The CRDT layer can merge the reclassification event with any local classification state writes that occurred during the disconnected window — applying the standard CRDT reconciliation logic. The result is a converged classification state that is consistent with both the shore-authoritative reclassification and any local classification decisions made during the disconnected window, with conflicts surfaced for human arbitration per the P2 §2 governance model.¹⁶

Classification lineage in the WAL also enables derivative classification tracking. When the AI Supervisor creates a derived object from source materials, it generates a WAL entry that records not only the derived object’s content but its provenance: which source objects it was derived from, their WAL entry references, and their classification states at derivation time. This provenance chain is the basis for automated derivative classification re-evaluation. When a source object’s classification state changes, the substrate can query the WAL for all derived objects that reference that source and flag them for classification re-evaluation. The flagging process produces a human governance alert, not an automated reclassification — because the derivative sensitivity calculation requires the domain judgment that only the H half can supply.¹⁷

The WAL’s append-only property is preserved throughout this design. Classification state writes are new entries, not modifications to existing entries. The historical record of what was believed at any point in time is recoverable by replaying the WAL to any timestamp. This is the evidentiary property that accreditation requires: the substrate can demonstrate, with cryptographic integrity, what classification state any object held at any given time and on which node. The WAL is not just a recovery mechanism; it is the classification audit trail.¹⁸

16 Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski, “Conflict-Free Replicated Data Types,” *Proceedings of SSS 2011*, Springer LNCS 6976, 386–400, https://doi.org/10.1007/978-3-642-24550-3_29 (CRDT convergence guarantees that all nodes eventually hold the same state given sufficient communication — classification state as CRDT enables convergence to authoritative classification state as link windows occur).

17 Sabelfeld and Myers, “Language-Based Information-Flow Security,” 8 (type-based flow tracking — the WAL provenance record is the substrate’s equivalent of the static type annotation: it records flow relationships that can be evaluated when classification changes propagate).

18 Mohan et al., “ARIES,” 95 (WAL as evidentiary ground truth: the log is the authoritative record of what happened, not a backup of the current state — this property makes the WAL the appropriate substrate for classification audit trail).



The five WAL entry types that classification state management requires are distinct from data write entries and must be explicitly specified to ensure consistent implementation across nodes in the substrate.

Entry Type

Key Fields

Produced By

Purpose

data-write

Object ID, content hash, classification tag (snapshot), write time, writer credential

Any authorized writer

Object content lifecycle

classification-state-write

Object ID, classification level, control markings, authority ID, event time, authority credential

Classification authority (human or delegated)

Classification lifecycle — separate from data lifecycle

derived-object-provenance

Derived object ID, source WAL entry refs[], source classification states at derivation time[],
AI Supervisor credential

AI Supervisor

Derivative classification tracking — enables re-evaluation triggers

classification-review-required

Object ID, review trigger type (aggregation / drift / derivation), flag time, queue priority

CRDT merge function or AI Supervisor

Human governance review queue — holds object at CDS boundary

classification-review-resolved

Object ID, reviewer credential, resolution type (uplift / downgrade / no-action), new
classification if changed, resolution time

Human reviewer



Classification audit trail — closes the review queue entry

Figure 1 — Classification Lineage in the Write-Ahead Log *WAL entry structure showing three distinct entry types — data write, classification state write, and derived-object provenance entry — as separate log streams on the same append-only WAL. The data write and classification state write have independent lifecycles: the data object can be updated without changing its classification state, and the classification state can be updated (by reclassification) without rewriting the data object. The derived-object provenance entry chains to both source WAL entries, enabling automated re-evaluation when source classification states change. [FIGURE SEED: assets/series-3-edo/figures-seed/p8-fig-01-classification-wal-lineage.png — full SVG diagram at v1.0-preprint]*

4.2 — CRDT Classification Merge Semantics

The CRDT layer's job is to converge divergent state across nodes that have been operating independently. For data objects, convergence means applying a deterministic merge function that produces the same result regardless of the order in which updates are applied. For classification state, convergence requires additional specification: the merge function must honor the classification system's fundamental ordering properties.

The standard CRDT merge rule for classification state is highest-water-mark: when two nodes hold different classification states for the same object, the merged result takes the more restrictive classification. SECRET//NOFORN merged with CONFIDENTIAL//REL TO USA produces SECRET//NOFORN. This rule is correct under most conditions and operationally conservative — it errs toward over-restriction rather than under-restriction, which is the right failure mode for a classification system.¹⁹

Highest-water-mark has two failure modes that require substrate-level treatment. The first is the phantom upgrade problem: if a node holds a classification state of SECRET//NOFORN because of a local classification error (the classifier made a mistake that was subsequently corrected), and the correction message downgraded the object to CONFIDENTIAL, highest-water-mark would preserve the erroneous SECRET//NOFORN classification even after the correction has propagated. The resolution is to give classification state writes temporal authority: a reclassification event from a recognized authority always supersedes a conflicting classification state, regardless of vector clock ordering, because classification authority is not symmetric — the authoritative classifier's decision overrides a local estimate.

The second failure mode is the tombstone problem for declassified or downgraded objects. When an object is declassified, the reclassification event must eventually reach all nodes holding the object. If a node never receives the declassification event — because its links were interrupted and the event was not retained in the DTN custody chain — it continues to hold and enforce the original classification indefinitely. The WAL's retention policy must ensure that classification state events are retained long enough to propagate to all potentially-affected nodes, which requires knowledge of the network topology and custody chain characteristics. This is a governance decision, not an engineering default.²⁰

¹⁹ David F.C. Brewer and Michael J. Nash, "The Chinese Wall Security Policy," *Proceedings of the IEEE Symposium on Security and Privacy 1989*, 206–214, <https://doi.org/10.1109/SECPRI.1989.36295> (conflict of interest classes and the principle that in conflict, the more restrictive policy governs — highest-water-mark inherits this principle for classification merge).

²⁰ Fall, "Delay-Tolerant Network Architecture," 30 (DTN custody transfer and the challenge of ensuring delivery to nodes that may be intermittently connected — the tombstone problem for



A concrete merge scenario illustrates how temporal authority and highest-water-mark interact. Suppose Node A holds SECRET//NOFORN for Object X, and Node B holds TS//HCS-P — the result of a reclassification event that reached Node B before the link dropped. Under highest-water-mark alone, reconciliation produces TS//HCS-P. Now suppose Node A subsequently received a downgrade from the same shore authority: CONFIDENTIAL//REL TO USA. At reconciliation, the substrate sees three signed events from a single recognized authority and must decide which governs:

Event 1 (Node B): TS//HCS-P reclassification, signed by shore authority, timestamp T_1

Event 2 (Node A): CONFIDENTIAL//REL TO USA downgrade, signed by same authority, timestamp $T_2 > T_1$

Temporal authority override — applied only to events bearing the same authority's credential — produces CONFIDENTIAL//REL TO USA: the most recent event from the recognized authority wins. Highest-water-mark without temporal authority would produce TS//HCS-P, which is operationally conservative but incorrect if the downgrade was intentional. The authority-identity check is the gate: only events carrying a verified authority credential exercise temporal override. An anonymous downgrade event, regardless of timestamp, does not supersede a signed upgrade event.

The WAL records both events, the resolution logic applied, and the final classification state — providing the audit trail that the accreditor reads when verifying that the merge decision was correctly governed.

Figure 2 — CRDT Classification Merge: Highest-Water-Mark with Temporal Authority Override *Merge resolution diagram showing two concurrent classification state event streams — Node A and Node B — for the same object, with a reclassification conflict. Left branch: highest-water-mark-only resolution (produces TS//HCS-P, ignores downgrade). Right branch: temporal authority override applied to same-authority events (produces CONFIDENTIAL//REL TO USA, most recent authoritative decision wins). Authority identity verification shown as the gate condition between the two resolution paths. [FIGURE SEED: assets/series-3-edo/figures-seed/p8-fig-02-classification-crdt-merge.png — full SVG diagram at v1.0-preprint]*

For aggregation detection, the CRDT layer's merge function must be extended with an aggregation sensitivity check. After merging two objects, the merge function applies a configurable aggregation rule set: a set of pattern-matching rules that identify combinations of data elements that should trigger a classification review. The rule set is human-defined and encodes the domain knowledge that the substrate cannot supply. The merge function does not automatically reclassify the merged object — it flags it for review. The flag is a WAL entry of type classification-review-required, which the AI Supervisor surfaces to the human governance interface at the next review checkpoint.²¹

4.3 — DTN Bundle Classification as Cryptographic Property

In the P1 Tactical Substrate, DTN bundles are the custody-carrying unit of transmission. Bundles are signed at the custody source and verified at each hop. For classification integrity,

classification events is the classification-domain instance of this general DTN challenge).

²¹ Bertino and Sandhu, "Database Security," 14 (aggregation detection as a classification challenge requiring rule-based pattern matching against schema-aware classification guidance — the CRDT aggregation rule set is the substrate-level expression of this requirement).



the bundle's classification tag must be a cryptographic property of the bundle, not a metadata annotation that can be modified in transit.

The implementation is straightforward within the existing DTN bundle protocol: the classification tag is included in the signed payload of the bundle, not in the unsigned extension block. A custody hop that cannot verify the bundle's cryptographic signature cannot verify the classification tag; the bundle is rejected as per the AE² fail-closed principle. A bundle whose classification tag has been modified in transit produces a signature verification failure and is rejected at the next custody hop.²²

This design creates a hard property: a bundle's classification tag, as known to the receiving node, is cryptographically bound to the tag as set by the originating custody source. Classification tag spoofing — which is a realistic threat in a contested electromagnetic environment where adversaries may attempt to inject bundles with false classification tags to manipulate routing — is structurally impossible if tag verification is enforced. A bundle claiming to be UNCLASSIFIED that originated as SECRET will fail signature verification at any hop configured to enforce bundle authentication.²³

The classification tag in the bundle is a snapshot — it reflects the classification state at the time the bundle was signed. For objects that may be reclassified in transit, the bundle carries the originating classification; any subsequent reclassification events must arrive as separate classification state write events and be reconciled via the WAL/CRDT process. This is the correct design choice: the bundle's classification tag is not the current authoritative classification of the object; it is the classification known to the sending node at the time of transmission. The receiving node must apply its locally-held classification state, which may differ from the bundle tag if reclassification events have been received, to determine the object's current classification.

Figure 3 — DTN Bundle Classification as Cryptographic Property *Bundle structure diagram showing two alternative designs: (left) classification tag as unsigned extension block — modifiable in transit without invalidating bundle signature; (right) classification tag embedded in the signed bundle payload — any modification to the tag breaks the signature verification at the next custody hop. Annotations show the attack path that the unsigned-extension design enables (adversary modifies extension block to change NOFORN to REL TO) and why it is structurally closed by the signed-payload design. The custody hop verification flow is shown for both designs. [FIGURE SEED: assets/series-3-edo/figures-seed/p8-fig-03-bundle-classification-signing.png — full SVG diagram at v1.0-preprint]*

4.4 — AI Supervisor Classification Awareness

The AI Supervisor's routing decisions must be classification-aware: the Supervisor must know the current classification state of each object it is considering routing, and it must route in accordance with the classification policy encoded by the H half. This is not simply a matter of reading the bundle's classification tag — for the reasons developed in §4.3, the bundle tag is a

²² Burleigh et al., “Delay-Tolerant Networking,” 132 (DTN bundle security and the requirement for bundle authentication at custody boundaries — the classification tag as signed payload rather than unsigned extension is a direct application of this security requirement).

²³ Leslie Lamport, Robert Shostak, and Marshall Pease, “The Byzantine Generals Problem,” *ACM Transactions on Programming Languages and Systems* 4, no. 3 (1982): 382–401, <https://doi.org/10.1145/357172.357176> (adversarial node behavior and the requirement for cryptographic authentication to distinguish authentic messages from adversarial injections — classification tag spoofing is a Byzantine fault on the classification channel).



historical snapshot. The Supervisor must maintain a classification state view that integrates bundle tags with subsequent reclassification events received via the WAL.

The classification-aware Supervisor maintains a classification state index: a compact representation of the current classification state of every object in the local WAL, updated in real time as classification state writes arrive. The index is not a separate database — it is a view derived from the WAL, and it can be rebuilt from the WAL at any time (important for recovery after Supervisor restart). The Supervisor's routing decisions reference the classification state index, not raw object metadata.²⁴

The Supervisor's curriculum includes classification-specific training data: historical classification state events, reclassification patterns, aggregation detections, and their operational outcomes. The Supervisor learns which classification state patterns are associated with which operational risks — for instance, which types of aggregation events historically produced classification reviews, and whether those reviews resulted in uplift, downgrades, or no-action decisions. This curriculum enables the Supervisor to weight its aggregation flagging more accurately, reducing false-positive alerts that consume human review capacity without improving classification integrity.

The Supervisor does not make classification decisions. Its role in the classification architecture is operational: route correctly according to current classification state, flag aggregations for human review, surface reclassification drift alerts, and enforce the H-half's routing policy within its autonomy envelope. The boundary between what the Supervisor decides autonomously and what it must escalate is a governance artifact specified in §6.1.

4.5 — Cross-Domain Solution Integration

The CDS is the enforcement mechanism at classification boundaries. In the substrate, the CDS is positioned between the tactical node and any network that operates at a different classification level. The CDS's job is to ensure that data does not transit a boundary without proper authorization, and that data transiting a boundary is processed in accordance with the boundary's specific handling rules.²⁵

The classification architecture in this paper extends the CDS's operating contract. The CDS does not simply evaluate object classification tags against the boundary's policy — it verifies that the tag is cryptographically bound to the object, that the tag is consistent with the classification state index (not just the historical bundle tag), and that the object's classification lineage is traceable in the WAL to a recognized classification authority. An object with a correct tag but no WAL-traceable lineage is rejected by the CDS as unverifiable — fail-closed, not fail-open.²⁶

24 Shapiro et al., "Conflict-Free Replicated Data Types," 390 (CRDT state view as a derived representation of the convergent state — the classification state index is a CRDT-derived view, not a separate database, with full reconstructibility from the WAL).

25 Kuiper, "HGC³AE² at the Degraded Edge," §4 (AE² half: authentication and enforcement applied to the substrate — CDS placement as an architectural, not advisory, enforcement mechanism).

26 Sandhu, "Lattice-Based Access Control Models," 15 (unverifiable security labels are a failure case in the lattice model that must be handled conservatively — fail-closed treatment of unverifiable classification tags applies the lattice model's conservative handling principle to the DTN classification context).



The CDS also enforces the aggregation review queue. Objects flagged as requiring classification review are withheld from boundary crossing until the review is resolved. This creates a classification review checkpoint at every CDS boundary — exactly the point in the operational flow where a human governance decision about aggregation sensitivity can be made with the lowest operational disruption. Objects in the review queue are held, not lost: they remain in the WAL and can transit the boundary as soon as the review is complete. The queue length and aging policy are human-governance parameters that trade operational tempo against classification review thoroughness.

The classification review queue has five operational states that the CDS and AI Supervisor manage jointly. The state machine is straightforward; the governance implication of each state is not.

Queue State

Entered When

Exited By

Operational Effect

pending-review

Object flagged by CRDT aggregation rule set or AI Supervisor drift alert

Human reviewer assigns or auto-assigns to review queue

Object held at CDS boundary; not visible to downstream systems

under-review

Human reviewer accepts assignment and opens the review

Human reviewer completes review (any resolution)

Object held; reviewer identity and start time logged in WAL

review-resolved-pass

Reviewer determines aggregation does not exceed classification of parts

Object released for CDS boundary crossing

WAL classification-review-resolved event with resolution=no-action; object cleared



review-resolved-uplift

Reviewer determines aggregate classification exceeds parts

Object receives new classification state write; CDS re-evaluates against updated classification

WAL classification-state-write event followed by new CDS evaluation pass

review-expired

Object in pending-review exceeds aging threshold without assignment

Governance alert escalated to senior reviewer or mission authority

Object held; escalation event logged in WAL; human governance response required

The review-expired state deserves specific attention in the DDIL context. When classification reviewer capacity is constrained — because the human expert with classification authority is not reachable during a connectivity blackout — the review queue can accumulate objects that genuinely require human judgment but cannot receive it. The aging threshold and escalation path are human-governance parameters that must account for this scenario: they define what happens operationally when the review queue exceeds the available human review capacity, and they must be set conservatively enough to prevent classification violations while permissively enough to not paralyze the operational picture during sustained blackout windows.

§5 — CONVERGENCE: HOW THE ARCHITECTURE RESOLVES THE CLASSIFICATION PROBLEM

The substrate architecture in §4 converges on a classification integrity model that addresses each of the structural failure modes in §2 without requiring connectivity that DDIL environments cannot guarantee.

Classification drift is addressed by WAL-based classification lineage and CRDT reconciliation. When a reclassification event arrives via the next link window, it is applied to the WAL as a classification state write. The CRDT layer reconciles it with any local classification decisions made during the disconnected window. Conflicts are surfaced for human arbitration. The node's classification view converges toward the shore-authoritative view as link windows occur. The convergence is not instantaneous — it depends on the frequency and reliability of link windows — but it is deterministic: given sufficient link windows, all nodes will converge to the same classification state, and the WAL provides the audit trail to verify convergence.²⁷

Derivative sensitivity collapse is addressed by WAL-based provenance tracking and automated re-evaluation triggers. When a source object's classification changes, the WAL query identifies all derived objects that reference the source. The Supervisor generates classification review alerts for each affected derived object. Human governance resolves each alert, producing a new classification state write for the derived object. The provenance chain in the WAL ensures that derived objects remain connected to their classification lineage indefinitely, not just at creation time.

²⁷ Shapiro et al., “Conflict-Free Replicated Data Types,” 395 (convergence theorem for CRDTs — eventual convergence to the same state is guaranteed given eventual delivery; the classification state CRDT applies this theorem to classification state management).



Enforcement against a stale baseline is addressed by treating the classification policy baseline as CRDT-managed distributed state, replicated through the same WAL/DTN mechanism as all other substrate state. Policy updates arrive from the shore-authoritative node via DTN bundle and are applied to the local policy CRDT. The CDS always enforces against the most recent converged policy state, not a fixed configuration artifact. Policy update events are audited in the WAL.²⁸

Aggregation-driven sensitivity uplift is addressed not by automating the aggregation judgment — which requires domain knowledge the substrate cannot encode — but by creating the substrate conditions under which the aggregation problem can be detected, surfaced, and resolved through human governance. The CRDT aggregation rule set flags candidate aggregations. The CDS holds flagged objects at boundaries. Human governance resolves flags at review checkpoints. The substrate closes the operational loop without attempting to substitute automated judgment for the domain expertise that classification aggregation decisions require.

The cumulative effect is an architecture in which classification integrity is maintained as a property of the substrate's distributed state model, not as a property of individual objects. Classification state is a first-class distributed state; it participates in the same replication, conflict resolution, and audit mechanisms as every other substrate state. An adversary or failure condition that compromises classification integrity — by injecting false classification tags, by corrupting reclassification events, or by preventing policy baseline updates from propagating — is visible in the WAL and detectable by the classification state index. The architecture makes classification integrity failures auditable, not just preventable.

The architecture does not make classification decisions. It creates the substrate conditions under which the H half's classification policy can be executed faithfully, the C³ half's classification history can be cataloged completely, and the AE² half's classification enforcement can be applied correctly against current state. This is what "HGC³AE² applied to the classification problem" means in practice: not a classification AI, but a classification-aware substrate that executes human classification governance without substituting automated judgment for human authority.²⁹

The following threat-model scenarios illustrate the architecture's practical effect. Each maps a realistic adversarial or operational

²⁸ Kuiper, "HGC³AE² Framework," §5 (Skipjack Protocol's live validation principle: dynamic operational facts must be queried when they matter and cannot be treated as equivalent to static reference data — classification policy baseline is a dynamic operational fact).

²⁹ Kuiper, "HGC³AE² Framework," §3 (the HGC³AE² framework proposes not a collection of best practices but a coherent operating architecture — the classification architecture in this paper is a domain-specific instance of that operating architecture, not an independent classification system).



failure condition to the architectural mechanism that addresses it.

Threat / Failure Scenario

Without Architecture

With Architecture

Adversary injects DTN bundle with forged UNCLASSIFIED tag to bypass CDS

Bundle accepted — tag is metadata, not cryptographically bound

Bundle rejected at first custody hop — signed-payload tag verification fails; AE² fail-closed

Node operating on 72-hour stale reclassification routes TS data as SECRET

Data routed at wrong classification; CDS enforces wrong policy silently

WAL classification state index shows pending reclassification; AI Supervisor flags for human review; CDS holds bundle

AI Supervisor creates fused track from source subsequently reclassified TS

Derived object carries SECRET classification permanently; no re-evaluation

WAL provenance trigger fires on source reclassification; derived object enters classification-review-required queue

CRDT merge of two FOUO logistics records produces SECRET-level aggregate

Merged object carries FOUO; crosses CDS boundary without review

CRDT aggregation rule set flags merged object; CDS holds at boundary; human reviewer resolves

Policy baseline not updated in 3 weeks; new compartment caveat unenforced

CDS enforces old policy; new caveat data crosses boundary it should not

Policy CRDT receives shore update via next DTN window; CDS enforces updated policy on next routing cycle

§6 — GOVERNOR APPLICATION: HGC³AE² AT THE CLASSIFICATION LAYER

This section applies the HGC³AE² framework to the classification problem as defined in this paper. It is not a summary of P2. It is an application of P2's framework at the scale of classification integrity. Readers arriving without P2 should complete P2 — “HGC³AE² at the Degraded Edge” — before proceeding.

§6.1 — H: Human Governance

Three classification decisions are unambiguously reserved for human governance in the tactical substrate, and a wrong delegation of any of them produces a failure mode the architecture cannot recover from.



The first is classification routing policy. Which objects, carrying which classification tags, may transit which boundaries under which conditions — this is a human governance decision. The CDS enforcement rule set is a formal expression of this policy. The AI Supervisor's routing table is derived from this policy. The classification state index prioritizes review flags in accordance with this policy. None of these mechanisms produces the policy; all of them execute it. A classification routing policy that was written by an AI inference engine, even one with sophisticated classification curriculum, has no authoritative accountability structure — it is, structurally, AI-defined governance, which is precisely the category error the HGC³AE² framework identifies as the dominant failure mode of ungoverned agentic systems.³⁰

The second is derivative classification decision authority. When the WAL-based provenance tracking system generates a derived-object review alert, the resolution of that alert requires a human derivative classifier to examine the derived object, its source materials, and the applicable classification guidance, and to make a derivative classification decision. This decision cannot be delegated to the AI Supervisor without converting a human classification judgment into an automated inference result. The derivative classifier's decision is recorded in the WAL as a human governance event, with the classifier's identity and the classification guidance applied.

The third is aggregation sensitivity decision authority. When the CRDT aggregation rule set flags a merged object for classification review, the resolution of the flag requires a human reviewer to apply domain judgment to the combination of elements and determine whether the aggregate exceeds the classification of its parts. This is a definitional exercise of the H half: only the human side of the context boundary can supply the domain knowledge necessary to evaluate aggregation sensitivity.³¹

§6.2 — C³: Curation, Cataloging, Curriculum

Curation, in the classification context, means maintaining the classification state index as a curated view of current classification reality rather than as a raw accumulation of historical events. The classification state index is not a direct read of the WAL — it is a derived view that applies the CRDT reconciliation logic and presents the current converged state of each object's classification. Curation decisions include which objects are included in the active classification state index (recently-accessed objects vs. archived objects), how long classification state events are retained in the WAL for propagation to nodes that have not yet received them, and how aggregation review flags are prioritized in the review queue.

Cataloging, in the classification context, means recording the full classification lifecycle of every object in the WAL with enough fidelity that the classification history is reconstructable. Every classification state write — original classification, reclassification, declassification, derivative classification decision, aggregation review resolution — is a WAL entry. The WAL is the classification audit trail. Without complete cataloging, the accreditation problem (P9) cannot

30 Kuiper, "HGC³AE² Framework," §2 (unconstrained operation: in the absence of governance, systems begin to treat possibility as permission — AI-defined classification routing policy is the classification domain's instance of this failure mode).

31 Kuiper, "HGC³AE² at the Degraded Edge," §2 (conflict arbitration policy as a human governance decision: when the system cannot automatically resolve a conflict because the operational semantics require judgment the substrate cannot encode, a human must arbitrate — aggregation sensitivity decisions are the classification domain's instance of this governance requirement).



be addressed, because accreditation audits require demonstration that classification was maintained correctly throughout the object's lifecycle.

Curriculum, in the classification context, means training the AI Supervisor on the history of classification events, review outcomes, and operational patterns that characterize the tactical node's classification environment. The Supervisor learns: which data sources and derivation patterns historically produce classification drift; which aggregation combinations historically trigger human review and result in uplift; which CDS boundary conditions are associated with classification enforcement failures. This curriculum enables the Supervisor to prioritize classification review flags by estimated risk and to route objects in ways that minimize classification integrity exposure. Curriculum does not give the Supervisor classification decision authority; it improves the Supervisor's ability to manage the operational flow within the policy envelope the H half defines.³²

§6.3 — AE²: Authentication and Enforcement

Authentication in the classification architecture covers two boundaries: the DTN bundle authentication described in §4.3 (classification tags are cryptographically bound to bundle payloads and verified at every custody hop), and the classification state write authentication (every classification state event in the WAL is signed by the authority that produced it, enabling downstream nodes to verify that a reclassification event came from a recognized authority rather than from a compromised intermediate node).

Enforcement is fail-closed at every classification boundary without exception. An object with an unverifiable classification tag does not transit a CDS boundary — it is rejected and placed in the exceptions queue for human review. An object in the aggregation review queue does not transit a CDS boundary until the review is resolved. An object whose WAL provenance trace cannot reach a recognized classification authority is treated as unclassifiable-at-current-state and withheld pending provenance resolution. There is no fail-open fallback, no operational urgency exception, no bypass pathway. The integrity of the classification architecture depends on the unconditional nature of its enforcement; a fail-closed rule that applies “most of the time” is operationally equivalent to no rule.

The kill-switch provision for the AI Supervisor's classification-aware routing autonomy is: any human operator with appropriate credentials can halt all AI-autonomous classification routing decisions and revert to human-reviewed routing. When the kill switch fires, the Supervisor continues to maintain the classification state index and to generate review flags, but it does not execute routing decisions. A human operator reviews and approves each routing decision. This provision is the outer boundary of AI autonomy in the classification domain, and it is enforced by the AE² mechanism, not by the Supervisor's own judgment about when to exercise caution.³³

32 Kuiper, “HGC³AE² at the Degraded Edge,” §3 (C³ half: curriculum builds the AI Supervisor's operational model from cataloged history — the classification curriculum is the C³ function applied to the classification domain's operational patterns).

33 Kuiper, “HGC³AE² at the Degraded Edge,” §4 (kill-switch provision: a human-accessible control that suspends AI-autonomous behavior and returns the system to manual operation — the kill switch for classification routing is the AE² mechanism that enforces the outer boundary of AI autonomy in the classification domain).



§6.4 — Reference to P2

See P2 — “HGC³AE² at the Degraded Edge” §2 (H Half: Human Governance Applied to the Substrate) for the substrate-wide treatment of classification routing policy governance and the drift threshold governance decision class. Classification routing policy is the direct substrate expression of the H half’s classification governance authority. §4 of this paper (AE² Half: Authentication and Enforcement) provides the substrate-wide authentication and enforcement model that §6.3 applies specifically to the classification layer. The mandatory-section contract in P2 §6 defines the template this section implements.

§7 — READING GUIDE: WHAT THIS PAPER LEAVES OPEN

This paper has addressed the classification problem at the level of the object: how to maintain the integrity of individual objects’ classification state across the DDIL lifecycle. It has done so by treating classification as a first-class distributed state and applying the substrate primitives — WAL, CRDTs, DTN bundle authentication, and AI Supervisor curriculum — to the specific requirements of classification integrity.

What this paper has not addressed, and what the classification architecture depends upon, is the question of node accreditation: the question of whether the nodes that hold, process, transmit, and enforce classification state are themselves authorized to do so. Classification integrity is a property of objects. Accreditation integrity is a property of nodes. An object with a correct classification state, properly enforced by a CDS with an up-to-date policy baseline, may still be routed to a node that is no longer accredited to hold it — because the accreditation regime of that node has changed since the classification routing policy was last updated.

The accreditation problem is the subject of P9. Classification (P8) establishes what the data is. Accreditation (P9) establishes whether a given node is authorized to receive it. The two problems are interdependent: a correct classification architecture is necessary but not sufficient for classification security, because it operates on top of an accreditation regime that must itself maintain integrity under DDIL conditions. Readers who hold P8 without P9 hold the object-level half of the classification security model. P9 supplies the node-level half.

The interoperability problem (P10) will extend both P8 and P9 to the case where two nodes operating under different accreditation regimes — different classification taxonomies, different accreditation authorities, different CDS implementations — must exchange state at the degraded edge. The classification architecture in this paper assumes a single, coherent classification governance regime. That assumption does not hold in coalition operations, which P10 addresses.

§8 — BIBLIOGRAPHY

Bell, David Elliott, and Leonard J. LaPadula. *Secure Computer System: Unified Exposition and Multics Interpretation*. ESD-TR-75-306. Bedford, MA: MITRE Corporation, 1976.

Bertino, Elisa, and Ravi Sandhu. “Database Security — Concepts, Approaches, and Challenges.” *IEEE Transactions on Dependable and Secure Computing* 2, no. 1 (2005): 2–19. <https://doi.org/10.1109/TDSC.2005.9>.

Brewer, David F.C., and Michael J. Nash. “The Chinese Wall Security Policy.” *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, 1989: 206–214. <https://doi.org/10.1109/SECPRI.1989.36295>.



Burleigh, Scott, Adrian Hooke, Leigh Torgerson, Kevin Fall, Vint Cerf, Bob Durst, Keith Scott, and Howard Weiss. "Delay-Tolerant Networking: An Approach to Interplanetary Internet." *IEEE Communications Magazine* 41, no. 6 (2003): 128–136. <https://doi.org/10.1109/MCOM.2003.1204759>.

Chong, Stephen, and Andrew C. Myers. "Security Policies for Downgrading." *Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS 2004)*, Washington, DC, 2004: 198–209. <https://doi.org/10.1145/1030083.1030110>.

Denning, Dorothy E. "A Lattice Model of Secure Information Flow." *Communications of the ACM* 19, no. 5 (1976): 236–243. <https://doi.org/10.1145/360051.360056>.

Fall, Kevin. "A Delay-Tolerant Network Architecture for Challenged Internets." *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (ACM SIGCOMM 2003)*, Karlsruhe, Germany, 2003: 27–34. <https://doi.org/10.1145/863955.863960>.

Goguen, Joseph A., and José Meseguer. "Security Policies and Security Models." *Proceedings of the 1982 IEEE Symposium on Security and Privacy*, Oakland, CA, 1982: 11–20. <https://doi.org/10.1109/SP.1982.10014>.

Kuiper, Justin H. "Mitigating Confident Misalignment in Agentic Systems: The HGC³AE² Framework." Non Sequitur Publishing, 2026. v1.0-preprint. <https://nonsequitur.tech/white-papers/hgc3ae2/>. [Self-cite; additional to peer-review count.]

Lamport, Leslie, Robert Shostak, and Marshall Pease. "The Byzantine Generals Problem." *ACM Transactions on Programming Languages and Systems* 4, no. 3 (1982): 382–401. <https://doi.org/10.1145/357172.357176>.

Mohan, C., Don Haderle, Bruce Lindsay, Hamid Pirahesh, and Peter Schwarz. "ARIES: A Transaction Recovery Method Supporting Fine-Granularity Locking and Partial Rollbacks Using Write-Ahead Logging." *ACM Transactions on Database Systems* 17, no. 1 (1992): 94–162. <https://doi.org/10.1145/128765.128770>.

Myers, Andrew C., and Barbara Liskov. "A Decentralized Model for Information Flow Control." *Proceedings of the Sixteenth ACM Symposium on Operating System Principles (SOSP 1997)*, Saint-Malo, France, 1997: 129–142. <https://doi.org/10.1145/268998.266669>.

Sabelfeld, Andrei, and Andrew C. Myers. "Language-Based Information-Flow Security." *IEEE Journal on Selected Areas in Communications* 21, no. 1 (2003): 5–19. <https://doi.org/10.1109/JSAC.2002.806121>.

Sandhu, Ravi S. "Lattice-Based Access Control Models." *IEEE Computer* 26, no. 11 (1993): 9–19. <https://doi.org/10.1109/2.241422>.

Sandhu, Ravi S., and Pierangela Samarati. "Access Control: Principles and Practice." *IEEE Communications Magazine* 32, no. 9 (1994): 40–48. <https://doi.org/10.1109/35.312842>.

Shapiro, Marc, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. "Conflict-Free Replicated Data Types." In *Proceedings of the 13th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS 2011)*, Grenoble, France, 2011: 386–400. Springer LNCS 6976. https://doi.org/10.1007/978-3-642-24550-3_29.

Kuiper, Justin H. "HGC³AE² at the Degraded Edge." *The Implications of Edge Degraded Ops*, P2, 2026. v0.1-seed. [Self-cite; additional to peer-review count.]

Kuiper, Justin H. "The Tactical Substrate." *The Implications of Edge Degraded Ops*, P1, 2026. v0.1-seed. [Self-cite; additional to peer-review count.]



v0.1-draft — Full prose draft. Figures 1–3 embedded as inline placeholders; full SVG commission deferred to v1.0-preprint pipeline. Bibliography cross-check deferred. Stamp deferred.
