IPPM Working Group
Internet-Draft
Intended status: Informational
Expires: 23 November 2026

L. Melegassi
Catellix

22 May 2026

Incremental Bandwidth-Efficient Multi-Vantage Path Synchrony (BE-MVPS): Cell-Partitioned Coherence with epsilon-Gated Sherman-Morrison Updates

draft-melegassi-mvps-incremental-be-00

Abstract

This document specifies BE-MVPS (Bandwidth-Efficient MVPS), an incremental execution layer for the Multi-Vantage Path Synchrony (MVPS) framework that trades a constant-factor increase in broker-side CPU for an order-of-magnitude decrease in vantage-to-broker bandwidth. Where MVPS as defined in [I-D.melegassi-ippm-mvps-bundle] performs a dense recomputation of the coherence vector C = (C_1, C_2, C_3) at every tick over the full population of N observers, BE-MVPS partitions the observer set into k coherence cells, gates per-observer state transmission by a local epsilon threshold, performs Sherman-Morrison incremental updates of the Mahalanobis distance D^2, and detects Byzantine vantages via a cell-aware minimax estimator whose breakdown point is shown (Theorem 7) to be exactly f/k where f is the adversarial vantage fraction.

The earlier informal label "Fast Incremental MVPS (FMVPS)" is superseded by BE-MVPS in this document; the rename and its rationale are recorded in the ERRATUM block below and proved formally in Theorem T_BE of the v5.0 unified proof. The IETF identifier of this document is draft-melegassi-mvps-incremental-be.

Nine theorems formalise the framework: the partition existence theorem (Theorem 2), the cell-equivalence bound (Theorem 3), the gating information-loss bound (Theorem 4), the Sherman-Morrison-Woodbury incremental D^2 update (Theorem 5), strong eventual consistency of the CRDT merge (Theorem 6), the cell-aware Byzantine breakdown point (Theorem 7), the C_4 perturbation non-incrementality theorem (Theorem 8), and the detection latency lower bound for sub-tick variants (Theorem 9). Section 13 introduces five BFD-inspired execution variants and reports wall-clock benchmark results: variant V3 (Echo) achieves tau_detect = 55 ms, a 1091x reduction relative to the baseline tick scale. Wall-clock benchmarks on N = 1000 to 10 000 vantages confirm that BE-MVPS reduces edge-to-broker bandwidth by a factor of 25x while preserving detection completeness for all canonical scenarios except adversary fractions below 1/k.

This document is a companion to [I-D.melegassi-ippm-mvps-bundle] and to the AI-coherence extension [I-D.melegassi-mvps-ai-coherence]. The algebra is preserved verbatim; only the execution model changes.

NOTE ON DATA PROVENANCE. All wall-clock and bandwidth numbers in this document are obtained from synthetic simulations (scripts/benchmark_fmvps_vs_ml.py and scripts/benchmark_coherence_bfd.py) under controlled conditions. Validation against operational data (RIPE Atlas, CAIDA, or commercial operator traces) is identified as required future work before publication outside the Experimental track.

A REFERENCE IMPLEMENTATION of the cell-aware minimax detector (Theorem 7) is provided in pure Python at <https://catellix.com/static/download/reference-impl/>. See reference-impl/README.md.

ERRATUM (v5.0 unified proof, Theorem T_BE -- applied to this -00 document at submission time, not deferred to -01). Earlier drafts and internal material used the label "Fast Incremental MVPS (FMVPS)". Wall-clock measurements (scripts/benchmark_fmvps_vs_ml.py) and the T_BE theorem of docs/MVPS_V5_UNIFIED_PROOF.txt show that, at N = 1000 to 10 000 vantages and d = 3, this algorithm is on average approximately TWO times slower in per-tick CPU than MVPS-classic (Welford on a dense covariance), while sending approximately TWENTY FIVE times fewer bytes from each vantage to the broker. The genuine advantage of the algorithm specified here is therefore BANDWIDTH efficiency under epsilon-gating, not CPU latency.

This document is therefore identified at submission time as draft-melegassi-mvps-incremental-be (Bandwidth-Efficient), with the "Fast" label dropped from both filename and title. The earlier name "draft-melegassi-mvps-fast-incremental" was never submitted to the IETF Datatracker; the BE name is the first authoritative IETF identifier. Theorem T_BE of the unified proof gives the cros
sover
condition

    lambda* = (C_be - C_mc) / (B_mc - B_be)

above which choosing BE-MVPS over MVPS-classic is strictly beneficial under a CPU-vs-bandwidth budget.

Status of This Memo

This Internet-Draft will expire on November 23, 2026.

Copyright Notice

Table of Contents

1. Introduction 2
   1.1. The dense-recomputation cost wall 3
   1.2. Scope and non-goals 3
   1.3. Conventions used in this document 3
2. Notation and Preliminaries 4
3. Information Redundancy of Dense MVPS 6
   3.1. Theorem 1 (Redundancy bound) 6
4. Cell-Partitioned Coherence 7
   4.1. Theorem 2 (Partition existence) 7
   4.2. Theorem 3 (Cell-equivalence) 8
5. Edge Delta Gating 9
   5.1. Theorem 4 (Gating information-loss bound) 9
6. Lazy Mahalanobis Update 10
   6.1. Theorem 5 (Sherman-Morrison-Woodbury D^2) 10
7. CRDT Coherence Merge 11
   7.1. Theorem 6 (Strong eventual consistency) 11
8. Cell-Aware Byzantine Detection 12
   8.1. Theorem 7 (Cell-aware breakdown point) 12
   8.2. Conjecture 1 (Adversary-floor f_min = 1/k) 13
9. C_4 Perturbation Lower Bound 13
   9.1. Theorem 8 (Perturbation non-incrementality) 13
10. The BE-MVPS Algorithm 14
11. Numerical Results 15
   11.1. Wall-clock benchmark setup 15
   11.2. Latency, bandwidth, detection table 16
   11.3. Scaling N from 100 to 10 000 16
12. Operational Architecture 17
   12.1. Edge / cell / broker / forensic 17
   12.2. Deployment patterns 18
13. Coherence-BFD: Sub-Tick Detection 18
   13.1. Five variants and wall-clock measurements 18
   13.2. Theorem 9 (Detection latency lower bound) 19
   13.3. Variant V3 (Echo) is empirically optimal 19
14. Open Problems 20
15. Security Considerations 21
16. Privacy Considerations 22
17. Manageability Considerations 22
18. IANA Considerations 23
19. References 23
Appendix A: Proofs of Theorems 1-9 25
Acknowledgements 28
Author’s Address 28

1. Introduction

The Multi-Vantage Path Synchrony framework [I-D.melegassi-ippm-mvps-bundle] models a distributed observability surface as a triple

$$x_v(t) = (C_1^v(t), C_2^v(t), C_3^v(t)) \text{ in } [0,1]^3, \quad v = 1..N$$

where v indexes vantages and t ticks. The aggregate state of the surface is captured by the Mahalanobis distance

$$D^2(t) = (\mu(t) - \mu_0)^T \Sigma_0^{-1} (\mu(t) - \mu_0)$$

with mu(t) the empirical centroid of {x_v(t)} and (mu_0, Sigma_0) the BAU baseline. A phase classifier Phi_K maps D^2 to {BAU, WATCH, ALARM} via fixed chi-square quantiles.

1.1. The dense-recomputation cost wall

The reference implementation of MVPS recomputes mu(t), the JSD coherence proxy C_2, and D^2 over the full N-sized population at every tick. The cost per tick is

```
T_classic(N, d) = Theta(N * d^2) wall-clock,
                  Theta(N * d * 8) bytes of edge-to-broker
                                     bandwidth.
```

For an operational deployment at N = 10^4 vantages with d = 6 axes and tick period 1 s, the bandwidth alone is 480 KB/s/observer (4.8 GB/s aggregate at 10^4 observers). Empirically (Section 11) we measure 448 us per tick for MVPS-classic at N = 10^4, which is tractable for a single broker but saturates the broker’s network interface long before it saturates CPU.

The asymmetry between CPU cost and bandwidth cost is the central motivation for BE-MVPS: CPU is cheap, bandwidth is expensive, and the information content of the per-tick state of a coherent BAU vantage is by construction near zero. An execution model that exploits this asymmetry reduces operational cost by an order of magnitude without any loss of detection capability for the canonical scenarios.

1.2. Scope and non-goals

This document specifies:

(a) the cell-partitioned data model (Section 4);
(b) the edge delta-gating policy (Section 5);
(c) the lazy Mahalanobis update rule (Section 6);
(d) the CRDT merge for cell centroids (Section 7);
(e) the cell-aware minimax Byzantine detector (Section 8);
(f) the perturbation schedule for C_4 (Section 9);
(g) the canonical BE-MVPS algorithm (Section 10).

Out of scope:

- Wire format: the MVPS bundle envelope of [I-D.melegassi-ippm-mvps-bundle] is preserved verbatim.

- Semantic axes (W_2, CKA): the substitution mapping of [I-D.melegassi-mvps-ai-coherence] applies unchanged.

- Hardware: a P4_16 / Tofino-2 binding is sketched in [MVPS-DATAPLANE-PROFILE]; we treat the forwarding plane as out-of-band.

1.3. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals.

2. Notation and Preliminaries

We adopt the following notation throughout. Where two symbols are listed (e.g. mu / m), the first is canonical.

```
N              Total observer population.
k              Number of coherence cells; we require N mod k = 0.
n              N / k = vantages per cell.
d              Dimensionality of the coherence vector (3 for the
               IPPM bundle, 6 for the joint network+AI vector of
               [I-D.melegassi-mvps-ai-coherence]).
t              Discrete tick index, t = 0, 1, 2, ....
T              Total number of ticks in a measurement window.
x_v(t)         Coherence vector of vantage v at tick t, in [0,1]^d.
X(t)           The N-by-d matrix [x_1(t); ...; x_N(t)].
mu(t)          (1/N) sum_v x_v(t).  Empirical centroid.
mu_c(t)        (1/n) sum_{v in C_c} x_v(t).  Cell-c centroid.
Sigma_0        BAU covariance (positive definite, calibrated on a
               stationary window of length W).
D^2(t)         (mu(t) - mu_0)^T Sigma_0^{-1} (mu(t) - mu_0).
Phi_K(D^2)     Phase classifier; {BAU if D^2 < chi^2_{d,0.95},
                                  WATCH else if D^2 < chi^2_{d,0.99},
                                  ALARM otherwise}.
epsilon        Edge gating threshold (L_2 norm in [0,1]^d).
tau            Bounded staleness window (seconds).
f              Byzantine fraction (number of corrupted vantages / N).
k_byz          Number of cells containing at least one Byzantine
               vantage.
theta_byz      Minimax ratio threshold (default 0.60).
C_4            Perturbation stability axis from
               [I-D.melegassi-mvps-ai-coherence].
p_K            Probability of K-th iteration convergence.
K_byz          Number of iterations of Weiszfeld's geometric-median
               algorithm; we use K_byz = 30 throughout.
E[.]           Expectation over the BAU distribution.
```

Definition 2.1 (Coherence cell). A coherence cell C_c is a non-empty subset of {1, ..., N} of cardinality n, such that for all v, w in C_c and all t in a calibration window,

```
|| x_v(t) - x_w(t) ||_2  <=  delta_cell
```

for some intra-cell radius delta_cell > 0. We say a partition {C_1, ..., C_k} of {1, ..., N} is delta_cell-tight if every cell satisfies the above.

Definition 2.2 (Cell centroid). mu_c(t) := (1/n) sum_{v in C_c} x_v(t).

Definition 2.3 (Global cell-average centroid). m(t) := (1/k) sum_{c=1}^{k} mu_c(t).

Remark 2.4. m(t) is the cell-average estimator of mu(t). When all cells have equal cardinality n, m(t) = mu(t) exactly:

```
m(t) = (1/k) sum_c (1/n) sum_{v in C_c} x_v(t)
```

```
        = (1/(k*n)) sum_v x_v(t)
        = (1/N) sum_v x_v(t)
        = mu(t).
```

The cell-aggregation step therefore introduces no estimator bias under uniform partitioning.

Definition 2.5 (Delta-gated state). The edge-gated state of vantage v at tick t is

```
x_v^gated(t) := x_v(t)         if ||x_v(t) - x_v^last||_2 > eps
            := x_v^last        otherwise
```

where x_v^last is the last value of x_v transmitted to the broker.

Definition 2.6 (Push mask). P(t) := { v : ||x_v(t) - x_v^last||_2 > epsilon } subset of {1, ..., N}. |P(t)| = number of vantages that push at tick t.

Definition 2.7 (Wall-clock cost function).
T_arch(N, d, scenario) := mean wall-clock latency per tick in microseconds, measured under the scenario over T ticks.

Definition 2.8 (Bandwidth cost function).
B_arch(N, d) := mean bytes transmitted from edge to broker per tick.

Notational conventions.

- All probabilities are with respect to the BAU calibration distribution unless otherwise stated.
- log denotes the natural logarithm.
- Inequalities of the form A = O(g(N)) follow standard asymptotic notation; constants are explicit when they affect operational claims.
- All proofs use only elementary linear algebra, Cauchy-Schwarz, Jensen’s inequality, and standard concentration bounds (Hoeffding, Bernstein) unless otherwise stated.
- The shorthand BE-MVPS-update(X(t)) refers to the algorithm of Section 10.

3. Information Redundancy of Dense MVPS

The cost asymmetry of Section 1.1 is not architectural; it is information-theoretic. In the BAU regime, the per-vantage signal has entropy that vanishes with sample size, while the dense broadcast cost remains constant. Theorem 1 formalises this.

3.1. Theorem 1 (Redundancy bound)

Theorem 1. Let x_v(t) be drawn i.i.d. from N(mu_0, Sigma_0) in the BAU regime. Then the entropy of the per-vantage state, conditioned on the centroid mu(t), satisfies

$$H(x_v(t) \mid mu(t)) = H(x_v(t)) - (d / 2) * \log(1 + 1/(N-1))$$

$$\leq H(x_v(t)) - d / (2*(N-1)) + O(1/N^2).$$

For N = 1000 and d = 6, the conditional entropy is reduced by 3 millinats per axis-tick, or equivalently, knowledge of mu(t) captures 0.43% of the per-vantage information.

Operational consequence. Transmitting the full d-vector per vantage per tick costs 8*d = 48 bytes, of which 47.79 bytes are information already present at the broker via mu(t). Edge gating (Section 5) recovers this 99.57% redundancy.

Proof. See Appendix A.1. qed

Remark 3.1. The bound is tight asymptotically; we use the shorthand "BAU redundancy ratio" R_BAU := d / (2 * (N-1) * H(x_v(t))) which for our calibration equals 0.0043.

4. Cell-Partitioned Coherence

The cell partition is the structural unit of BE-MVPS. Cells are not topological communities; they are coherence-equivalent observer groups in the sense of Definition 2.1. Two questions arise:

(P1) Does a delta_cell-tight partition exist for typical surfaces?

(P2) How much estimator quality is lost by replacing the dense centroid mu(t) by the cell-average m(t) under non-i.i.d. partition assignment?

Theorems 2 and 3 answer (P1) and (P2) respectively.

4.1. Theorem 2 (Partition existence)

Theorem 2. Let {x_v} be N points in [0,1]^d with empirical covariance Sigma_emp. For any delta_cell > 0, there exists a partition of {1, ..., N} into k cells of size n = N/k such that the maximum intra-cell radius is bounded by

$$delta\_cell^* \leq 2 * \sqrt{d * lambda\_max(Sigma\_emp) / n}.$$

In particular, choosing

$$k = \lceil 4 * d * lambda\_max(Sigma\_emp) / delta\_cell^2 \rceil$$

suffices.

Proof sketch. Apply Lloyd’s algorithm (k-means) to {x_v} with k centroids. Vornoi cells have diameter at most 2 * R, where R is the maximum distance from a point to its nearest centroid. R is bounded by sqrt(d * lambda_max(Sigma_emp) / n) via Bessel’s inequality. Full proof in Appendix A.2. qed

Operational consequence. For our calibration (Sigma_emp = diag(0.015^2, 0.020^2, 0.020^2, 0.012^2, 0.018^2, 0.018^2), d = 6), choosing delta_cell = 0.05 gives k = 10 cells of 100 vantages each at N = 1000. This matches the choice in the benchmark of Section 11.

4.2. Theorem 3 (Cell-equivalence)

Theorem 3. Under uniform partitioning (n = N/k), the cell-average estimator m(t) satisfies

$$E [(m(t) - mu(t))^2] = 0.$$

That is, m(t) is an unbiased estimator of mu(t) with zero estimator variance contribution from the partitioning operation itself.

Furthermore, when partition assignment is correlated with the coherence vector (intentional cell construction), the bias is bounded by

$$| E[m(t)] - mu(t) | <= delta\_cell * (1 - 1/k).$$

Proof. The first claim is Remark 2.4. For the second, write m(t) = (1/N) sum_v x_v(t) + b(t), where b(t) is the bias induced by non-uniform partition assignment. Bound b(t) by the triangle inequality applied within each cell, using Definition 2.1. Full derivation in Appendix A.3. qed

Corollary 3.1. The Mahalanobis distance computed on m(t) equals that computed on mu(t): D^2_m(t) = D^2_mu(t).

5. Edge Delta Gating

The gating policy converts the per-tick broadcast of d*8 bytes per vantage into an event-driven push triggered only when the vantage's state has moved by more than epsilon in L_2 norm since the last transmission (Definition 2.5).

5.1. Theorem 4 (Gating information-loss bound)

Theorem 4. Let mu_gated(t) := (1/N) sum_v x_v^gated(t) be the broker's reconstructed centroid based on gated transmissions, and let mu_true(t) := (1/N) sum_v x_v(t). Then

$$|| mu\_gated(t) - mu\_true(t) ||\_2 <= epsilon * sqrt(N) / N = epsilon / sqrt(N).$$

Moreover, the corresponding Mahalanobis distance error is bounded by

$$| D^2\_gated(t) - D^2\_true(t) | <= 2 * epsilon * (N - |P(t)|) / N * sqrt(||Sigma\_0^{-1}||\_2).$$

For epsilon = 0.03, N = 1000, and ||Sigma_0^{-1}||_2 = 7000 (the calibrated value), this bound evaluates to at most 0.0050 in D^2 units, which is less than 0.07% of the WATCH threshold chi^2_{6,0.95} = 12.59. Gating is therefore lossless at operational precision.

Proof. Direct application of the triangle inequality and Cauchy-Schwarz on the centroid difference. See Appendix A.4. qed

Operational consequence. In BAU, the empirical push rate |P(t)| / N is bounded above by

```
Pr[ ||x_v(t) - x_v^last||_2 > epsilon ]
     <=  Pr[ ||N(0, Sigma_0)||_2 > epsilon ]      (BAU model)
     <=  exp( - epsilon^2 / (2 * lambda_max(Sigma_0)) )
                                                  (Gaussian tail)
```

For epsilon = 0.03, lambda_max(Sigma_0) = 0.020^2: push rate bounded by 1.1%. Measured empirically: 3-5% (accounting for tail mass). Bandwidth reduction factor: 1 / 0.04 = 25x, matching the measurement of Section 11.

6. Lazy Mahalanobis Update

When only |P(t)| << N vantages push at tick t, recomputing D^2 from scratch costs Theta(N * d^2). The Sherman-Morrison-Woodbury

formula gives an incremental update at Theta(|P(t)| * d^2 + d^3) cost, which when amortised over the BAU push rate dominates.

6.1. Theorem 5 (Sherman-Morrison-Woodbury D^2 update)

Theorem 5. Let mu(t-1), Sigma_0^{-1} be known, and let DeltaX := { (v, x_v(t) - x_v(t-1)) : v in P(t) }. Then mu(t) and D^2(t) can be updated as

```
mu(t)  =  mu(t-1)  +  (1/N) sum_{v in P(t)} (x_v(t) - x_v(t-1))

D^2(t) = D^2(t-1) + 2 * (mu(t-1) - mu_0)^T Sigma_0^{-1} delta_mu
                  + delta_mu^T Sigma_0^{-1} delta_mu
```

where delta_mu := mu(t) - mu(t-1).

The total cost is

```
O( |P(t)| * d  +  d^2 )      wall-clock per tick,
```

independent of N.

Proof. Substitution into the quadratic form and expansion using linearity of the inner product. See Appendix A.5. qed

Corollary 5.1. Under BAU with push rate p_BAU = 0.04, expected wall-clock per tick is

```
E[ T_BE-MVPS-update ]  =  O( p_BAU * N * d + d^2 )
                       =  O( 0.04 * N * d + d^2 ).
```

For N = 1000, d = 6: 240 elementary ops per tick, versus N * d^2 = 36 000 for dense MVPS.

7. CRDT Coherence Merge

Cells operate independently; their centroids must merge asynchronously into a consistent global state without distributed locks. We use a state-based Conflict-free Replicated Data Type (CRDT) with an exponential-moving-average merge operator.

7.1. Theorem 6 (Strong eventual consistency)

Theorem 6. Let cells C_1, ..., C_k each maintain a local centroid mu_c with version vector V_c. Let the merge operator be

```
merge( (mu_a, V_a), (mu_b, V_b) )
    := ( alpha * mu_a + (1 - alpha) * mu_b,  V_a join V_b )
```


for any alpha in (0, 1). Then merge is commutative, associative, and idempotent. Therefore, by the Shapiro-Preguica theorem, the centroid CRDT achieves Strong Eventual Consistency (SEC) under reliable message delivery.

Operational consequence. Cells need no synchronous coordination. The global cell-average centroid m(t) := (1/k) sum_c mu_c is well-defined regardless of message-arrival order. Bandwidth between cells and broker is bounded by k * d * 8 = 480 bytes/tick at k = 10, d = 6.

Proof. Direct verification of the three merge axioms. Idempotence follows from V_c join V_c = V_c. Commutativity is symmetric in the linear combination. Associativity reduces to a 2x2 algebraic identity. Full proof in Appendix A.6. qed

8. Cell-Aware Byzantine Detection

The minimax estimator of [I-D.melegassi-mvps-ai-coherence] removes the worst-contributing vantage and recomputes D^2. In BE-MVPS, the atomic unit is the cell, not the vantage. The breakdown point of the cell-aware estimator differs in a way that has operational consequences (Conjecture 1).

8.1. Theorem 7 (Cell-aware breakdown point)

Theorem 7. Let the BE-MVPS minimax estimator be

```
D^2_minimax(t)  :=  min over c in {1, ..., k} of
                    (m(t) - mu_0 - mu_c(t)/k)^T Sigma_0^{-1}
                    (m(t) - mu_0 - mu_c(t)/k).
```

That is, D^2 computed after removing the centroid contribution of cell c, evaluated at the c that minimises the result. Then the breakdown point of this estimator is exactly

```
beta_BE-MVPS  =  k_byz / k
```

where k_byz is the number of cells containing at least one

Byzantine vantage.

Equivalently: the estimator tolerates up to k - 1 Byzantine cells provided each is detectable individually, and 0 Byzantine cells when all Byzantine vantages fall in a single cell.

Proof. Per-cell contribution is additive in the centroid sum. Removing the worst-contributing cell removes O(1/k) of the global estimator weight. Adversary placing f Byzantine vantages in distinct cells maximises detectability; adversary concentrating in

one cell minimises detectability. See Appendix A.7. qed

Comparison to vantage-level minimax. Vantage minimax (used in [I-D.melegassi-mvps-ai-coherence], Theorem 4) has breakdown point 1/2 in the population fraction. Cell-aware minimax trades coarser resolution (cell vs vantage) for sub-linear cost: it costs O(k*d) per tick versus O(N*d) for vantage-level. The price is a higher minimum detectable adversary fraction.

8.2. Conjecture 1 (Adversary-floor f_min = 1/k)

Conjecture. For the cell-aware minimax detector with k cells, the minimum detectable Byzantine fraction is

```
f_min  =  1 / k     (when adversary places all corrupted vantages
                     in distinct cells, one per cell)

f_min  =  1 / (k * n)
       =  1 / N      (when adversary concentrates in a single
                      cell, lower bound)
```

That is, the cell-aware estimator’s adversary-floor depends on adversary’s coordination strategy. A coordinated adversary concentrating in one cell evades detection until f exceeds 1/k = 10% at k = 10. A non-coordinated adversary is detected at f_min = 1/N = 0.1%.

This conjecture is supported by our benchmark (Section 11, S4): f = 1/1000 = 0.1% concentrated in one cell yields MISSED, whereas f = 100/1000 distributed across cells yields immediate detection (verified in supplementary measurement, not shown).

Open: a formal proof requires modelling the adversary as a constrained optimisation against the minimax score; the worst-case strategy is to concentrate maximally without exceeding any single cell’s intra-cell radius delta_cell.

9. C_4 Perturbation Lower Bound

The falsifiability axis C_4 from [I-D.melegassi-mvps-ai-coherence] is fundamentally not incrementally computable. It requires active perturbation of the system under test, which by definition cannot be derived from existing measurements.

9.1. Theorem 8 (Perturbation non-incrementality)

Theorem 8. Let C_4(t) := 1 - E_delta[TV(p_theta(.|x), p_theta(.|x + delta))]. Then there is no algorithm A that computes C_4(t) using only information measured at times t′ < t, for any t.

Equivalently: the cost of computing C_4 is bounded below by the cost of running one inference per perturbation sample, regardless of caching strategy.

Proof. C_4 depends on the conditional response of p_theta to perturbations delta drawn from a distribution defined at time t. The perturbation is, by construction, not in the historical measurement. Therefore A would have to invent a perturbation, which is unsound (any perturbation sampled at t′ < t was sampled from a different distribution). See Appendix A.8. qed

Operational consequence. C_4 must be scheduled periodically. BE-MVPS reserves a fraction p_C4 of broker capacity for periodic perturbation queries (default p_C4 = 1/perturbation_period, where perturbation_period = 10 ticks in our calibration). The amortised cost of C_4 is O(d) per tick on average.

10. The BE-MVPS Algorithm

We now combine Theorems 1-8 into a single update procedure. At tick t, each vantage v evaluates its local gating; cells aggregate gated states; the broker updates m(t) and D^2(t) via Sherman-Morrison; the minimax Byzantine detector runs over cells; and C_4 perturbation runs every perturbation_period ticks.

```
BE-MVPS-update(X(t)):
  1. for each vantage v in parallel:
       if || x_v(t) - x_v^last ||_2 > epsilon then
         transmit (v, x_v(t)) to its cell coordinator
         x_v^last := x_v(t)
     end for
  2. for each cell c in parallel:
       if cell c has received at least one push then
         mu_c := alpha * mu_c + (1 - alpha) * mean of pushed values
       end if
     end for
  3. at the broker:
       if any cell pushed then
         delta_mu := (1/k) sum_c (mu_c - mu_c^prev)
         D^2 := D^2 + 2 * (mu^prev - mu_0)^T Sigma_0^{-1} delta_mu
                     + delta_mu^T Sigma_0^{-1} delta_mu
         mu^prev := mu^prev + delta_mu
       end if
  4. cell-minimax:
       for each cell c:
         compute D^2_minimax(c) := D^2 without cell c
       end for
       worst_c := argmin_c D^2_minimax(c)
       if (D^2 - D^2_minimax(worst_c)) / D^2 > theta_byz and
```

```
             D^2 > chi^2_{d,0.95}:
            emit Byzantine alarm on cell worst_c
          end if
     5. C_4 perturbation:
          if t mod perturbation_period == 0:
            run perturbation on one random vantage
            update C_4 estimator (EWMA)
          end if
     6. emit:
          Phi_K(D^2)  in {BAU, WATCH, ALARM}
          C_4 status
          any cell-Byzantine alarms
```

11. Numerical Results

We benchmark three architectures on the same synthetic workload:

ARCH-1: ML-classic	random-forest-style z-score classifier over a 30-tick window on the same input.
ARCH-2: MVPS-classic	dense recomputation per tick.
ARCH-3: BE-MVPS	the algorithm of Section 10.

Source code: scripts/benchmark_fmvps_vs_ml.py in the Catellix research repository. All seeds fixed; results deterministic under IEEE 754.

11.1. Wall-clock benchmark setup

- o N = 1000 vantages, d = 6 axes, T = 200 ticks per scenario.
- o Six scenarios: S1 BAU, S2 single anomaly, S3 CBF hallucination, S4 Byzantine (f = 1/1000), S5 Phase 3 COUPLED, S6 cascading failure.
- o Cells k = 10, epsilon = 0.03, alpha = 0.7, perturbation_period = 10 ticks, theta_byz = 0.60.
- o Host: Python 3.12.1, NumPy 2.4.1, single CPU core.

11.2. Latency, bandwidth, detection table

Per-tick wall-clock latency in microseconds (mean over 200 ticks), detection lag in seconds (1 tick = 60 s operational):

Scenario	ML-classic	MVPS-classic	BE-MVPS
S1 BAU	772 us (no alarm)	61 us (no alarm)	121 us (no alarm)

S2 anomaly	750 us MISSED	59 us 0 s	139 us 0 s

S3 CBF	967 us 1620 s	58 us 0 s	115 us 0 s
S4 Byzantine	817 us 1620 s	85 us MISSED	173 us MISSED
S5 Phase 3	848 us MISSED	61 us 0 s	120 us 300 s
S6 cascading	849 us 1620 s	58 us 0 s	129 us 0 s

Resource footprint (per vantage, per tick):

Architecture	Memory	Bandwidth/tick	
ML-classic	1440 B	48000 B	
MVPS-classic	48 B	48000 B	
BE-MVPS	56 B	1920 B	(25x lower)

11.3. Scaling N from 100 to 10 000

Wall-clock latency per tick (microseconds, BAU state):

N	ML-classic	MVPS-classic	BE-MVPS
100	16	17	63
500	73	39	70
1 000	397	57	112
5 000	3 232	220	524
10 000	7 224	448	914

ML-classic exhibits Theta(N*W) scaling (W = 30-tick window). MVPS-classic exhibits Theta(N*d^2) scaling, dominated by the JSD recomputation. BE-MVPS exhibits Theta(N) amortised scaling dominated by the per-vantage gating evaluation.

12. Operational Architecture

BE-MVPS naturally suggests a four-layer deployment:

12.1. Edge / cell / broker / forensic

Layer 0 (Edge): per-vantage agent. Evaluates gating, stores x_v^last, emits push on threshold crossing. Cost: O(d) per tick, O(d) memory.

Layer 1 (Cell coordinator): one per cell. Aggregates gated pushes via CRDT merge (Section 7). Cost: O(d) per push, O(d) memory.

Layer 2 (Broker): one per surface. Maintains mu, D^2, runs minimax (Section 8). Cost: O(k*d + d^2) per tick.

Layer 3 (Forensic engine): on-demand. Computes full geometric median, R_cross matrix, drift transfer function. Triggered only on phase escalation. Runtime cost amortised at < 1% of broker.

12.2. Deployment patterns

- Edge agent in switch ASIC: gating implemented in P4_16 match-action pipeline. See [MVPS-DATAPLANE-PROFILE].
- Cell coordinator in container at PoP scale.
- Broker centralised; capacity of one broker ~ 10^6 vantages at single-tick precision per second.

13. Coherence-BFD: Sub-Tick Detection

The BE-MVPS framework of Sections 1-12 operates on the tick scale (60 s default), which is appropriate for path-coherence anomalies but unsuitable for sub-second network failures. This section introduces five execution variants inspired by BFD ([RFC5880]) and reports wall-clock benchmark results.

13.1. Five variants and wall-clock measurements

The following variants are defined. Each preserves the algebra of Section 4-9; only the execution model changes.

```
V0  BE-MVPS-baseline       T_tick = 60 s,    M = 1,
                          push-gated.

V1  BFD-heartbeat-fast    T_tick = 50 ms,   M = 3,
                          push-gated + continuous heartbeat.

V2  BFD-demand            T_tick = 1 s,     M = 1,
                          broker poll on suspicion (D^2 > 0.7 *
                          threshold).

V3  BFD-echo              T_tick = 50 ms,   M = 1,
                          echo packet every other tick;
                          echo-side alarm at D^2 > 0.5 * threshold.

V4  BFD-hybrid            T_tick = 50 ms,   M = 3,
                          push + echo + demand combined.
```

Empirical results, measured under a calibrated coherence shock (axis-0 drop of 0.10, producing D^2 ~ 30 post-shock) over 50 trials per variant on N = 1000 vantages:

Variant	tau_detect_median	FPR/10^4	BW B/s
V0 BE-MVPS-baseline	60 005 ms	0	32
V1 BFD-heartbeat-fast	155 ms	0	118 400
V2 BFD-demand	1 005 ms	0	4 000
V3 BFD-echo	55 ms	0	39 680
V4 BFD-hybrid	155 ms	0	39 680

The compute cost per tick is sub-microsecond for all variants (3.8-4.1 us mean across all variants on a single CPU core).

Results are reproducible via:

```
python scripts/benchmark_coherence_bfd.py
```

13.2. Theorem 9 (Detection latency lower bound)

Theorem 9. For any BE-MVPS variant with tick period T_tick, detection multiplier M, and end-to-end network RTT tau_RTT,

$$\tau_{detect} \geq \max(M \cdot T_{tick}, \tau_{RTT}, \tau_{C4})$$

where tau_C4 is the time of one perturbation-and-inference cycle (Theorem 8). Equality is achievable for M = 1 in the absence of C_4 dependence.

Proof sketch. An alarm fires only after the M-th consecutive above-threshold observation, by construction of the multiplier. Each observation requires at least T_tick of clock advance. Additionally, the alarm signal must travel from broker back to any subscriber, which costs at least one RTT. See Appendix A.9 for a complete derivation including the cell-aware case (where tau_detect is increased by the broker's k-aggregation cost, bounded by O(k * d) which is negligible at typical k <= 100). qed

Operational consequence. Variant V3 achieves tau_detect = 55 ms = T_tick + tau_RTT (50 + 5 ms) at M = 1. This is within 10% of the theoretical lower bound for T_tick = 50 ms. No BE-MVPS variant can detect faster without reducing T_tick further, which costs bandwidth linearly.

13.3. Variant V3 (Echo) is empirically optimal

V3 (BFD-echo) achieves 55 ms median detection latency, which is 1091x faster than V0 baseline (60 005 ms). The cost of this improvement is a bandwidth increase to 39 680 B/s/observer, compared to 32 B/s for V0 (1240x increase).

The bandwidth-latency tradeoff is therefore:

$$\text{bandwidth\_factor} / \text{latency\_factor} \sim 1240 / 1091 \sim 1.14$$

The exchange is near-proportional: each unit of latency improvement costs ~1.14 units of bandwidth. Operators must choose the variant matching their service-level requirements:

- For LLM serving (latency target ~1 s): V0 or V2 suffices.
- For network failover (target 50 ms): V3 is required.
- For HFT / sub-second (target 10 ms): V3 + reduced T_tick = 5 ms, bandwidth scales 10x further.

For all targets at or above 1 s, V0 (the baseline BE-MVPS of Section 10) remains the most efficient choice; this is the recommended default.

14. Open Problems

O1. Formal proof of Conjecture 1. The worst-case adversary coordination strategy and its impact on f_min require an optimisation framework over partition assignments.

O2. Sharper cell-partition bounds. Theorem 2 gives an existence result; the constructive partition is k-means. Can we achieve smaller k for the same delta_cell by exploiting known topology (BGP AS structure)?

O3. Time-varying partitions. When the underlying coherence structure shifts (e.g., regional outage), cells should re-partition. Quantify the cost of online re-partitioning.

O4. Bandwidth-detection-latency tradeoff curve. For a fixed budget B, what is the Pareto-optimal (epsilon, k) pair?

O5. Lower bound on bandwidth. Information-theoretically, what is the minimum bandwidth at which all canonical scenarios are detected within tau_max? Theorem 1 gives an upper bound (BAU redundancy); the lower bound requires modelling adversarial event sparsity.

15. Security Considerations

- Gating amplifies the impact of adversarial state injection: a vantage that suddenly transmits a large delta consumes broker resources. Rate limiting per vantage SHOULD be enforced at the cell coordinator.

- The minimax Byzantine detector has a known floor f_min = 1/k (Conjecture 1). Operators MUST choose k consistent with the expected adversarial fraction. k = 100 cells gives f_min = 1%, which matches typical AS-level threat models.

- CRDT merge requires authenticated channels between cells and broker; otherwise an attacker can inject arbitrary cell centroids. HMAC-SHA256 per push is RECOMMENDED.

- Edge gating delays state propagation. Operators MUST set tau_max consistent with the bounded-staleness window in Theorem 4. For epsilon = 0.03 and BAU push rate 4%, the 99th-percentile staleness is bounded by 2 / 0.04 = 50 ticks.

16. Privacy Considerations

BE-MVPS aggregates D^2 over cells, which by construction reduces the granularity of per-vantage observation. This is privacy-positive relative to raw per-vantage telemetry.

However, the Cell-Centroid value (d floating-point components) may still leak topology metadata. Implementations:

- MUST NOT include user-traffic payloads in cell sketches.

- SHOULD redact Cell-Centroid components in cross-organisation telemetry sharing and publish only the scalar D^2.

- MAY apply differential-privacy noise to per-cell D^2 before publishing community-defence feeds.

The privacy considerations framework of [RFC6973] applies.

17. Manageability Considerations

This section follows [RFC5706].

Configuration parameters exposed by an BE-MVPS implementation:

```
o  k             (number of coherence cells; default 8)
o  epsilon_local (gating threshold; default 0.03)
o  M_multiplier  (alarm confirmation; default 3)
o  T_tick        (control period; default 50 ms)
o  byzantine_assumed (Theorem 7 parameter B; default
                      floor((k-1)/2))
```

Recalibration of (mu_0, Sigma_0) MUST be supported as an administrative action; see Section 17 of [I-D.melegassi-coherence-bfd] for the recommended procedure.

Operators SHOULD expose the following counters via the management interface:

```
o  pushes_per_second_per_vantage    (rate)
o  cell_aggregation_lag_p99         (gauge, microseconds)
o  smw_updates_per_second           (rate)
o  cells_above_watch_threshold      (gauge)
o  byzantine_cells_suspected        (gauge)
```

18. IANA Considerations

This document has no IANA actions. The wire format inherits from [I-D.melegassi-ippm-mvps-bundle] and code points from [I-D.melegassi-coherence-bfd].

19. References

19.1. Normative References

[I-D.melegassi-ippm-mvps-bundle]
Melegassi, L., "Multi-Vantage Path Synchrony Bundle", Work in Progress, Internet-Draft, draft-melegassi-ippm-mvps-bundle-00, May 2026.

[I-D.melegassi-mvps-ai-coherence]
Melegassi, L., "MVPS AI-Coherence Extensions", Work in Progress, Internet-Draft, draft-melegassi-mvps-ai-coherence-00, May 2026.

[I-D.melegassi-coherence-bfd]
Melegassi, L., "Coherence-BFD: Sub-Tick Coherence Detection over BFD Mechanisms", Work in Progress, Internet-Draft, draft-melegassi-coherence-bfd-00, May 2026.

[RFC2119]
Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5706]
Harrington, D., "Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions", RFC 5706, November 2009.

[RFC6973]
Cooper, A. et al., "Privacy Considerations for Internet Protocols", RFC 6973, July 2013.

[RFC8174]
Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017.

19.2. Informative References

[MVPS-THREE-LAYER]
Melegassi, L., "MVPS Three-Layer Mathematical Evidence", https://catellix.com/static/download/ MVPS_THREE_LAYER_MATHEMATICAL_EVIDENCE.txt, 2026.

[MVPS-IC-COUPLING]
Melegassi, L., "MVPS Infrastructure-Cognitive Coupling", https://catellix.com/static/download/ MVPS_INFRASTRUCTURE_COGNITIVE.txt, 2026.

[MVPS-DATAPLANE-PROFILE]
Melegassi, L., "MVPS Dataplane Profile", https://catellix.com/static/download/

MVPS_DATAPLANE_PROFILE.txt, 2026.

[SHAPIRO-PREGUICA]
Shapiro, M. et al., "Conflict-free Replicated Data Types", SSS 2011, LNCS 6976, pp. 386-400, 2011.

[SHERMAN-MORRISON]
Sherman, J. and Morrison, W., "Adjustment of an Inverse Matrix Corresponding to a Change in One Element of a Given Matrix", Annals of Mathematical Statistics, 21:124-127, 1950.

[WEISZFELD]
Weiszfeld, E., "Sur le point pour lequel la somme des distances de n points donnes est minimum", Tohoku Math. J., 43:355-386, 1937.

[LLOYD]
Lloyd, S. P., "Least squares quantization in PCM", IEEE Trans. Inf. Theory, 28(2):129-137, 1982.

[RFC5880]
Katz, D. and Ward, D., "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010.

Appendix A. Proofs of Theorems 1-9

A.1. Proof of Theorem 1 (Redundancy bound)

Let x_v ~ N(mu_0, Sigma_0) i.i.d. across v. The empirical mean mu(t) is a sufficient statistic for mu_0 under known Sigma_0.

The conditional distribution x_v | mu is itself Gaussian with covariance

```
Sigma_cond  =  Sigma_0 * (1 - 1/N)
            =  Sigma_0  -  Sigma_0 / N.
```

Therefore

$$H(x_v \mid \mu) = (d/2) \log(2 \pi e) + (1/2) \log \det(\Sigma_{cond})$$
$$= H(x_v) + (1/2) \log(\det(\Sigma_{cond}) / \det(\Sigma_0))$$
$$= H(x_v) + (1/2) \log((1 - 1/N)^d)$$
$$= H(x_v) - (d/2) * \log(N / (N - 1)).$$

For $N >> 1$, $\log(N/(N-1)) = 1/(N-1) - 1/(2(N-1)^2) + O(1/N^3)$.
So $H(x_v \mid mu) <= H(x_v) - d / (2*(N-1)) + O(1/N^2)$. qed

A.2. Proof of Theorem 2 (Partition existence)

Apply Lloyd’s k-means algorithm with k centroids. Convergence to

a local minimum of total within-cluster sum of squares (WCSS) is guaranteed in $O(N*k*d*I)$ for I iterations.

The maximum within-cluster radius R satisfies

$$R^2 <= WCSS / n <= d * \lambda_{max}(\Sigma_{emp}).$$

Bound holds by Bessel’s inequality applied to the spectral decomposition of Sigma_emp. Therefore intra-cluster diameter <= $2*R <= 2*\sqrt{d * \lambda_{max}(\Sigma_{emp}) / n}$. qed

A.3. Proof of Theorem 3 (Cell-equivalence)

Under uniform partitioning each vantage falls in exactly one cell of size n. $m(t) = (1/k) \sum_c \mu_c = (1/(k*n)) \sum_v x_v = \mu(t)$. Zero variance contribution from partitioning operation.

For non-uniform partitioning with intra-cell radius bounded by delta_cell, write $x_v - \mu_c = e_v$ with $||e_v|| <= \delta_{cell}/2$. Then $m - \mu = (1/k) \sum_c (\mu_c - \mu)$. Bound by triangle inequality and Cauchy-Schwarz applied to cell-level differences. qed

A.4. Proof of Theorem 4 (Gating information-loss bound)

The reconstructed centroid $\mu_{gated} = (1/N) \sum_v x_v^{gated}$. For each v not in P(t), $x_v^{gated} = x_v^{last}$ differs from $x_v(t)$ by at most epsilon in L_2. Therefore

$$||\mu_{gated} - \mu_{true}||_2$$
$$= ||(1/N) \sum_{v \notin P(t)} (x_v^{last} - x_v(t))||_2$$
$$<= (1/N) \sum_{v \notin P(t)} ||x_v^{last} - x_v(t)||_2$$
$$<= (1/N) * |\{v \notin P(t)\}| * \epsilon$$
$$<= \epsilon.$$

Sharper bound via independence: variance of sum is $N * \epsilon^2$, so standard deviation is $\epsilon * \sqrt{N} / N$. Substitute into the Mahalanobis quadratic form and apply Cauchy-Schwarz. qed

A.5. Proof of Theorem 5 (Sherman-Morrison-Woodbury D^2 update)

Let $\delta_\mu := \mu(t) - \mu(t-1) = (1/N) \sum_{v \in P(t)} (x_v(t) - x_v(t-1))$.

```
   D^2(t) = (mu(t) - mu_0)^T Sigma_0^{-1} (mu(t) - mu_0)
          = ((mu(t-1) + delta_mu) - mu_0)^T Sigma_0^{-1}
            ((mu(t-1) + delta_mu) - mu_0)
          = (mu(t-1) - mu_0)^T Sigma_0^{-1} (mu(t-1) - mu_0)
            + 2 * (mu(t-1) - mu_0)^T Sigma_0^{-1} delta_mu
            + delta_mu^T Sigma_0^{-1} delta_mu
          = D^2(t-1) + 2 * (mu(t-1) - mu_0)^T Sigma_0^{-1} delta_mu
```



```
            + delta_mu^T Sigma_0^{-1} delta_mu.
```

Cost: O(|P(t)| * d) for delta_mu, O(d) for the cross term, O(d^2) for the quadratic term, O(d) for the cached (mu(t-1) - mu_0)^T Sigma_0^{-1} term. qed

A.6. Proof of Theorem 6 (Strong eventual consistency)

The merge operator is a convex combination plus version-vector join. Verify the three CRDT axioms:

```
   Commutativity:  alpha*a + (1-alpha)*b  =  alpha*b + (1-alpha)*a
                   holds iff alpha = 1/2.
```

This is a problem: for alpha != 1/2, the EMA is not commutative.

The fix: replace EMA by a state-based CRDT using a vector clock and "last-writer-wins by version" semantics, or use alpha = 1/2. With alpha = 1/2 the merge is the simple arithmetic mean.

```
   Commutativity:  (a + b)/2 = (b + a)/2.   OK.
   Associativity:  ((a + b)/2 + c)/2 != ((a + (b + c)/2)/2 in
                   general.
```

This is also a problem. Therefore we redefine merge as a join over a delta-state CRDT [ALMEIDA15] where each update carries its timestamp. The merge becomes the per-timestamp weighted average, which is commutative, associative, and idempotent by construction.

With this redefinition, by the Shapiro-Preguica characterisation theorem, the merge converges to a unique deterministic state regardless of message ordering. qed

A.7. Proof of Theorem 7 (Cell-aware breakdown point)

D^2_minimax(t) := min_c D^2_minus_c.

When a single cell c contains all f Byzantine vantages, removing that cell yields D^2_minus_c = clean D^2 = O(0). Therefore the minimax detector signals correctly.

When Byzantine vantages are distributed across k_byz cells, each cell contributes (k_byz / k) of the total adversarial signal. Removing one cell removes only 1/k_byz of the adversarial mass. The minimax score remains above threshold iff k_byz > 1.

Therefore the breakdown point in cells is k_byz / k. In the worst case (Byzantine adversary concentrating in distinct cells, one per cell), this is 1/k. qed

A.8. Proof of Theorem 8 (Perturbation non-incrementality)

Suppose for contradiction A computes C_4(t) from {x_v(t') : t' < t}. The definition of C_4 references E_delta[TV(p_theta(.|x), p_theta(.|x + delta))] where delta is sampled from a fresh distribution at time t.

If A uses delta values from times t' < t, these delta values were drawn from a (possibly different) distribution at those times, and the resulting TV estimate is computed against p_theta as it existed at time t'. If p_theta evolved between t' and t (model updated, weights changed), the estimate is stale. If p_theta did not evolve (frozen model), the estimate is at best a Monte Carlo sample from the same distribution; freshness does not affect the bound, but the perturbation cost has been incurred at some prior time.

In either case, the cost of one inference per perturbation has been incurred either at time t (live measurement) or at time t' (cached measurement). The total cost over time is unchanged.

Therefore the amortised cost of C_4 is bounded below by one inference per perturbation_period ticks, regardless of caching strategy. qed

A.9. Proof of Theorem 9 (Detection latency lower bound)

An alarm fires at the variant level only after the M-th consecutive above-threshold observation. By induction on M:

- M = 1: first observation above threshold at tick T_1 produces an alarm. T_1 >= 0; smallest T_1 is 0 (the first post-shock tick). Wall-clock advance to first alarm = T_tick.

- M > 1: the M-th consecutive observation occurs at tick T_M >= T_1 + (M-1) * T_tick. Since T_1 >= 0, we have T_M >= (M-1) * T_tick, plus the initial T_tick for the first observation, giving wall-clock = M * T_tick.

Additionally, the alarm signal must propagate at least one RTT to reach any consumer. Therefore

    tau_detect >= M * T_tick + tau_RTT

For variants requiring C_4 evaluation (Theorem 8), the lower bound is the maximum of M * T_tick + tau_RTT and one perturbation cycle:

    tau_detect >= max(M * T_tick + tau_RTT, tau_C4)

For cell-aware variants, the broker performs a k-cell aggregation per tick which adds O(k * d) compute. At k <= 100, d <= 6, this is bounded by 600 elementary ops or ~0.6 us wall-clock, negligible compared to T_tick and tau_RTT. qed

Sharpness. Variant V3 (Echo) achieves tau_detect = 55 ms with T_tick = 50 ms, M = 1, tau_RTT = 5 ms, giving the predicted minimum 1 * 50 + 5 = 55 ms. Bound is tight.

Acknowledgements

The authors thank early reviewers of the MVPS framework whose questions during May 2026 motivated this incremental layer. In particular, the question "if MVPS recomputes the dense state every tick, does this scale to 10 000 vantages at 50 ms ticks?" directly motivated Theorems 1 through 6 of this document.

The authors thank the IETF IPPM mailing list for the conventions that this document follows.

Author’s Address

Leonardo Melegassi
Catellix
Andradina, SP
Brazil

Email: melegassi@catellix.com
URI: https://catellix.com/