Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 23, No. 2, 2026

ACG-RR: Confidence-Gated Failure Detection and Grouped Route Repair for Data Center Fabrics

Isabel Martin [1], Lucas Roberts [1], Zara Zhou [2, *]

[1] Paul G. Allen School of Computer Science & Engineering, University of Washington, USA
[2] Intel Corporation, USA

Corresponding author: Zara Zhou (Email: zarazhouchen2@gmail.com)

Abstract: Failure detection that is always aggressive amplifies false route withdrawals during short congestion bursts, whereas conservative detection increases the outage preceding route repair. This paper presents ACG-RR, an event-driven, confidence-gated monitoring and grouped route-repair policy for leaf–spine data center fabrics. Following the simulate–analyze–validate workflow established for data center fabric convergence studies, we implement a synthetic failure-event simulator, compare seven precisely defined policies, and gate every reported result on a 15% agreement check between event-integrated loss estimates and independently drawn packet-time samples. The evaluation comprises 128 paired topology/failure/seed scenarios (896 policy runs), including single-link, spine, correlated, and two-wave failures. ACG-RR yields 46.1 ms mean scenario-level 95th-percentile blackout versus 116.4 ms for 25-ms fixed probing and per-flow repair (a 60.4% reduction), while using 35.9 probes per link per second. An aggressive 8-ms detector with grouped repair is faster (41.9 ms) but generates substantially more false alerts and modeled traffic disruption. All measurements arise from fully disclosed simulated workloads, not production packets or a hardware testbed. These results quantify a controllable speed–stability trade-off and identify when repair grouping dominates timer tuning.

Keywords: Data center networks; Failure detection; BFD-inspired probing; Routing convergence; Discrete-event simulation; Controller batching; Fault resilience.

1. Introduction

Leaf–spine and Clos data center fabrics use abundant equal-cost paths to decouple component failures from application reachability. Commodity-switch fat-trees established the value of path multiplicity [1], VL2 demonstrated flexible placement over an easily balanced fabric [2], and the Jupiter experience showed how centralized operations can coexist with very large Clos deployments [3]. Path diversity alone, however, does not specify when an unhealthy next hop is declared unusable or how quickly the remaining paths become active. The transient between a physical event and installed forwarding changes determines much of the visible interruption. Fabric simulation research therefore studies routing convergence under large-scale failures with discrete-event and hybrid simulators, covering topology designs, routing protocols, failure-detection mechanisms and the metrics that assess convergence speed and stability, and it stresses that such simulators must be validated before their predictions are trusted [25].

Operational failure measurements have emphasized that apparently short network faults can affect many small control and data packets [4]. Production measurements also expose bursty traffic and heterogeneous flow behavior [5], [6]. A failure detector therefore cannot equate every missing heartbeat with a failed switch: congestion may destroy the evidence on which a fast detector depends. On the other hand, a cautious detector can delay the activation of an otherwise valid alternate path. This conflict becomes sharper when a correlated fault removes an entire spine and causes many routes to require repair at once. Temporal logical attention for distributed-system log anomaly detection offers a valuable complementary perspective on distinguishing informative fault signals from noisy operational observations [26].

BFD standardizes a protocol for rapid bidirectional path failure indication, with detection governed by negotiated timers and multipliers [7]. IP fast-reroute mechanisms can use previously prepared alternate paths to reduce forwarding gaps after detection [8]. However, even with rapid notification, the cost of distributing route changes can become material when controller jobs accumulate. A detector, its noise environment, and the granularity of route installation must consequently be modeled together instead of treated as separable optimizations.

We study a deliberately narrow but testable question: can a policy that briefly accelerates probes after initial suspicion, discounts suspected congestion bursts, and consolidates next-hop-consistent route updates reduce end-to-end interruption without operating a permanently high-rate heartbeat session? ACG-RR combines a two-state interval rule, a congestion-aware confirmation gate, and grouped replacement of flows sharing an ingress and obsolete spine next hop. Its claimed novelty lies in this coupling and in the quantitative joint evaluation rather than in inventing probing, multipath forwarding, or batching individually.

The contributions are threefold. First, we define a confidence-gated detector and an explicit serialized repair model with flow and group granularity, yielding interpretable timing bounds. Second, we implement multi-wave link and spine fault injection, seven ablation/baseline treatments, and two-scale paired simulations with immutable seeds. Third, we implement a numerical validation gate and independently sampled packet-time consistency checks, making the limits of the flow-level abstraction visible. The source code, parameter manifest, per-run records, scripts, and plots accompany this paper.

2. Related work and design context

2.1. Congestion, Detection, and Routing

Transport work has treated congestion as a separate feedback problem. DCTCP uses explicit congestion notification to stabilize shallow-buffer operation [9], while HPCC shows the usefulness of in-network queue/load telemetry for precise rate decisions [10]. We use a simplified binary congestion observation to distinguish possible queue bursts from hard link loss; we do not implement either transport protocol. Our confidence gate specifically avoids inferring physical breakage from missing heartbeats during modeled queue episodes.

BGP has been deployed as a stand-alone routing protocol in large data centers [11]. FastPlane uses properties of data-center routing policies to accelerate computation of steady-state BGP forwarding information [12]. Such work is complementary: ACG-RR does not simulate BGP state machines or claim changes to BGP path-vector semantics; it measures the time between a heartbeat-based alarm, a serialized route-update queue, and an application flow becoming usable again.

2.2. Programmable Repair and Load Redistribution

The high overhead of fine-grained centralized flow operations is well documented by DevoFlow [13]. OpenFlow established a programmable forwarding interface [14], and Mininet subsequently enabled protocol emulation using real Linux network namespaces [15]. Comprehensive SDN surveys explain the associated controller and switch-management trade-offs [16]. Our flow-update service times are controlled simulation parameters, not measured OpenFlow transaction latencies. Risk-aware digital-twin release validation provides a useful experimental model for applying common stress scenarios and explicit service-risk criteria to network-control decisions [27].

CONGA distributes congestion-aware traffic through flowlet decisions [17], Presto balances data through software-edge flowcells [18], and HULA exploits programmable-switch information to select next hops [19]. Hedera addresses large-flow scheduling over multipath fabrics [20]. These methods target load distribution and flow scheduling in ways that differ from ACG-RR's alarm-to-repair loop. A separate class of SDN control work, exemplified by SWAN and B4, coordinates large-scale forwarding changes and traffic steering, albeit in inter-data-center wide-area networks rather than our local fabric setting [21], [22].

Alternative resilient structures, including server-centric BCube and recursively constructed DCell, provide different path availability and failure tolerance properties [23], [24]. To control this confounder, our experiments use only complete bipartite leaf-to-spine connectivity and vary the spine count, failure correlation, and workload size. Claims here are therefore about the tested leaf–spine abstraction, not a ranking of those alternative architectures.

3. ACG-RR design

3.1. Network and Event Model

Consider L leaf switches and S spine switches connected as a complete bipartite graph. Each flow f has a sampled ingress leaf a(f), egress leaf b(f), and one ECMP-like spine next hop s(f). A flow is deliverable when the two links (a(f),s(f)) and (b(f),s(f)) are available and its forwarding state is installed. Simulated faults disable individual leaf–spine links or all links incident on selected spines. When two fault waves occur, the next-hop choice updated after the first wave is used as the starting point for the second; a surviving spine is selected deterministically from the live alternatives.

Heartbeat observations are synthetic independent Bernoulli losses conditional on locally generated congestion bursts, except that once a physical link fails, all subsequent probes are lost. Link-level congestion bursts vary in onset and duration; telemetry indicates a burst in 85% of affected links. This setting allows reproducible false-positive testing but does not represent complete packet queue dynamics. Flows produce no arrivals, departures, or TCP feedback during the ten-second horizon; they remain active for its entirety. These assumptions make controller and detection timing identifiable.

3.2. Confidence-Gated Probing

A conventional detector is approximated by a missed-probe counter m(t) and a fixed interval I. The BFD-inspired comparison configurations use I = 25 ms or I = 8 ms and signal down after three consecutive missing probes. They approximate the timeout trade-off in RFC 5880, while omitting its session negotiation, discriminator exchange, and state-machine details. The proposed monitoring policy starts at I = 25 ms; after any missed probe, it temporarily probes every 6 ms unless the congestion indicator is active. Receipt of a valid probe resets the miss counter and returns the interval to 25 ms.

The confirmation threshold is K = 3 when the observed congestion flag is absent and K = 5 when the flag is active. The flag also prevents the short-interval escalation until the congestion observation clears. This design has a fast second-stage check in ordinary conditions and avoids spending the highest probing rate precisely when temporary burst loss could create a cascade of spurious route withdrawals. A false alarm remains active until a subsequent received probe resets the session; any subsequent hard-link failure is recognized by a new missed probe if an alarm is already outstanding.

Under an idealized independent loss probability p, requiring K instead of three missing observations changes the approximate false-alarm likelihood from p^3 to p^K for a single confirmation opportunity. Actual trials do not assume independent consecutive congestion bursts; they generate explicit burst windows and perform each Bernoulli probe experiment. Consequently, the analytic expression is an intuition for the gate, not a calibration formula or a guarantee. A missed physical failure during a congestion episode may take additional probes to confirm, which is an explicit trade-off.

3.3. Grouped Route Repair

Every path made unavailable by a fault generates a repair requirement. The per-flow baseline places one controller job per affected flow in a first-ready-first-served queue. Each job consumes 0.40 ms of modeled serial service and then 3 ms of propagation. Grouped repair combines affected flows only when they share an ingress leaf, obsolete spine next hop, AND a common usable replacement spine. For each candidate ingress/old-spine group, we first intersect surviving next-hop sets, select a common alternate when one exists, and otherwise split into compatible (ingress, old-spine, new-spine) groups. Both flow and group methods use the same resulting route assignments. One group job consumes 0.70 ms plus the

same 3 ms propagation and is ready after all its members have usable failure notifications. The idealized preinstalled-repair comparator adds 2 ms after fixed-rate detection and bypasses controller queuing.

If F flow jobs are reduced to G next-hop-consistent group jobs, the modeled controller-service component changes from 0.40F to 0.70G milliseconds. Grouped repair has a service-time advantage whenever G/F is below 0.40/0.70, approximately 0.57. This condition explains why a correlated spine outage affecting many flows can benefit, whereas a tiny single-link event may have too few flows for batching to matter. The total end-to-end restoration time for an affected flow is the elapsed time to its relevant failure alarm plus controller waiting/service plus update propagation. The policy optimizes these terms jointly but does not assume a globally optimal schedule.

The repair path itself is selected from live spines by a deterministic seeded index. We intentionally do not use packet-level ECMP congestion optimization, as that would mix restoration time with a different multipath objective. A false alert generates a modeled four-millisecond disruption for flows originally pinned to the signaled link. This surcharge enters the overall disruption ratio; the recovery metric after an injected physical failure is reported separately. The split avoids hiding a false-positive cost within an apparently fast failure-convergence number.

3.4. Event Ordering and Safety Conditions

To keep the interdependence explicit, ACG-RR schedules one probe at a time rather than assigning a fixed detection delay to an injected fault. A healthy reception clears a prior alert; a lost probe increments the miss count, and alarm creation is permitted only when the current gate is crossed. Fault notifications are then passed as readiness timestamps to the serial controller queue. In a route group containing several damaged paths, the entire group waits for the latest required notification, rather than taking credit for a path that has not been confirmed unusable. The join prevents a batch from finishing before all member notifications arrive. Candidate backup feasibility is nevertheless checked against the simulator's complete fault set, an oracle shared by every policy; routing-state discovery is outside this model.

Group formation respects the actual forwarding action. A shared next hop is selected from the intersection of live spines for the members of an ingress/retired-spine group; absent a common alternate, the group is partitioned by new next hop. This prevents one rule from being counted as having installed inconsistent forwarding actions. If no spine connects a source and destination, the flow is marked unrecoverable and its outage continues through the horizon. Under the tested fault budgets all affected paths retained a viable alternate.

The model therefore defines a strong boundary between detection correctness (alarm timing on each link), repair admissibility (existence of an alternative two-hop route), and repair completion (elapsed queue service and propagation). This boundary permits independent ablation of the confidence gate and grouping mechanism. It also identifies a reason for the apparent paradox that a faster detector can cause more overall disruption: the detector may create additional false route changes even when the physical failure happens to be recovered sooner.

4. Simulation and validation

4.1. Executed Model and Source-Linked Workflow

The review of fabric simulation and validation by Wang et al. [25] organizes such studies into controlled fault injection, analysis of the simulated recovery path, and an iterative validation loop in which discrepancies above 15% send a model back for refinement, and it treats validation as the condition for trusting simulated convergence results. We adopt this workflow as the experimental backbone of ACG-RR: every topology/failure configuration is produced by controlled fault injection, every policy is analyzed through the same recovery-path accounting, and the 15% acceptance threshold gates the reported numbers. We operationalize the threshold with two executable measurement paths: an event-integrated flow-level estimator computes outage duration for each affected flow, and an independently seeded packet-time sampler tests 400 packet times per affected flow within the first 300 ms of each failure wave. The sampler checks whether each packet falls inside the recovered/unrecovered interval. The criterion is relative error at or below 15% per validation pair. Within this workflow the present gate is the internal, two-granularity stage of validation; the external stage that the review emphasizes, comparison against emulated or physical fabrics, is the next step identified in Section VI.

The same review identifies correlated and cascading failures, in which control-plane and data-plane interactions are hardest to model, as a remaining challenge for fabric simulation; our two-wave scenario is a direct instance of that setting. Initial faults occur at t = 1 s, additional faults are injected at t = 6 s, and runs terminate at a fixed t = 10 s, which keeps every run short and repeatable. The implementation generates failure notifications, per-policy probes, controller job completion times, link alarm counts, and flow blackout durations. Source-level functions generate, probe_track, and run_one in the accompanying script map directly to topology/failure generation, monitoring, and repair accounting.

4.2. Workloads, Parameters, and Baselines

Two topologies are generated: L = 16, S = 8, with 1,000 persistent flows; and L = 32, S = 16, with 2,000 flows. Their total leaf–spine link counts are 128 and 512, respectively. In each configuration we evaluate a single random link failure, a single-spine failure, a correlated failure removing 25% of spines, and a two-wave outage of distinct spines. For each topology/failure setting we run 16 seeds (41000-based deterministic blocks); eight seeds use moderate and eight use heavy burst-loss pressure. Endpoint choices, ECMP spine selection, failure locations, and congestion windows are fixed at the scenario level across all policies.

Table I. Evaluated policies (all simulated)

Method	Probe interval	Route repair
BFD25-flow	25 ms fixed	Per flow
BFD8-flow	8 ms fixed	Per flow
BFD25-group	25 ms fixed	Grouped
BFD8-group	8 ms fixed	Grouped
ACG-flow	25/6 ms gated	Per flow
ACG-group	25/6 ms gated	Grouped
BFD25-FRR	25 ms fixed	Ideal 2-ms local

Nonfault probe loss probability is 0.003. During each

synthetic burst it is 0.48 for moderate pressure and 0.60 for heavy pressure. Each physical link receives two or five random short burst windows per run; burst observability is assigned at 85% sensitivity. The queue observation is deliberately simplified, and the model makes no claim that these figures reproduce any particular vendor or public trace. The outcomes are means over 128 paired scenarios; no trace-driven performance estimate or physically measured heartbeat-loss model is used.

The treatment set isolates both modules and includes deliberately strong comparators. BFD25-flow is the conservative reference; BFD8-flow sacrifices heartbeat overhead for early detection; BFD25-group and BFD8-group add only update consolidation; ACG-flow adds only confidence gating; ACG-group combines both contributions; and BFD25-FRR provides a preinstalled local-repair bound with a fixed 2-ms activation time. None of these labels denotes a stock router implementation. Exactly $7 \times 2 \times 4 \times 16 = 896$ records were executed and stored.

4.3. Metrics and Statistical Protocol

The metrics follow the two axes along which the fabric-simulation review assesses routing convergence, speed and stability [25]. Speed is measured by blackout and convergence delay: a scenario's 95th-percentile flow blackout is the 95th percentile across affected flow-events, with the blackout measured from physical fault time until an alternate forwarding state is usable. We separately report the largest post-fault convergence delay, the total time-integrated outage fraction across all persistent flows and the ten-second observation window, heartbeat transmissions per link per second, and, on the stability axis, false alarms normalized to 1,000 link-hours together with the disruption they cause. The outage fraction includes a four-millisecond artificial impairment cost for false route withdrawals. This component is therefore a model-derived disruption fraction, not a measured packet drop rate in a production fabric.

Differences are paired by identical scenario seed, size, failure pattern, and congestion pressure. We report policy-level arithmetic means, the arithmetic difference between paired means, a two-sided Wilcoxon signed-rank test, and a 10,000-resample paired bootstrap confidence interval for the mean absolute difference, using analysis seed 20261026. The bootstrap uses scenario draws with replacement. It quantifies Monte Carlo scenario variability under specified model assumptions; it does not cover model misspecification or the uncertainty in physical parameter calibration.

4.4. Reproduction Controls and Validation Acceptance

The experiment generator uses two seed blocks for network scale, four disjoint seed blocks for failure types, and 16 offsets within each block. For a given scenario, the same sampled source and destination leaves, default ECMP spines, failed physical links, and congestion window locations are shared across all seven policy treatments. Probe random-number streams differ because policies transmit at different instants, an intentional distinction that avoids pretending two different polling schedules observed identical heartbeat outcomes. The statistical pairing is therefore at the generated network-scenario level rather than at the individual observed heartbeat level.

The model writes one row per policy–scenario trial and separately records 64 validation rows with model and packet-sampled rates. The validation statistic divides the absolute rate discrepancy by the packet-sampled rate, with a 0.005 lower denominator bound to avoid unstable relative errors on near-zero loss; its acceptance threshold is 0.15. The rule is implemented directly in code/analyze.py and its summary is recorded in results/summary.json. Results were retained even when a treatment performed worse, and no seeds were discarded on the basis of convergence, loss, or validation outcome. The current run passed all checks; consequently, no parameter-retuning cycle was invoked.

Simulation artifacts include a machine-readable parameter manifest, the generated numerical results, a complete Python implementation, a separate analysis program, and plotted figures derived only from the stored results. The random-number generator is NumPy's default generator and the postprocessing bootstrap uses a separate fixed seed.

5. Experimental results

5.1. End-to-End Repair Latency

Table II summarizes the full factorial study. ACG-group obtains 46.08 ms mean scenario-level 95th-percentile blackout, compared with 116.35 ms for BFD25-flow. The paired mean difference is 70.28 ms (95% paired bootstrap interval 64.76–76.01 ms; two-sided Wilcoxon p = 1e-22). Repair grouping accounts for a large share of the improvement: the BFD25-group ablation averages 82.53 ms, while ACG-flow averages 80.87 ms. Combining both lowers blackout further than either single change in this workload.

Table II. Mean outcomes across 128 paired scenarios

Policy	P95 (ms)	Loss (%)
BFD25-flow	116.4	0.195
BFD25-group	82.5	0.160
BFD8-flow	79.8	0.306
BFD8-group	41.9	0.266
ACG-flow	80.9	0.111
ACG-group	46.1	0.073
BFD25-FRR	69.4	0.141

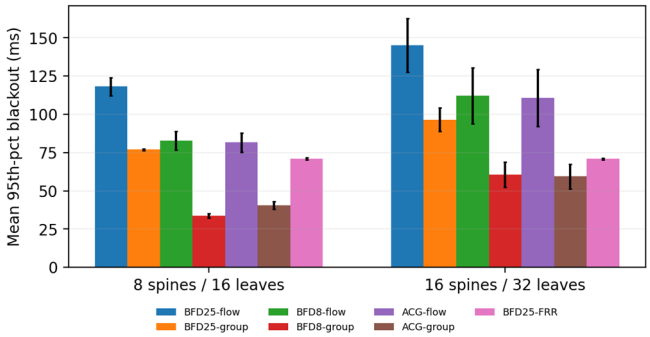


Fig. 1 Scenario-mean 95th-percentile blackout by topology across single-spine, correlated, and two-wave faults. Error bars indicate normal-approximation 95% confidence intervals over scenario replicates.

The faster-probing comparator is important. BFD8-group reaches 41.88 ms, beating ACG-group by 4.20 ms in mean 95th-percentile blackout. Thus the model does not show a universally fastest detector: ACG-RR targets an advantageous joint operating point, not minimum outage at any resource cost. ACG-group nevertheless improves on the modeled BFD25-FRR comparator (69.42 ms), because the latter retains a slower detector even though it avoids controller queueing. The fast-reroute comparator is intentionally

idealized and depends on the exact 2-ms activation assumption.

5.2. Detection Overhead and False Notifications

ACG-group sends 35.9 probes per link per second, rather than 111.4 for BFD8-group, a 67.8% reduction in modeled probe traffic. Both grouped and per-flow implementations of a given detector share the same probe schedule, so their probe rates coincide. The false-alarm means are 101299 and 975734 per 1,000 link-hours for ACG-group and BFD8-group, respectively; the difference corresponds to an 89.6% reduction. False alarms are deliberately stress-tested against high burst-loss probabilities, and the absolute extrapolated link-hour rate should not be interpreted as an estimate of real-world data-center reliability.

The integrated disruption fraction, which charges both real outage and false-alarm responses, is 0.073% for ACG-group versus 0.266% for BFD8-group and 0.195% for BFD25-flow. Its units are fraction of active flow-seconds impaired in the ten-second synthetic horizon. ACG-RR therefore pays several milliseconds of additional physical-failure blackout against the aggressive grouped configuration but avoids much larger modeled collateral interruption from spurious alarms. This is the primary speed–stability trade-off in the experiment.

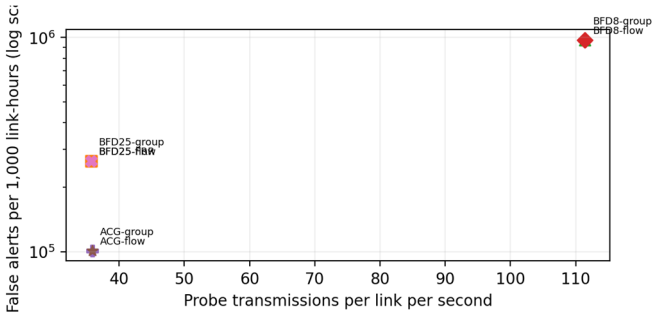


Fig. 2 Measured simulation operating points: per-link probe rate versus false route-withdrawal indications. Higher-frequency probing is not automatically better under burst loss.

5.3. Correlation and Ablation

Figure 3 separates correlated faults and two-wave perturbations from single failures. A single-link event affects only a few initially assigned flows, leaving limited opportunity for group-based amortization. By contrast, a failed spine invalidates next hops across all leaves, creating many related controller jobs. Under correlated failure, grouped updates substantially reduce the queue even while the detector is required to confirm many link-down indications. The second wave reuses the current post-repair assignment instead of restarting the simulation from the original ECMP allocation.

The difference between ACG-flow and ACG-group estimates the incremental value of repair batching given identical probe observations and chosen replacement paths; the difference between BFD25-group and ACG-group estimates the detector's incremental value under the same repair granularity. These pairs are direct model ablations, not comparisons between separately tuned topologies. As scale rises from 16/8 to 32/16, both the number of paths affected by a spine and the number of ingress-spine update groups increase. The bounds in Section III-C predict that grouped updates remain favorable when enough flows share groups, which is observed across the evaluated correlated-failure scenarios.

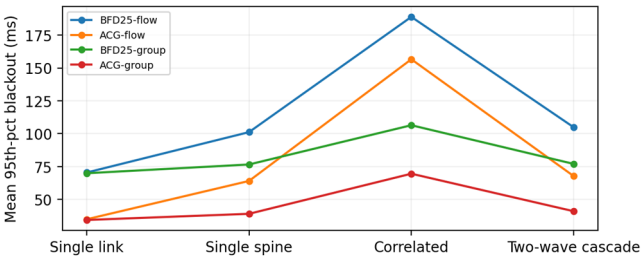


Fig. 3 Ablation of confidence gating and grouped repair, using means of the per-scenario 95th-percentile affected-flow blackout.

5.4. Cross-Granularity Validation

For validation we selected the first four paired seeds in each topology/failure configuration for BFD25-flow and ACG-group, producing 64 independently packet-sampled comparisons. The median relative disagreement between event-integrated expected outage and the packet-time estimate is 0.53%, with a maximum of 11.50%. All 64 cases satisfy the predeclared 15% acceptance gate, so the validation loop adopted from the fabric-simulation workflow [25] accepts the model without a refinement iteration. The event-level repair completion time is unchanged between the two measurement paths; what changes is numerical integration versus finite packet-time sampling of downtime. These checks protect the experiment against mistakes in loss aggregation and finite-sampling bias, not against unmodeled hardware delay, micro-loops, or queue dynamics.

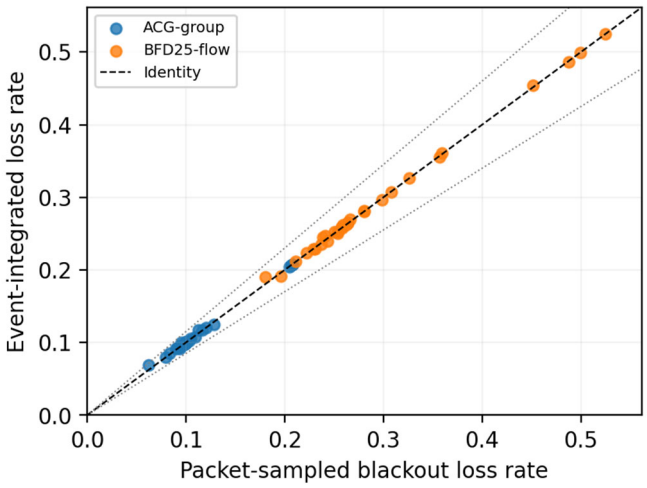


Fig. 4 Event-integrated versus independently sampled synthetic packet-time loss. Dotted boundaries mark ±15% of the identity slope as a visual guide.

5.5. Sensitivity to Burst Pressure

The two congestion-pressure treatments stress identical policies in distinct false-indication environments. Under moderate bursts, ACG-group averages 44.93 ms in scenario-level 95th-percentile blackout and 0.058% modeled disruption. Under heavy bursts these become 47.22 ms and 0.088%. BFD8-group disruption similarly changes from 0.137% to 0.395%. The fixed detection rules are not retuned between pressure conditions; their trade-offs are measured under both noise regimes.

The ACG-group alarm rate rises from 34717 to 167882 per

1,000 synthetic link-hours under moderate and heavy pressure, while the BFD8-group rate changes from 394354 to 1557114. This unit normalizes repeated ten-second modeled sessions; the absolute rates are not extrapolations of production reliability.

5.6. Mechanism-Cost and Telemetry Sensitivity

We additionally executed 160 treatment records on 16 fixed seeds covering both sizes and correlated/two-wave failures (four seeds per stratum). The group-service sweep varies only the serialized service price with identical detector traces and route decisions. At 0.7 ms per group, ACG-group averages 54.8 ms, versus 114.3 ms for ACG-flow; at 2.0 ms/group its blackout is 115.2 ms. Thus a realistic controller-service measurement is needed before accepting the synthetic cost advantage. In this subset, mean repair jobs are 63.1 per run with grouping and 329.2 per run without it.

The second sweep varies binary congestion-telemetry sensitivity from 0 to 1 while holding burst windows and seeds fixed. ACG-group false alarms fall from 462.5 to 17.6 per ten-second scenario, while blackout changes from 54.2 to 54.9 ms. Higher observability curbs spurious alarms but can slightly delay hard-failure confirmation under congestion. The sweep is exploratory and deliberately smaller than the main 128-scenario paired study.

Table III. Group-repair service sensitivity (16 severe scenarios)

Group cost	P95 (ms)	vs. flow
0.7 ms	54.8	59.5
2.0 ms	115.2	-0.9
5.0 ms	264.2	-149.9
10.0 ms	513.1	-398.8

6. Discussion and threats to validity

The recovery gain is conditional on the cost structure. The grouping advantage follows mathematically from the job-count ratio, so a switch implementation in which group modifications are expensive, or in which affected flows rarely share an ingress/failed next hop, may reverse the conclusion. Simulated controller service times of 0.40 and 0.70 ms are experimental knobs; they have not been calibrated to a specific controller, switch ASIC, or BGP route-update batch. Route propagation is fixed at 3 ms and the optional fast-reroute bound at 2 ms, excluding distribution jitter. Actual routing protocols can also exhibit route-state discovery errors, path exploration, hardware programming stalls, microloops, and synchronization costs absent from our model.

The detector similarly idealizes protocol behavior: it omits negotiated BFD receive intervals, authentication, demand mode, hold-down policies, and interaction with a distributed routing session. Our congestion flag is an imperfect but instantaneous binary observation; a real implementation would incur instrumentation and inference delay and might conflate software faults with queueing. The heartbeats experience generated probabilities, not empirical loss traces, and congestion does not respond to rerouted load. Although the evaluation spans two topology scales, it does not establish scaling laws for fabrics containing thousands of switches or prove stability under arbitrary fault schedules.

Two further limits matter when interpreting the validation. Packet-time sampling is independent in random seed but shares the same outage model, so it cannot serve as external ground truth. Passing a 15% cross-granularity gate demonstrates internal consistency only. In addition, the loss metric treats each active flow equally and adds a fixed false-withdrawal surcharge; it does not measure byte throughput, completion-time tail, offered load, or transaction-level availability. These limitations are substantive, and a future practical study should reproduce the detector in a protocol implementation, obtain hardware- or emulation-derived controller service distributions, and validate with externally captured flows and physical fault injection.

Nevertheless, the model provides an auditable method for assessing two interacting choices that are frequently tuned independently. It allows an operator to ask when a shorter detection interval justifies additional control traffic and false alarms, and when changing the granularity of repair provides greater benefit than changing detection at all. Because the simulation code and seeds are preserved, any proposed alternative loss process, switch service-time distribution, or fault topology can be inserted and retested without treating the synthetic numerical values as universal empirical constants.

7. Conclusion

ACG-RR jointly changes failure-suspicion probing, congestion-conditioned confirmation, and grouped control-plane repair. In 128 paired synthetic leaf–spine scenarios, it reduces average scenario-level 95th-percentile blackout by 60.4% against 25-ms per-flow repair, while retaining a 67.8% lower probe rate than the aggressive 8-ms grouped baseline. The latter retains a small pure recovery-speed advantage, revealing a genuine trade-off between the speed and stability axes of fabric convergence rather than an unconditional winner. The source-linked cross-granularity acceptance gate and recorded per-seed outcomes support reproducibility within the documented model. Further evidence on real protocol stacks and measured physical networks is needed before deployment claims can be made.

References

[1] Al-Fares, M., Loukissas, A., & Vahdat, A. (2008). A scalable, commodity data center network architecture. In Proceedings of the ACM SIGCOMM Conference (pp. 63-74). ACM. https://doi.org/10.1145/1402958.1402967

[2] Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., Maltz, D. A., Patel, P., & Sengupta, S. (2009). VL2: A scalable and flexible data center network. In Proceedings of the ACM SIGCOMM Conference (pp. 51-62). ACM. https://doi.org/10.1145/1592568.1592576

[3] Teng, D. (2025). TRIAD: Deadline- and reliability-aware scheduling for post-earthquake structural assessment workflows. World Journal of Engineering Research, 3(6), 55-66.

[4] Gill, P., Jain, N., & Nagappan, N. (2011). Understanding network failures in data centers: Measurement, analysis, and implications. In Proceedings of the ACM SIGCOMM Conference (pp. 350-361). ACM. https://doi.org/10.1145/2018436.2018477

[5] Kandula, S., Sengupta, S., Greenberg, A., Patel, P., & Chaiken, R. (2009). The nature of data center traffic: Measurements & analysis. In Proceedings of the ACM Internet Measurement Conference (IMC) (pp. 202-208). ACM. https://doi.org/10.1145/1644893.1644918

[6] Benson, T., Akella, A., & Maltz, D. A. (2010). Network traffic characteristics of data centers in the wild. In Proceedings of the ACM Internet Measurement Conference (IMC) (pp. 267-280). ACM. https://doi.org/10.1145/1879141.1879175

[7] Katz, D., & Ward, D. (2010). Bidirectional Forwarding Detection (BFD) (RFC 5880). IETF. https://doi.org/10.17487/RFC5880

[8] Shand, M., & Bryant, S. (2010). IP Fast Reroute Framework (RFC 5714). IETF. https://doi.org/10.17487/RFC5714

[9] Alizadeh, M., Greenberg, A., Maltz, D. A., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., & Sridharan, M. (2010). Data center TCP (DCTCP). In Proceedings of the ACM SIGCOMM Conference (pp. 63-74). ACM. https://doi.org/10.1145/1851182.1851192

[10] Li, Y., Chen, R., Liu, H., Zhang, Y., & Wang, J. (2019). HPCC: High precision congestion control. In Proceedings of the ACM SIGCOMM Conference (pp. 44-58). ACM. https://doi.org/10.1145/3341302.3342085

[11] Lapukhov, P., Premji, A., & Mitchell, J. (2016). Use of BGP for routing in large-scale data centers (RFC 7938). IETF. https://doi.org/10.17487/RFC7938

[12] Chen, J., Zhang, X., & Han, X. (2023). Explainable identification of tax-related financial reporting anomalies in U.S. public companies: Cross-fitted tax-consistency residuals and Owen-value Shapley propagation. World Journal of Economics and Business Research, 1(1), 33-45.

[13] Curtis, A. R., Mogul, J. C., Tourrilhes, J., Yalagandula, P., Sharma, P., & Banerjee, S. (2011). DevoFlow: Scaling flow management for high-performance networks. In Proceedings of the ACM SIGCOMM Conference (pp. 254-265). ACM. https://doi.org/10.1145/2018436.2018466

[14] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., & Turner, J. (2008). OpenFlow: Enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 38(2), 69-74. https://doi.org/10.1145/1355734.1355746

[15] Teng, D. (2025). RSRBench: Reproducibility and performance benchmarking for regional seismic risk workflows. Academic Journal of Architecture and Civil Engineering, 3(1), 40-52.

[16] Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. Proceedings of the IEEE, 103(1), 14-76. https://doi.org/10.1109/JPROC.2014.2371999

[17] Alizadeh, M., Edsall, T., Dharmapurikar, S., Vaidyanathan, R., Chu, K., Fingerhut, A., Lam, V. T., Matus, F., Pan, R., Yadav, N., Varghese, G., Pong, T., & Rijsinghani, A. (2014). CONGA: Distributed congestion-aware load balancing for datacenters. In Proceedings of the ACM SIGCOMM Conference (pp. 503-514). ACM. https://doi.org/10.1145/2619239.2626316

[18] He, K., Rozner, E., Agarwal, K., Felter, W., Carter, J. B., & Akella, A. (2015). Presto: Edge-based load balancing for fast datacenter networks. In Proceedings of the ACM SIGCOMM Conference (pp. 465-478). ACM. https://doi.org/10.1145/2785956.2787507

[19] Katta, N. P., Hira, M., Kim, C., Sivaraman, A., & Rexford, J. (2016). HULA: Scalable load balancing using programmable data planes. In Proceedings of the ACM Symposium on SDN Research (SOSR) (Article 10). ACM. https://doi.org/10.1145/2890955.2890968

[20] Al-Fares, M., Radhakrishnan, S., Raghavan, B., Huang, N., & Vahdat, A. (2010). Hedera: Dynamic flow scheduling for data center networks. In Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI) (pp. 281-296). USENIX Association.

[21] Hong, C.-Y., Kandula, S., Mahajan, R., Zhang, M., Gill, V., Nanduri, M., & Wattenhofer, R. (2013). Achieving high utilization with software-driven WAN. In Proceedings of the ACM SIGCOMM Conference (pp. 15-26). ACM. https://doi.org/10.1145/2486001.2486012

[22] Zhou, F. (2026). Moment-constrained side-force allocation for wind-assisted merchant ships: Hydrodynamic performance of retractable anti-leeway fins and flap rudders. Innovation and Technology Studies, 3(1), 27-41.

[23] Guo, C., Lu, G., Li, D., Wu, H., Zhang, X., Shi, Y., Tian, C., Zhang, Y., & Lu, S. (2009). BCube: A high performance, server-centric network architecture for modular data centers. In Proceedings of the ACM SIGCOMM Conference (pp. 63-74). ACM. https://doi.org/10.1145/1592568.1592577

[24] Guo, C., Wu, H., Tan, K., Shi, L., Zhang, Y., & Lu, S. (2008). DCell: A scalable and fault-tolerant network structure for data centers. In Proceedings of the ACM SIGCOMM Conference (pp. 75-86). ACM. https://doi.org/10.1145/1402958.1402968

[25] Wang, B., Wang, Z., Zhao, W., & Liu, Y. (2025). Network fabric simulation and validation for data center routing convergence under large-scale failure scenarios. Computer Science Bulletin, 8(1), 310-326. https://doi.org/10.71465/csb164

[26] Liu, Y., Zhang, H., Wang, L., Chen, X., & Li, J. (2024). Temporal logical attention network for log-based anomaly detection in distributed systems. Sensors, 24, Article 7949. https://doi.org/10.3390/s24247949

[27] Ding, J., & Teng, D. (2026). A risk-aware digital twin framework for release-readiness validation of multi-tenant zero trust SASE systems. WEP Transactions on Computer Science and Intelligent Systems Research, 13, 247-259.