CLOCKWORK.IO

Clockwork Fleet Monitoring: A Layered Architecture for GPU Network Observability

Executive Summary

Large-scale GPU fabrics do not fail like ordinary networks. They fail partially, directionally, and in ways that are amplified by synchronized collective communication. A single hot spine corridor, marginal optic, unstable NIC, congested traffic class, or asymmetric ECMP path can turn into a job-wide straggler while every traditional health signal still looks green: links are up, switches are reachable, utilization is plausible, and no single counter names the problem.

Clockwork Fleet Monitoring is built around the thesis that GPU network observability must start from what the workload actually experiences: directional one-way delay between NICs, measured continuously, at high cadence, with synchronized clocks and hardware timestamps. The vision we are executing against is a layered diagnostic fabric for AI infrastructure. The Clockwork Probemesh is the detection backbone: it tells us, continuously and directionally, which GPU-to-GPU paths are becoming slower or lossy. It turns thousands of edge measurements into a fabric-wide view showing where delay is accumulating. Finally, switch, NIC, optical, congestion, and control-plane telemetry attach the cause. The value is operational clarity. Silent gray failures are found before they become customer-visible incidents. Slow training jobs can be triaged quickly by pinpointing network, host, or workload issues. When the fabric is at fault, the alert names the likely link, switch, port, traffic class, or NIC with corroborating evidence attached.

What used to require multi-team forensics becomes a precise, actionable verdict for the on-call engineer.

© 2026 CLOCKWORK. All rights reserved.

GPU networks need edge latency and switch telemetry

GPU networks fail in partial, directional, and workload-amplified ways

At the scale of a 16,384-GPU backend fabric, with roughly 50,000 active links, 100,000 transceivers, and thousands of ECMP groups, partial failure is the steady state. Classical availability metrics such as link up/down, switch reachability, and average port utilization remain necessary, but are systematically insufficient. The operational difficulty is that most of these failures do not present as clean outages. A leaf may lose a subset of uplinks and remain reachable. A spine may become a repeatable slow path while still forwarding correctly. A transceiver may drift toward its receive-power threshold long before a link-down occurs. A switch may mark ECN or FECN, or generate credit stalls, without dropping a single packet. In a synchronized GPU workload, all of these surface to the operator as the same symptom: "the network is slow".

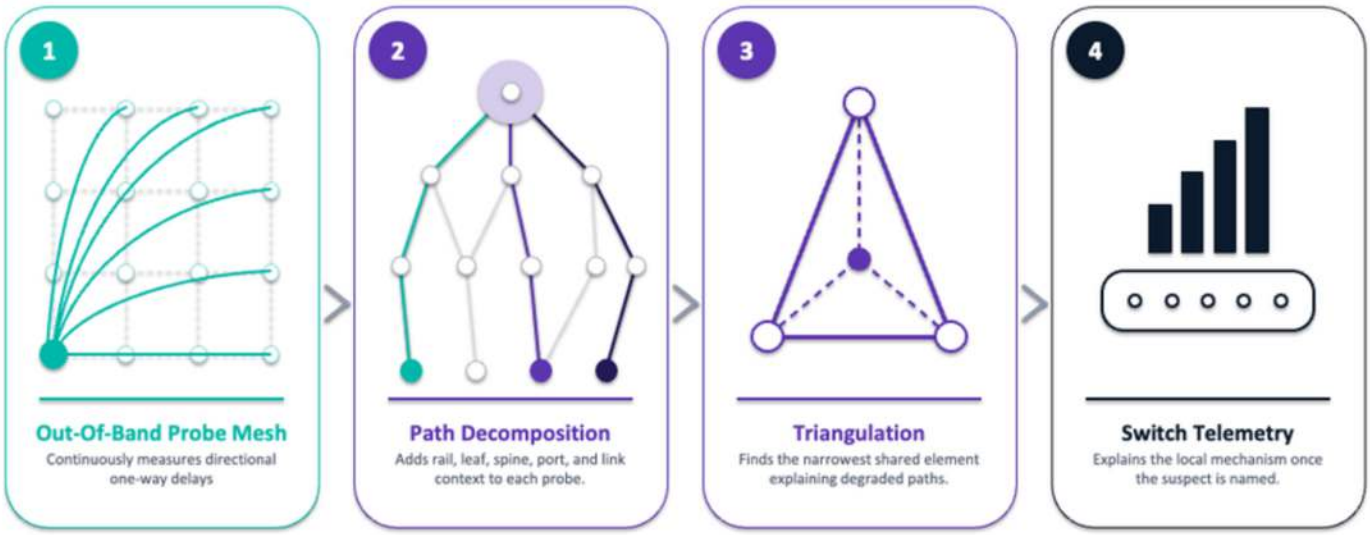
Operators, accordingly, ask for help across a recognizable set of failure shapes:

- **Congestion and micro-congestion:** oversubscribed paths, queue buildup, head-of-line blocking, and congestion-control or QoS interactions reduce effective throughput while links continue to look healthy.
- **Topology-aware degradation**: Only certain rails or node-to-node combinations underperform because of asymmetric mappings, cabling inconsistencies, or other path-specific issues, where the right answer requires reasoning about the topology, not about a single device.
- **Gray failures:** A switch or fabric component remains up but silently delivers degraded performance on only a subset of paths, defeating standard health checks.
- **Workload-visible slowdowns** with unclear root cause, where higher-level schedulers and basic health checks report no fault even as the distributed job runs slowly, and the issue traces back to partial fabric, path, or NIC-related problems that no single component is in a position to surface.

The monitoring system must therefore answer four questions at once. First, is there a latency symptom? Second, is the symptom directional? Third, which tier or path segment explains it? Fourth, which physical, queueing, control-plane, or host-side condition caused it? No single telemetry source can answer all four.

The remainder of this chapter develops the case for Clockwork's thesis of a layered architecture: directional edge OWD as the detection backbone, path decomposition as the localization layer, triangulation to pinpoint the narrowest shared element, and switch telemetry as the attribution layer. It explains why the combination, not the substitution of one for the other, is the only architecture that closes gaps at scale.

© 2026 CLOCKWORK. All rights reserved.



Switches are the best place to collect fabric-local evidence

Switches expose rich, ground-truth telemetry and provide the authoritative, fabric-local evidence required for definitive root-cause analysis. Specifically, switch telemetry excels across several critical domains:

- **Port and link state.** Switches detect hard state changes, such as link down, carrier transitions, speed or width degradation, port-state changes, and topology updates - the ground truth on whether a port is up, at its negotiated speed and width, and continuously connected since its last reset.
- **Drop and error reason codes.** Switches name packet-handling outcomes through structured drop-attribution telemetry such as NVIDIA's What Just Happened (WJH) framework on Spectrum which can tell you a packet was dropped by an ACL on egress port 17 of leaf-3, or by a buffer tail-drop on a particular traffic class with a particular destination prefix.
- **Queue and buffer telemetry.** Per-port and per-traffic-class queue occupancy and shared-buffer watermarks tell the operator whether bytes were sitting in queue at a specific device, and whether a peak watermark within the polling window captured a microburst.
- **Congestion signals**. Switches track pre-drop congestion pressure: per-priority PFC tx_pause and rx_pause counters and ECN-marked packet counts on Spectrum; PortXmitWait, per-VL XmitWait, and per-SL FECN and BECN receive counters on InfiniBand. These signals give operators a leading indicator that the fabric is approaching its operating envelope.
- **Physical layer and optics.** Switches track the physical-layer with comprehensive counters, such as CRC, FCS, alignment, symbol errors, FEC and BER trends. These counters are the ground truth on physical-medium health.

© 2026 CLOCKWORK. All rights reserved.

- **Control-plane health.** Control-plane stress signals, such as switch CPU and memory spikes, daemon restarts, OpenSM sweep activity, UFM events, and routing-engine changes, distinguish a fabric-wide latency event caused by a control-plane stall from one caused by data-plane saturation.

Platforms such as NVIDIA Spectrum and Quantum, Broadcom Tomahawk and Jericho, and the management planes of Arista, Cisco, and Cumulus-based switches may have different collection mechanisms but they all surface fabric-local truth that only they possess; edge probes can only detect the consequence of a misbehaving switch.

Switch counters are not packet-latency measurements

A switch can tell that a queue was non-empty, that a watermark was reached, that a port was credit-stalled, or that a drop occurred. It cannot, by counter alone, tell the one-way delay of a particular probe packet from source NIC to destination NIC across the whole path. Five structural blind spots follow from this.

- **No end-to-end latency**. Switch counters do not measure the latency a packet experienced across the path. A packet that traverses three switches each individually within its normal operating envelope can still have anomalous cumulative latency, and no single switch counter will surface it. End-to-end latency is the metric the workload actually feels, and it has no switch-native equivalent.
- **Counter attribution ambiguity**. Switch counters are aggregated by port, queue, priority, traffic class, virtual lane, or drop category. Counter deltas can tell us that a local resource experienced pressure but cannot identify the specific packet, flow, Queue Pair (QP), GPU pair, or training job that experienced the impact. A queue watermark does not preserve the identity of the packets behind that queue.
- **Temporal ambiguity.** Many switch and control-plane signals are reported at polling or event cadence, often once in many seconds or tens of seconds, while GPU tail-latency events often happen at microsecond-to-millisecond timescales.
- **No fabric-wide path localization.** A switch sees only its own local forwarding behavior, and not the complete end-to-end path taken by a probe or workload flow, and it cannot determine that a set of slow GPU pairs all share the same spine, leaf uplink, egress queue, traffic class, or ECMP member.
- **No workload correlation**. Switch counters describe what the fabric did, not what the workload felt. Two flows with identical 5-tuples can experience very different application-level performance, and switch counters cannot distinguish a network problem from a workload-side problem.

© 2026 CLOCKWORK. All rights reserved.

- **Silent and gray failures** are invisible to the switch. TCAM corruption can install a black-hole rule, a fabric-module bit flip can produce random per-flow drops, and an ASIC flaw can mis-forward specific 5-tuples - all without switch counters incrementing or indicating any health issues.

These gaps are why edge-based latency monitoring is necessary.

Why edge-based one-way delay monitoring is necessary

The gaps in switch-only telemetry are the gaps that an always-on edge-based Probemesh closes. By instrumenting every NIC in the fleet and having NICs probe each other, the operator obtains end-to-end measurements that are workload-independent, directionally attributable when probes are timestamped against synchronized clocks, and capable of surfacing silent failures that no switch will ever report on its own.

The empirical record is the strongest case for this approach. Microsoft's Pingmesh, deployed across Microsoft's data centers in the early 2010s, demonstrated that a fleet of TCP/HTTP probes between every server pair was operationally feasible and that the resulting always-on data was the deciding factor in roughly half of incident triage cases. R-Pingmesh, deployed at ByteDance starting in 2023 and reported at SIGCOMM 2024, demonstrated the same point on RoCE fabrics: the system detected 207 problems with 85% overall accuracy, including 100% on the 157 switch-network problems specifically. Among the categories it caught were silent fiber-corruption-induced drops, PFC deadlock events on which the switches' own deadlock watchdogs had failed, and ECMP hash-collision-driven uplink congestion that no individual switch flagged because no individual switch was over its envelope.
These are precisely the failure classes that switch counters cannot surface alone, and they are the classes that an edge-based mesh catches by construction.

Two qualifiers shape what kind of edge-based probing is needed for GPU fabrics specifically.

- Latency, not just connectivity, is the primary signal. Connectivity probing through pings between endpoints to confirm reachability, is a much weaker form of edge measurement and misses the gray-failure regime entirely. A switch port that is 100% reachable but adding 50 microseconds of queueing delay to every packet that crosses it is operationally a network problem; a connectivity-only mesh will report it as healthy. The signal that exposes gray failures, micro-congestion, hot spines, and ECMP imbalance is latency, and the probing system must measure it natively at probe granularity rather than infer it from loss or reachability.

© 2026 CLOCKWORK. All rights reserved.

- Directional attribution is operationally essential. Several of the most consequential GPU-network problems are inherently directional. A receiver-side incast raises inbound delay to a specific NIC and its leaf-facing port without affecting outbound delay from that same NIC; a hot spine appears as a consistent set of elevated paths crossing the same spine in one direction, not as a generic latency increase across the fabric; head-of-line blocking distinguishes itself as same-egress HOL versus cross-port HOL through the directional pattern of correlated spikes across flows that share a queue or an upstream pause domain. Resolving these requires separate forward and reverse latency, combined with topology and switch-side evidence.

These two qualifiers together, latency as the primary signal, measured rectionally, define what an edge-based monitoring layer must produce to be useful for GPU fabrics.

Edge and switch telemetry are complementary

The two preceding sections establish that edge-based one-way-delay probing is necessary because switch counters cannot produce end-to-end, directional, or workload-attributable measurements, and that switch telemetry is necessary because edge probing cannot produce physical-layer attribution, queue-level reason codes, optical leading-indicator data, or microsecond-resolution queue dynamics.

The architectural conclusion follows directly: the two modalities are complementary, not redundant, and the gaps in each are precisely the strengths of the other. The pattern of complementarity is visible in direct comparison in the next table.

© 2026 CLOCKWORK. All rights reserved.

Edge and switch telemetry are complementary

Capability	Edge-based probing	Switch telemetry
End-to-end latency	Directly measured between any two NICs at probe granularity.	Not measurable. Switches see only their own queue residency and credit-stall time.
Directional attribution	Available with synchronized clocks. Each direction is measured independently.	Per-port and per-direction counters exist, but not per-flow-direction across the path.
Silent / black-hole drops	Detected by sent-minus-acked divergence. The defining use case.	Not detected. Counters report known drops; but silent drops are silent.
Physical-layer attribution	Cannot distinguish a marginal optic from congestion once both produce loss.	Authoritative. The only place where physical-layer events are observed.
Optical / DOM monitoring	Cannot observe.	The only vantage point. Predicts optic failure hours to days in advance.
Drop reason codes	Cannot observe. Only sees that a probe was lost.	Authoritative when WJH or equivalent is enabled.
PFC / credit-stall attribution	Inferred from latency elevation patterns. Imprecise.	Authoritative. PFC tx_pause / rx_pause and PortXmitWait localize backpressure.
ECN / FECN congestion marking	Probes can carry ECT and report marks, but marking is determined by the switch.	Authoritative. Per-port marked-packet counters identify the exact marking point.
Microburst detection	At 10 ms probe cadence, microbursts shorter than 10 ms are missed.	Buffer watermark counters can capture microbursts directly.
Workload correlation	Strong: probes share the fabric with workloads and feel the same conditions.	Weak: counters describe fabric state, not workload-perceived performance.
Fabric-interior path localization	Possible with hierarchical probing and path decomposition	Single-switch only. No native correlation across switches.

© 2026 CLOCKWORK. All rights reserved.

The pattern is consistent across the table: edge-based probing dominates on detection, end-to-end attribution, and silent-failure surface area; switch telemetry dominates on root-cause attribution, physical-layer ground truth, and microsecond-resolution queue dynamics. Neither modality alone is viable.

Layer	Primary signal	What it answers
Edge Probemesh	Directional and tier-specific one-way delay, probe loss, ECN/FECN marks.	Did a NIC-to-NIC path get slower, by how much, in which direction, and at which fabric tier?
Path decomposition	Spectrum PathFinder observations, on-demand hop-limit traces, IB LFT path walks, topology/version windows.	Which switches, ports, traffic classes, virtual lanes, and tiers are shared by the bad edges?
Switch/fabric telemetry	Drops, queues, credit stalls, PFC/ECN, DOM/BER/FEC, CoPP, UFM events.	Why did the shared path become slow or lossy?
Workload context	Job placement, NCCL communicator and QP path, throughput, application probes.	Which jobs are affected, and did the network explain the workload symptom?

Clockwork Probemesh

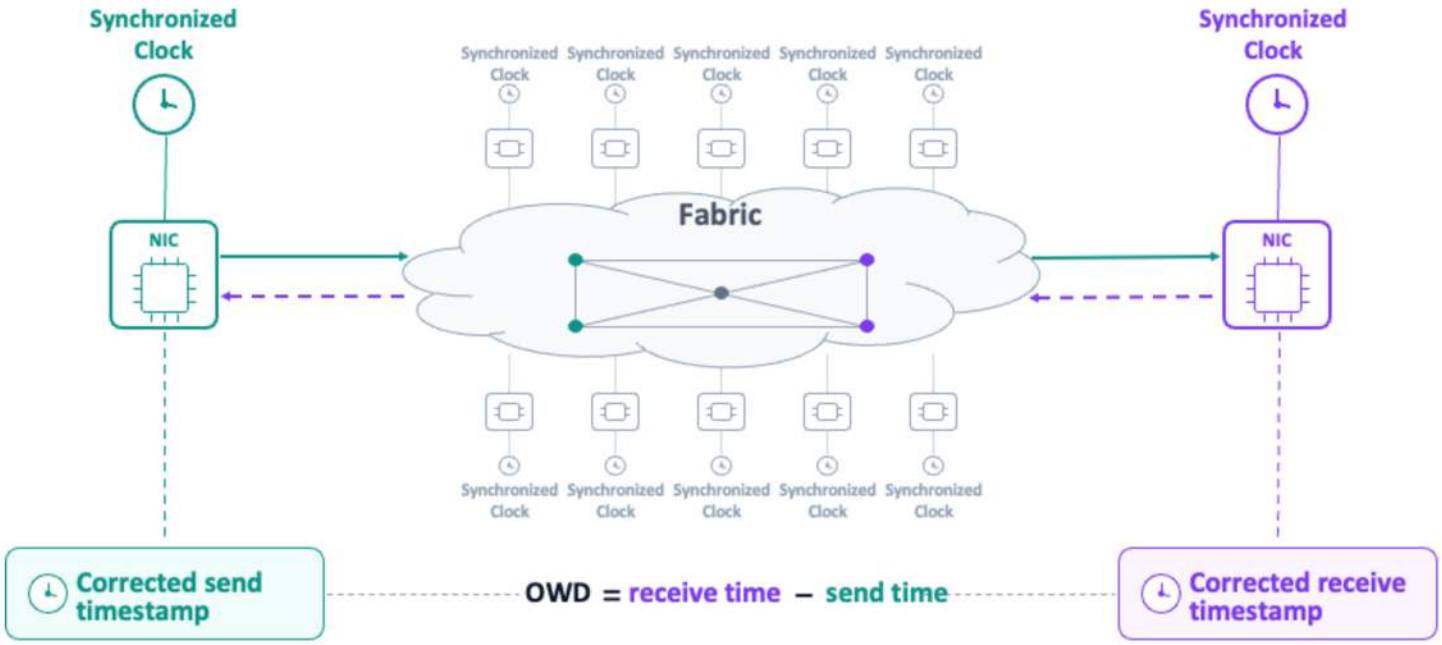
Clockwork's Fleet Monitoring detection backbone is a fleet-wide out-of-band Probemesh that measures one-way delay (OWD) between NICs in the cluster at a default cadence of ten milliseconds. The mesh is workload-independent: probes are generated by Clockwork agents on every node, exchanged across the same physical paths used by production traffic, and consumed by Clockwork agents on the receiving side, with no dependence on whether a training job is active.

The probes use synchronized clocks established by Clockwork's ClockSync subsystem, which maintains sub-microsecond global time accuracy across the fleet and is what enables true directional latency measurement rather than an RTT/2 approximation.

T**he entire goal of the Probemesh, along with full path decomposition, is to enable rapid detection and triangulation down to individual switches, or even specific switch ports and links, so that switch telemetry can then be leveraged in a targeted manner to narrow down the specific root-cause.**

Synchronized clocks: The foundation of Clockwork's Probemesh

Clockwork's measurement model rests on a single architectural commitment: every NIC in the fleet has a clock that can be related to every other NIC's clock with sub-microsecond precision in a common time reference. Without that commitment, one-way delay is not a measurable quantity; the only thing two unsynchronized hosts can compute ogether is round-trip time, which collapses forward and reverse path delays into a single number and hides every directional fault the GPU fabric is most prone to. With it, an edge measurement becomes a property of the network rather than an artifact of the host that sourced it: the receive timestamp at one end, minus the send timestamp at the other end, both already in the same global frame. Forward and reverse OWDs become two independent numbers, asymmetric problems become first-class observations, and probe paths become quantitatively comparable across tiers and across the fleet. Every capability described later in this document: hierarchical tier localization, always-on path decomposition, and link-level OWD tomography, inherits its meaning from this property of the underlying measurement.



Clockwork ClockSync - synchronized clocks with sub-microsecond accuracy

Architecturally, ClockSync is built entirely in software as a coordinator-agent system. Agents run on every node, send and receive synchronization probes, capture timestamps, estimate pairwise clock relationships, and apply clock corrections where enabled. The coordinator orchestrates probe sessions, tracks agent and edge health, reasons over the synchronization graph, computes global offsets, and distributes corrections and time bounds. Probe timestamps are hardware-captured where the platform supports it, so the measurement is anchored as close to the NIC as possible rather than being polluted by userspace scheduling, kernel delay, interrupt timing, or host CPU contention. Hardware timestamping is a key mechanism that makes ClockSync's global time frame useful for network measurement.

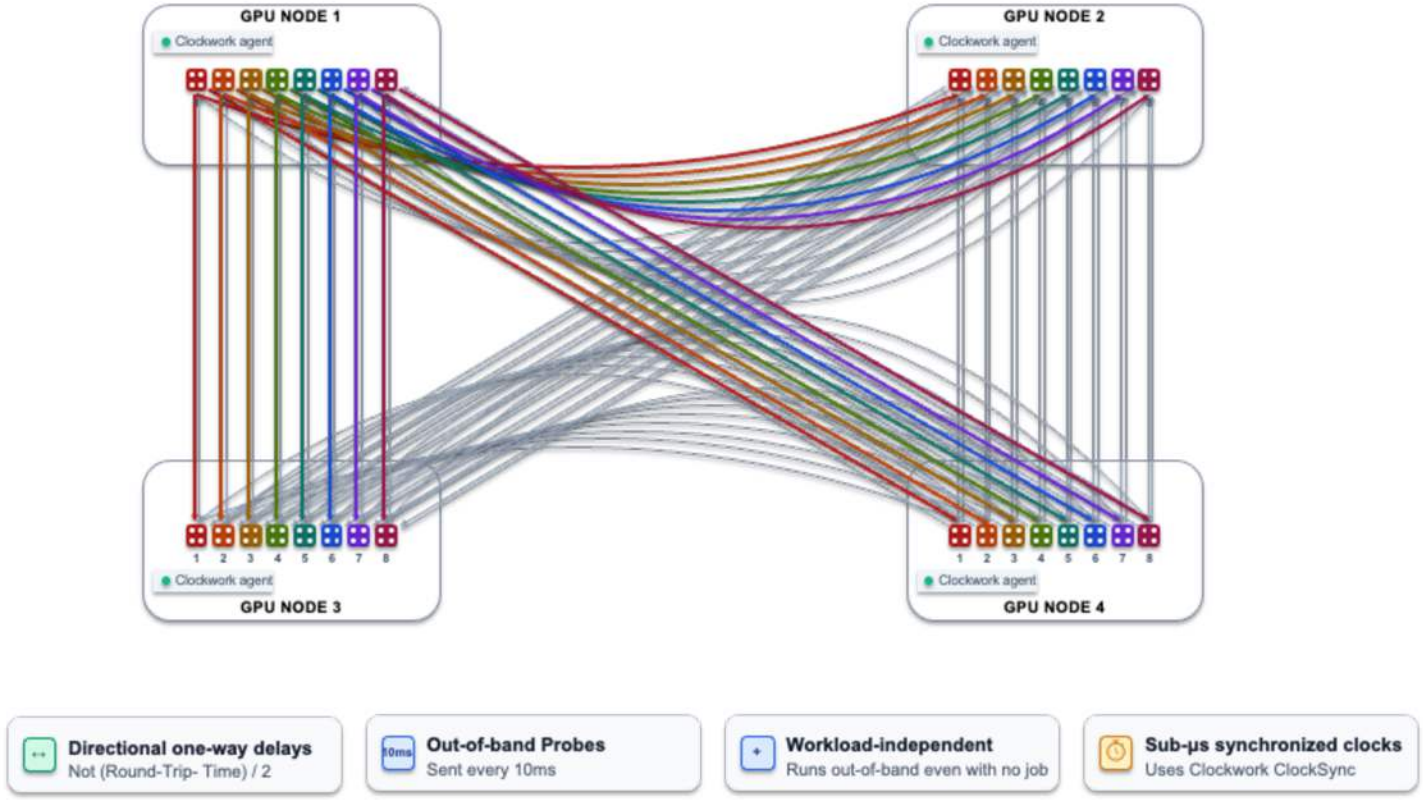
ClockSync turns noisy network timing samples into a fleet-wide time reference with explicit error bounds. The underlying methodology, including data filtering, statistical estimation, and graph-level reasoning, is described in the NSDI 2018 Paper.

The result is not merely a best-effort estimate of time; it is a fleet-wide timing substrate with explicit correction state and error bounds. Probe samples from different tiers, racks, pods, and fabrics become comparable because they share a common time reference. ClockSync is what makes Clockwork's monitoring architecture possible.

The Probemesh is not powerful merely because it sends probes frequently, nor because those probes are lightweight, nor because they traverse many paths. It is powerful because every probe is measured in a shared, bounded, sub-microsecond time frame.

The out-of-band Probemesh

The out-of-band Probemesh is the continuous baseline for fabric health. It sends lightweight synthetic probes between NICs independent of any training workload, and reports directional OWD, latency quantiles, sent and acknowledged counts, probe loss, ECN or FECN-style congestion marks where available, and NIC and host timing breakdowns where available. Because the mesh is workload-independent, it produces a baseline whether or not any training job is running, which means it can detect problems before workloads notice them and continue to detect them when workloads have stalled.



The out-of-band Probemesh measures directional one-way-delays between NIC pairs every 10ms

Probe transport. On RDMA fabrics, probes use Unreliable Datagram (UD) queue pairs established by the Clockwork agent on each NIC. UD is the right transport for three reasons. First, UD supports one-to-many probing without requiring a dedicated connected QP per peer, which keeps NIC QP-context-cache pressure low even when each NIC probes hundreds of peers.

Second, UD has no transport-level retransmission, so a lost probe is genuinely lost and visible to the monitoring system rather than silently retransmitted by hardware. Third, UD probes can be made AR-eligibility-controlled at the NIC, which is required for the deterministic-path design described in the next section. On Ethernet and front-end fabrics, UDP probes provide a comparable workload-independent signal between Ethernet interfaces and share their packets with the ClockSync infrastructure: a single packet serves both as a clock-synchronization sample and as a network-latency sample, which materially reduces probe traffic volume on the front-end network.

Probe structure. Probe packets are small datagrams (on the order of 44 bytes on the backend mesh) carrying an edge identifier, a destination GID and port number, and a serial number for matching requests with acknowledgments. The compact size keeps per-probe bandwidth cost negligible. Typical per-host overhead is on the order of tens of kilobits per second, so that the mesh can operate continuously at high cadence without measurable impact on production traffic.

Metrics produced. Each NIC pair produces, over each one-second reporting window, forward and reverse OWD (min, p50, p90, p99, max), probes-sent and probes-acked counts, ECN-marked-probe counts, kernel delay, and NIC delay. Aggregated across the mesh, these primitives roll up into the metric families that downstream consumers (alerting, dashboards, RCA automation) actually use as captured in the next table:

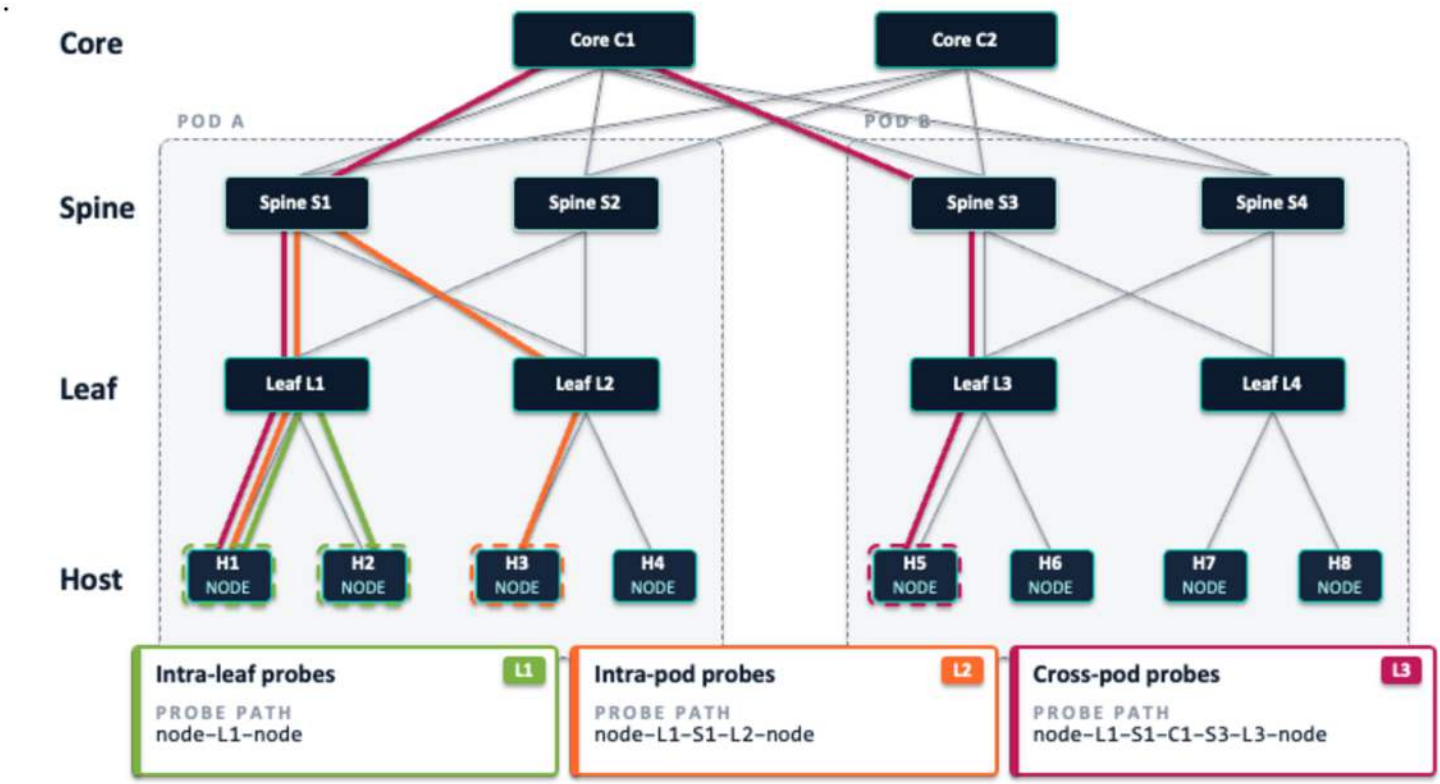
Metric family	Meaning	Why it matters
Forward/reverse OWD	Directional latency from source NIC to destination NIC and back	Detects asymmetric congestion, incast, egress queueing, and route-specific delay.
Probe loss	Sent minus acknowledged probes per edge or NIC.	Detects drops, black-holes, flaps, hard failures, and severe congestion.
Latency distribution	Min/mean/p50/p90/p99/max over the reporting window.	Separates baseline path length from tail queueing and microburst symptoms.
Congestion marks	ECN on Ethernet and RoCE; FECN-style indicators in IB where exposed.	Provides pre-drop congestion evidence on the paths being probed.
Tier-specific OWD	OWD measured with probes constrained to L1, L2, or L3 fabric tiers.	Localizes the symptom to access, spine, or deeper fabric tiers before full path decomposition.
Clock quality	Offset, error bounds, and clock convergence status.	Prevents latency false positives caused by clock divergence.

The mesh, in other words, is intentionally more than a connectivity test. It is a controlled, continuously running, directionally attributed measurement of the fabric — produced by the same primitives that the audit, the runtime monitor, and the RCA pipeline all consume.

Plane-aware probing. Modern large-scale GPU fabrics are increasingly built as multi-plane topologies, where each NIC connects to multiple independent fabric planes. The Probemesh targets every plane of a multi-plane fabric, with per-plane OWD baselines reported alongside per-tier baselines so that plane-specific degradations surface as their own signal.

Hierarchical probing enables fault localization to a switch tier

Hierarchical probing is the mechanism by which the operator controls and tracks which tier of the fabric a given probe traverses. In a three-tier fat-tree (node–leaf–spine–core-spine-leaf–node), a probe between two NICs may traverse only the L1 access tier (when both endpoints are under the same leaf), the L2 leaf–spine–leaf tier (when endpoints are on different leaves but the same spine pod), or the full L3 inter-pod tier. The same probe-path measurement gives the operator different information at each level: an L1 probe isolates leaf-local issues, an L2 probe isolates spine and inter-leaf-uplink issues, and an L3 probe isolates higher-tier transit. The point is not to replace full path tracing which comes later. The point is to make the first alert specific enough that the investigation starts in the right place.



Hierarchical probes tracks which tier of the fabric each probe traverses

The Clockwork agent's coordinator assigns probe edges according to a hierarchical schedule that ensures coverage at each tier. Endpoints under the same leaf are assigned L1 edges to each other; endpoints across leaves but within the same spine domain are assigned L2 edges; endpoints across spine domains are assigned L3 edges. The assignment is computed centrally so that every link at every tier is exercised by some set of probe edges, and the fanout per NIC is bounded so that probe traffic remains a small fraction of NIC bandwidth even at fleet scale.

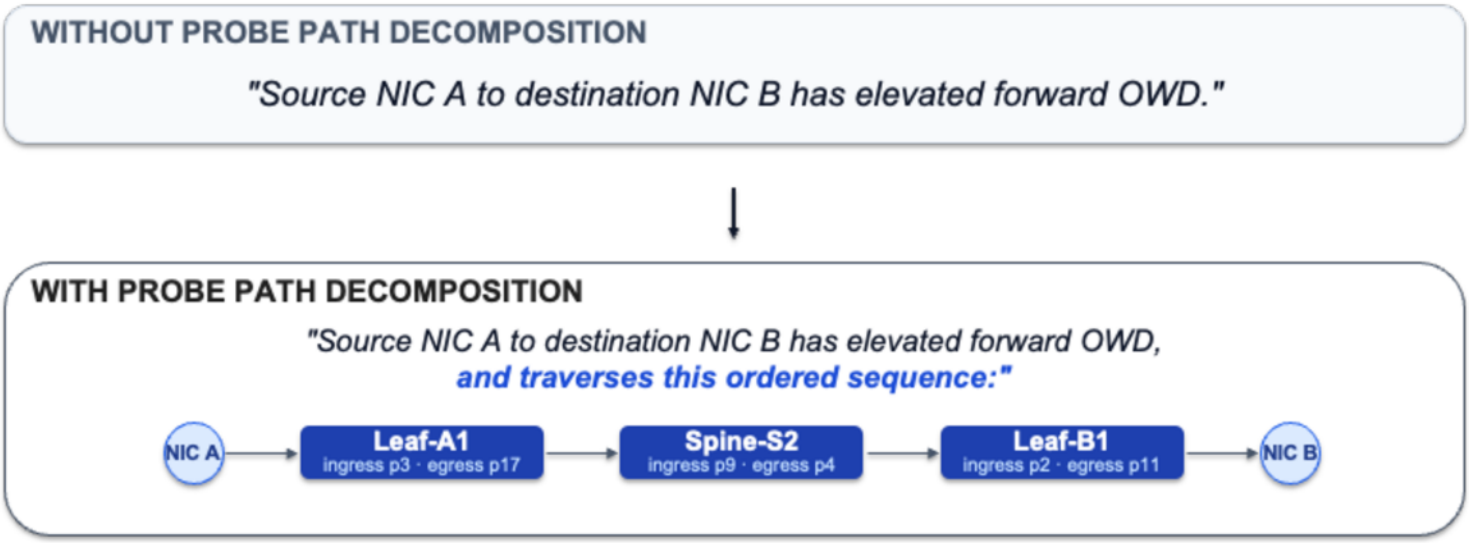
The detection consequence of hierarchical probing is that an OWD elevation observed on a particular tier's probe set localizes the fault to that tier without requiring further investigation.

This is qualitatively different from a flat all-to-all mesh, in which the operator must perform post-hoc correlation across many endpoint pairs to infer which tier the fault is in. With hierarchical probing, the tier is part of the alert, not a downstream inference and the first alert is therefore actionable in a way that a flat-mesh alert is not.

Path decomposition and triangulation to specific switches and links

Probe path decomposition

Clockwork's synchronized Probemesh tells us which NIC-to-NIC paths are slow, in which direction, and by how much. Hierarchical probing takes the first localization step by constraining probes to L1, L2, or L3 fabric tiers, so an alert can immediately distinguish access-tier, leaf-spine, and inter-pod symptoms. But tier localization is not the end goal. The operational goal is to identify the specific switch, port, link, queue, traffic class, virtual lane, or endpoint that explains the elevated one-way delay. That requires a second layer: probe path decomposition.



Probe path decomposition is the process of attaching every relevant OWD sample to the path the probe traversed. A decomposed probe is no longer just “source NIC A to destination NIC B had elevated forward OWD”. It becomes “source NIC A to destination NIC B had elevated forward OWD, and that sample traversed this ordered sequence of switches, ingress ports, egress ports, links, traffic class or service level, and path-map version”. Once this binding exists, latency stops being an edge-only signal and becomes a fabric-structured observation. The system can then ask which physical elements are common to the degraded probes and not present in the healthy probes.

Clockwork supports two operating modes for path decomposition. On-demand path decomposition runs only during a troubleshooting epoch, typically after a sustained OWD alert. Always-on path decomposition runs continuously, attaching path metadata to every probe sample whether or not it is currently degraded. The two modes are not competing concepts; they are two ways to populate the same evidence stream. The difference is coverage, freshness, and analytical power.

On-demand path decomposition

On-demand path decomposition is the alert-triggered mode. It is used in two situations:

1. **Automatic fallback:** The always-on substrate (described below) has a validity gap for a particular probe sample (a switch agent has not yet started reporting on a recently deployed switch, or the IB forwarding-state bundle has been invalidated by a fabric event and the next SUBNET UP marker has not yet arrived) and the system needs path information for that sample's edge despite the gap.
2. **Operator-initiated:** an operator wants an active trace during a troubleshooting window even when always-on coverage is nominally available. In either case the trigger is a sustained Clockwork alert, where one edge or a group of edges exceeds an OWD threshold for a configured duration and decomposition runs only for the duration of the troubleshooting epoch.

On Spectrum and RoCE fabrics, Clockwork actively traces the alerting probe's path, ensuring that the path decomposition is accurate even when Adaptive Routing or Dynamic Load Balancing is active for data traffic. The trace is constructed so that it follows the same forwarding decisions the alerting probe took, and it identifies the responding switches along the way. Where supported by the switch platform, the trace also names the front-panel ingress port on each hop.

On InfiniBand, where there is no equivalent of hop-limit traceroute, the path is resolved through the subnet manager's installed forwarding state. With probe traffic configured to bypass adaptive routing, the resolved path matches the path the probe traversed.

Always-on path decomposition

Always-on path decomposition generalizes the same idea to every probe in the mesh. Instead of waiting for an alert and tracing only the degraded edges, Clockwork continuously attaches path metadata to all probe samples. The analytics layer therefore receives a rolling stream of tuples: probe edge, direction, timestamp, OWD, path, and path-validity state. This changes the localization problem from episodic troubleshooting to continuous fabric inference.

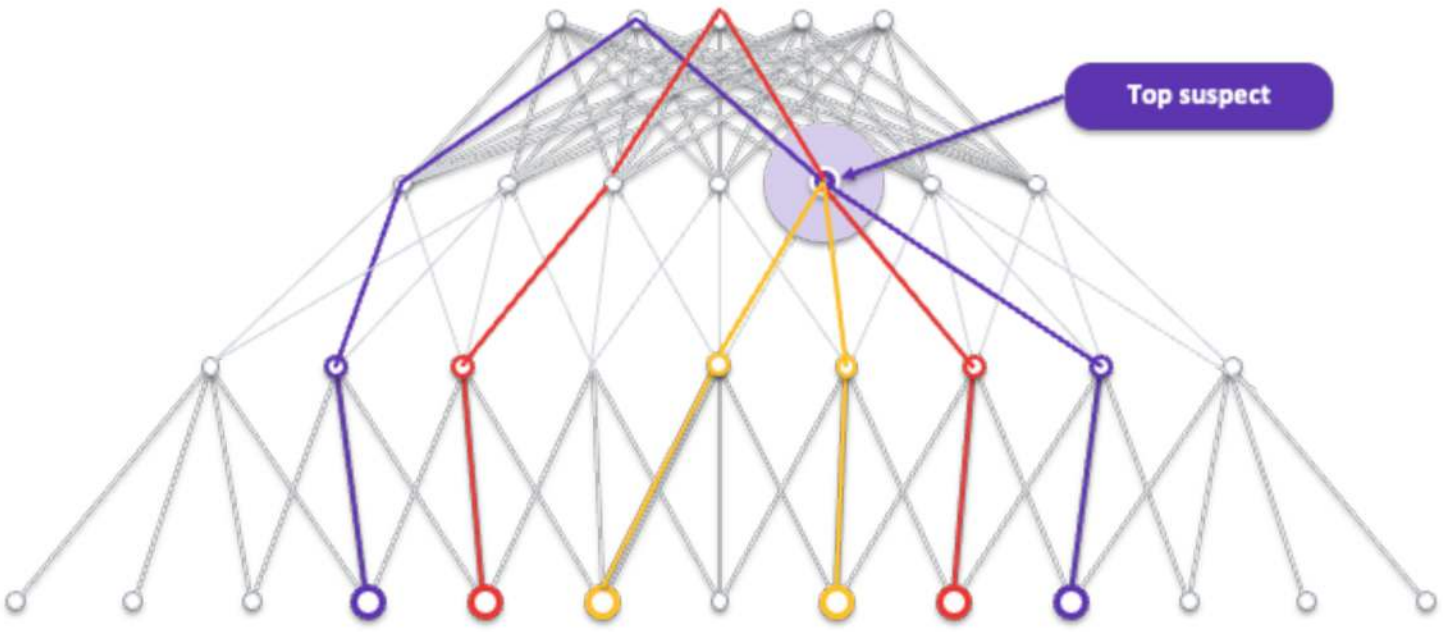
On Ethernet, a userspace switch agent runs on each switch and observes mesh probes passively. For every Clockwork probe that passes through the switch, the agent records which port it entered on and when. The switch reports these per-probe sightings to the central analytics layer, along with per-port queue and link telemetry. The analytics layer then assembles the sightings from every switch in the fabric into the complete path each probe traversed.The agent's footprint is fixed and small, benchmarked at 1.5 - 2.0% the switch CPU and 30 MB RAM on Spectrum-3 / Cumulus.

InfiniBand uses no on-switch agent, as the SM (Subnet Manager) owns the control plane and IB switches do not host third-party daemons. Instead, Clockwork subscribes to the subnet manager's forwarding state and updates its cached view of the fabric as routing or topology events occur. Probe paths are resolved against this cached forwarding state, and corroborating per-port counters stream from UFM telemetry.

In both fabrics, the always-on substrate has well-defined failure modes that determine when the system falls back to on-demand. Coverage gaps can occur when a switch agent is restarting, when a recently deployed switch has not yet begun reporting, or when InfiniBand forwarding state is transitioning between routing events. In any of these cases, the affected probe's path is stamped "unknown," and on-demand decomposition is invoked if the sample belongs to an active alert. The two modes therefore compose: always-on is the steady state, on-demand is the targeted fallback for substrate gaps.

Triangulation: from decomposed paths to suspect links

Once paths are available, either on-demand or always-on, Clockwork can triangulate. Triangulation operates at three granularities in parallel: switch, port, and switch-plus-port-pair; and reports the narrowest stable element that explains the degraded set. Each granularity carries a distinct interpretation. If all degraded probes share a switch but not a specific port, the likely cause is switch-wide: control-plane stress, ASIC behavior, or shared resource pressure. If they share an egress port, the likely cause is port- or queue-specific: congestion, PFC or credit stalls, ECN/FECN marking, buffer pressure, or misconfiguration. If they share a bidirectional port pair, the likely cause is link-local: cable, transceiver, lane quality, FEC/BER degradation, or link instability. The granularity at which the verdict is sharpest is the granularity at which the verdict is reported, which means the same triangulation step can localize to a switch, a port, or a link depending on what the path data actually supports.



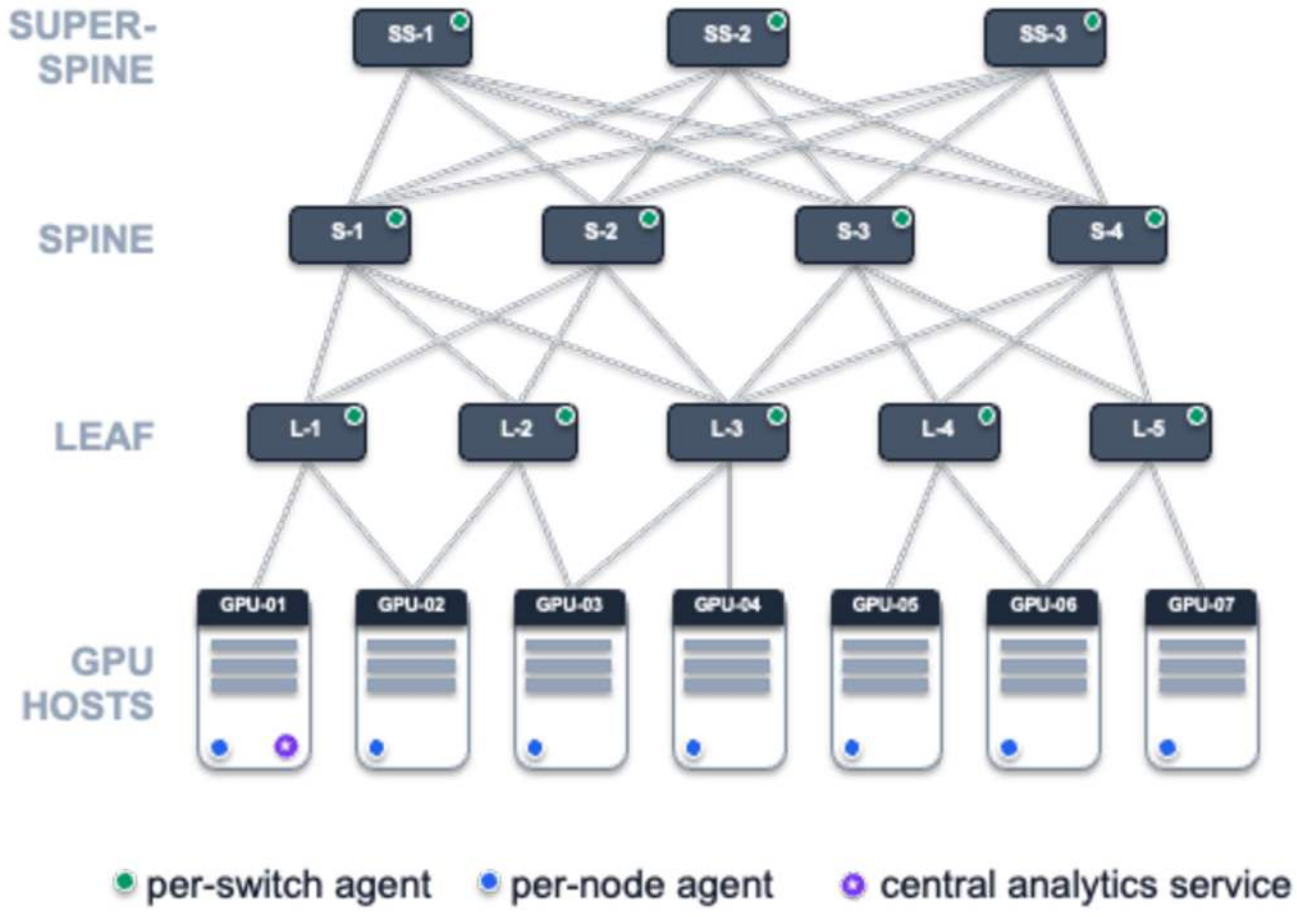
Triangulation pinpoints down to the switch, port or switch-plus-port-pair.

In on-demand mode, triangulation is primarily structural. It overlays the decomposed paths of the alerting edges, computes common elements at the three granularities, and reports the narrowest element that explains the degraded set.

With always-on decomposition, triangulation becomes substantially stronger. Every probe in the mesh contributes both a path and an OWD measurement. - Slow probes point at suspect paths; healthy probes rule out the links they cross. Across a rolling window, Clockwork combines both into a continuously updated map of per-link latency, calling out each bottleneck link and the delay it's adding.This is the key advantage of always-on decomposition: healthy probes become first-class evidence. On-demand decomposition asks, primarily, “what do the degraded paths have in common?” Always-on decomposition asks a stronger question: “which link-delay explanation best fits both the degraded probes and the healthy probes across the whole mesh?”

Deployment and operational integration Of Clockwork Probemesh

Clockwork Fleet Monitoring deploys as three components — a per-node agent, a per-switch agent (on Ethernet and RoCE fabrics), and a centralized analytics service. They communicate over standard channels and impose modest resource and access requirements.



Note: Central analytics service can be deployed anywhere and does not have to be part of the active fabric.

Per-node agent. Runs as a userspace process on each GPU node and operates two distinct probe streams: the Clockwork Probemesh, which traverses the backend RDMA NICs out-of-band to measure NIC-to-NIC OWD; and ClockSync, which maintains the fleet-wide synchronized time reference. Both use hardware NIC timestamps where available. The agent reports per-edge metrics to the central analytics service. CPU usage is a fraction of a single core; memory is bounded.

Per-switch agent (Ethernet / RoCE only). Runs as a userspace process on each leaf and spine. As Clockwork probes pass through the switch, the agent records which port each probe entered on and when, and reports those observations along with per-port queue and link telemetry. The central analytics service uses these observations to stitch together the complete path each probe traversed. Footprint is single-digit-percent CPU and tens of megabytes of memory on representative platforms.

Central analytics service. Aggregates probe data, switch-agent reports, and (on InfiniBand) UFM telemetry. Maintains the synchronized clock graph, computes per-tier baselines, performs path decomposition, and emits operator-facing alerts and dashboards.

Detections enabled by the stack

The following scenarios describe a set of high-value detections enabled by the Clockwork Fleet Monitoring stack and delivered as operator-oriented narratives. Each scenario follows the same causal chain: the condition that matters to a GPU workload and the Clockwork signal that detects and localizes it. Once the issue is localized to a specific switch or link, the final root cause analysis needs to be based on switch telemetry and physical integrity.
The conditions are categorized as:

●	**Congestion and micro-congestion**: oversubscribed paths, queue buildup, head-of-line blocking, and congestion-control or QoS interactions reduce effective throughput while links continue to look healthy.
●	**Topology-aware degradation**: Only certain rails or node-to-node combinations underperform because of asymmetric mappings, cabling inconsistencies or other path-specific issues where the right answer requires reasoning about the topology, not about a single device.
●	**Gray failures**: A switch or fabric component remains up but silently delivers degraded performance on only a subset of paths, defeating standard health checks.
●	**Workload-visible slowdowns** with unclear root cause, where higher-level schedulers and basic health checks report no fault even as the distributed job runs slowly, and the issue traces back to partial fabric, path, or NIC-related problems that no single component is in a position to surface.
●	**Hard failure** detectable with existing telemetry, but requires a method to rapidly determine the blast radius.

Condition	Detection logic with probe OWDs and path decomposition
Hot spine switch or spine corridor: A spine switch, or a small set of spine-facing links, becomes a repeatable slow path for cross-leaf traffic. The fabric remains connected, so the customer sees stragglers, slower collectives, or an unexplained latency floor rather than a clean outage.	• L2 or L3 probe edges show sustained directional OWD elevation over the rolling confirmation window. • Path decomposition shows degraded probes share the same spine switch, spine-facing port, or leaf–spine–leaf corridor. • Comparable probes that avoid that spine or corridor remain at baseline. • If many unrelated corridors through the same spine are degraded, classify as suspected spine-wide degradation. • If only one narrow corridor is degraded, classify as a suspected corridor, port, link, traffic-class, SL, or VL issue rather than a spine-wide issue.
Leaf-to-spine uplink capacity reduction: A leaf loses one or more spine-facing uplinks, or an uplink remains up but operates at reduced effective capacity. The leaf itself stays in service, but traffic from nodes behind that leaf now shares fewer healthy paths and becomes more sensitive to bursts.	• Probe edges sourced from, or destined to, nodes behind the same leaf show sustained OWD elevation across multiple remote leaves. • Path decomposition shows affected probes share the same leaf's spine-facing fanout rather than one destination leaf or one spine. • Always-on path decomposition shows traffic concentrating on fewer surviving uplinks, or link-level OWD rising on the remaining leaf-to-spine paths. • Other leaves using the same spines remain at baseline, which argues against a spine-wide issue.
Silent bad NIC or degraded access link: A NIC, optic, cable, or host access path remains technically up but becomes slow, lossy, or unstable. This is the classic gray failure: the component passes basic health checks while every workload touching it sees degraded performance.	• All or most probe edges touching the same NIC, GPU-facing port, or access link show elevated OWD, probe loss, or high jitter; adjacent NICs or access links on the same leaf are normal. • Forward versus reverse OWD separates inbound-to-NIC degradation from outbound-from-NIC degradation. • Path decomposition localizes the symptom to the first hop or last hop rather than the spine fabric. • If all fabric-interior links on those paths remain at baseline, classify the suspect as endpoint, access link, NIC, or host-adjacent rather than core fabric.
Cross-leaf pair or corridor-specific degradation: Only certain node pairs, rails, QPs or placement combinations slow down because they traverse a particular source-leaf to destination-leaf corridor. This can happen due to queueing/congestion on the specific path, physical layer issues or AR/hash skews	• Only specific source-leaf to destination-leaf pairs, rails, QPs, or placement combinations show elevated OWD. • Degraded probes share a narrow decomposed path segment, such as source leaf uplink → spine → destination leaf downlink. • The same source leaf to other destination leaves remains normal; Other source leaves to the same destination leaf remain normal. • Other leaf pairs through the same spine remain normal, which down-ranks a spine-wide cause and points to a corridor-specific condition.
Persistent link flaps or unstable recovery: A link flaps, returns at a degraded speed or width, or repeatedly re-enters recovery. The operational danger is not only the outage interval; it is the residual instability after the link appears to have recovered.	• Probe loss, missing acknowledgments, missing path observations, or path-map invalidations align with a topology-change window. • After the link returns, Clockwork measures time to first probe recovery and time to clean OWD recovery. • Classify as transient if OWD, loss, and jitter return to baseline within the recovery window. • Classify as persistent if OWD, loss, or jitter remains elevated after the topology appears recovered. • Classify as pattern-of-flapping if the same path element repeatedly enters and exits the decomposed path set across observation windows

Condition	Detection logic with probe OWDs and path decomposition
● **Hard cable, NIC, leaf, or spine failure**: A component fails cleanly. The important monitoring task is not merely to detect that something disappeared, but to identify the blast radius quickly: one cable, one NIC, one leaf, one spine, or a larger control-plane event	• Probe acknowledgments disappear abruptly rather than merely showing elevated OWD. • Path decomposition, span observations, or IB path resolution show missing links, missing endpoints, rerouted paths, or path-unknown samples after invalidation. • Group affected probes by blast radius: one access path implies cable or access-link failure; all probes touching one NIC imply NIC or endpoint failure; many probes behind one leaf imply leaf or leaf-facing failure; many paths through one spine imply spine failure.
● **Inter-node network bottleneck versus host-side bottleneck**: A distributed job slows down, but the initial symptom does not reveal whether the cause is the fabric, the host, GPUDirect, PCIe, NUMA placement, QP/CQ behavior, or the application runtime.	• For the affected job or placement, identify the NIC pairs and fabric tiers the job exercises. • Compare job-time OWD against the rolling baseline for those same edges, tiers, or decomposed paths. • If all traversed fabric paths remain at baseline during the slowdown window, classify the monitored fabric as unlikely to be the cause. • If any traversed tier, path segment, or link-level OWD estimate is sustained-elevated during the slowdown window, classify the network path as suspect and name the affected path element. • Use peer jobs or nearby placements as negative evidence: if they share the same fabric path and remain healthy, down-rank the fabric hypothesis.
● **Head-of-line blocking and pause propagation:** Head-of-line blocking appears in two forms. Same-egress HOL is a queue problem on one egress port and traffic class. Cross-port HOL occurs when lossless backpressure propagates upstream, causing unrelated flows to wait behind packets destined for a congested port.	• Same-egress HOL suspicion: degraded probes share the same switch egress direction, while probes through the same switch using other egresses remain at baseline. • Cross-port HOL suspicion: multiple unrelated flows through the same switch show correlated OWD spikes even when their final egress ports differ. • Directional OWD separates upstream blocking from downstream blocking. • Path decomposition must include ingress and egress context; otherwise report the result as suspected HOL pattern rather than confirmed HOL mechanism.
● **Incast congestion at a receiver:** Many senders converge on one receiver faster than the receiver-facing link, NIC, or host stack can absorb the traffic. The defining symptom is directional: inbound traffic to the receiver slows down, while outbound traffic from the same receiver remains normal	• Many source NICs show simultaneous forward OWD elevation toward the same destination NIC, destination leaf, or receiver-facing access path. • Reverse OWD from that destination back to the same sources remains at baseline or is materially lower; the common feature across degraded edges is the destination, not the source. • Path decomposition shows diverse sender-side paths converging on the same receiver-side path element. • If only one source group or one corridor is degraded, classify as corridor degradation rather than incast.

The table above narrows down the scope of the issue, with further corroboration from switch telemetry including: (i) Link state and negotiated capacity; (ii) Traffic load and utilization balance; (iii) Queueing and buffer pressure; (iv) Congestion signaling and backpressure; (v) Drops, discards, and packet-handling reason codes; (vi) Physical link integrity and optics; (vii) Control-plane, routing-plane, and platform health; and (viii) Path, topology, and forwarding-state validity.

Conclusion: From network telemetry to operational verdicts

GPU cluster networking has outgrown traditional network monitoring. At scale, the operator's problem is not whether the fabric is generally alive; it is whether a partial, directional, path-specific degradation is turning into a workload-visible straggler. The failures that matter most do not announce themselves as clean outages. They appear as hot corridors, marginal optics, unstable NICs, asymmetric paths, microbursts, credit stalls, queue buildup, or gray failures that remain invisible to any one counter, dashboard, or health check. In a synchronized training workload, any one of those can limit the speed of the entire job.

Clockwork Fleet Monitoring is built to close this gap. The core architectural move is simple but powerful: start with what the workload feels, then work backward to the cause. Synchronized edge OWD provides the detection backbone. Hierarchical probing tells us which tier is implicated. Path decomposition maps the symptom to the switches, ports, links, traffic classes, service levels, and virtual lanes that the probes actually traversed. Triangulation uses both degraded and healthy paths to identify the smallest fabric element that explains the observation. Switch, NIC, optical, congestion, and control-plane telemetry then attach the reason code. The result is not another pile of metrics; it is a causal chain from symptom to path to cause to action.

This is the step beyond Pingmesh and R-Pingmesh that GPU clusters require. Fleet-wide probing proved the right organizing principle. RDMA-native probing proved that the principle applies to modern lossless fabrics. Clockwork takes the next step by making the measurement directional, clock-synchronized, hardware-timestamped, path-attributed, and ultimately workload-correlated. The system is not merely asking, "Is there latency somewhere?" It is answering: which GPU pairs slowed down, in which direction, across which tier, through which path segment, because of which physical, queueing, congestion-control, forwarding-state, or host-adjacent condition.

The operational value is decisive. Silent gray failures are surfaced before they become customer-visible incidents. Slow jobs are triaged quickly as network, host, or workload problems instead of becoming multi-team forensic exercises. When the network is innocent, Clockwork helps prove that and moves the investigation to the right layer. When the fabric is at fault, the alert names the likely switch, port, link, corridor, traffic class, NIC, or access path, with corroborating evidence attached.
Our vision is not just better monitoring. It is a fabric diagnostic system for AI infrastructure, that enables the on-call engineer to see the measured impact together with the evidence that matters, rather than having to infer impact from thousands of counters.

Appendix: Classes of switch telemetry

Telemetry class	What it answers	Spectrum/Ethernet	InfiniBand
Link state and negotiated capacity	Is the port up, stable, and operating at the expected speed and width?	Admin/oper state, carrier transitions, negotiated speed, MTU/config mismatch, port state transitions	Port state, active link speed, active link width, link-down events, SM/UFM port-state changes
Traffic load and utilization balance	Is the implicated link or switch carrying disproportionate load relative to peers?	Per-port bytes/packets, utilization, peer uplink imbalance, traffic-class utilization where available	PortXmitData, PortRcvData, per-port utilization, per-SL/VL traffic where exposed
Queueing and buffer pressure	Was traffic actually queued at the implicated switch, port, queue, traffic class, SL, or VL?	Per-port/per-TC queue occupancy, shared-buffer occupancy, watermarks, buffer histograms, estimated queueing latency	Egress queue-depth indications or histograms where available, ideally per SL/VL
Congestion signaling and backpressure	Did congestion-control or flow-control mechanisms activate on the implicated path element?	PFC tx_pause / rx_pause per priority, ECN-marked packet counters, CNP-related indicators where available	PortXmitWait, per-SL FECN/BECN counters where available, per-VL wait indicators where exposed
Drops, discards, packet-handling reason codes	Were packets dropped or discarded, and why?	WJH drop categories and packet-header context, buffer tail drops, ACL drops, policy drops, interface discards	PortXmitDiscards, discard counters, drop/reason telemetry where available

Appendix: Classes of switch telemetry

Telemetry class	What it answers	Spectrum/Ethernet	InfiniBand
Physical link integrity and optics	Is the medium itself degrading?	CRC/FCS/frame/alignment/symbol errors, FEC/BER trends, DOM Rx/Tx power, laser bias, transceiver temperature/voltage	SymbolErrorCounter, LinkErrorRecoveryCounter, LinkDownedCounter, BER/FEC where exposed, DOM/optic telemetry where available
Control-plane, routing-plane, and platform health	Is the switch or fabric control plane causing correlated latency, reroutes, or instability?	CoPP counters, CPU/memory, systemd/daemon state, routing daemon events, platform sensors, ASIC resource pressure	OpenSM/UFM state changes, SM failover, sweep/reroute activity, routing-engine changes, switch CPU/platform health where exposed
Path, topology, and forwarding-state validity	Did the path, topology, or forwarding state change during the event? Is the path map trustworthy for this sample?	LLDP / neighbor state, FDB / ARP / NDP adjacency, ECMP membership, route changes, link up/down events, switch-agent topology observations	UFM link/node/SM/routing events, OpenSM SUBNET UP, reroute, sweep, LFT-set events, LFT snapshot validity, AR/HBF/SHIELD/PLFT state, SL/VL mapping

Glossary: Acronyms

Acronym	Definition
ACL	Access Control List: A list of rules used to filter network traffic.
ACS	Access Control Services: Typically relates to security or control features, likely within the host or PCIe fabric.
BER	Bit Error Rate: A measure of physical-layer health, often used for optical links.
CoPP	Control Plane Policing: A security feature on switches used to limit traffic sent to the switch CPU.
CQ / CQE	Completion Queue / Completion Queue Entry: Data structures used in RDMA (like InfiniBand or RoCE) to notify the host about the completion of network operations.
DSCP	Differentiated Services Code Point: A field in an IP header used to classify and manage network traffic.
ECMP	Equal-Cost Multi-Path: A routing strategy that allows network traffic to be load-balanced across multiple paths of equal cost.
ECN / FECN	Explicit Congestion Notification / Forward ECN: Signaling mechanisms used to communicate network congestion between endpoints (FECN is specific to InfiniBand/RDMA).
eBPF	Extended Berkeley Packet Filter: A framework used to run custom programs in the Linux kernel, used here for service-aware tracing.
EOS	Extensible Operating System: The network operating system used on Arista switches.
GID	Global Identifier: An address used for routing in an InfiniBand network.
GPU	Graphics Processing Unit: The primary compute engine used in these clusters, whose synchronized communication drives the monitoring requirements.
GRH	Global Routing Header: A header in InfiniBand packets that enables routing across subnets.

Glossary: Acronyms

Acronym	Definition
GUID	Global Unique Identifier: A unique, persistent hardware address for InfiniBand nodes and switches.
HCA	Host Channel Adapter: The InfiniBand or RDMA-capable network interface card on a server.
HOL	Head-of-Line: Refers to a blocking condition (Head-of-line blocking) where a packet at the front of a queue prevents subsequent packets from being processed.
ICMP	Internet Control Message Protocol: Used in networking diagnostics, here specifically for handling time-exceeded replies during path tracing.
LIDs	Local Identifiers: Short-form addresses used for routing within a single InfiniBand subnet.
NCCL	NVIDIA Collective Communications Library: A library optimized for high-speed, multi-GPU communication, which is the primary workload being monitored.
NIC	Network Interface Card: The hardware component used by nodes to connect to the network fabric.
NUMA	Non-Uniform Memory Access: A memory architecture on a host, relevant for host-side bottleneck analysis.
PFC	Priority Flow Control: A link-level flow control mechanism used in lossless Ethernet (RoCE) to prevent frame loss due to congestion.
PSU	Power Supply Unit: A hardware component that is part of the switch's health checks.

Glossary: Acronyms

Acronym	Definition
QP	Queue Pair: A communication endpoint used in RDMA fabrics to send and receive data.
QoS	Quality of Service: Mechanisms for prioritizing traffic and guaranteeing performance levels.
RCA	Root-Cause Analysis: The process of identifying the fundamental cause of a detected failure.
RNIC	RoCE Network Interface Card: A network card that supports the RoCE protocol.
RoCE	RDMA over Converged Ethernet: A protocol that enables Remote Direct Memory Access (RDMA) functionality over a standard Ethernet network.
SL2VL	Service Level to Virtual Lane: A mapping configuration in InfiniBand used to assign traffic prioritized by Service Level to specific Virtual Lanes.
SVM	Support Vector Machine: A machine learning model used here as an estimator for extracting clock offset and drift in the ClockSync subsystem.
ToR	Top-of-Rack: The leaf switch at the top of a server rack.
VL	Virtual Lane: A mechanism in InfiniBand and RoCE for traffic segmentation and flow control on a single physical link.