Shrinking the Observability Bill

Smart Strategies for Cost-Effective Kubernetes Monitoring

Aritra Ghosh

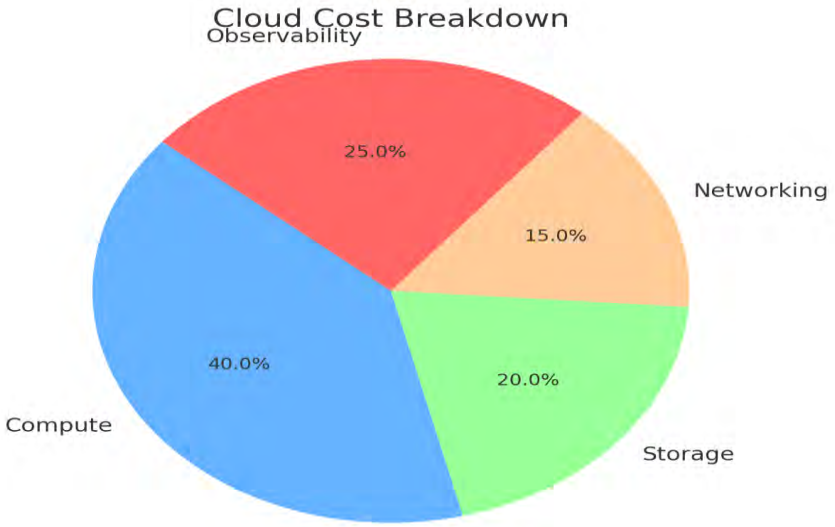
CONF42

Agenda

1. Why Observability Costs Are Rising
2. A Framework to Cut the Waste
3. Deep Dives: Control Plane, Nodes, Pods, Network
4. Tools & Dashboards That Help
5. Key Takeaways & Action Plan

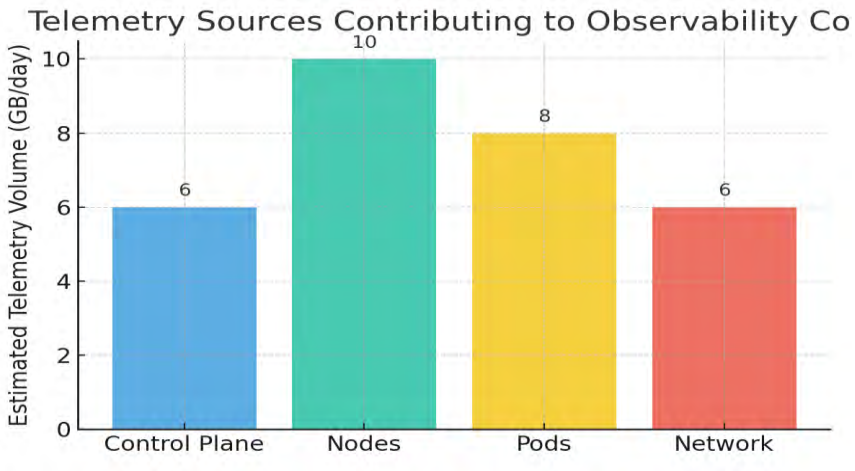
Why This Talk Matters

- Observability can cost up to 40% of your cloud bill (even more).
- Too much data, not enough insight.
- We need smart, efficient monitoring strategies.



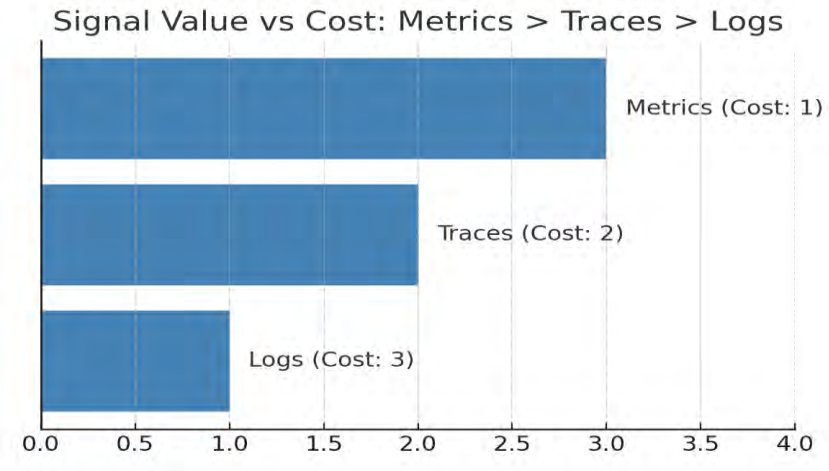
Where the Dollars Go

- High-cardinality custom metrics.
- Long log retention and verbose logging.
- Multiple tools with duplicate data pipelines.
- Expensive egress and storage fees.



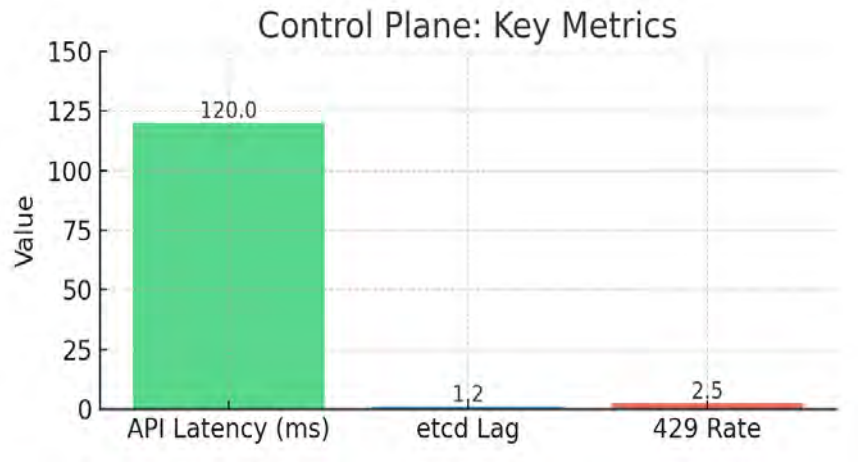
A Framework for Cost-Effective Observability

- Collect what matters: SLOs > all logs.
- Pick the right signal: metrics > traces > logs.
- Tune retention and aggregation.
- Consolidate and offload wherever possible.



Control Plane Monitoring: Precision, Not Volume

- Use managed service metrics (e.g., API latency, 429s, etcd health).
- Avoid ingesting full logs — sample or summarize.
- Alert based on deviations, not static thresholds.



Smart Node Monitoring

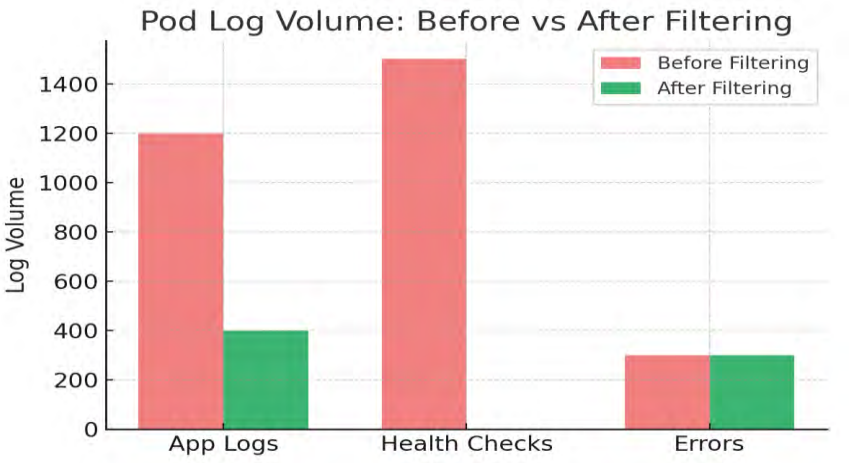
- Track only key metrics: CPU, memory pressure, disk IO, readiness.
- Avoid per-process metrics unless debugging.
- Identify node degradation via heartbeats and NotReady transitions.

Optimizing Metrics with OpenTelemetry

- Use explicit aggregation temporality (delta/sum).
- Filter labels to reduce cardinality.
- Limit custom metrics to business-critical paths.

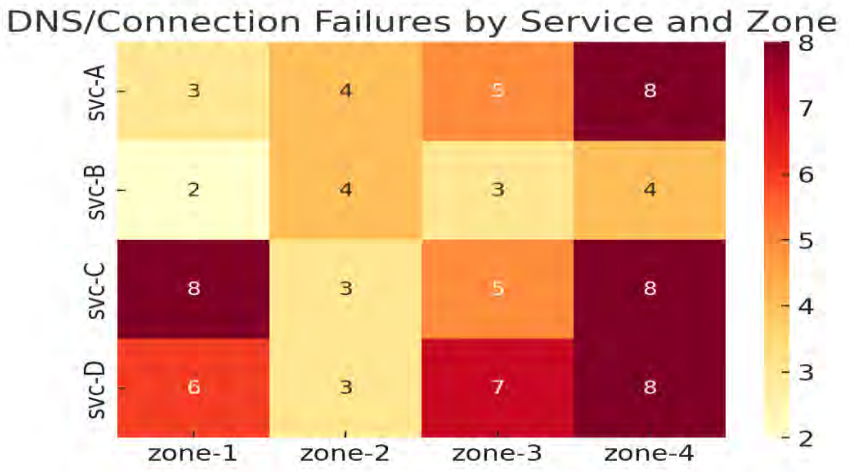
Pod-Level Visibility Without the Noise

- Filter logs at source — exclude health checks, static endpoints.
- Use labels to correlate telemetry with services.
- Adopt OpenTelemetry for auto-tagging and structured tracing.



Network: Catch the Big Issues Only

- Track connection failures and DNS resolution issues.
- Use sampling for flow logs (NetFlow, IPFIX).
- Adopt eBPF-based tooling where supported.



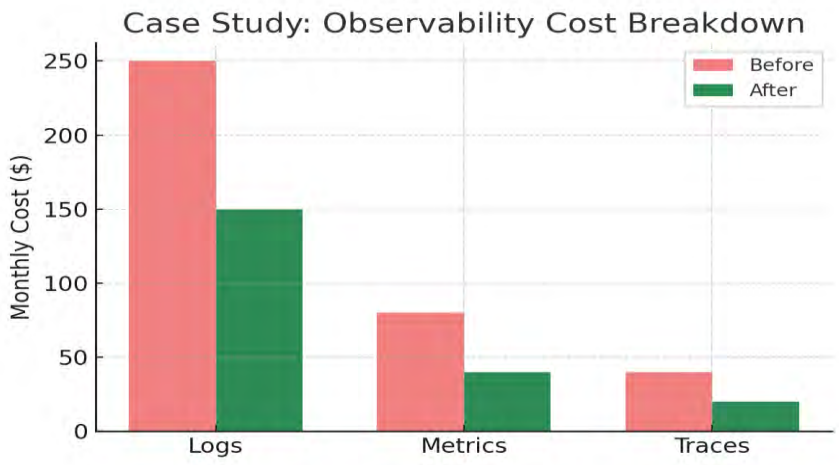
Third-Party Tooling: Consolidate and Compress

- Avoid duplicating telemetry across tools.
- Ship filtered logs using grep, drop, or Fluent Bit parsers.
- Choose tools based on $/GB for metrics vs logs vs traces.

Tool	Metrics ($/GB)	Logs ($/GB)	Traces ($/GB)
Prometheus	0.0	0.0	0.0
Grafana Cloud	0.1	0.05	0.08
Datadog	0.3	0.6	0.4
New Relic	0.28	0.55	0.38
Azure Monitor	0.2	0.45	0.3

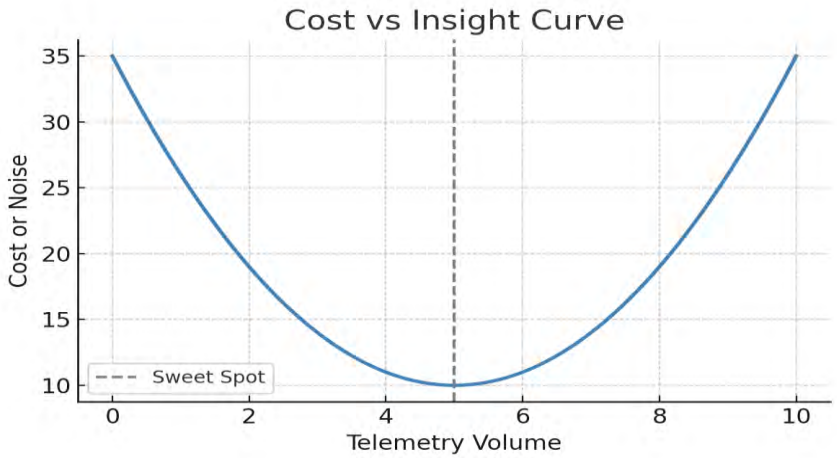
Case Study: 40% Cost Reduction Without Losing Signal

- Dropped health check logs, shrank verbosity levels.
- Switched from raw latency logs to Prometheus histograms.
- Moved cold observability data to low-cost object storage.



Know What You're Giving Up

- Sampling ≠ Losing visibility — it's strategic reduction.
- Shorter retention works for most active monitoring cases.
- Balance fidelity with engineering and cost trade-offs.



Cost-Aware Observability Dashboard

- Visualize cost per telemetry type: logs, metrics, traces.
- Add tags for team ownership and cost attribution.
- Integrate with billing export APIs for full transparency.

Build Your Observability Budget

- Cap $/cluster/month for telemetry
- Set log volume and custom metric quotas
- Use Fluent Bit, Loki limits for enforcement

Your Cost Optimization Checklist

Drop	Shorten	Filter	Consolidate	Tune
Drop unused metrics and logs.	Shorten retention wherever possible.	Filter telemetry at source.	Consolidate tools to avoid duplication.	Tune alert rules to minimize noise.

High-Cardinality Metric Pitfalls

- Avoid labels like userid or UUID in metrics
- Use tier, region, status codes instead
- Reduce series count → lower Prometheus costs

Alert Noise Detox

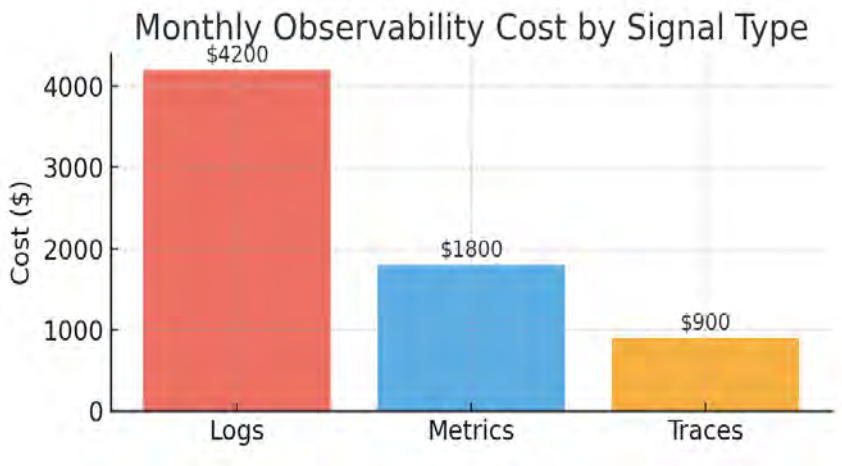
- -Funnel: Raw → Filtered → Routed → Actionable
- -Only alert on SLO violations, customer impact
- Reduce fatigue to improve MTTR

Shrink the Bill, Keep the Insight

- Observability ≠ Log everything.
- Focus on valuable signals, not raw volume.

End-to-End Observability Flow

- Ingest → Process → Store → Alert → Optimize
- Tag value at each stage to measure ROI.
- Make optimization a continuous loop.



Recommended tools to start with

- Prometheus with Thanos or Cortex for low-cost metrics
- Fluent Bit for log filtering at source
- OpenTelemetry SDKs and Collector
- Grafana for dashboards with cost overlays
- eBPF-based tools (e.g., Cilium, Pixie) for efficient network observability

Closing Call to Action

Common Pitfalls to Avoid

- Logging everything without sampling or filtering
- Using default retention settings for all telemetry
- Duplicate telemetry sent to multiple destinations
- Unmonitored cost spikes from high-cardinality metrics
- Alerting on static thresholds without context

Metric	Before Cleanup	After Cleanup
Log Volume	1.5 TB	600 GB
Alert Count	1800/day	900/day
Ingestion Cost	$12,000	$4,500
MTTR	45 mins	20 mins

“Better signals. Lower costs. Happier engineers.”

Thank you