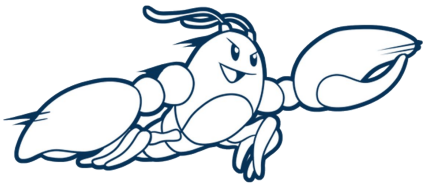


Buoyant Cloud – See What Your Mesh Is Telling You



Agenda

- What is Buoyant Cloud?
- Architecture and how data flows
- Deploying the agent
- Dashboard walkthrough (metrics, topology, events, health)
- Golden metrics

- Alerts and integrations
- Multi-cluster visibility
- Data privacy and security
- Best practices
- Demo / Q&A



The Problem: Why Do You Need Buoyant Cloud?

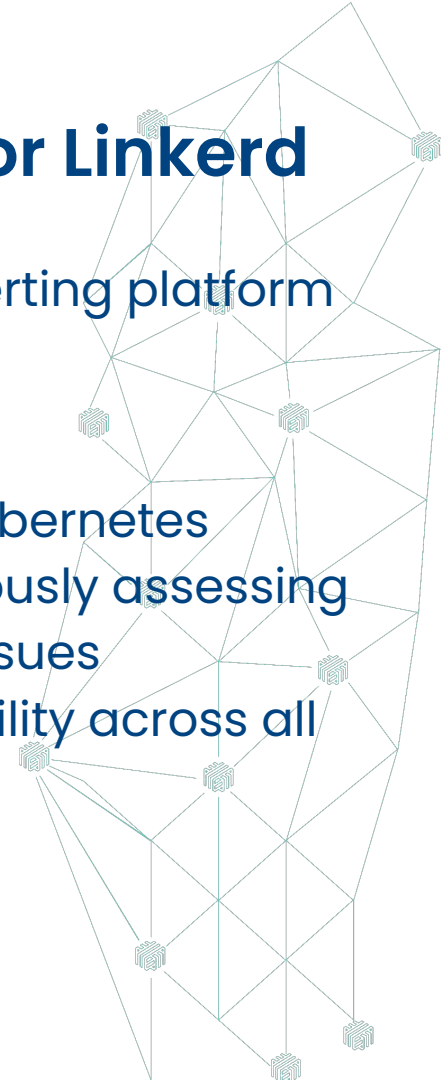
- **Brings multiple service mesh** deployments into a single pane of glass
- **Lack of unified visibility** across multiple clusters
- **Reactive troubleshooting** instead of proactive health management
- **Teams need SRE tooling** without building everything from scratch
- **Pain points:** "How do I know my mesh is healthy? How do I keep it healthy?"



Buoyant Cloud: Your Command Center for Linkerd

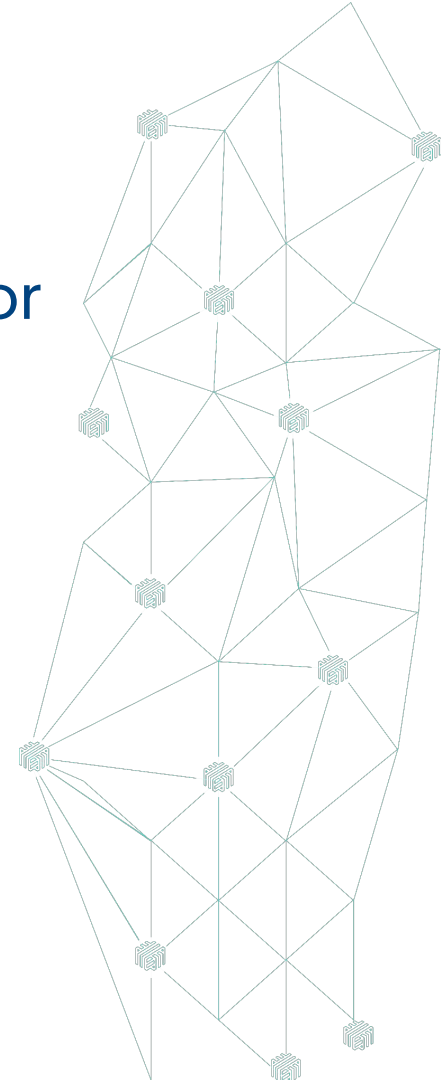
Buoyant Cloud is the SaaS dashboard, monitoring, and alerting platform for Buoyant Enterprise for Linkerd (BEL).

It provides a global, cross-cluster health dashboard for Kubernetes environments running Linkerd, automatically and continuously assessing the health of Linkerd deployments, pinpointing potential issues proactively, and providing global platform-wide observability across all connected clusters.



What is Buoyant Cloud?

- Dashboard, monitoring, and alerting tool for Buoyant Enterprise for Linkerd (BEL)
- Provides a single pane of glass across all your Linkerd-managed clusters
- Proactive diagnostics
- Performance management



Architecture Overview

- **Agent pushes data out** — no inbound connections required
- **Agent is lightweight** — minimal resource footprint
- **Deployed via separate Helm chart**, integrates with standard release cycle tooling



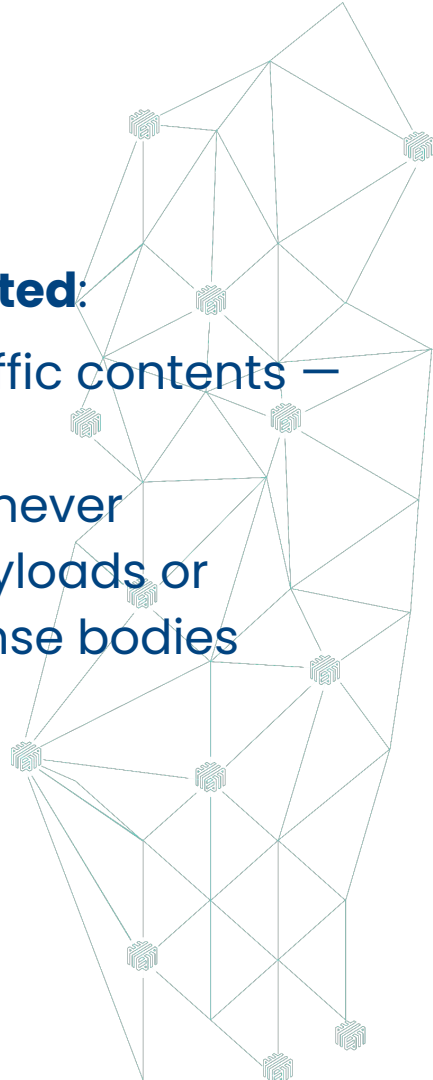
What Data Does Buoyant Cloud Collect?

What IS collected:

- Metadata about Deployments, StatefulSets, DaemonSets
- Linkerd CRDs (ServerAuthorizations, Servers, Multiclusters etc)
- Proxy metrics about application traffic (success rates, latencies, request volumes)
- Proxy mTLS public key info (NOT private keys)

What is **NOT** collected:

- Application traffic contents – never
- Private keys – never
- Application payloads or request/response bodies



Deploying the Buoyant Cloud Agent

Prerequisites:

- BEL Strategic Plan
- Linkerd control plane installed
- Helm

Steps:

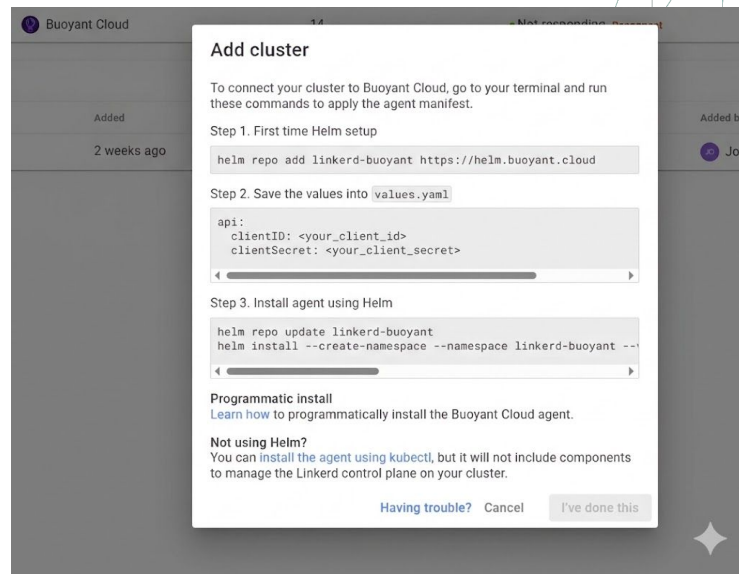
- Retrieve **client ID** and **secret** from Buoyant Cloud Dashboard and set these variables, along with the **cluster name**

```
terminal
helm repo add linkerd-buoyant https://helm.buoyant.cloud
helm install --create-namespace \
  --namespace linkerd-buoyant \
  --set api.clientID = $CLIENT_ID \
  --set api.clientSecret = $CLIENT_SECRET \
  --set metadata.agentName = $CLUSTER_NAME \
  linkerd-buoyant linkerd-buoyant/linkerd-buoyant
> _
```



Agent Configuration

- **Custom values file** from Buoyant Cloud Clusters page
- **Configurable** resource requests/limits for the agent pod
- **Network policies:** agent needs outbound HTTPS to Buoyant Cloud endpoints
- **Upgraded through Helm**



Add cluster

To connect your cluster to Buoyant Cloud, go to your terminal and run these commands to apply the agent manifest.

Step 1. First time Helm setup

```
helm repo add linkerd-buoyant https://helm.buoyant.cloud
```

Step 2. Save the values into `values.yaml`

```
api:
  clientId: <your_client_id>
  clientSecret: <your_client_secret>
```

Step 3. Install agent using Helm

```
helm repo update linkerd-buoyant
helm install --create-namespace --namespace linkerd-buoyant --
```

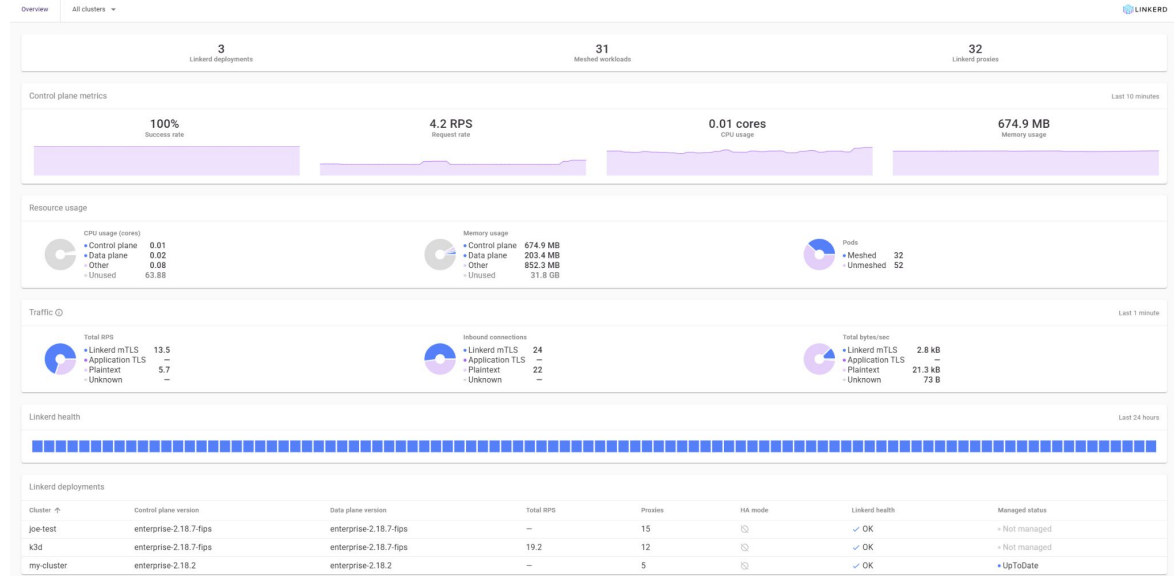
Programmatic install
Learn how to programmatically install the Buoyant Cloud agent.

Not using Helm?
You can install the agent using kubectl, but it will not include components to manage the Linkerd control plane on your cluster.

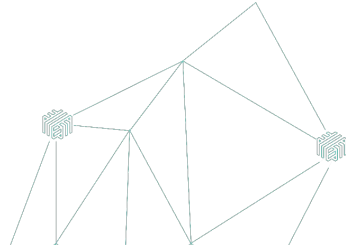
Having trouble? Cancel I've done this

Dashboard: Overview

- **Unified** view across all connected clusters
- **At-a-glance** health status of the mesh
- **Quick indicators:** control plane health, data plane health, version info, resource usage
- **Number of meshed** vs unmeshed workloads
- **Recent events** and alerts



Dashboard: Workload Metrics



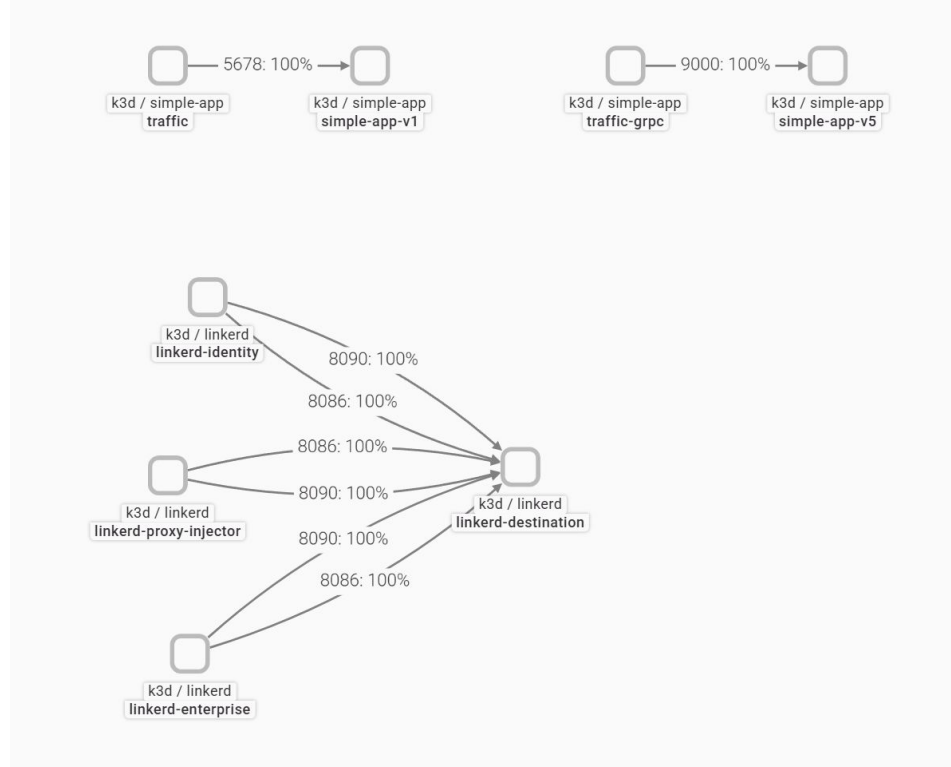
Golden metrics for every meshed workload:

- **HTTP/gRPC/TCP** success rates
- **Latency percentiles** (p50, p95, p99)
- **Request volume** (RPS)
- **Real-time** CPU and memory usage for any Kubernetes workload (meshed or not)
- **Historical graphs** – trend analysis
- **Drill down** from cluster to namespace to workload
- **Quickly identify** workloads in trouble



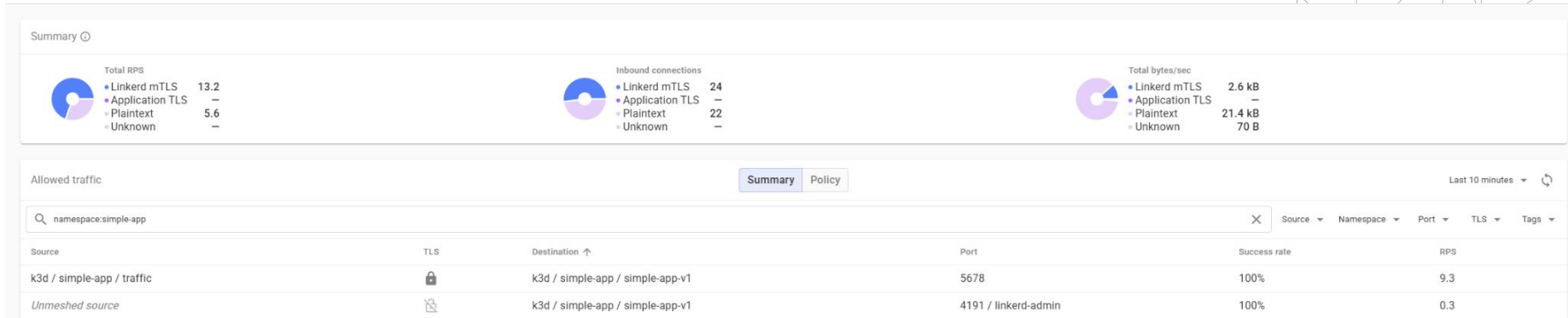
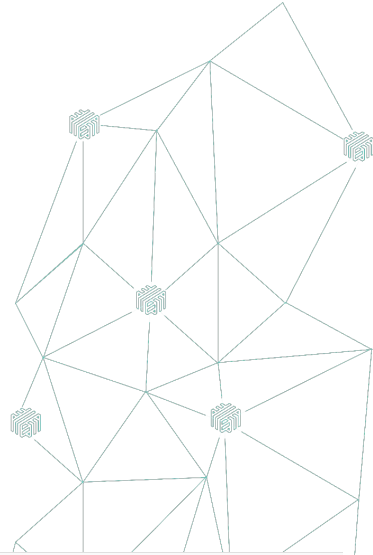
Dashboard: Topology

- **Visual dependency map** between workloads
- **See which services call which** – clients and dependencies
- **Detect new dependencies** or new clients automatically
- **Filter by namespace**, cluster, workload
- **Pin nodes** to focus on specific service chains
- **Use case:** "Where does traffic go after it hits my API gateway?"



Dashboard: Traffic

- **Traffic view** with success rate column
- **Time window selector**
- **Policy columns** showing authorization policy status
- **Same underlying data** source as the Topology page

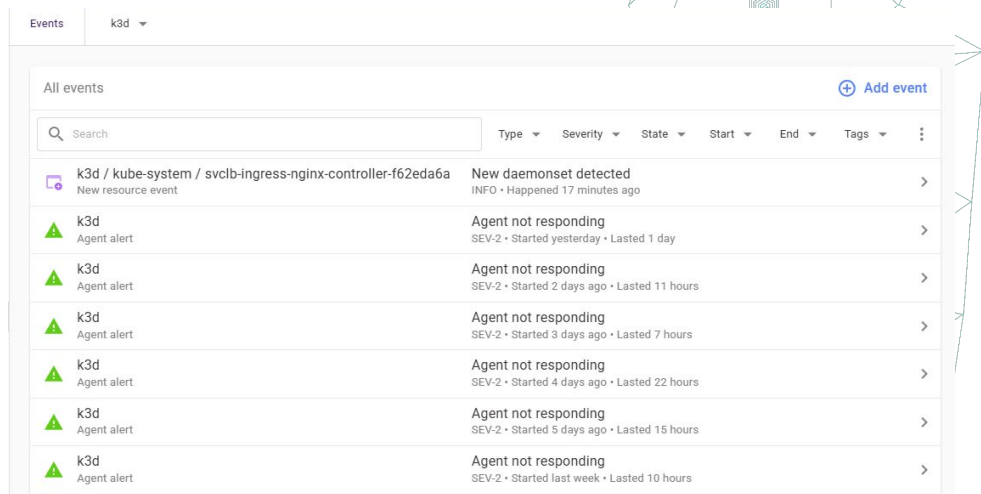


Dashboard: Events and Health Checks

- **Continuous health assessment** of the Linkerd deployment
- **Proactive diagnostics** – issues flagged before they become incidents

Event types:

- Version incompatibilities (control plane vs data plane)
- Certificate expiration warnings
- Configuration drift
- Rollout events (deployments, restarts)
- New workload meshed / unmeshed
- Sanity-checking mesh installs and upgrades



The screenshot shows the Linkerd dashboard's 'Events' section for a cluster named 'k3d'. The interface includes a search bar and filters for Type, Severity, State, Start, End, and Tags. The event list contains the following entries:

Type	Source	Message	Severity	State	Start	End	Tags
New resource event	k3d / kube-system / svclb-ingress-nginx-controller-f62eda6a	New daemonset detected	INFO	Occurred	17 minutes ago		
Agent alert	k3d	Agent not responding	SEV-2	Occurred	Started yesterday	Lasted 1 day	
Agent alert	k3d	Agent not responding	SEV-2	Occurred	Started 2 days ago	Lasted 11 hours	
Agent alert	k3d	Agent not responding	SEV-2	Occurred	Started 3 days ago	Lasted 7 hours	
Agent alert	k3d	Agent not responding	SEV-2	Occurred	Started 4 days ago	Lasted 22 hours	
Agent alert	k3d	Agent not responding	SEV-2	Occurred	Started 5 days ago	Lasted 15 hours	
Agent alert	k3d	Agent not responding	SEV-2	Occurred	Started last week	Lasted 10 hours	

Alerts and Integrations

Alert on:

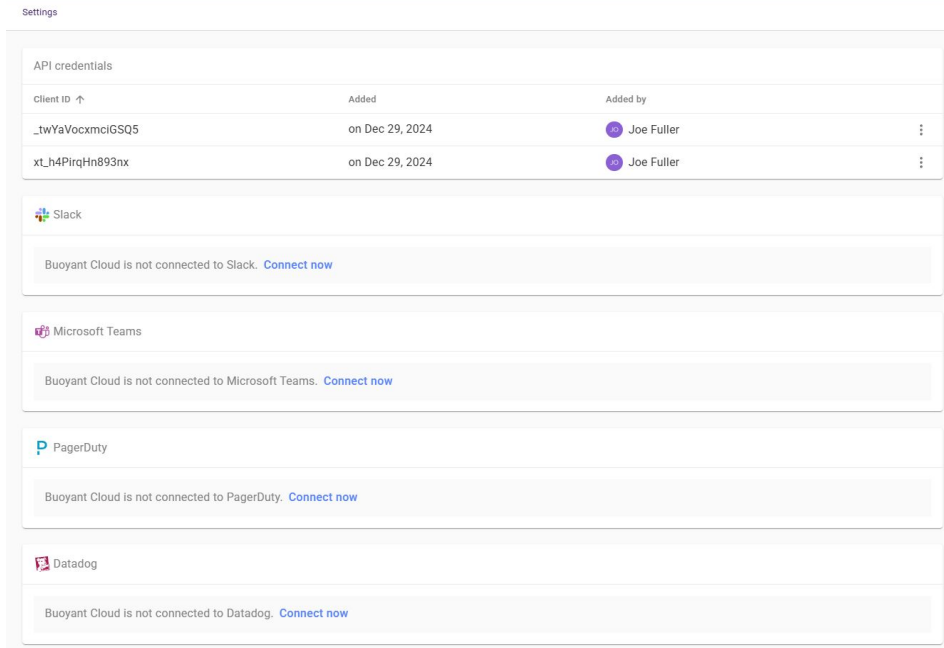
- Linkerd health issues
- Workload traffic anomalies
- Rollout events
- SLO breaches

Notification channels:

- Slack
- Pagerduty
- Microsoft Teams
- Datadog

Integrate into your existing incident response workflows

Tip: start with broad alerts, refine over time



Settings

API credentials	Client ID ↑	Added	Added by	
	_twYaVocxmciGSQ5	on Dec 29, 2024	Joe Fuller	⋮
	xt_h4PirqHn893nx	on Dec 29, 2024	Joe Fuller	⋮

Slack

Buoyant Cloud is not connected to Slack. [Connect now](#)

Microsoft Teams

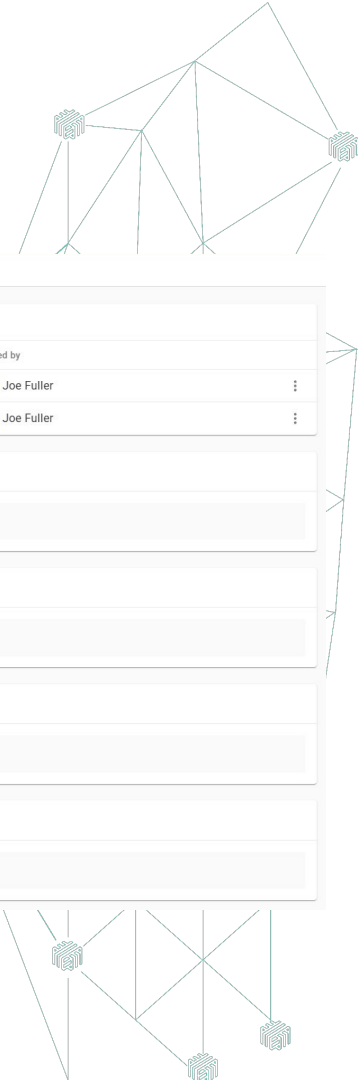
Buoyant Cloud is not connected to Microsoft Teams. [Connect now](#)

PagerDuty

Buoyant Cloud is not connected to PagerDuty. [Connect now](#)

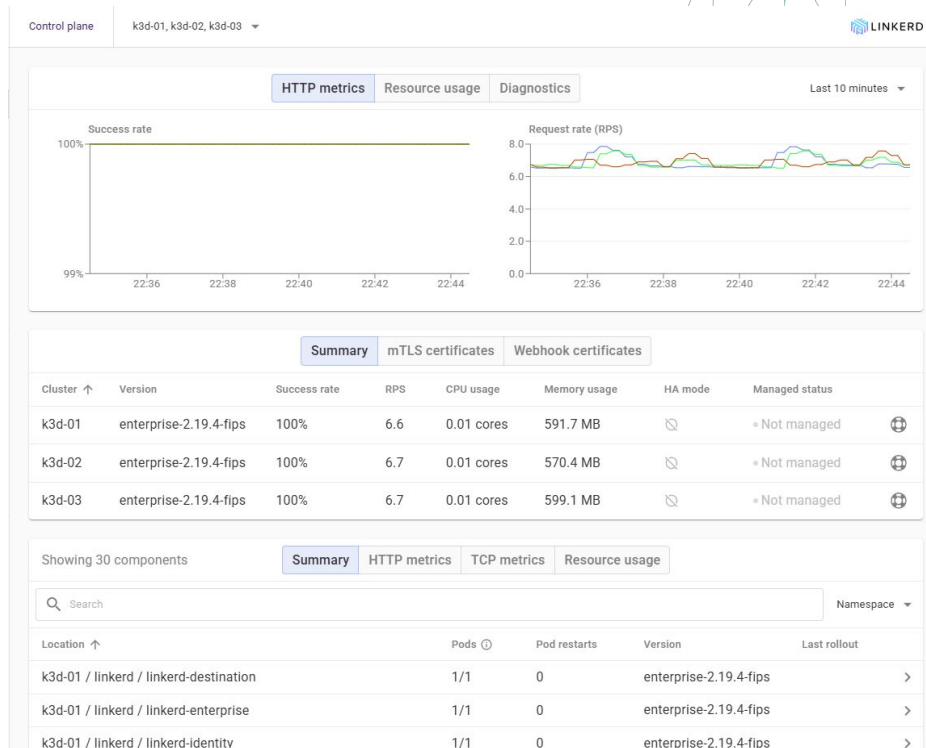
Datadog

Buoyant Cloud is not connected to Datadog. [Connect now](#)



Multi-Cluster Visibility

- **Single dashboard** across all clusters
- **Compare health and metrics** across environments (dev, staging, prod)
- **Identify cross-cluster** version drift
- **Multi-cluster** service mirroring visibility
- **Use case:** "Are all my production clusters running the same Linkerd version?"



Common Pitfalls / Troubleshooting

- **Agent can't connect:** check network policies / egress rules
- **Cluster name mismatch:** renaming after registration can cause confusion
- **Stale data:** agent pod not running or crashlooping
- **Missing metrics:** workloads not meshed (no sidecar = no proxy metrics)
- **Certificate issues:** ensure `buoyant-cloud-org-credentials` secret is correct
- **Resource:** link to Buoyant troubleshooting docs



Best Practices

- **Naming clusters thoughtfully** – the `CLUSTER_NAME` persists in Buoyant Cloud; use a convention (e.g., `env-region-purpose`)
- **Deploy the agent to every cluster** – even dev/staging, for full visibility
- **Integrate alerts from day one** – connect Slack/PagerDuty before you need it
- **Use workload exclusion** – exclude noisy or irrelevant workloads to reduce clutter
- **Review topology regularly** – catch unexpected dependencies early
- **Keep the agent updated** – upgrade via Helm when new versions are released
- **Verify agent artifacts** – use signature verification for supply chain security
- **Use "Send Diagnostics"** – for troubleshooting with Buoyant support



Q&A

