Linkerd 2.8

Multi-cluster Kubernetes made simple and secure by default

Oliver Gould @olix0r

LINKERD

An open source **service mesh** for Kubernetes.

- 🔥 **46+** months in production
- 🔥 CNCF since **January 2017**
- 🔥 **3,500+** Slack channel members
- 🔥 **10,000+** GitHub stars
- 🔥 **100+** contributors
- 🔥 *Edge* releases **weekly**
- 🔥 *Stable* releases every **~2 months**

CLOUD NATIVE COMPUTING FOUNDATION

NORDSTROM CHASE

GoDaddy Walmart

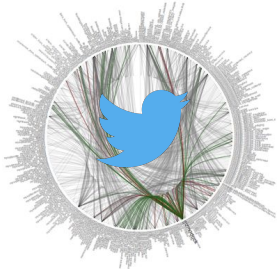
STRAVA BIGCOMMERCE

Expedia OfferUp

ebay Cisco webex

COMCAST planet.

History of Linkerd



LINKERD 0.1

CLOUD NATIVE COMPUTING FOUNDATION

LINKERD 2.0

2010-2015 | 2/2016 | 1/2017 | 9/2018

Two parallel branches of development:

- 🚀 **Linkerd 2.x:** ultralight, zero-config, Kubernetes-first (active)
- 🚀 **Linkerd 1.x:** JVM-based and multi-platform (maintenance)

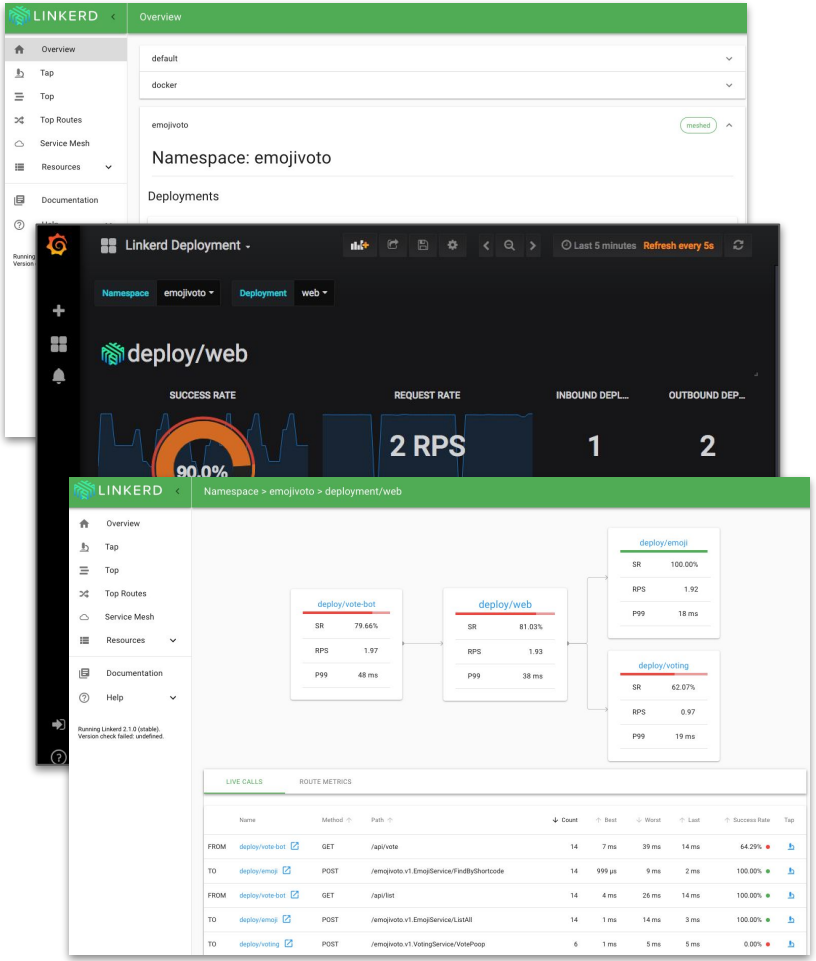
What does Linkerd do?

⚡ **Observability:** *Golden metrics*: success rates, latencies, throughput; Service topologies; Distributed and ad-hoc tracing.

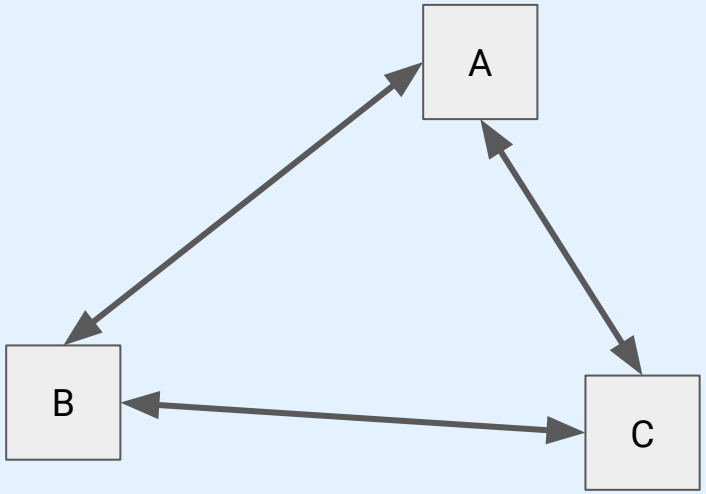
⚡ **Connectivity:** Load balancing, retries, timeouts, **multi-cluster**

⚡ **Security:** Transparent mTLS, cert management and rotation, policy*

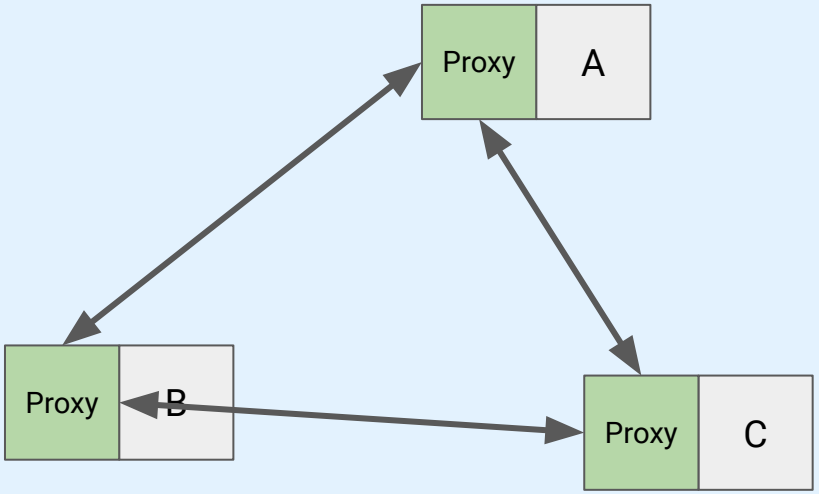
Focused on **operational simplicity**



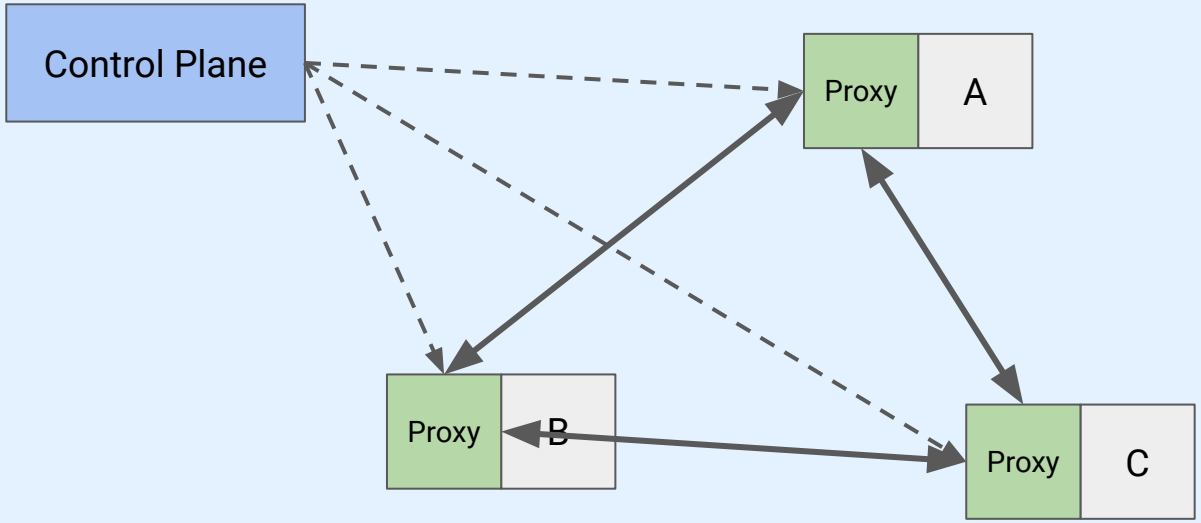
Microservices



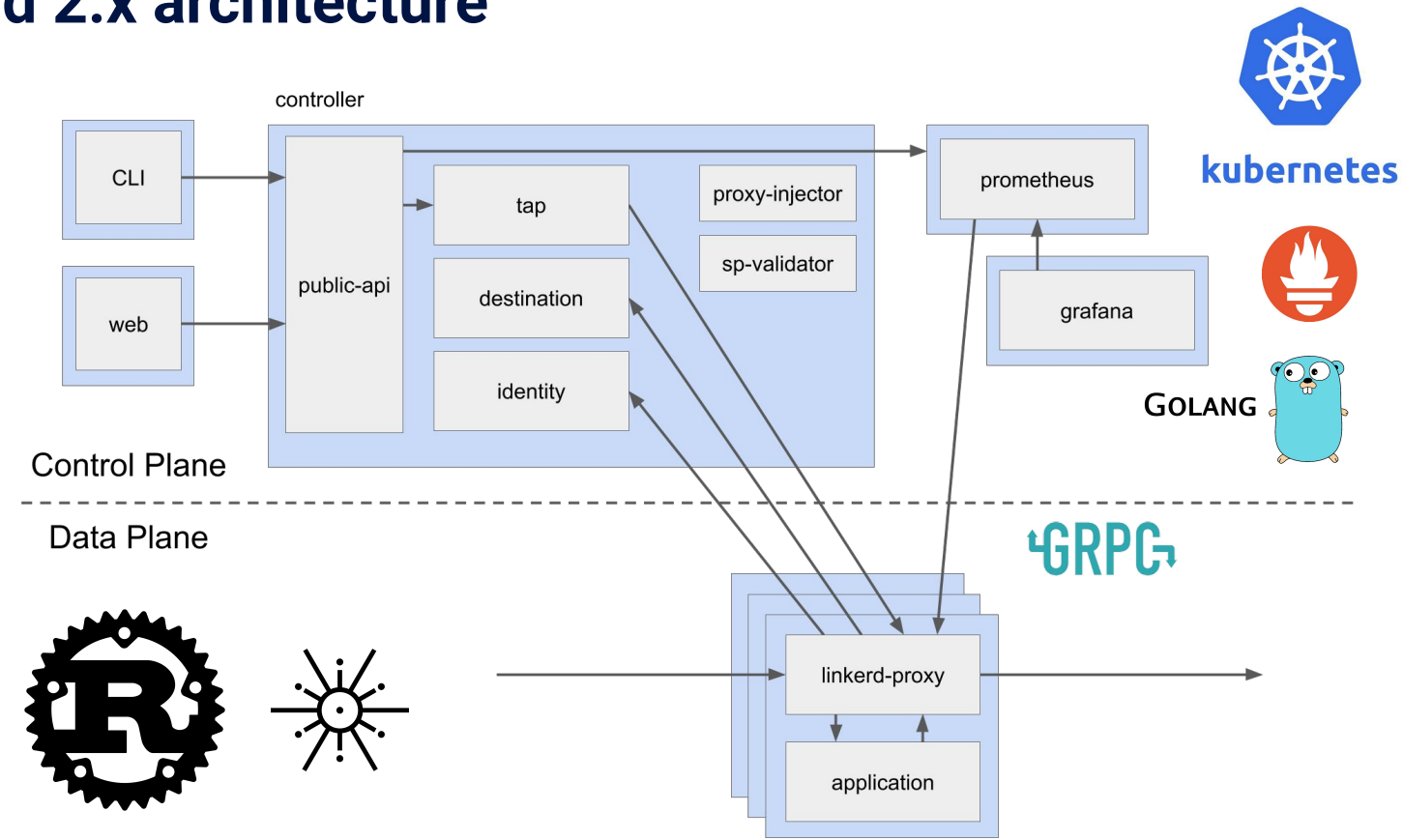
Service Mesh: Data Plane



Service Mesh: Control Plane



Linkerd 2.x architecture



How is Linkerd designed?

Zero-config, out-of-the-box functionality

Minimal latency and resource overhead

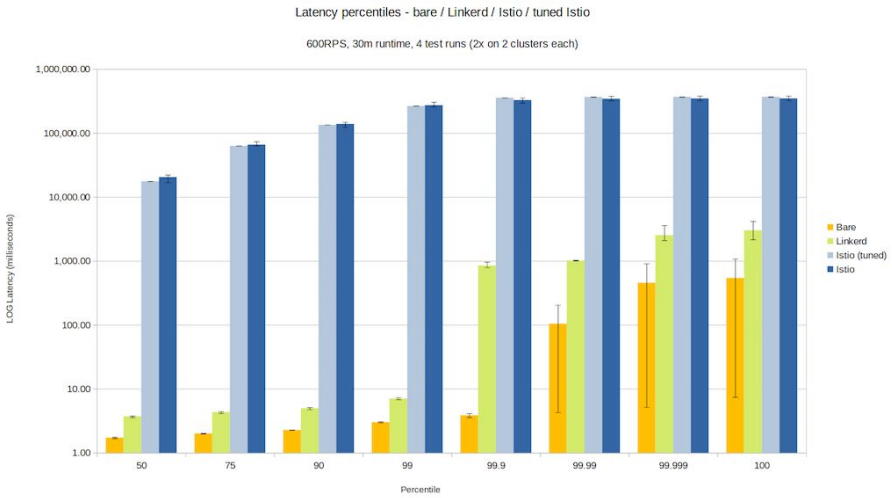
Kubernetes-native; integrates with the ecosystem

Control plane: Go. ~200mb RSS (excluding Prometheus). (Repo: linkerd/linkerd2).

Data plane: Rust. <20mb RSS, ~1ms p99. (Repo: linkerd/linkerd2-proxy)

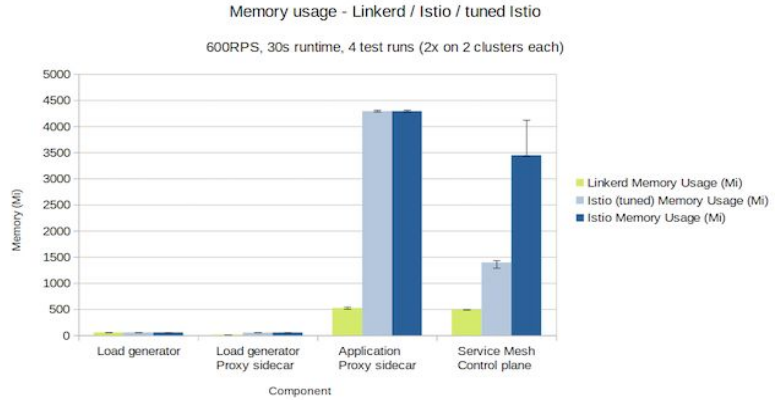
Background reading: Linkerd v2: How Lessons from Production Adoption Resulted in a Rewrite of the Service Mesh (InfoQ)

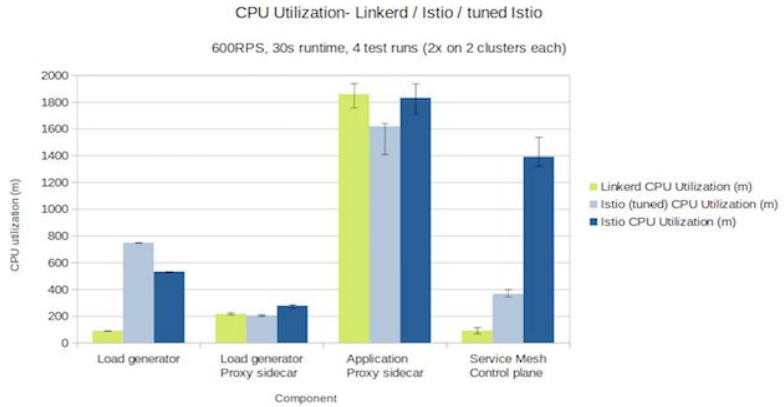
How fast/small is it?



Tl;dr: really fast. Not as fast as "no service mesh", but *significantly* smaller and faster than Istio.

Source: https://kinvolk.io/blog/2019/05/performance-benchmark-analysis-of-istio-and-linkerd/

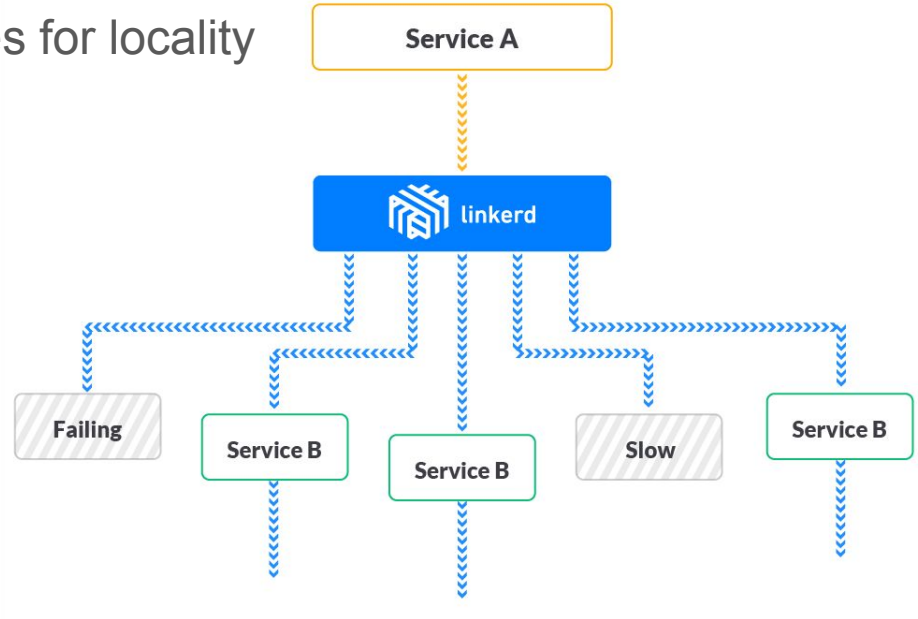


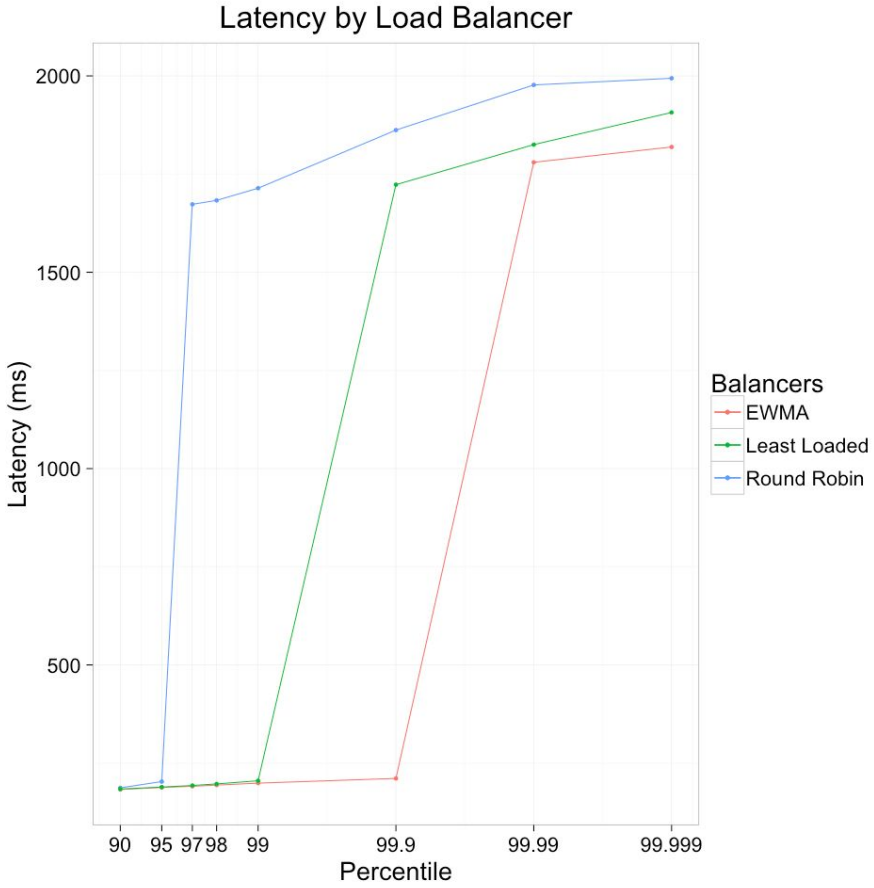


What Does it Do?

Peak-EWMA HTTP/gRPC Balancing

- Efficiently distributes requests across k8s `Deployments`, etc
- Client-side: No centralized balancer state
- Latency-aware: Automatically optimizes for locality
- Backed by k8s `Services`
- Bypasses kube-proxy
- No application changes




Latency by Load Balancer
2000
1500
1000
500
Latency (ms)
90
95
97
98
99
99.9
99.99
99.999
Percentile
Balancers
EWMA
Least Loaded
Round Robin

Automatic, transparent mutual TLS

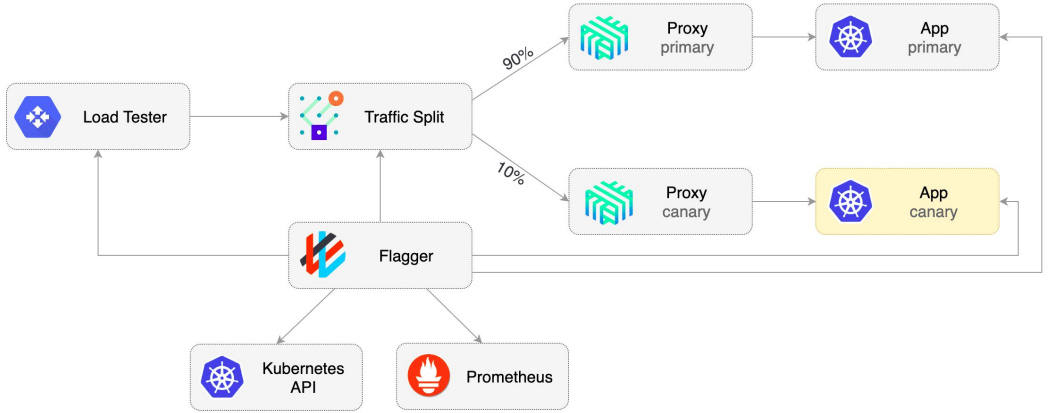
- All meshed HTTP traffic automatically secured
- Extends *workload identity* for zero-trust communication
  - Bootstrapped from k8s `ServiceAccounts`
- Automatic pod certificate rotation
- Can bootstrap from `cert-manager`
- Does not conflict with Ingress/Application TLS
- mTLS for arbitrary protocols coming in stable-2.9
- No application changes

Transparent HTTP/2 Multiplexing

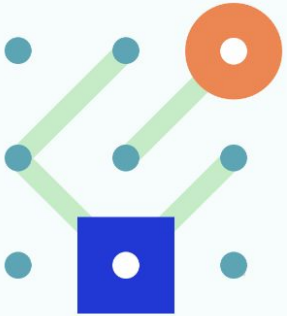
- All meshed HTTP/1 traffic over HTTP/2 between pods
- Amortizes connection overhead (TCP, mTLS)
- No application changes

Traffic Splitting

- For canary and blue/green
- Splits requests between k8s `Services`
- Uses the Service Mesh Interface's `TrafficSplit` API
- Can be driven by Flagger



The Service Mesh Interface



What SMI covers

Service Mesh Interface is a specification that covers the most common service mesh capabilities:

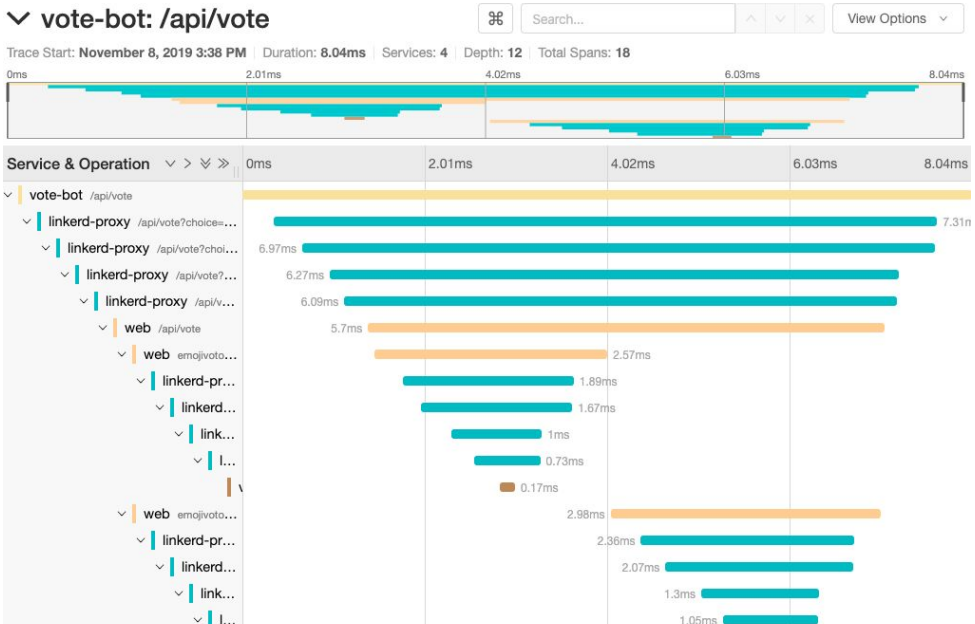
- Traffic policy – apply policies like identity and transport encryption across services
- Traffic telemetry – capture key metrics like error rate and latency between services
- Traffic management – shift traffic between different services

High-fidelity Prometheus Visibility

- **Uniform**: Every pod gets the same, app-independent traffic metrics
- HTTP- and gRPC-aware
- Rich k8s workload metadata
- Raw latency histograms: no `avg` on latencies
- Can be enhanced with **OpenAPI** (Swagger) & **gRPC** (Protobuf) specs
- No application changes

Distributed Tracing with OpenCensus

- Linkerd participate in your application's OpenCensus tracing
- *Application changes required*



Ad-hoc tracing with Linkerd Tap

- Tap into the request stream at runtime
- Authorized via k8s RBAC
- No application changes

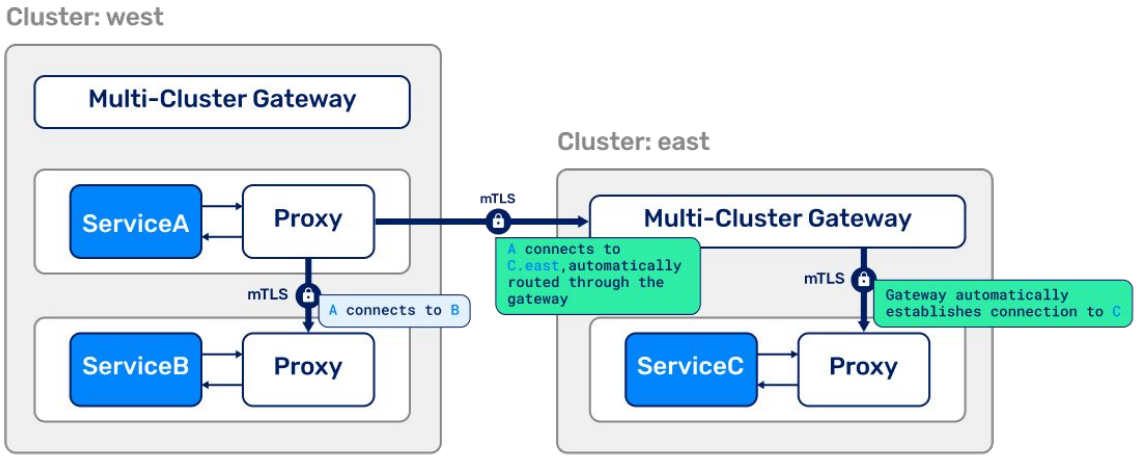
(press q to quit)
(press a/LeftArrowKey to scroll left, d/RightArrowKey to scroll right)

Source	Destination	Method	Path	Count	Best	Worst	Last	Success Rate
linkerd-prometheus-5dd896954c-g7snn	10.244.0.219	GET	/metrics	6	1ms	3ms	2ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.4.222	GET	/metrics	5	2ms	3ms	2ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.1.16	GET	/metrics	5	2ms	3ms	2ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.4.221	GET	/metrics	5	1ms	4ms	3ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.2.82	GET	/metrics	5	2ms	4ms	4ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.3.116	GET	/metrics	4	1ms	3ms	1ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.3.115	GET	/metrics	4	1ms	3ms	3ms	100.00%
10.244.4.1	linkerd-grafana-548d67bdd-ftv62	GET	/api/health	4	448µs	547µs	530µs	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.4.220	GET	/metrics	4	2ms	4ms	4ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	10.244.0.220	GET	/metrics	3	1ms	1ms	1ms	100.00%
10.244.2.1	linkerd-destination-6d9d9dfbf6-fq6hd	GET	/ready	3	395µs	629µs	395µs	100.00%
linkerd-prometheus-5dd896954c-g7snn	linkerd-sp-validator-77f8b989-g6bjq	GET	/metrics	3	2ms	2ms	2ms	100.00%
10.244.3.1	linkerd-web-55bfcf9698-5wxwf	GET	/ping	3	449µs	723µs	472µs	100.00%
linkerd-prometheus-5dd896954c-g7snn	linkerd-controller-78844b9b87-z8sgl	GET	/metrics	3	2ms	2ms	2ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	linkerd-grafana-548d67bdd-ftv62	GET	/metrics	3	2ms	3ms	2ms	100.00%
linkerd-prometheus-5dd896954c-g7snn	linkerd-destination-6d9d9dfbf6-fq6hd	GET	/metrics	3	2ms	3ms	3ms	100.00%
10.244.0.1	linkerd-sp-validator-77f8b989-g6bjq	GET	/ready	3	466µs	885µs	573µs	100.00%
linkerd-prometheus-5dd896954c-g7snn	linkerd-proxy-injector-648d6864b6-f8fqt	GET	/metrics	2	2ms	2ms	2ms	100.00%
10.244.1.1	linkerd-controller-78844b9b87-z8sgl	GET	/ping	2	346µs	578µs	578µs	100.00%
10.244.3.1	linkerd-web-55bfcf9698-5wxwf	GET	/ready	2	453µs	614µs	453µs	100.00%
10.244.1.1	linkerd-prometheus-5dd896954c-g7snn	GET	/-/healthy	2	459µs	468µs	468µs	100.00%
linkerd-prometheus-5dd896954c-g7snn	linkerd-web-55bfcf9698-5wxwf	GET	/metrics	2	2ms	2ms	2ms	100.00%
10.244.3.1	linkerd-proxy-injector-648d6864b6-f8fqt	GET	/ping	2	461µs	490µs	490µs	100.00%
10.244.0.1	linkerd-sp-validator-77f8b989-g6bjq	GET	/ping	2	375µs	532µs	375µs	100.00%
10.244.1.1	linkerd-controller-78844b9b87-z8sgl	GET	/ready	2	432µs	446µs	432µs	100.00%
10.244.1.1	linkerd-prometheus-5dd896954c-g7snn	GET	/-/ready	2	646µs	668µs	668µs	100.00%
10.244.2.1	linkerd-destination-6d9d9dfbf6-fq6hd	GET	/ping	2	537µs	614µs	614µs	100.00%
10.244.3.1	linkerd-proxy-injector-648d6864b6-f8fqt	GET	/ready	2	602µs	969µs	602µs	100.00%

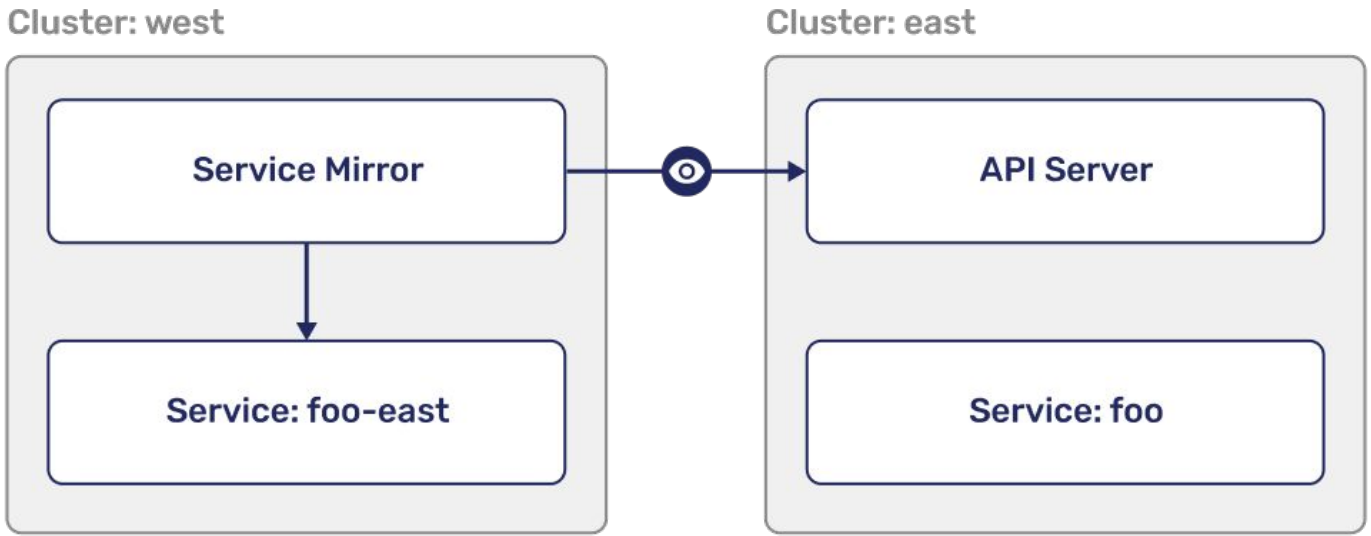
Zero-Trust Multi-Cluster

Zero-Trust Multi-Cluster Connectivity

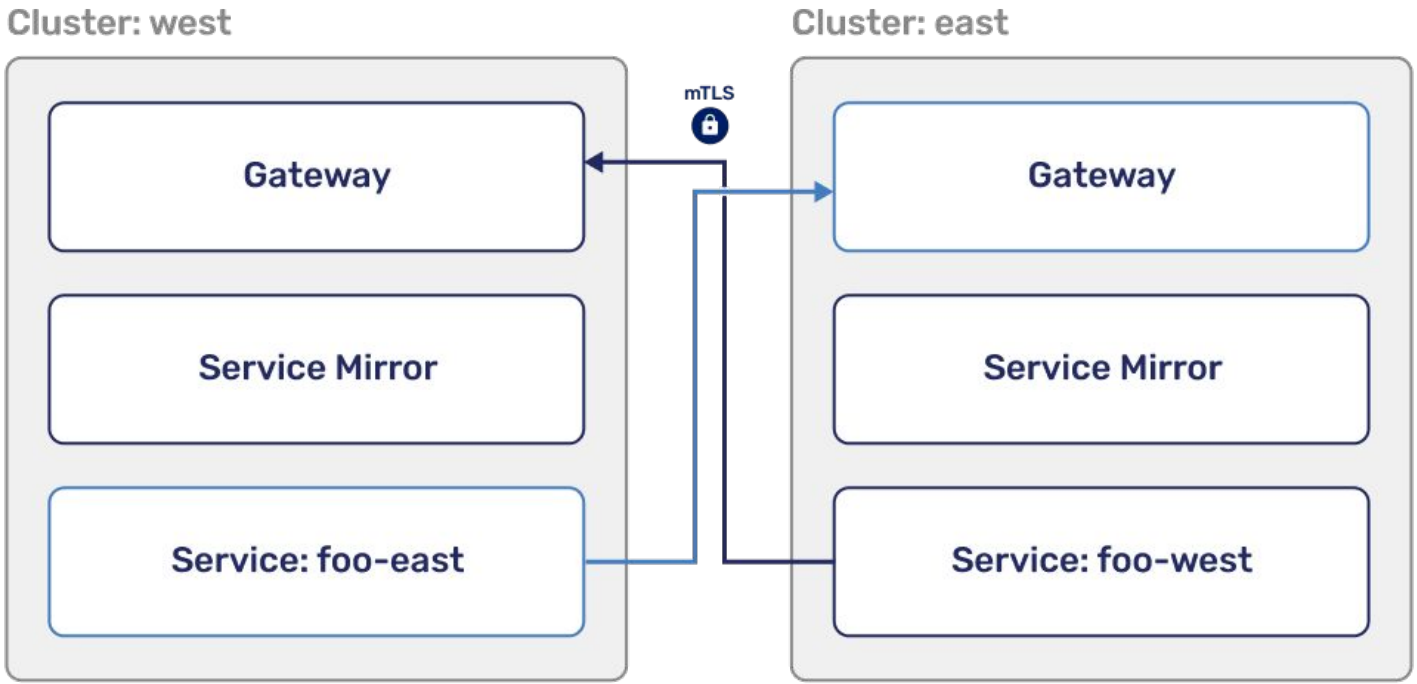
- Backed by k8s `Services`
- Better with SMI `TrafficSplit`
- Secured by mTLS
- Optional component
- No application changes



Service Mirroring

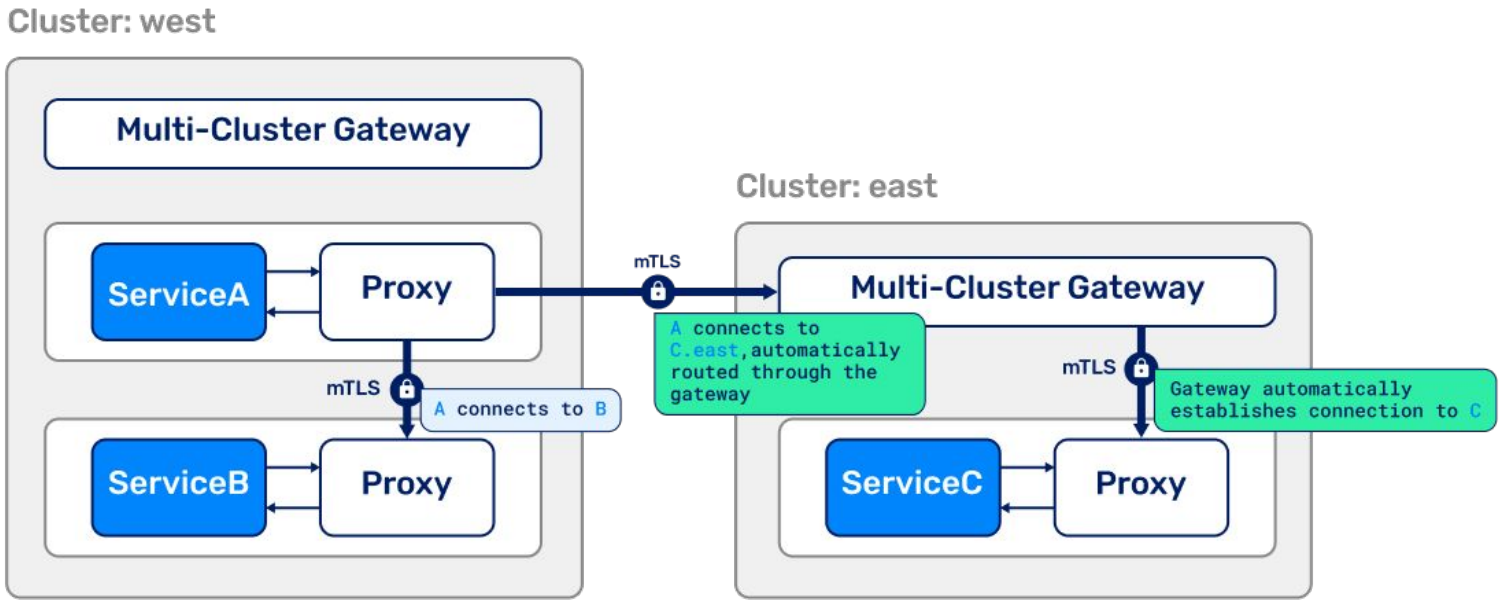


Service Mirroring



Zero-Trust Multi-Cluster Demo

- Demo!



Looking Forward

Roadmap

- Minimized modular control plane
  - Optional/configurable Prometheus, Web, ..
- Mandatory TLS
  - Arbitrary TCP protocols
  - TCP load balancing
- `proxy_wasm`

Join us for KubeCon + CloudNativeCon Virtual

Event dates: August 17-20, 2020

Schedule: Now available!

Cost: $75

Register now!

Get involved!

- 💚 Development is all on GitHub
- 💚 New RFC process
- 💚 Thriving community in the Slack
- 💚 Formal announcements on the CNCF mailing lists
- 💚 Monthly community calls
- 💚 Formal 3rd-party security audits

Linkerd has a friendly, welcoming community! Join us!

Linkerd is 100% Apache v2 licensed, owned by a neutral foundation (CNCF), and is committed to