Overview & State of Linkerd

KubeCon NA 2021

Oliver Gould  @olix0r
Project lead

LINKERD

Agenda

- A brief overview of Linkerd
- A peek at what's new & upcoming
- Special guest appearance
- Questions!?

Ultralight, ultrafast, ultra-simple **service mesh** for Kubernetes.

- **4+** years in production
- **5,000+** Slack channel members
- **10,000+** GitHub stars
- **200+** contributors
- Linkerd 1.x donated to **CNCF in 2016**
- **Linkerd 2.x** started in 2017

And many more...

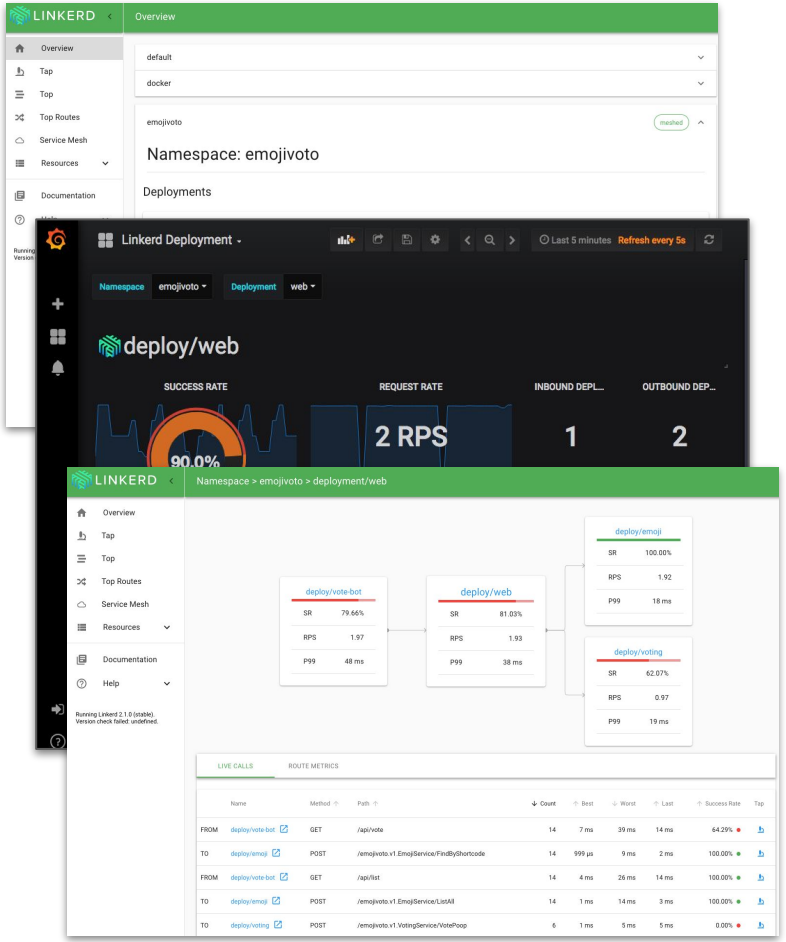
What does Linkerd do?

⚡ **Observability:** Service-level *golden metrics*: success rates, latencies, throughput. Service topologies. Tracing.

⚡ **Reliability:** *Request-level load balancing*, retries, timeouts, traffic shifting

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

In an ultralight package focused on **operational simplicity** first and foremost.



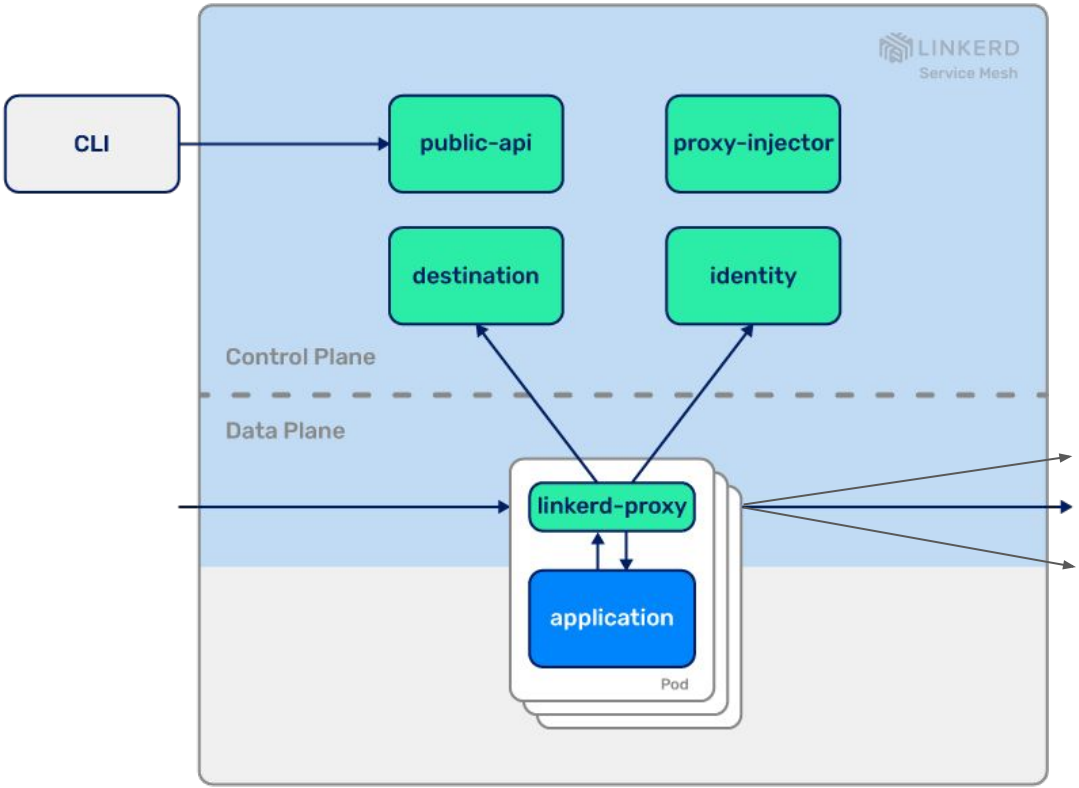
Design principles

"Less is more, and more is less."

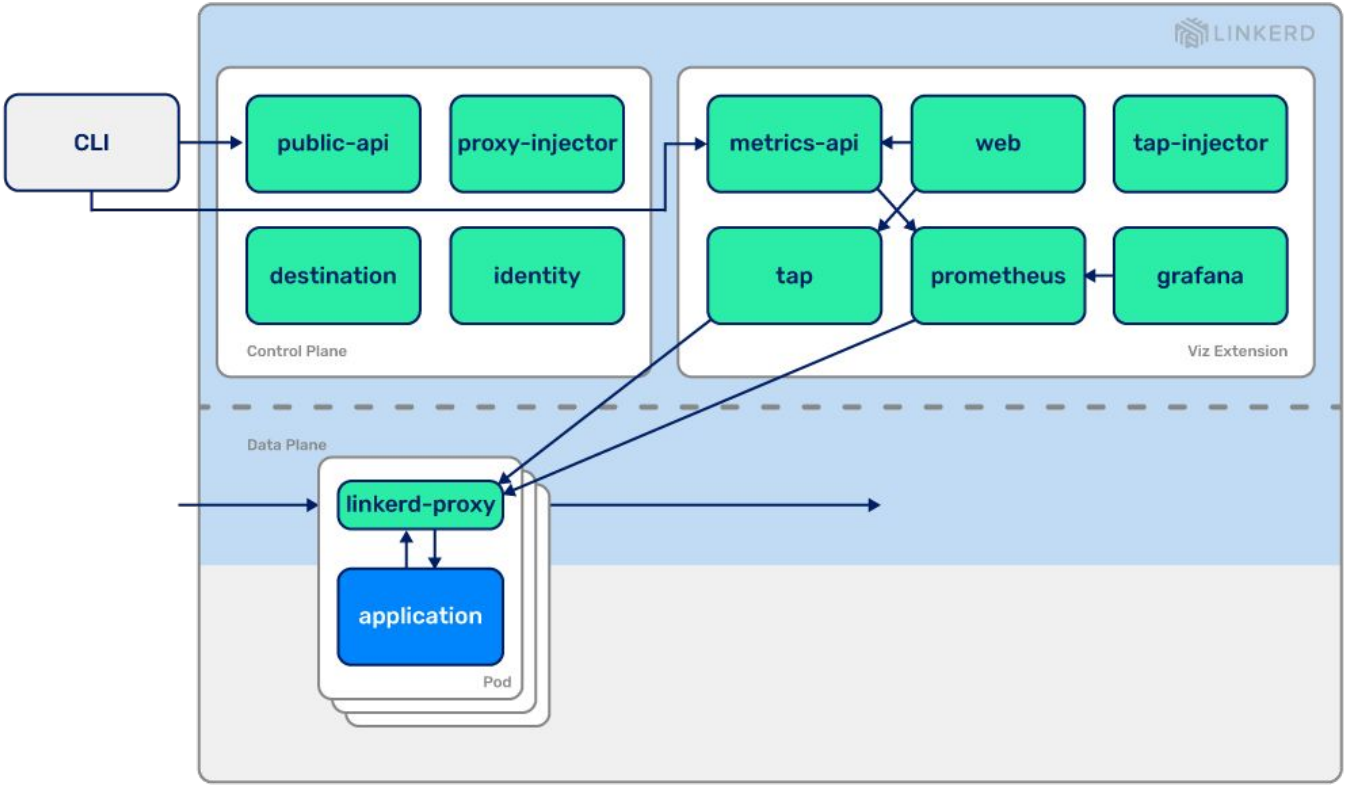
- 💪 **Just works** for any Kubernetes app
- 💪 **Ultralight:** Bare minimum perf and resource cost
- 💪 **Simple:** builds on Kubernetes primitives to reduce operational complexity
- 💪 **Security out of the box**: make security the *default*

Control plane: *Rust* + Go. ~200mb footprint. (github.com/linkerd/linkerd2).
Data plane: *Rust*. <20mb footprint, <1ms p99 (github.com/linkerd/linkerd2-proxy)
Background reading: Linkerd v2: How Lessons from Production Adoption Resulted in a Rewrite of the Service Mesh (InfoQ)

The service mesh architecture



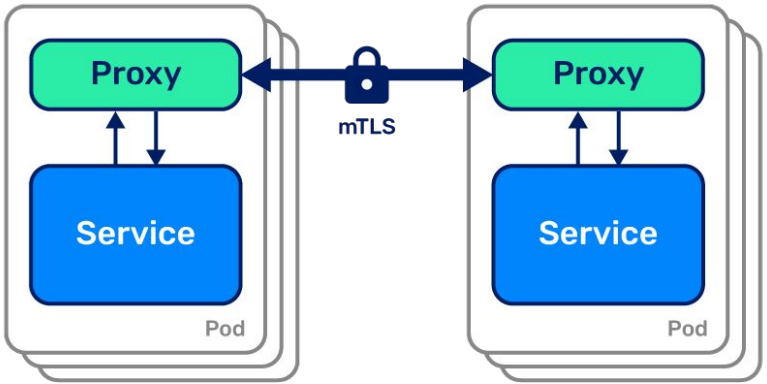
Ready-made observability with linkerd-viz



LINKERD

Transparent, automatic, mutual TLS

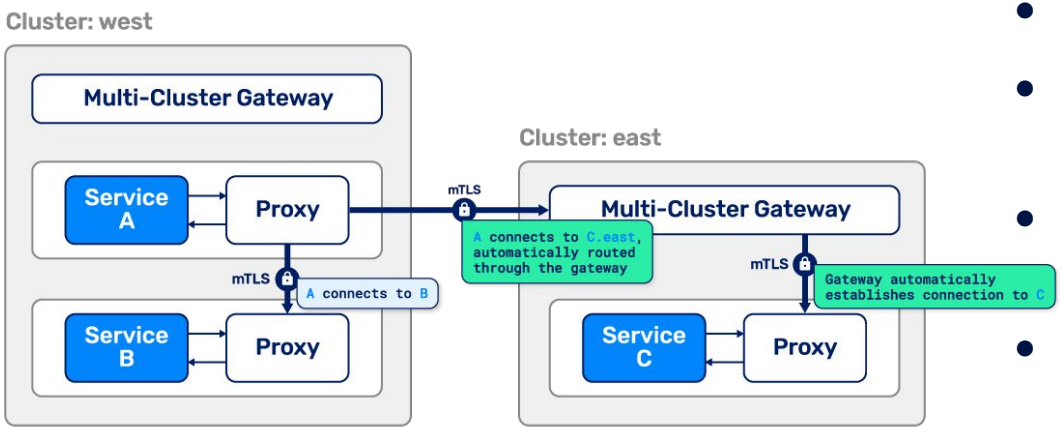
Workload *identity* is established via mTLS for all meshed communication.



- Good defaults:
  - TLSv1.3
  - ECDSA P-256/SHA256
  - No OpenSSL/BoringSSL
- Private keys never leave pod's memory
- Automatic certificate rotation
- Bootstraps from pod `ServiceAccount`
- No application changes necessary

Cross-cluster connectivity with linkerd-multicluster

Connects Kubernetes services across cluster boundaries in a way that's secure, fully transparent to the application, and independent of network topology.



- **Unified trust domain** across all clusters
- **Separate failure domains** so there's no SPOF
- **Works over the open Internet** so no difficult L3/L4 requirements
- **A unified communication model** with in-cluster communication

LINKERD

The Linkerd Sidecar Proxy

A "micro-proxy" called simply Linkerd2-proxy. (Not Envoy!)

- 🚀 **Security first:** Rust lets us avoid an entire class of memory vulnerabilities.
- 🚀 **Ultralight, ultrafast:** Rust compiles to native code. No runtime env/GC!
- 🚀 **State-of-the art network stack:** Built on Tokio, Hyper, H2, Tower, and the rest of the modern Rust async networking stack.
- 🚀 **Built for the mesh:** Automatic protocol detection, mTLS, etc

Philosophy: proxy should be an implementation detail. You should not have to become an operational expert in the proxy just to operate the service mesh! (See Why Linkerd Doesn't Use Envoy for more.)

Linkerd vs Istio

A common question! It comes down to: what do you *need* in a service mesh?

Istio

Theme: Every feature, every situation, no matter the complexity.

Pros:

- Has all the features
- Your boss has read a blog post about it
- No one ever got fired for choosing IBM
- Supports complex use cases

Cons:

- Big, complex, hard to operate

Linkerd

Theme: the bare minimum to build a secure, reliable, flexible Kubernetes platform.

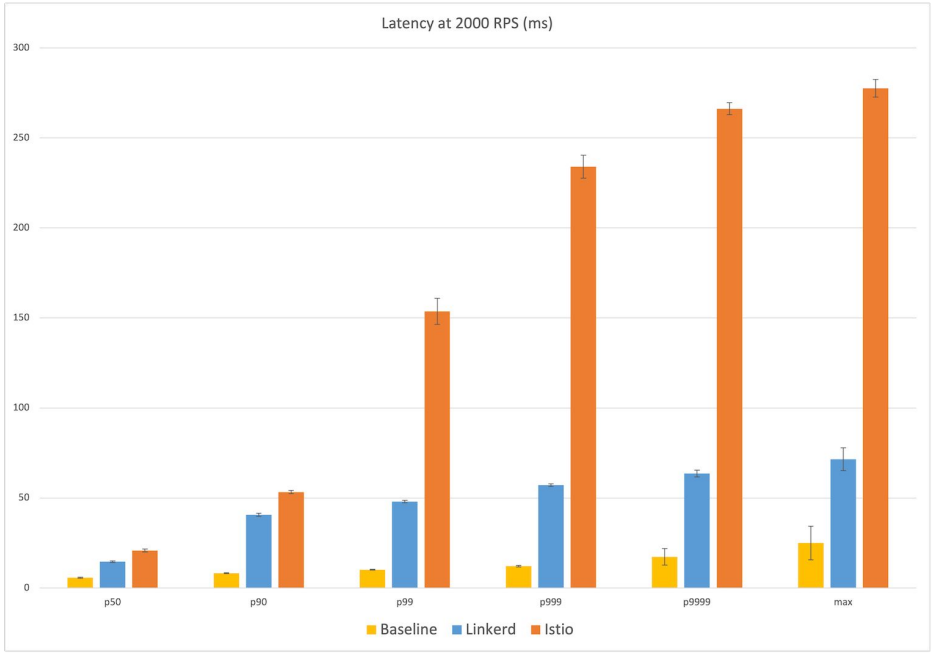
Pros:

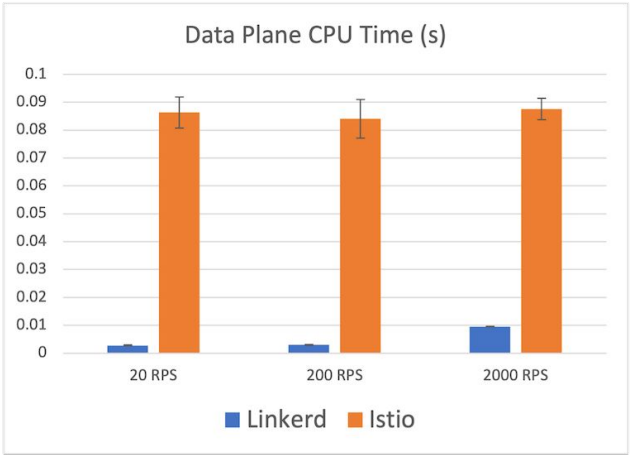
- Much easier to understand and operate
- Smaller, lighter, faster
- Secure by default
- Kubernetes native

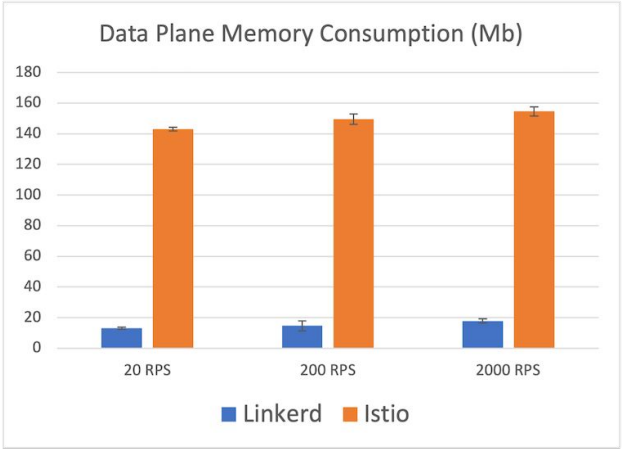
Cons:

- Only works on Kubernetes

How fast/small is Linkerd?

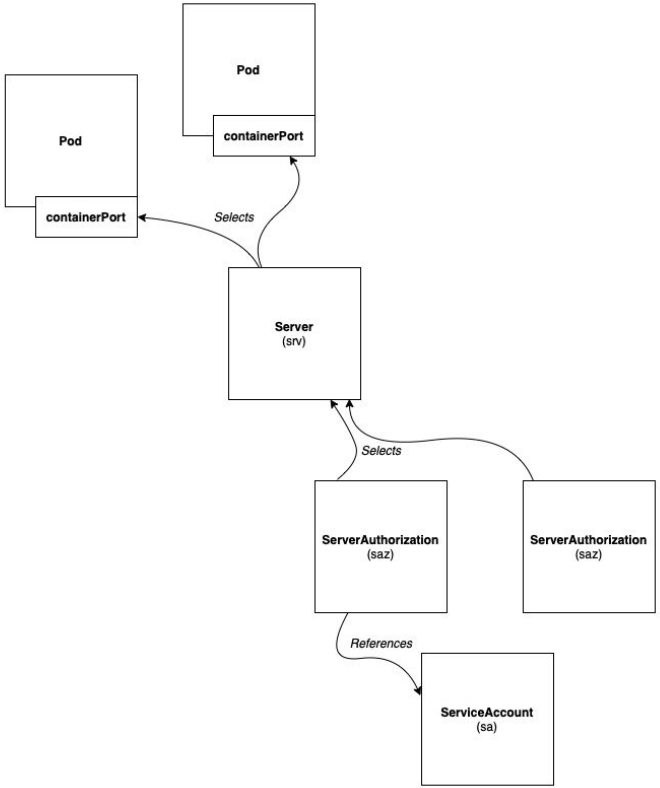






Really fast and really small. *Significantly* faster and smaller than Istio. (2021 source)

New in Linkerd 2.11: Authorization Policies



`Servers` select pods/ports.

`ServerAuthorizations` select `Servers`, authorizing connections/requests to:

- Require mTLS
- Restrict client access

LINKERD

`ServerAuthorization` example

```yaml
apiVersion: policy.linkerd.io/v1beta1
kind: Server
metadata:
  namespace: emojivoto
  name: emoji-grpc
  labels:
    app-kind: internal
spec:
  podSelector:
    matchLabels:
      app: emoji  # matches pods labeled app=emoji
  port: grpc  # matches the containerPort named grpc
  proxyProtocol: gRPC  # configures protocol detection
```

```yaml
apiVersion: policy.linkerd.io/v1beta1
kind: ServerAuthorization
metadata:
  namespace: emojivoto
  name: emoji-grpc
spec:
  server:
    selector:
      app-kind: internal  # matches servers by label
  client:
    meshTLS:  # authorizes mTLS'd connections from web
      serviceAccounts:
        - name: web
```

LINKERD

In the oven...

- Richer outbound policies
  - Circuit-breaking/Failover
  - Improved traffic splitting
- HTTP access logging
- `TPROXY` support to preserve client IPs (for ingresses)
- `linkerd-smi` extension (split SMI from core functionality)
- Separate Helm chart for CRDs…
- Proxy: `io-uring`
- FIPS 140-2
- … and more!

Linkerd Community Updates

As Linkerd grows, so does its community, here are a few awesome updates:

- 💚 **CNCF Graduation** 🎉
- 💚 Linkerd Steering Committee with production users
- 💚 Rolled out the Linkerd Community Anchor program
- 💚 Developer Discord Server
- 💚 Weekly *edge* channel releases
- 💚 Monthly community meetings

Linkerd talks at KubCon

Oct 11

- **A Crash Course in mTLS and Authorization for Kubernetes with Linkerd**

Oct 12

- **Replatforming a $4B Retailer Onto Kubernetes and Linkerd**
- **Service Mesh War Stories - Lessons Learned the Hard Way**

Oct 13

- Virtual office hours
- **Overview and State of Linkerd** (This talk)

Oct 14

- Virtual AMA (Buoyant booth)
- Virtual office hours

Oct 15

- **Real-time Kubernetes: How Entain Australia 10x'd throughput with Linkerd**.

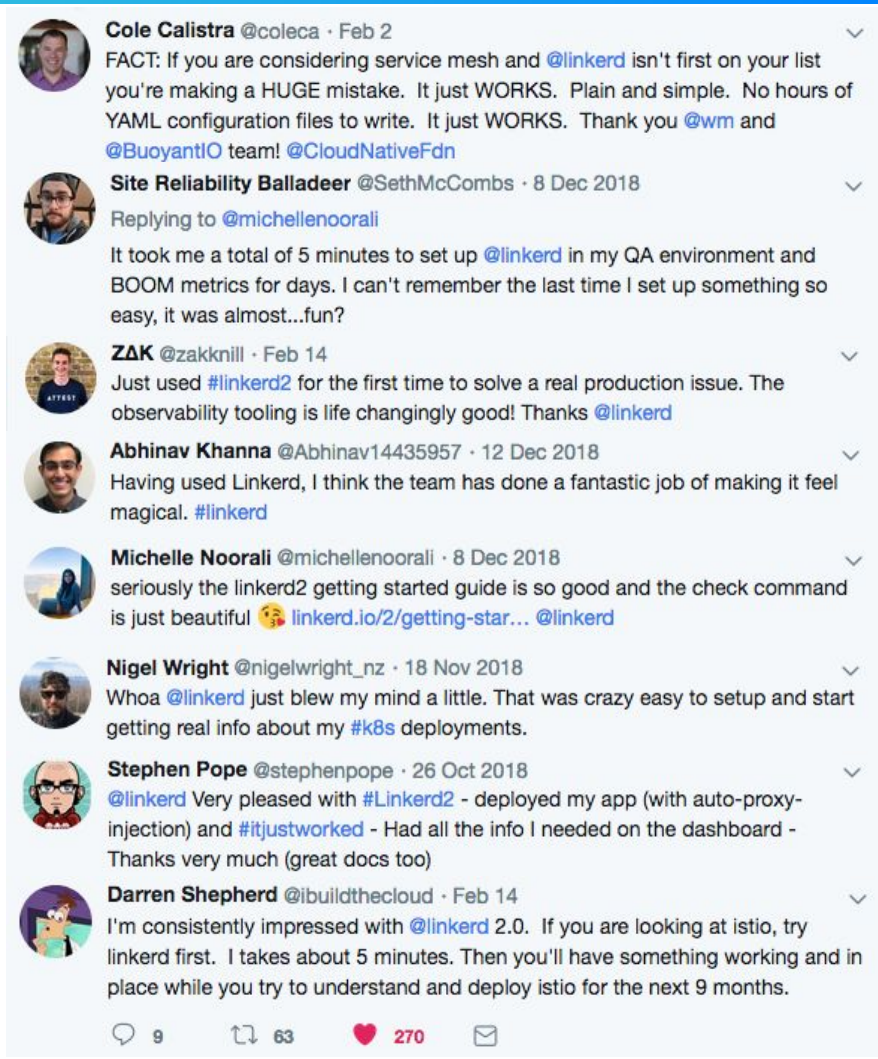
LINKERD

Get involved!

- 💚 Development is all on GitHub
- 💚 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 open governance.



Cole Calistra @coleca · Feb 2
FACT: If you are considering service mesh and @linkerd isn't first on your list you're making a HUGE mistake. It just WORKS. Plain and simple. No hours of YAML configuration files to write. It just WORKS. Thank you @wm and @BuoyantIO team! @CloudNativeFdn

Site Reliability Balladeer @SethMcCombs · 8 Dec 2018
Replying to @michellenoorali
It took me a total of 5 minutes to set up @linkerd in my QA environment and BOOM metrics for days. I can't remember the last time I set up something so easy, it was almost...fun?

ΖΔΚ @zakknill · Feb 14
Just used #linkerd2 for the first time to solve a real production issue. The observability tooling is life changingly good! Thanks @linkerd

Abhinav Khanna @Abhinav14435957 · 12 Dec 2018
Having used Linkerd, I think the team has done a fantastic job of making it feel magical. #linkerd

Michelle Noorali @michellenoorali · 8 Dec 2018
seriously the linkerd2 getting started guide is so good and the check command is just beautiful 😘 linkerd.io/2/getting-star… @linkerd

Nigel Wright @nigelwright_nz · 18 Nov 2018
Whoa @linkerd just blew my mind a little. That was crazy easy to setup and start getting real info about my #k8s deployments.

Stephen Pope @stephenpope · 26 Oct 2018
@linkerd Very pleased with #Linkerd2 - deployed my app (with auto-proxy-injection) and #itjustworked - Had all the info I needed on the dashboard - Thanks very much (great docs too)

Darren Shepherd @ibuildthecloud · Feb 14
I'm consistently impressed with @linkerd 2.0. If you are looking at istio, try linkerd first. I takes about 5 minutes. Then you'll have something working and in place while you try to understand and deploy istio for the next 9 months.

9 63 270

Linky

LINKERD