KubeCon | CloudNativeCon

Europe 2023

Emissary-ingress: Self-Service APIs and the Kubernetes Gateway API

Lance Austin

Principal Engineer

ambassador

Flynn

Technical Evangelist

BUOYANT

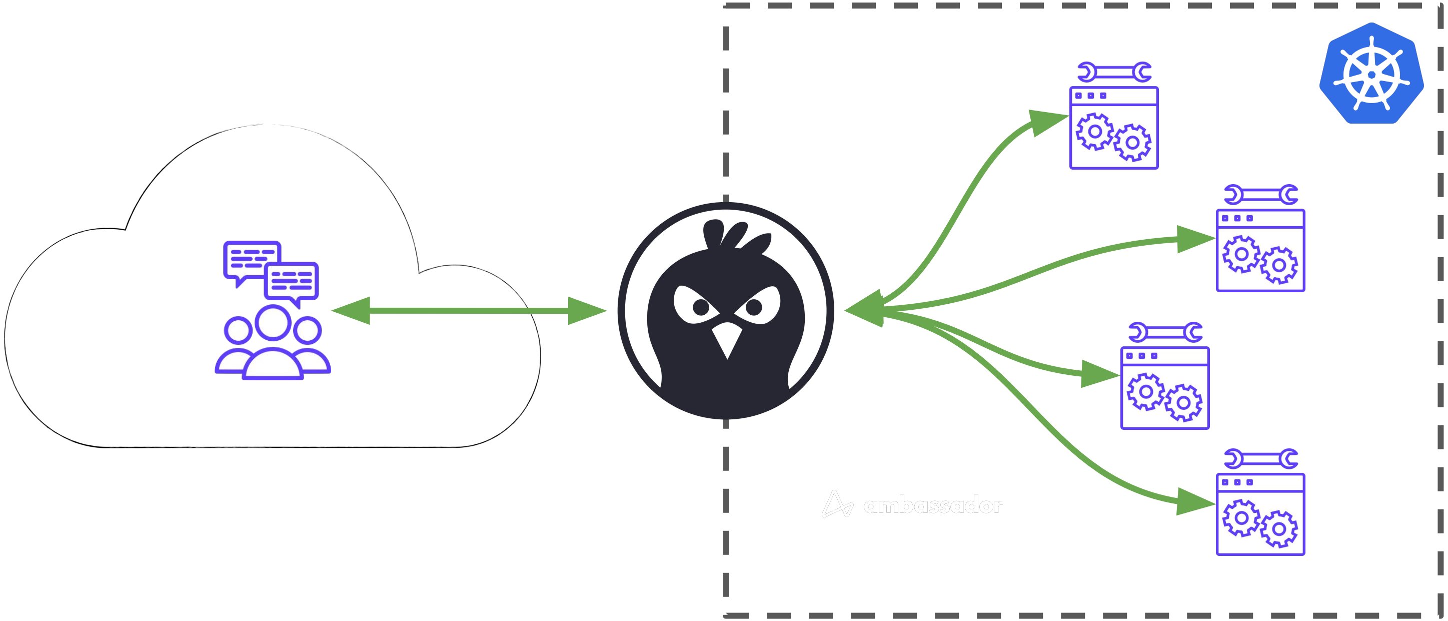
Emissary-ingress: Self-Service APIs and the Kubernetes Gateway API

- Intro & Recap
- Self-Service Configuration
- Gateway API

(Quick) Intro

EMISSARY INGRESS

Emissary-ingress is an API Gateway

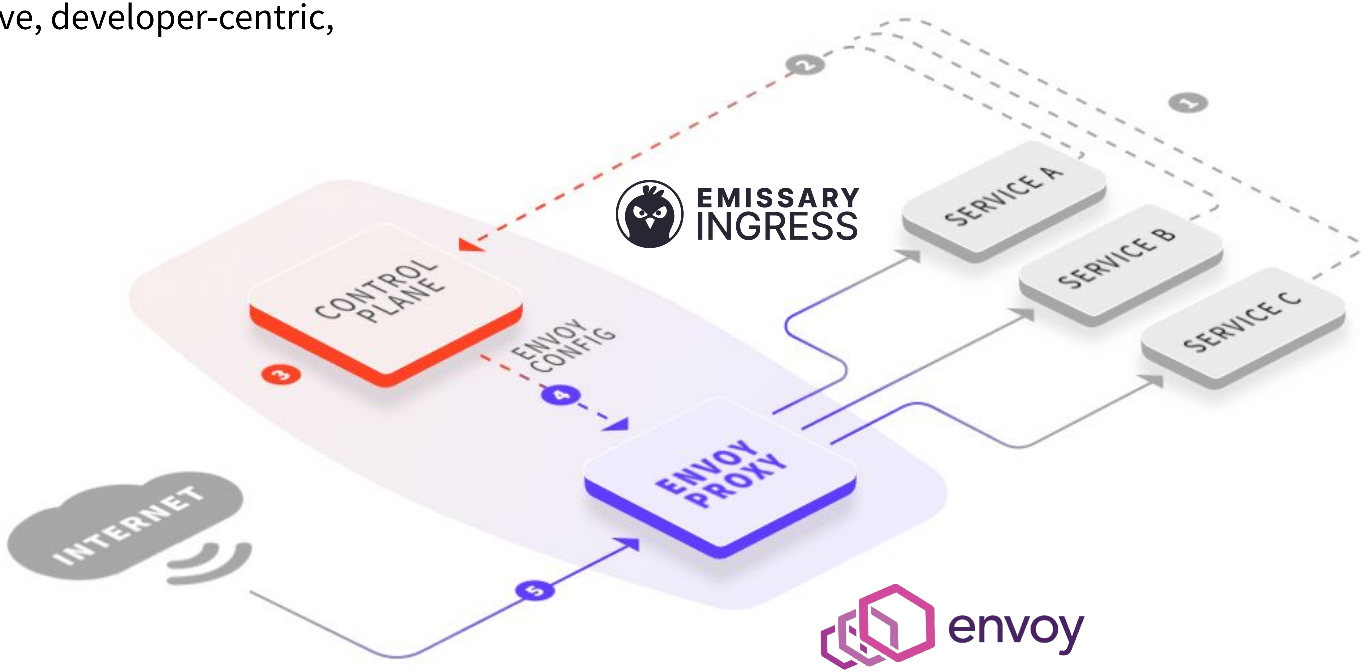


Emissary-ingress is powered by Envoy

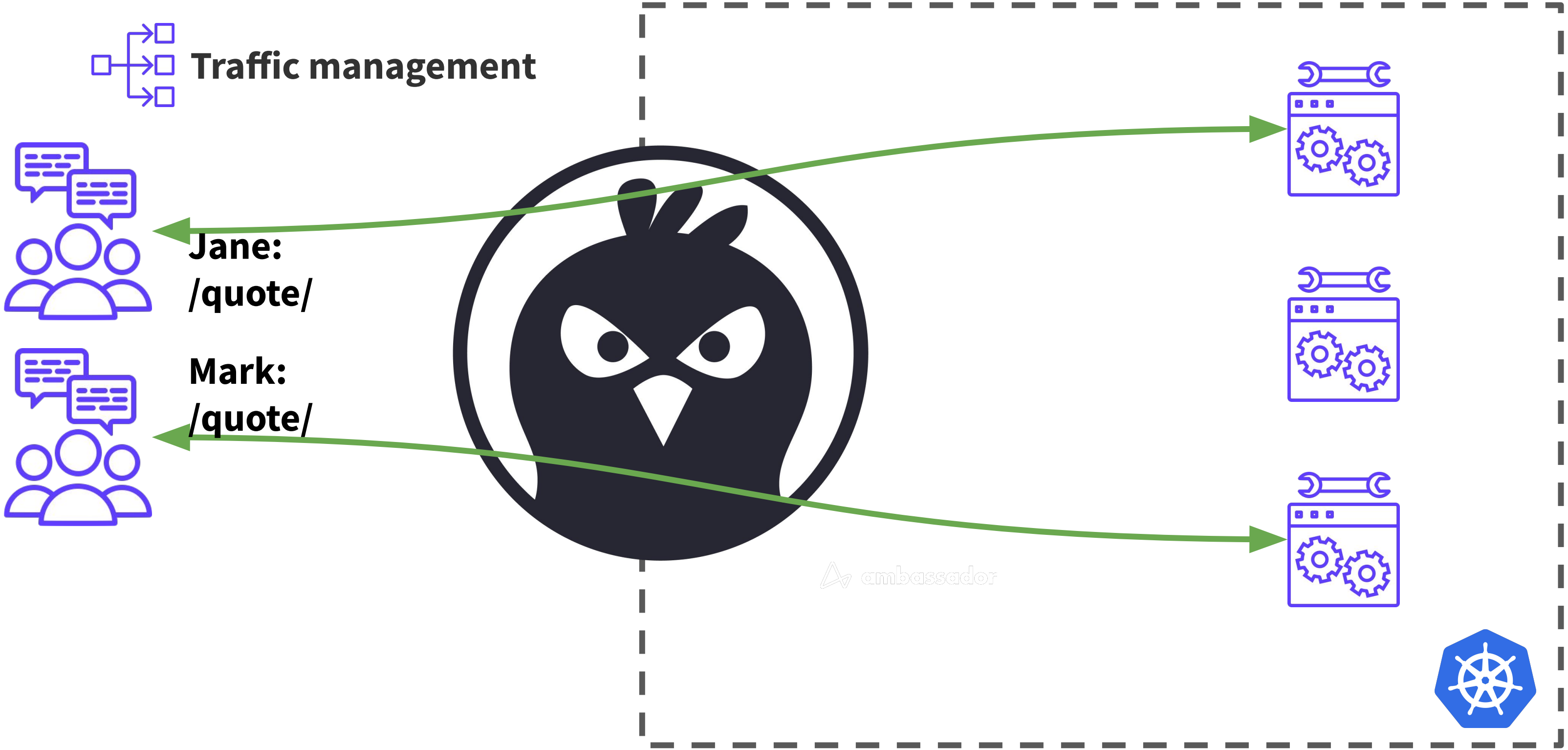
Open-source, cloud-native, developer-centric, self-service API Gateway

CNCF Incubating project

Powered by Envoy



Emissary-ingress is an API Gateway

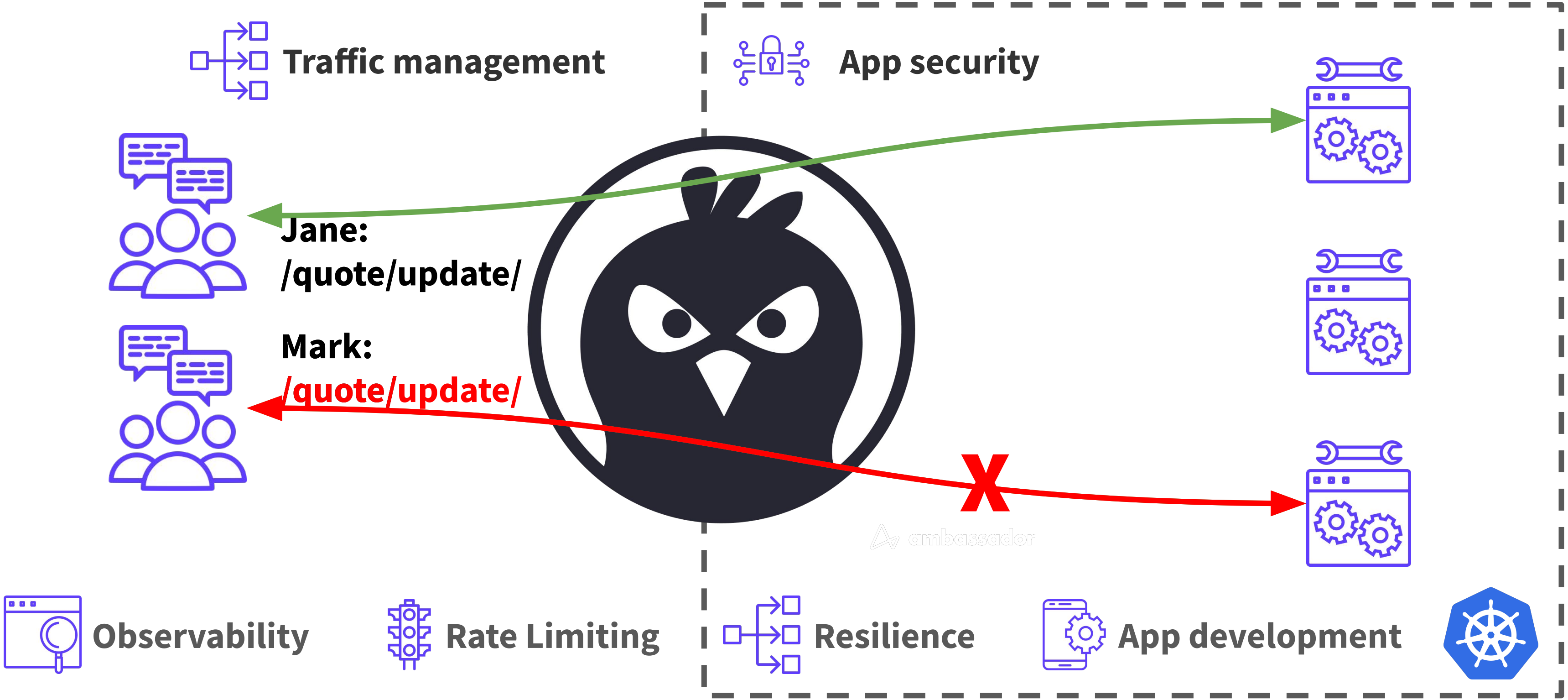


Traffic management

Jane:
/quote/

Mark:
/quote/

Emissary-ingress is not just a proxy



Traffic management

App security

Jane:
/quote/update/

Mark:
/quote/update/

Observability

Rate Limiting

Resilience

App development

Emissary Core Features

Resilience

- Advanced load balancing
- Circuit breakers
- Automatic retries
- Timeouts
- Auth (via ext_authz)
- Rate limiting (via rls.proto)

Observability

- Distributed tracing
- Real-time metrics
- Logs (+Envoy output)

Cloud-native

- Service discovery with Kubernetes and Consul
- Zero downtime configuration
- Stateless high availability architecture

L7 support

- HTTP/1.0, HTTP/1.1, HTTP/2, HTTP/3
- gRPC, gRPC-Web
- TCP, WebSockets
- Server Name Indication (SNI)
- Gzip compression

Self-Service Opinionated
Developer-Centric API Gateway

Recap

EMISSARY INGRESS

Recap: Since Kubecon Detroit

- New releases!
  - 3.4, 3.5, and 3.6: new features, better security, easier migrations
  - 2.5: critical fixes
- Major dependency updates:
  - Envoy 1.25.4
  - Golang 1.20.3

Recap: Since Kubecon Detroit

- Key new Emissary 3 features:
    - OpenTelemetry Tracing (thanks, @psalaberria002!)
    - Resolve named ports in Ingress (thanks, @antonu17!)
    - Direct TLS support for Ingress (thanks, @olemarkus!)
    - Non-blocking ready listener (thanks, @fabrice & @tomas!)
    - Active upstream health checks
    - `getambassador.io/v1` => reduced migration friction

Thanks to our Community!

175 Commits

10
First-Time Contributors

~9000
Members on Slack

4K
Github Stars

Shout out!

- Pier
- Paul Salaberria
- Anton Ustyuzhanin
- Dmitry Golushko
- Tomas
- Fabrice

Self-Service Configuration

EMISSARY INGRESS

Self-Service Developer-Centric Configuration

EMISSARY INGRESS

Why do we talk about this every time?

- Kubernetes is a **means**, not an **end**:
    - Developers have goals beyond just “run Kubernetes”.
- “Self-service” is about achieving those goals **faster**.
- “Developer-centric” is about achieving those goals **more easily**.
- “Opinionated” is about achieving those goals **in a straightforward way**.

Configuring the Control Plane

```
apiVersion: getambassador.io/v3alpha1
kind: Listener
metadata:
  name: listener-8443
spec:
  port: 8443
  securityModel: XFP
  hostBinding:
    selector:
      matchLabels:
        my-listener: example
```

```
apiVersion: getambassador.io/v3alpha1
kind: Host
metadata:
  name: example-host
  labels:
    my-listener: examp
spec:
  hostname: foo.exampl
  tlsSecret:
    name: sslexample-c
```

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
metadata:
  name: quote-mapping
spec:
  hostname: foo.example.com
  prefix: /quote/
  service: quote
```

Configuring the Control Plane

```
apiVersion: getambassador.io/v3alpha1
kind: Listener
metadata:

sp
```

```
apiVersion: getambassador.io/v3alpha1
kind: Host
metadata:

sp
```

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
metadata:
  name: quote-mapping
spec:
  prefix: /quote/
  service: quote
```

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    kubernetes.io/ingress.class: ambassador
  name: example-ingress
spec:
  tls:
  - hosts:
    - foo.example.com
    secretName: sslexample-cert
  rules:
  - host: foo.example.com
    http:
      paths:
      - path: /quote/
        backend:
          serviceName: quote
          servicePort: 80
```

Self-Service Configuration

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
me
sp
```

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
meta
  na
spec
  pr
  se
  we
```

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
metadata:
  name: restricted-mapping
spec:
  host: restricted.example.com
  prefix: /restricted/
  rewrite: /a/very/safe/path/
  rewrite_host: safe.example.com
  ervice: dangerous-service
```

```
apiVersion: getambassador.io/v3alpha1
kind: Listener
me
sp
```

```
apiVersion: getambassador.io/v3alpha1
kind: Host
me
spe
```

```
apiVersion: getambassador.io/v3alpha1
kind: AuthService
metadata:
  name: extauth-service
spec:
  auth_service: example-auth
  path_prefix: “/extauth”
  allowed_request_headers:
    – “x-example-session”
  allowed_authorization_headers:
    – “x-example-session”
    – “x-example-userid”
```

Separation of Concerns

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
metadata:
sp
```

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
meta
  na
spec
  pr
  se
  we
```

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
metadata:
  name: restricted-mapping
spec:
  host: restricted.example.com
  prefix: /restricted/
  rewrite: /a/very/safe/path/
  rewrite_host: safe.example.com
   ervice: dangerous-service
```

```
apiVersion: getambassador.io/v3alpha1
kind: Listener
metadata:
sp
```

```
apiVersion: getambassador.io/v3alpha1
kind: Host
metadata:
spe
```

```
apiVersion: getambassador.io/v3alpha1
kind: AuthService
metadata:
  name: extauth-service
spec:
  auth_service: example-auth
  path_prefix: “/extauth”
  allowed_request_headers:
      “x-example-session”
  ed_authorization_headers:
      x-example-session”
      x-example-userid”
```

Best Practices

EMISSARY INGRESS

Self-Service Configuration Best Practices

Self-service requires trust.

- Maximum benefit with independent teams doing independent releases (microservices 101)
- Maximum benefit when developers don’t need to be stuck behind ops

Trust goes both ways.

- Developers benefit from trusting ops to handle infrastructure.
  - TLS, authentication, rate limiting, circuit breaking, etc.
- Ops benefits from trusting developers to handle applications.
  - Routes, URLs, etc.
- Both benefit from trusting the other to act in good faith.

Self-Service Configuration Best Practices

Trust doesn’t have to be blind.

- Kubernetes RBAC can give developers guardrails.
  - https://github.com/postfinance/kubectl-sudo
- It’s easy to let everyone audit configurations.
  - kubectl get -o yaml
- GitOps / infrastructure-as-code can add control points to configuration
  - Submit a pull request which requires review and approval
- CD tools can help here too.
  - Automatically validate configurations before rollout

Gateway API

EMISSARY
INGRESS

What is the Gateway API?

- Standardized set of resources for service networking
- Successor to Ingress resource
- Community driven
- Role-oriented resources
- Taking learnings from projects like Emissary-ingress, Contour and the wider Kubernetes ecosystem

https://gateway-api.sigs.k8s.io/

How does it compare?

```
apiVersion: getambassador.io/v3alpha1
kind: Listener
metadata:
```

```
apiVersion: getambassador.io/v3alpha1
kind: Host
metadata:
```

```
apiVersion: getambassador.io/v3alpha1
kind: Mapping
metadata:
  name: quote-mapping
spec:
  prefix: /quote/
  service: quote
```

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: GatewayClass
```

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: Gateway
```

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: qotm-route
spec:
  parentRefs:
  - name: prod-gateway
  hostnames:
  - “foo.example.com”
  rules:
   - backendRefs:
      - name: quote
        port: 80
```

What does it look like?

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: GatewayClass
metada
  name
spec:
  contr
```

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: Gateway
metadat
  name:
spec:
  gatew
  liste
  - nam
    pro
    por
    hos
   tls
     mo
      c
      -
        name: wildcard-example-com-cert
```

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: qotm-route
spec:
  parentRefs:
  - name: prod-gateway
  hostnames:
  - “foo.example.com”
  rules:
  - backendRefs:
     - name: quote
       port: 80
```

Separation of Concerns

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: qotm-route
spec:
  parentRefs:
  - name: prod-gateway
  hostnames:
  - “foo.example.com”
  rules:
  - backendRefs:
    - name: quote
      port: 80
```

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: GatewayClass
metadata:

s
```

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: Gateway
metadata:
  name: prod-gateway
spec:
  gatewayClassName: cluster-gateway
  listeners:
  - name: prod-gw-8443
    protocol: HTTPS
    port: 8443
    hostname: "*.example.com"
    tl
           Terminate
          ificateRefs:
          nd: Secret
          oup: ""
          me: wildcard-example-com-cert
```

Emissary Configuration

Does Emissary support Gateway API?

Yes but not really…

What about Emissary **v3** CRDs?

We want to hear from you!

ContribFest: Emissary-Ingress - Bugs, Deprecations, and Features, Oh My!

Wednesday, April 19 • 16:30 - 18:00

K101-102 | First Floor | Congress Centre

Summary

EMISSARY INGRESS

Summary

- **Emissary-ingress** focuses on self-service ingress because it's a great way to let everyone get things done faster.
  - It takes trust, but it works really well.
- It makes sense to start looking at the Gateway API at this point.
  - We'll keep supporting the Emissary-ingress input language.
  - We need your feedback on where the Gateway API will work - or not - for you.

Emissary-ingress and the Community

- Emissary-ingress would never have made it this far without the amazing community that's grown up around it.
- First-ever Emissary ContribFest is happening later today, and we'd love to see you there: 4:30 PM in room K101 (first floor).
- To get involved with development or ask questions, join us on Slack at a8r.io/slack (@Alice Wasko, @flynn, @Lance Austin).

TAG

Contributor Strategy

Creating a **cross-project community of maintainers** to connect, exchange ideas, and support each other!

- → Maintainer’s Circle
- → Templates, how-to guides, and other resources
- → Guidance on engaging with contributors
- → And more!

👉 Learn about the TAG and how to participate
(cncf.io/contributor-strategy)

Thanks!

Lance Austin

Principal Engineer

Ambassador Labs

laustin@datawire.io

@LanceAustinDEV

Flynn

Technical Evangelist

Buoyant

flynn@buoyant.io

@flynn on Emissary Slack
(https://a8r.io/slack)