#argocon

Demystifying Argo Events:

An Architectural Deep Dive

J.P. Zivalich
CTO & Founder, Pipekit
jp@pipekit.io

Becky Pauley
Solutions Engineer, Jetstack Consult
becky.pauley@venafi.com

Image credit: Phil Botha, Unsplash

What is Argo Events? What's in a name?

Argo Workflows

Kubernetes-native workflow engine supporting DAG and step-based workflows

Argo CD

Declarative continuous delivery with a fully-loaded UI

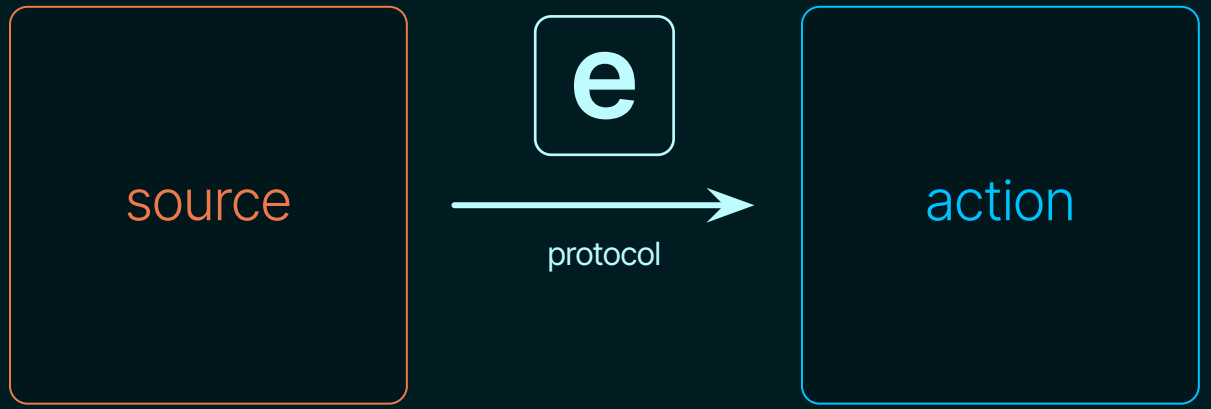
Argo Events

Event based dependency management for Kubernetes.

Argo Rollouts

Advanced Kubernetes deployment strategies such as Canary and Blue-Green made easy.

An event is...(an action and context)



Source:
https://github.com/cloudevents/spec/blob/main/cloudevents/primer.md

An event is...(an action and context)

An event has two parts:

1. an action, and
2. its context

Sent from an event producer (the source) to event consumers

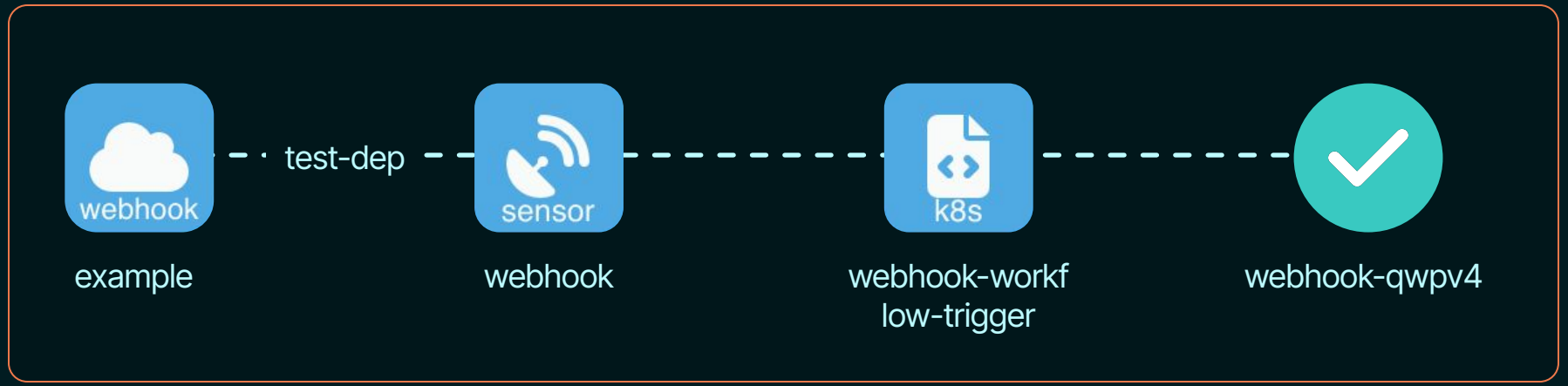
Unopinionated - Routing can be based on information in the event, but an event will not identify a specific routing destination.

```
{
   "context": {
     "type": "type_of_event_source",
     "specversion": "cloud_events_version",
     "source": "name_of_the_event_source",
     "id": "unique_event_id",
     "time": "event_time",
     "datacontenttype": "type_of_data",
     "subject": "name_of_the_configuration_within_event_source"
   },
   "data": {
       "body": "Body is the Github event data",
       "headers": "Headers from the Github event",
   }
}
```

What is Argo Events? (definition)

Argo Events is an event-driven workflow automation framework.

The Argo Events UI lives within Argo Workflows



What is Argo Events? (use cases)

CI/CD Pipelines

Event-based data processing

General Automation

What is Argo Events? (components)

4

components

Event Source

Describes configuration for consuming events FROM external sources

Event Bus

A transport layer that connects the Event Source to the Sensor.

May or may not be driven by Ms. Frizzle

Sensor

Defines the connection between the Event Source and the resource being triggered

Controller Manager

Observes creation/changes to Event Source, Event Bus and Sensor custom resources.

Installing Argo Events... in a few steps

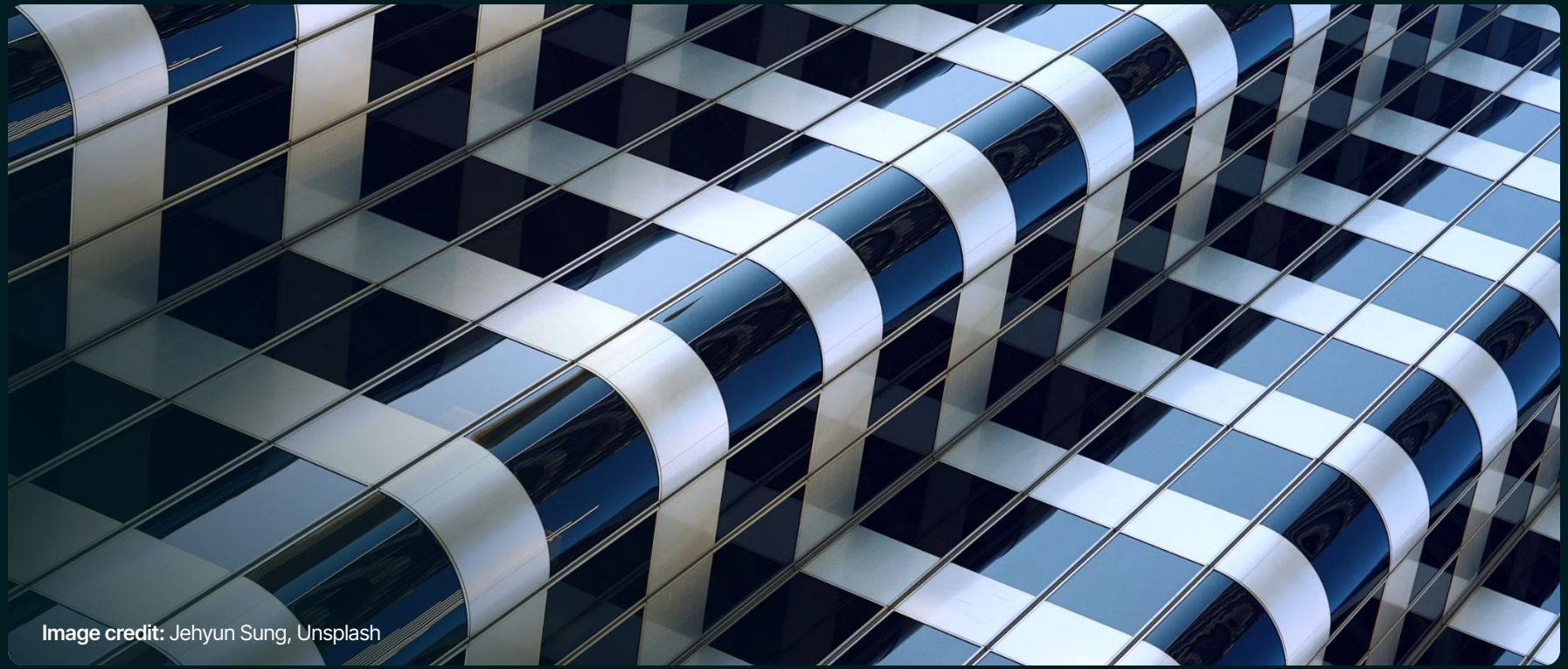


Image credit: Jehyun Sung, Unsplash

What exists in my cluster?

```
customresourcedefinition.apiextensions.k8s.io/eventbus.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/eventsources.argoproj.io created
customresourcedefinition.apiextensions.k8s.io/sensors.argoproj.io created
serviceaccount/argo-events-sa created
clusterrole.rbac.authorization.k8s.io/argo-events-aggregate-to-admin created
clusterrole.rbac.authorization.k8s.io/argo-events-aggregate-to-edit created
clusterrole.rbac.authorization.k8s.io/argo-events-aggregate-to-view created
clusterrole.rbac.authorization.k8s.io/argo-events-role created
clusterrolebinding.rbac.authorization.k8s.io/argo-events-binding created
configmap/argo-events-controller-config created
deployment.apps/controller-manager created
```

CRDs

Controller Manager

Custom Resource Definitions

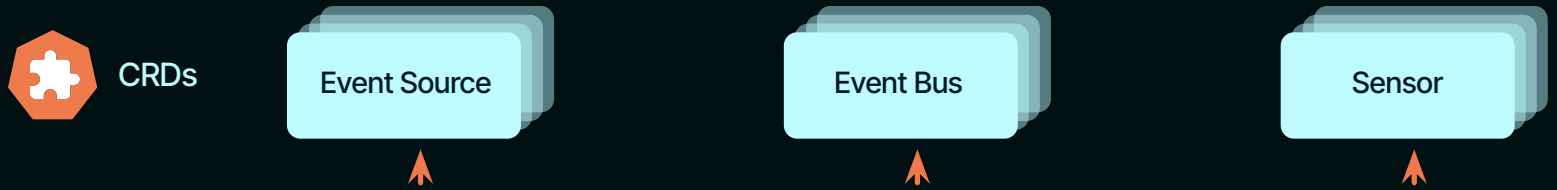
```yaml
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: eventsources.argoproj.io
spec:
  group: argoproj.io
  names:
    kind: EventSource
    listKind: EventSourceList
    plural: eventsources
    shortNames:
      - es
    singular: eventsource
  scope: Namespaced
  versions:
    - name: v1alpha1
      schema:
        openAPIV3Schema:
          properties:
            apiVersion:
              type: string
            kind:
              type: string
            metadata:
              type: object
            spec:
              type: object
```

CRDs

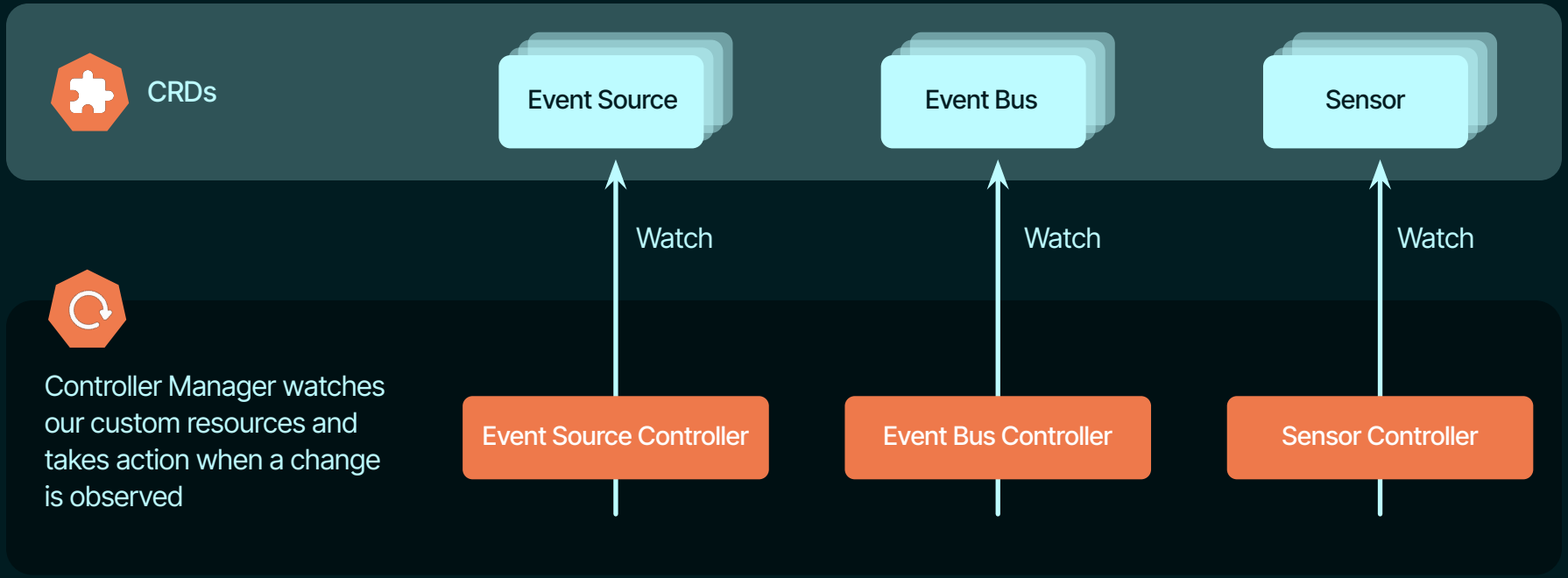
```yaml
apiVersion: argoproj.io/v1alpha1
kind: EventSource
metadata:
  name: webhook
spec:
  service:
    ports:
      - port: 12000
        targetPort: 12000
  webhook:
    example:
      # port to run HTTP server on
      port: "12000"
      # endpoint to listen to
      endpoint: /example
      # HTTP request method to allow.
      # In this case, only POST requests are accepted
      method: POST
```

Custom Resource Definitions

```
NAME
eventbus.argoproj.io
eventsources.argoproj.io
sensors.argoproj.io
```



Controller Manager



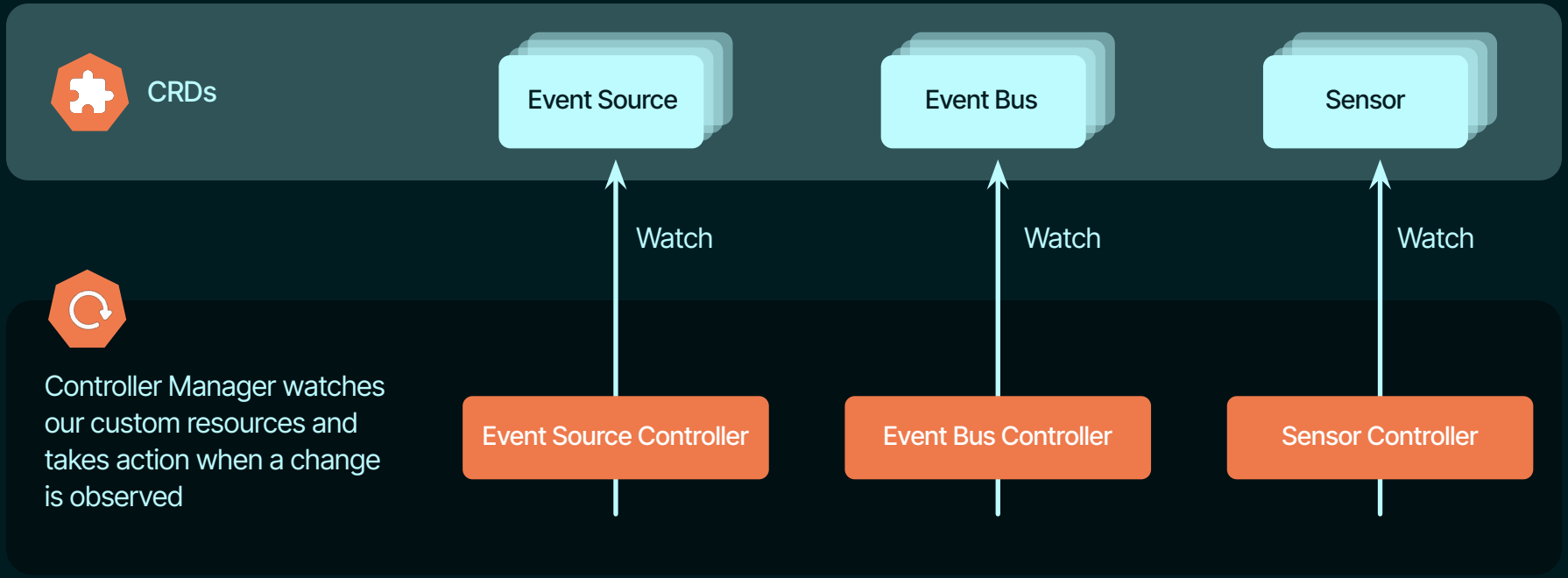
The case of the missing EventBus...

```
kubectl get eventbus -n argo-events
No resources found in argo-events namespace.

kubectl apply -n argo-events -f
https://raw.githubusercontent.com/argoproj/arg
o-events/stable/examples/eventbus/native.yaml
```

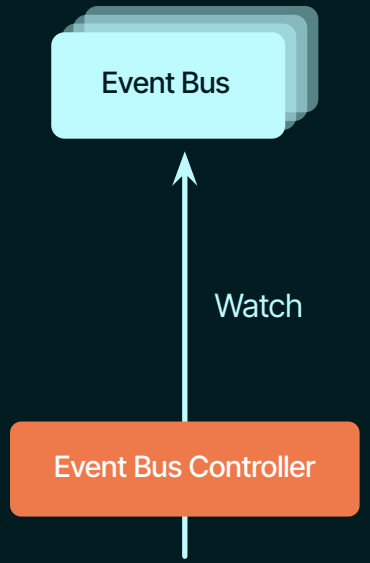
```
apiVersion: argoproj.io/v1alpha1
kind: EventBus
metadata:
  name: default
spec:
  nats:
    native:
      replicas: 3
      auth: token
```

Controller Manager

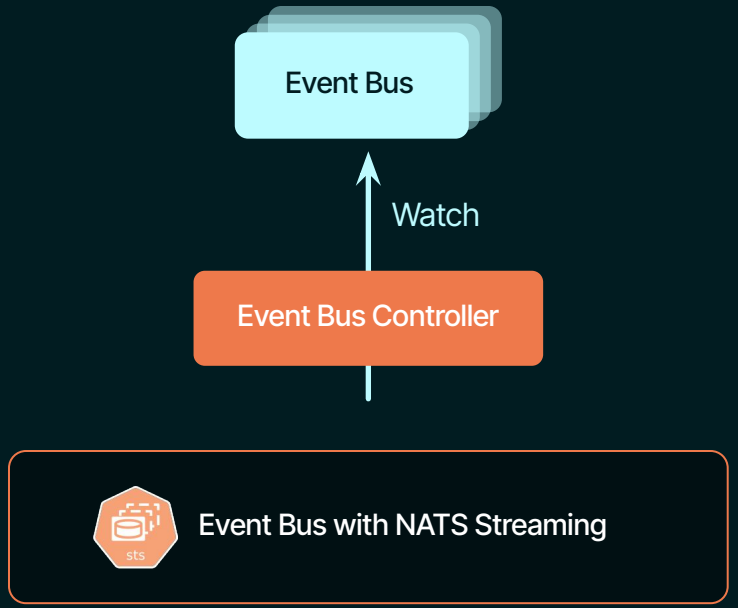


The case of the missing EventBus...


```
{“level”:“info”,“ts”:1725264481.6412816,“logger”:“argo-events.eventbus-controller.nats”
,“caller”:“installer/nats.go:180”,“msg”:“Service is
created”,“namespace”:“argo-events”,“eventbus”:“default”,“serviceName”:“eventbus-default
-stan-svc”}
{“level”:“info”,“ts”:1725264481.6484525,“logger”:“argo-events.eventbus-controller.nats”
,“caller”:“installer/nats.go:221",“msg”:“Created
configmap”,“namespace”:“argo-events”,“eventbus”:“default”,“configmapName”:“eventbus-def
ault-stan-configmap”}
{“level”:“info”,“ts”:1725264481.6610892,“logger”:“argo-events.eventbus-controller.nats”
,“caller”:“installer/nats.go:314”,“msg”:“Created server auth
secret”,“namespace”:“argo-events”,“eventbus”:“default”,“serverAuthSecretName”:“eventbus
-default-server”}
{“level”:“info”,“ts”:1725264481.6640878,“logger”:“argo-events.eventbus-controller.nats”
,“caller”:“installer/nats.go:343",“msg”:“Created client auth
secret”,“namespace”:“argo-events”,“eventbus”:“default”,“clientAuthSecretName”:“eventbus
-default-client”}
{“level”:“info”,“ts”:1725264481.6714368,“logger”:“argo-events.eventbus-controller.nats”
,“caller”:“installer/nats.go:399”,“msg”:“Statefulset is
created”,“namespace”:“argo-events”,“eventbus”:“default”,“statefulsetName”:“eventbus-def
ault-stan”}
```

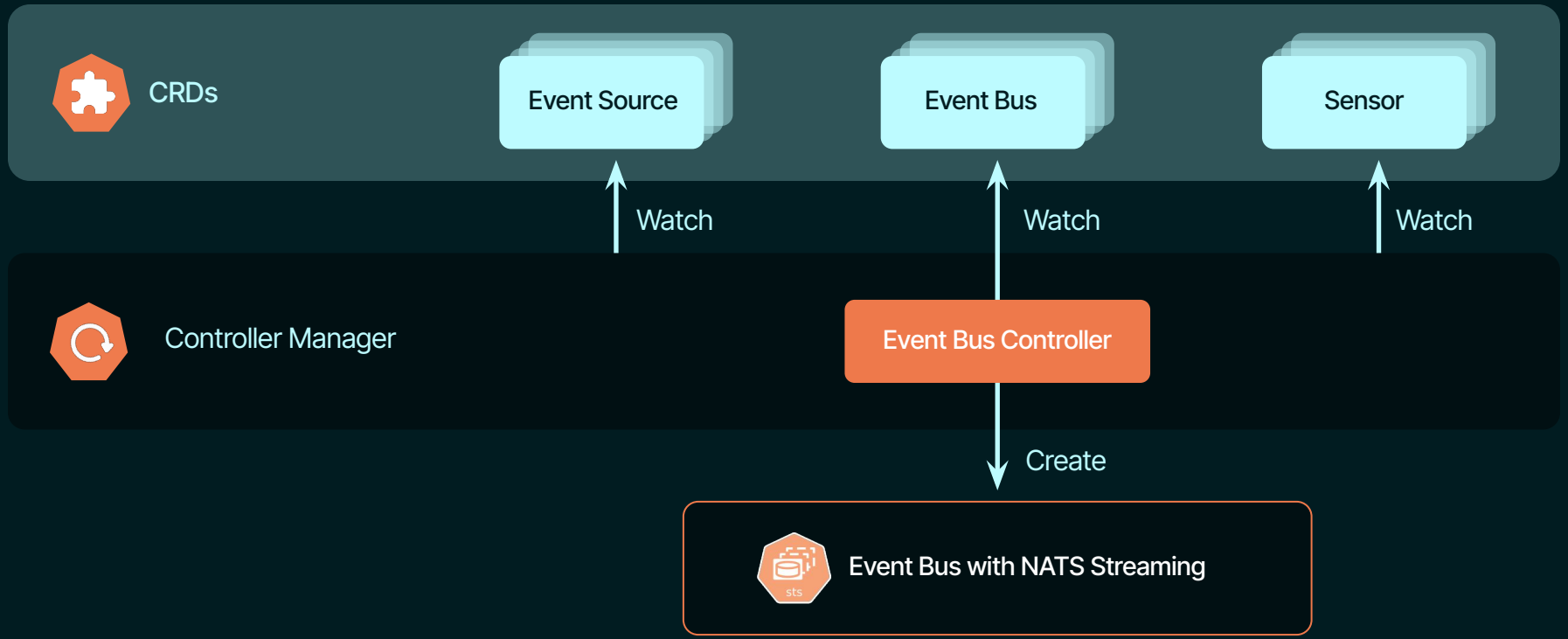



The case of the missing EventBus...



bitly

Recap of where we are in the install

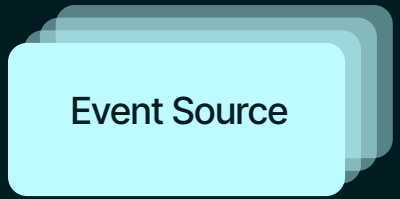


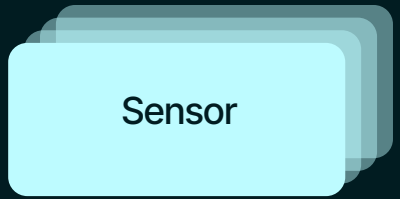
The Life of an Event - from JSON blob to tests

Workflow as test runner - parameters from the event e.g. PR ID

For each event...

We must define:





Controller Manager

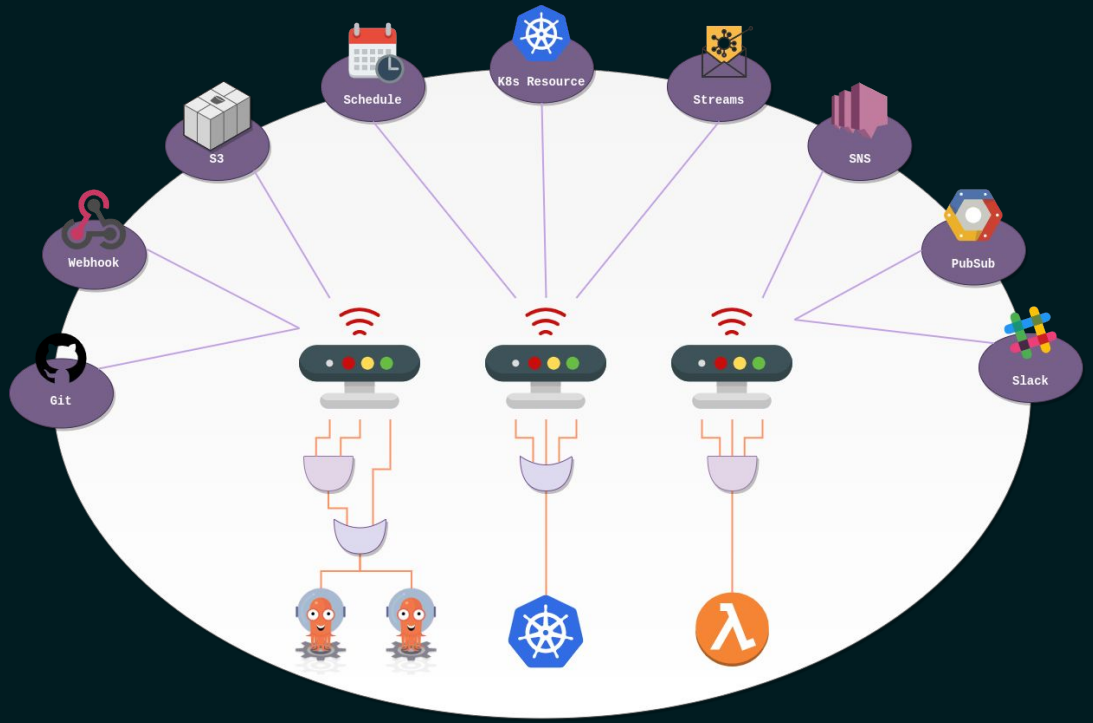
Where do I see Events?

Argo Workflows

Kubernetes-native workflow engine supporting DAG and step-based workflows

1. The Life of an Event: Before it reaches Argo Events

Something **produces** an event.



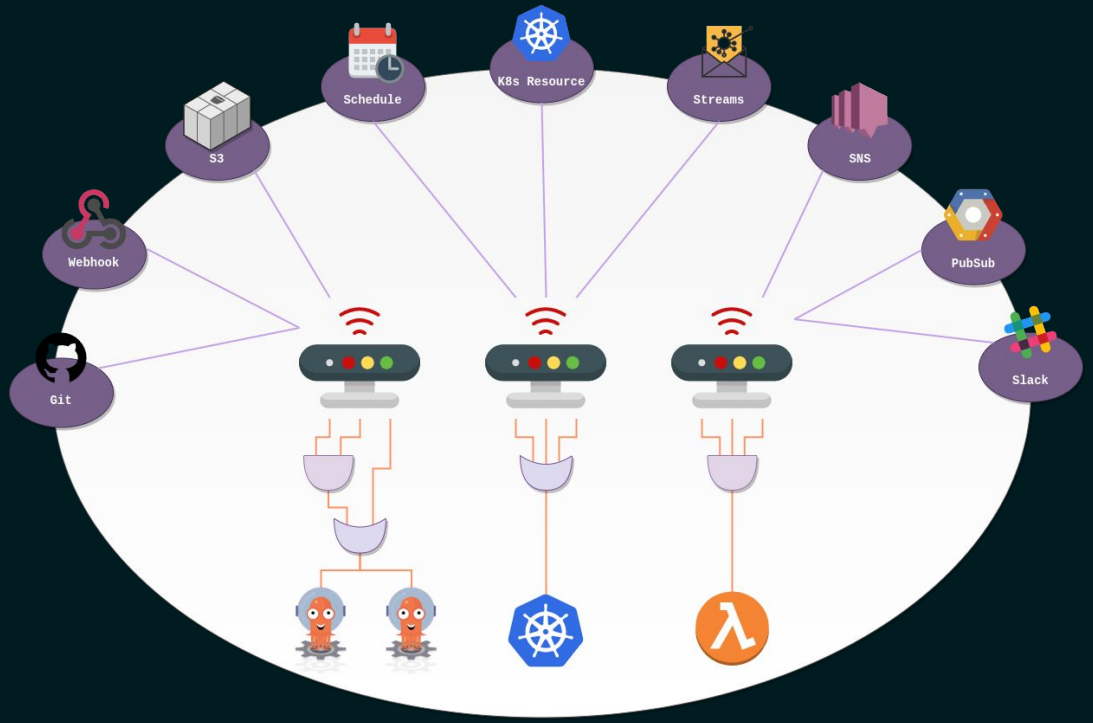
2. The Life of an Event: Event Source

Something produces an event.

We configure an event source so argo-events to be able to consume that event.

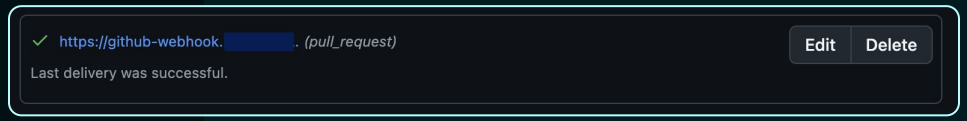
The Event Source:

- transforms the event into a standard format (see cloudevents)
- dispatches the event onto the eventbus.

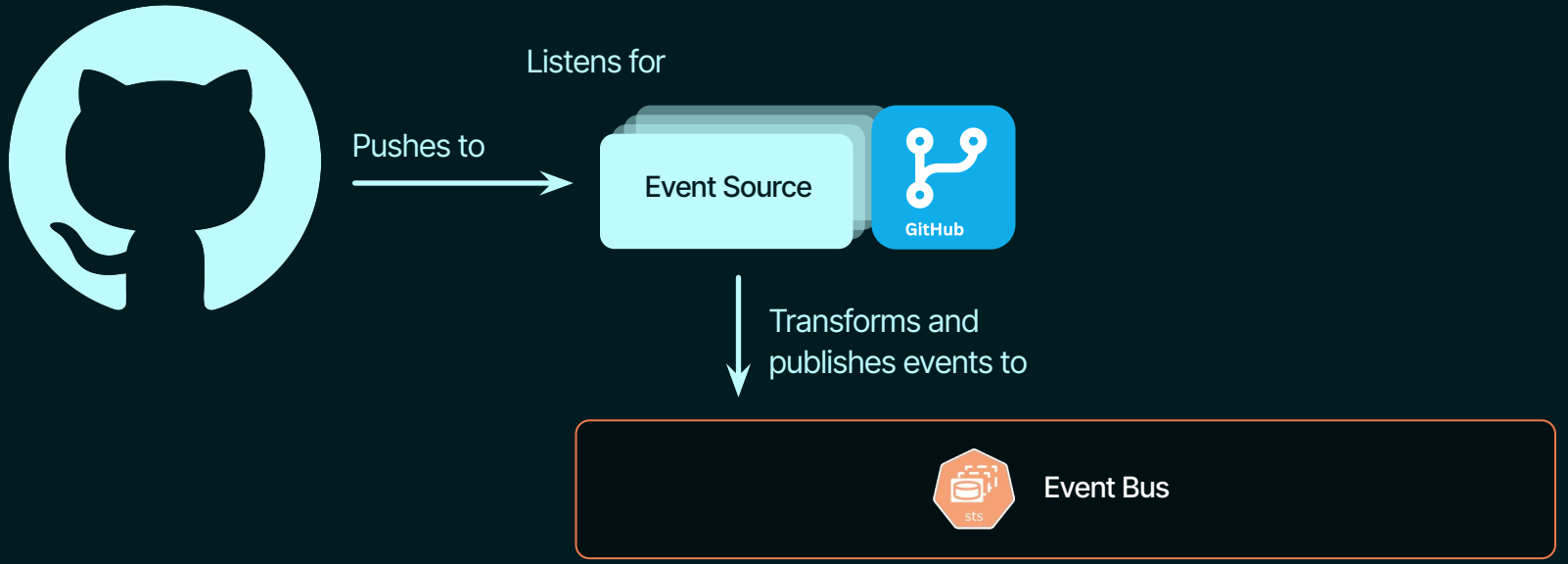


2. The Life of an Event: Event Source

```yaml
apiVersion: argoproj.io/v1alpha1
kind: EventSource
metadata:
 name: github-event-source
 namespace: argo-events
Spec:
 # this can be omitted if using the default eventbus
 eventBusName: default
 github:
   argo-events-example:
     owner: BeckyPauley
     repository: github-argo-events-demo
     events:
     - pull_request
     webhook:
       endpoint: /pr
       port: "12000"
       method: POST
       url: https://github-webhook.my-domain.com ...
```



3. The Life of an Event: Event Bus



4. The Life of an Event: Sensor - Dependencies

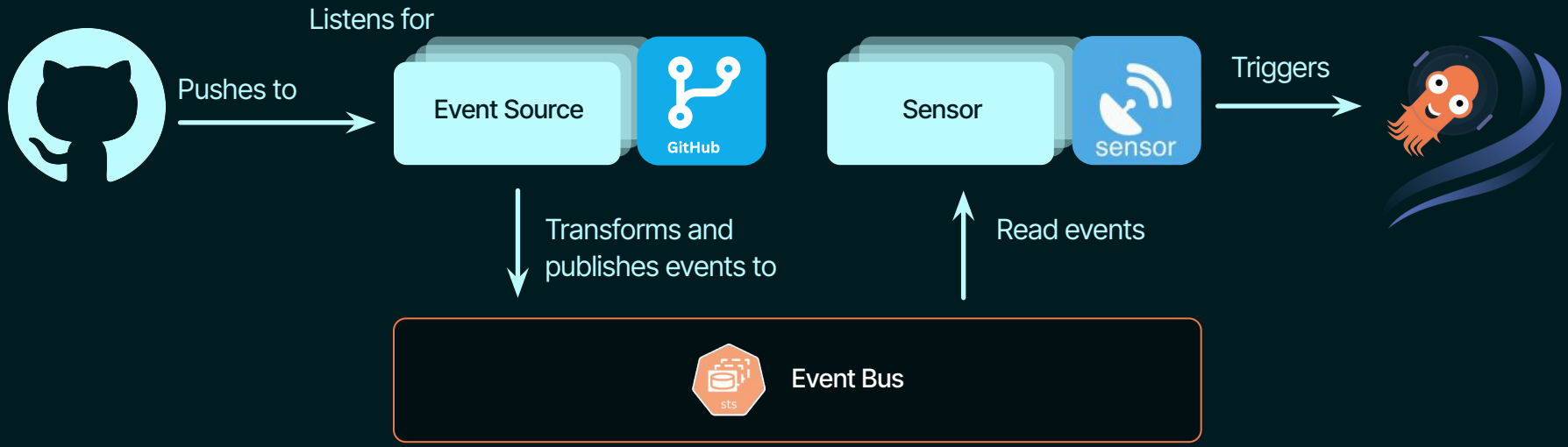
```
apiVersion: argoproj.io/v1alpha1
kind: Sensor
metadata:
 name: github-pr-sensor
 namespace: argo-events
spec:
 template:
   serviceAccountName: operate-workflow-sa
 dependencies:
   - name: pr
     eventSourceName: github-event-source
     eventName: github-event
```

Note: we *could* use filters here.
See: https://argoproj.github.io/argo-events/sensors/filters/data

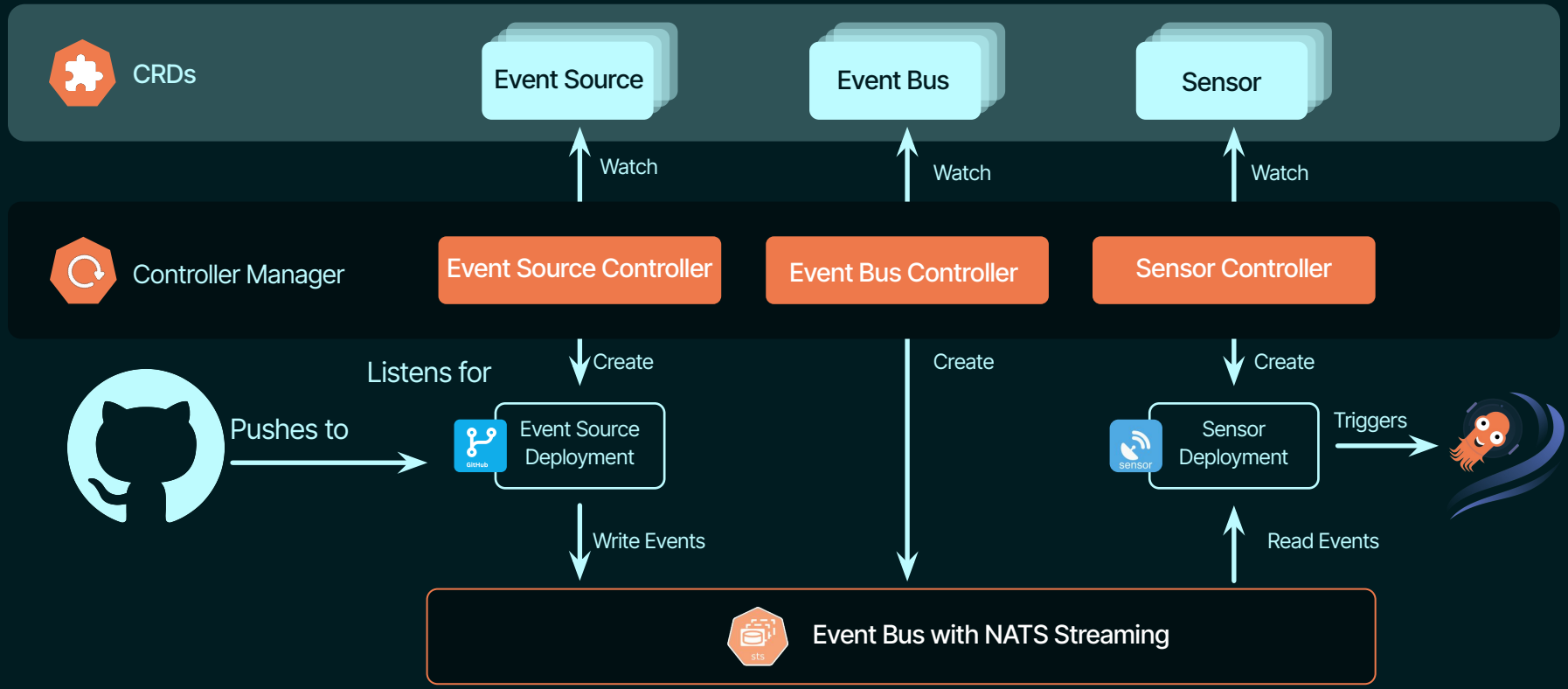
4. The Life of an Event: Sensor - Triggers

```yaml
triggers:
  - template:
      name: github-workflow-trigger
      # NB: We could use argoWorkflows or k8s trigger type here
      k8s:
        group: argoproj.io
        version: v1alpha1
        resource: workflows
        operation: create
        source:
          resource:
            apiVersion: argoproj.io/v1alpha1
            kind: Workflow
            metadata:
              name: pr-
              namespace: argo-events
              labels:
                workflows.argoproj.io/workflow-template: pr-build-and-test
            spec:
              arguments:
                parameters:
                  - name: pr_number
                    value: ''
                  - name: git_sha
                    value: ''
              workflowTemplateRef:
                name: pr-build-and-test
```

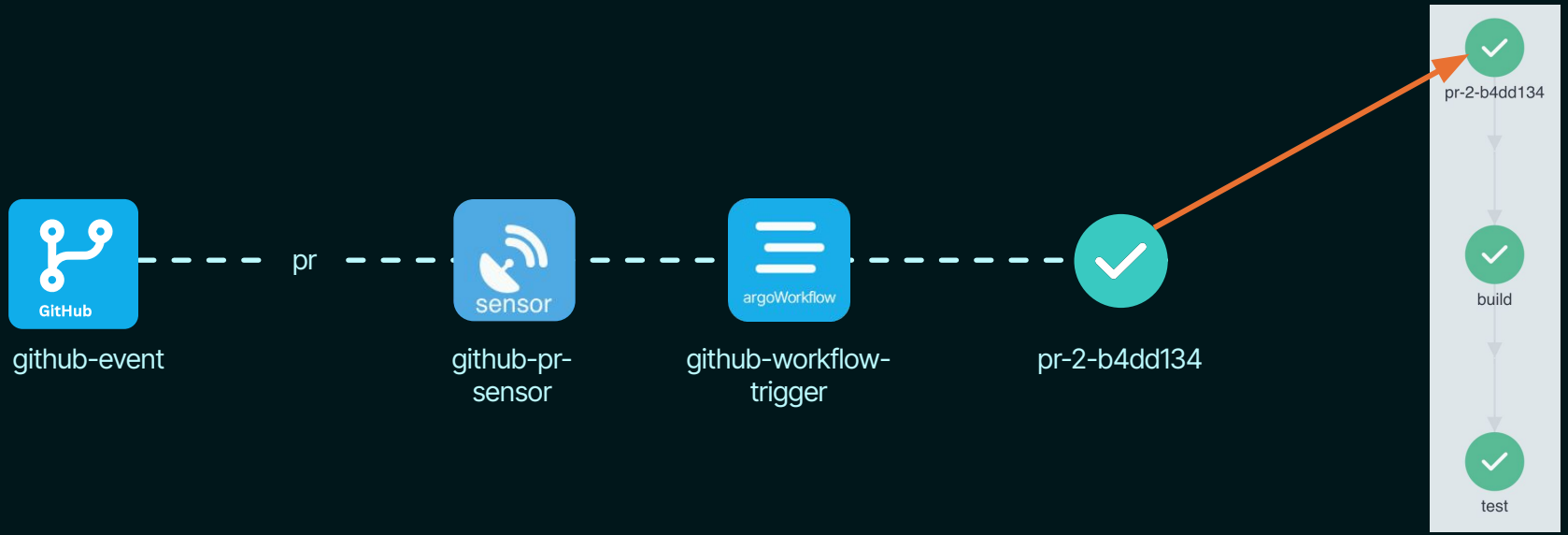
4. The Life of an Event: Sensor



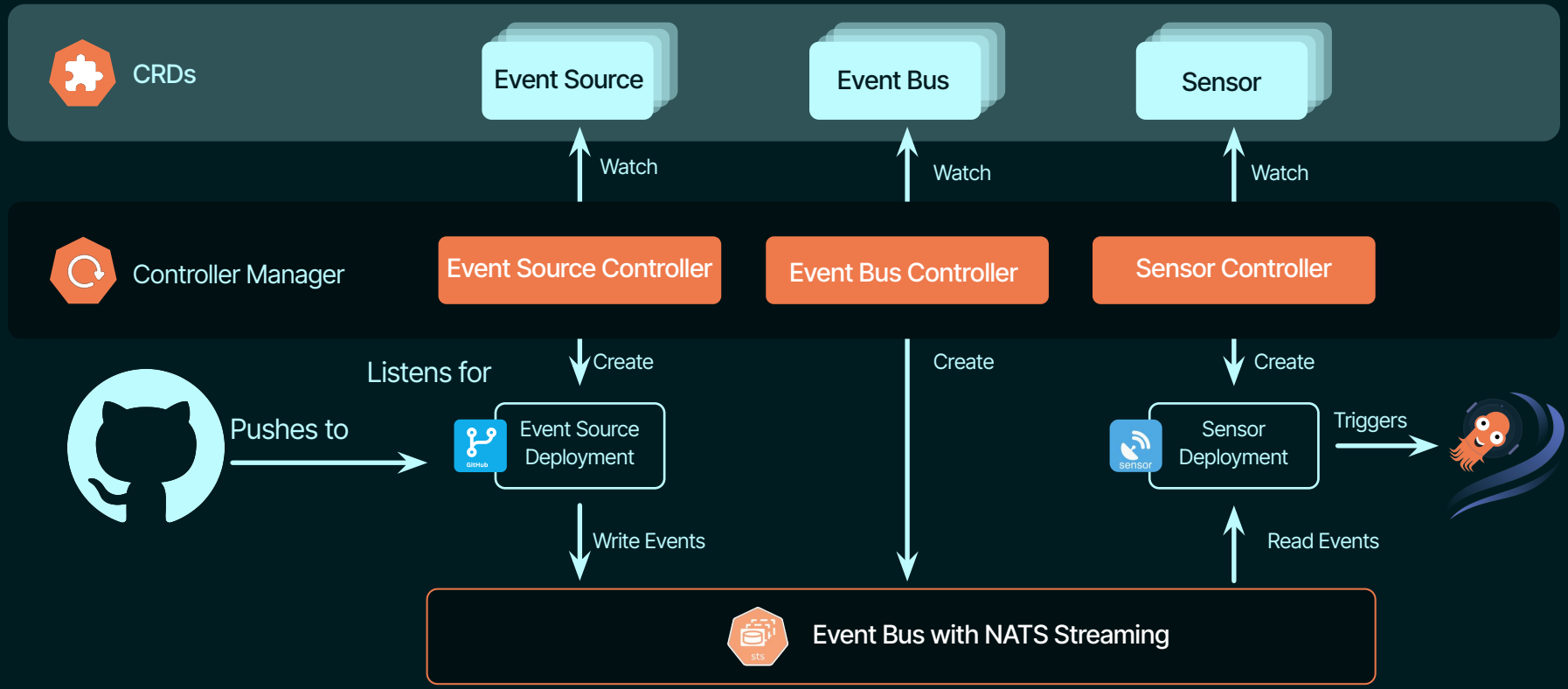
The Life of an Event: The Big Picture



The Life of an Event: Argo Workflows UI



The Even Bigger Picture



The Even Bigger Picture

Normally : one Event Bus

We could have:

- Many Event Sources → one Event Bus → many Sensors
- One Event Source for all activity in an entire GitHub org, many Sensors

Don’t know exactly what sensors you’ll need in the future?

Gather up information in your Event Source, build the Sensors you want today.

For Further Investigation

Event Bus Options

- There are three supported Event Bus options:
  - NATS
  - NATS Jetstream
  - Kafka

Validating admission webhook

- Kicks an error if your Argo Events configuration is invalid

Conclusion

- Recap of the lifecycle
- Customization options

Want to learn more?

Try the Getting Started guide and tutorials
https://argoproj.github.io/argo-events/quick_start/

Read through the docs
https://argoproj.github.io/argo-events/

GitHub

Visit GitHub (especially the examples directory)
https://github.com/argoproj/argo-events

Stay up to date with our blogs
Pipekit Blog
Cloud Native - Venafi Machine Identity Security

About Pipekit

Scale Argo & Kubernetes with Pipekit

Direct support from 40% of the active Argo Workflows maintainers in the world

Save engineering time and up to 60% on compute costs

Add 3 Argo maintainers and 7 Argo contributors to your team

Serving startups & Fortune 500 enterprises since 2021:

> **Enterprise Support for Argo:**
> Ideal for Platform Eng teams scaling with Argo

> **Control Plane for Argo Workflows:**
> Ideal for data teams, granular RBAC, and multi-cluster architectures

About Jetstack Consult

Jetstack Consult provides strategic advisory services, engineering and training to help businesses, large and small, solve five key platform engineering challenges:

- Platform optimisation
- Cloud diversification
- Infrastructure modernisation
- Innovation at scale
- Security by default

Jetstack Consult Services Overview | Venafi

Thank you

J.P. Zivalich
CTO & Founder, Pipekit
jp@pipekit.io

Becky Pauley
Solutions Engineer, Jetstack Consult
becky.pauley@venafi.com

Feedback

Q&A