ANSIBLE OPERATOR FOR KUBERNETES

SURVEY
Ansible Experience?

SURVEY
Kubernetes Experience?

SURVEY
K8s Operator Experience?

Operators are application aware Kubernetes objects.

Active throughout the application's lifecycle, they manage instantiation, ongoing state, and destruction.

FROM VISION TO PROBLEM

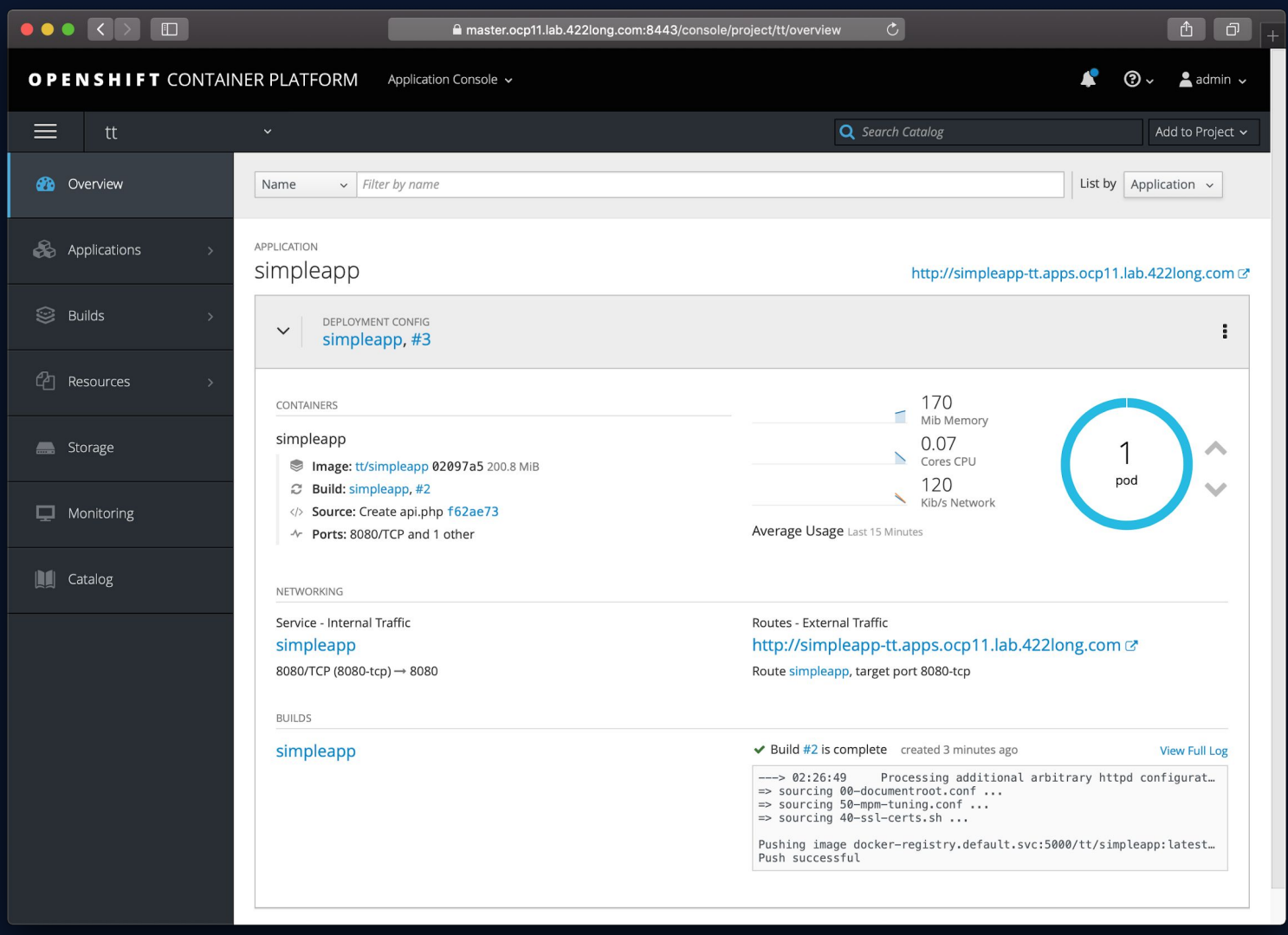
problem:

turnkey management of stateless application

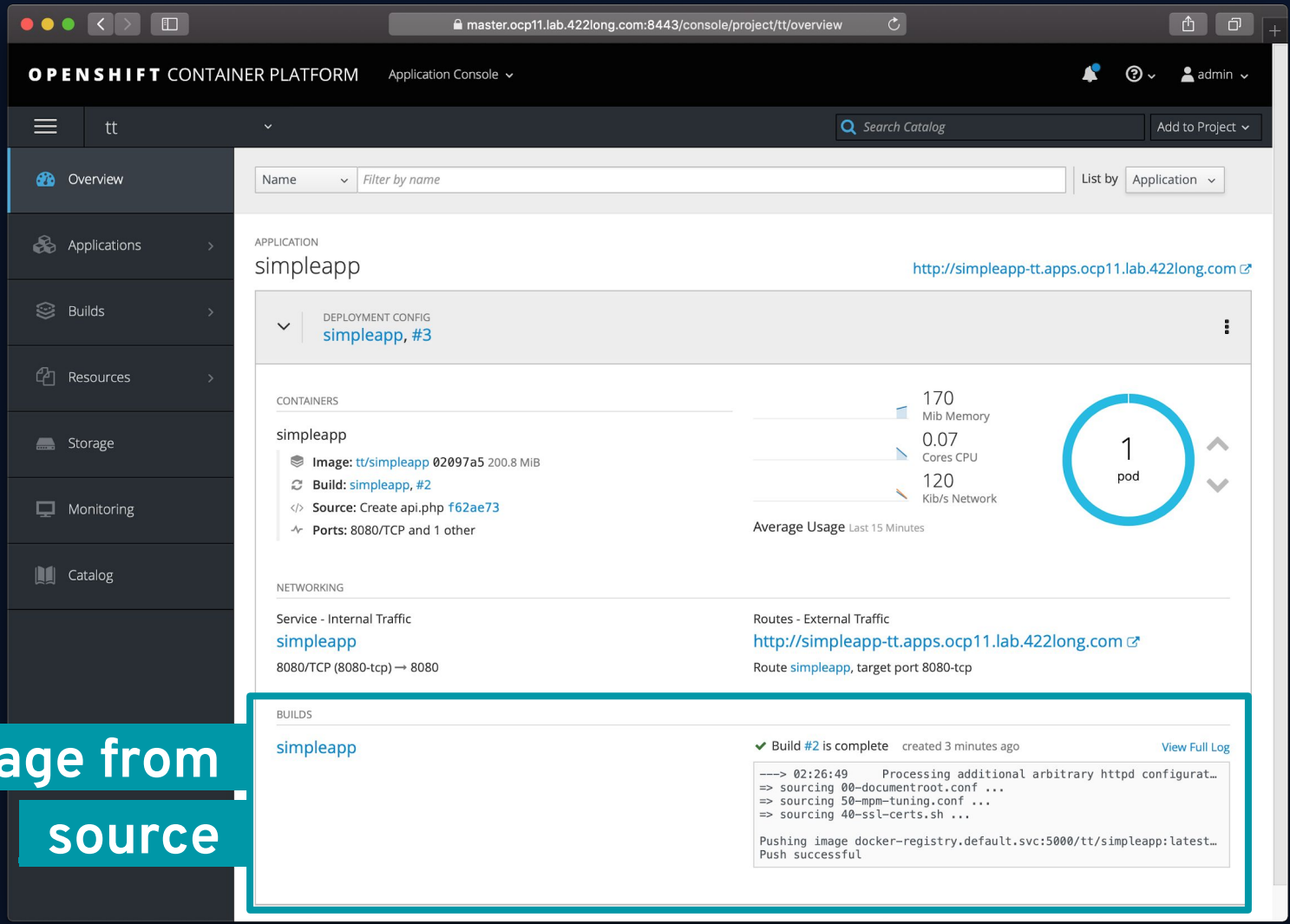
solution:

kubernetes (we just saw this)

S2I, Helm

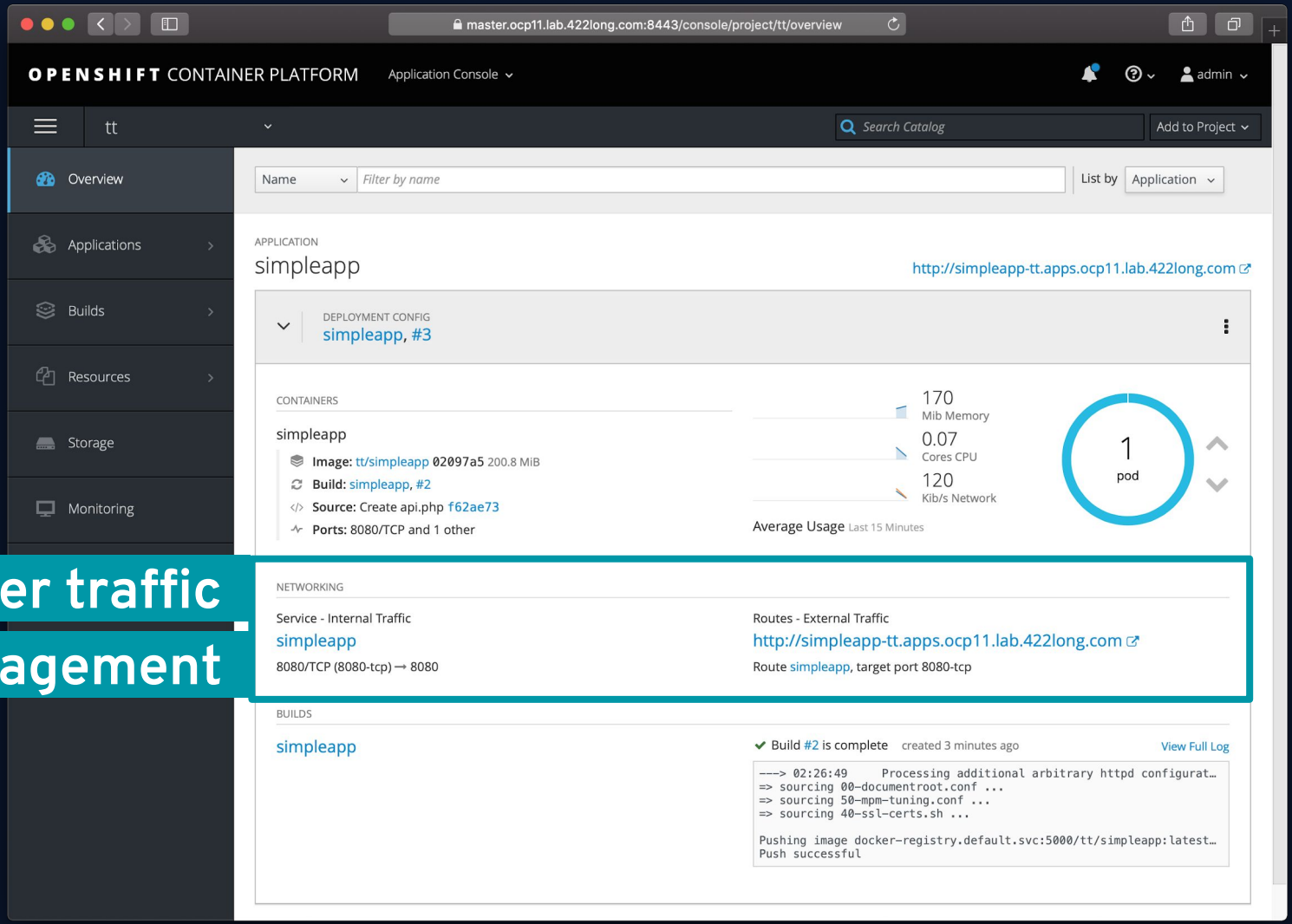

master.ocp11.lab.422long.com:8443/console/project/tt/overview
OPENSHIFT CONTAINER PLATFORM
Application Console
admin
tt
Search Catalog
Add to Project
Overview
Applications
Builds
Resources
Storage
Monitoring
Catalog
Name
Filter by name
List by
Application
APPLICATION
simpleapp
http://simpleapp-tt.apps.ocp11.lab.422long.com
DEPLOYMENT CONFIG
simpleapp, #3
CONTAINERS
simpleapp
Image: tt/simpleapp 02097a5 200.8 MiB
Build: simpleapp, #2
Source: Create api.php f62ae73
Ports: 8080/TCP and 1 other
170
Mib Memory
0.07
Cores CPU
120
Kib/s Network
Average Usage Last 15 Minutes
1
pod
NETWORKING
Service - Internal Traffic
simpleapp
8080/TCP (8080-tcp) → 8080
Routes - External Traffic
http://simpleapp-tt.apps.ocp11.lab.422long.com
Route simpleapp, target port 8080-tcp
BUILDS
simpleapp
Build #2 is complete
created 3 minutes ago
View Full Log
---> 02:26:49 Processing additional arbitrary httpd configurat…
=> sourcing 00-documentroot.conf ...
=> sourcing 50-mpm-tuning.conf ...
=> sourcing 40-ssl-certs.sh ...
Pushing image docker-registry.default.svc:5000/tt/simpleapp:latest…
Push successful

@KeithResar



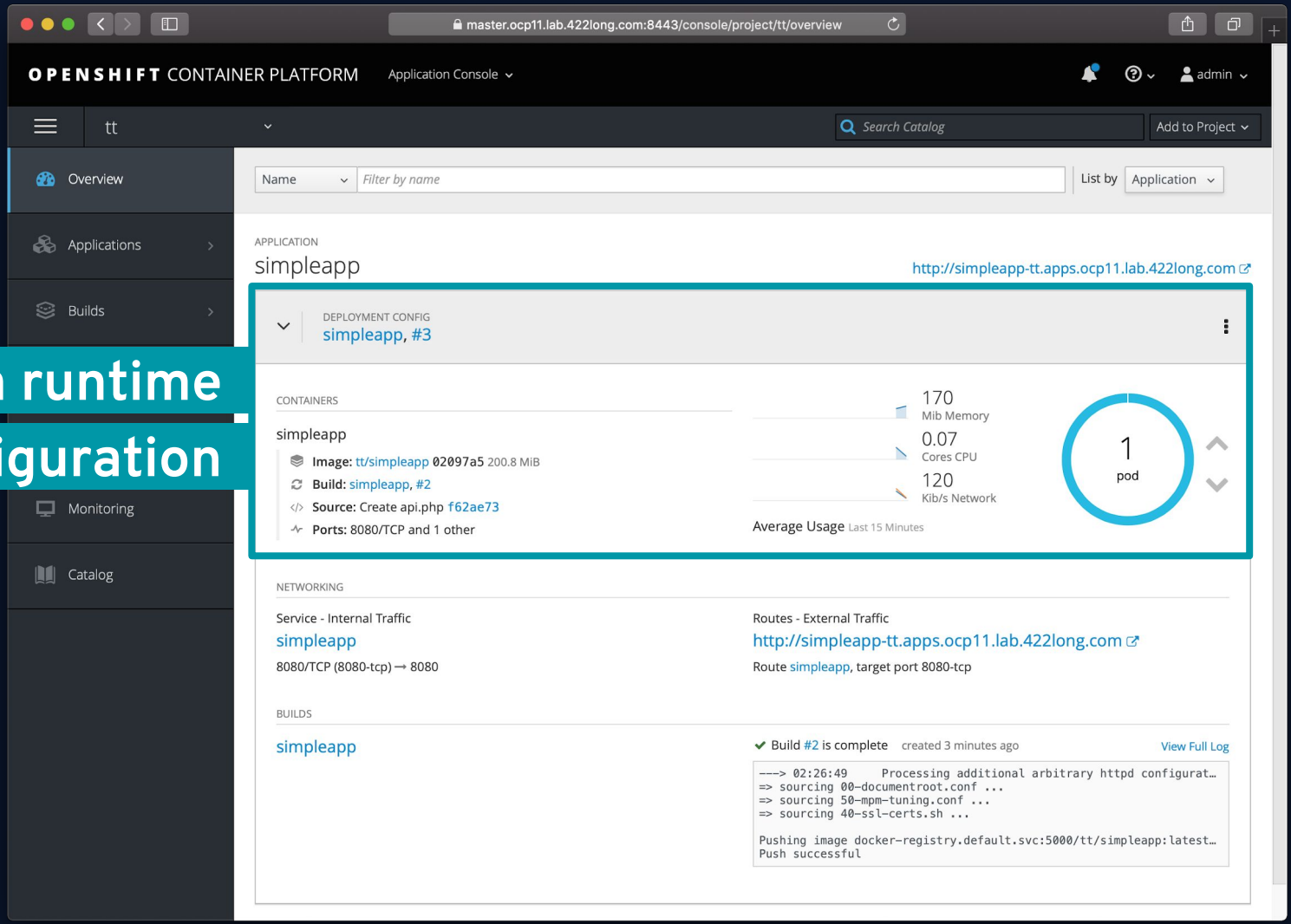
build image from source

@KeithResar



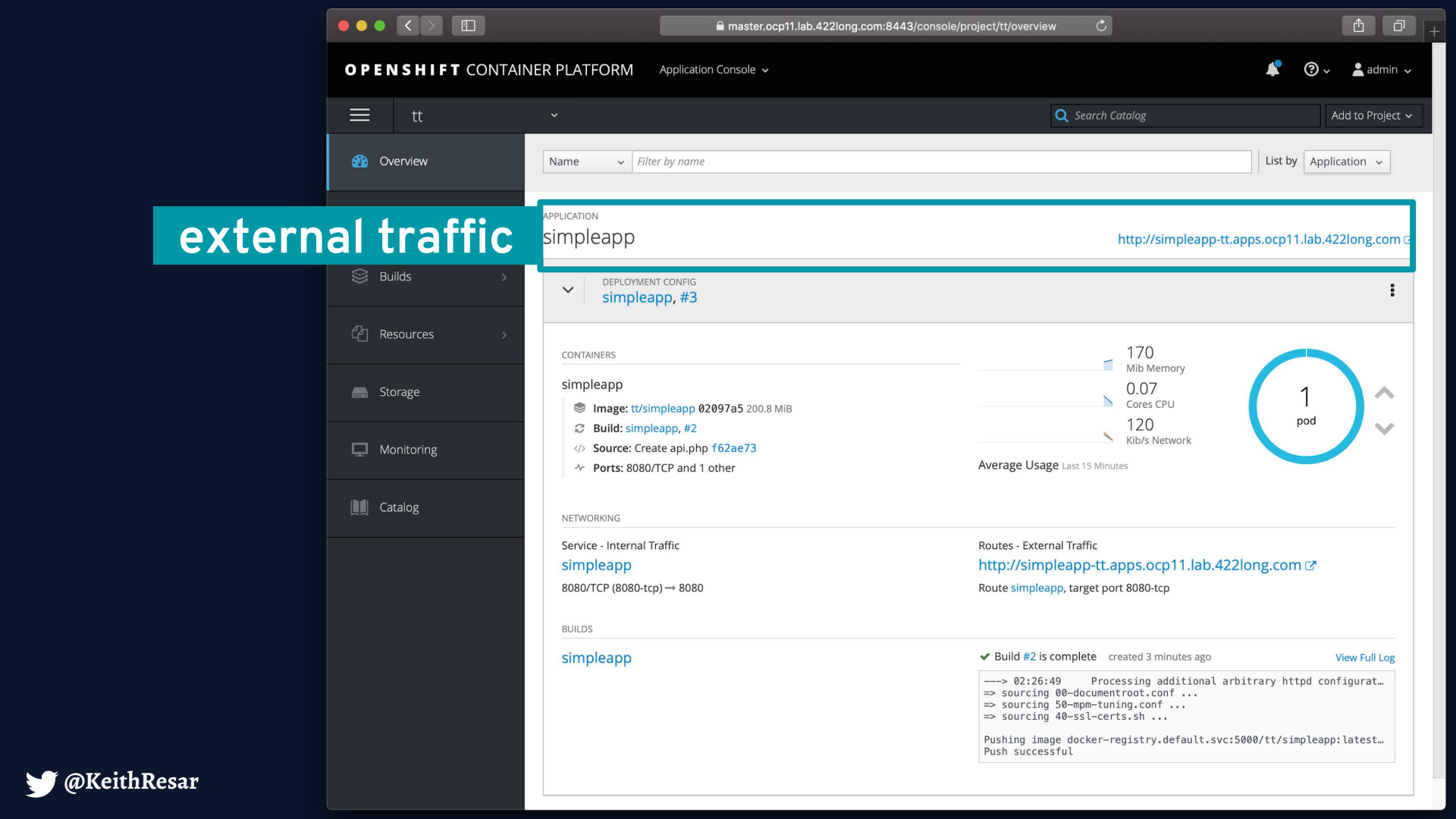
intra-cluster traffic management

@KeithResar



application runtime configuration

@KeithResar


external traffic
master.ocp11.lab.422long.com:8443/console/project/tt/overview
OPENSHIFT CONTAINER PLATFORM
Application Console
admin
tt
Search Catalog
Add to Project
Overview
Builds
Resources
Storage
Monitoring
Catalog
Name
Filter by name
List by
Application
APPLICATION
simpleapp
http://simpleapp-tt.apps.ocp11.lab.422long.com
DEPLOYMENT CONFIG
simpleapp, #3
CONTAINERS
simpleapp
Image: tt/simpleapp 02097a5 200.8 MiB
Build: simpleapp, #2
Source: Create api.php f62ae73
Ports: 8080/TCP and 1 other
170
Mib Memory
0.07
Cores CPU
120
Kib/s Network
Average Usage Last 15 Minutes
1
pod
NETWORKING
Service - Internal Traffic
simpleapp
8080/TCP (8080-tcp) → 8080
Routes - External Traffic
http://simpleapp-tt.apps.ocp11.lab.422long.com
Route simpleapp, target port 8080-tcp
BUILDS
simpleapp
Build #2 is complete
created 3 minutes ago
View Full Log
---> 02:26:49 Processing additional arbitrary httpd configurat…
=> sourcing 00-documentroot.conf ...
=> sourcing 50-mpm-tuning.conf ...
=> sourcing 40-ssl-certs.sh ...
Pushing image docker-registry.default.svc:5000/tt/simpleapp:latest…
Push successful

@KeithResar

problem:

I'm a vendor or I create data service apps,

kubernetes doesn't know anything about me

Stand-in for your app

etcd is a **distributed key value store** that provides a reliable way to store data across a cluster of machines.

Stand-in for your app

Create and Destroy • Resize • Failover

Rolling upgrade • Backup and Restore

problem:

I'm a vendor or I create data service apps,

kubernetes doesn't know anything about me

solution:

create custom resource definitions

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: simpleapp
spec:
  ports:
  - name: 8080-tcp
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    deploymentconfig: simpleapp
  sessionAffinity: None
  type: ClusterIP
```

Defining a service resource

service resources are a built in object type.

```yaml
---
apiVersion: etcd.database.coreos.com/v1beta2
kind: EtcdCluster
metadata:
  name: example-etcd-cluster
spec:
  size: 3
  version: "3.2.13"
```

Defining an `EtcdCluster` resource

Our custom resource looks pretty similar.

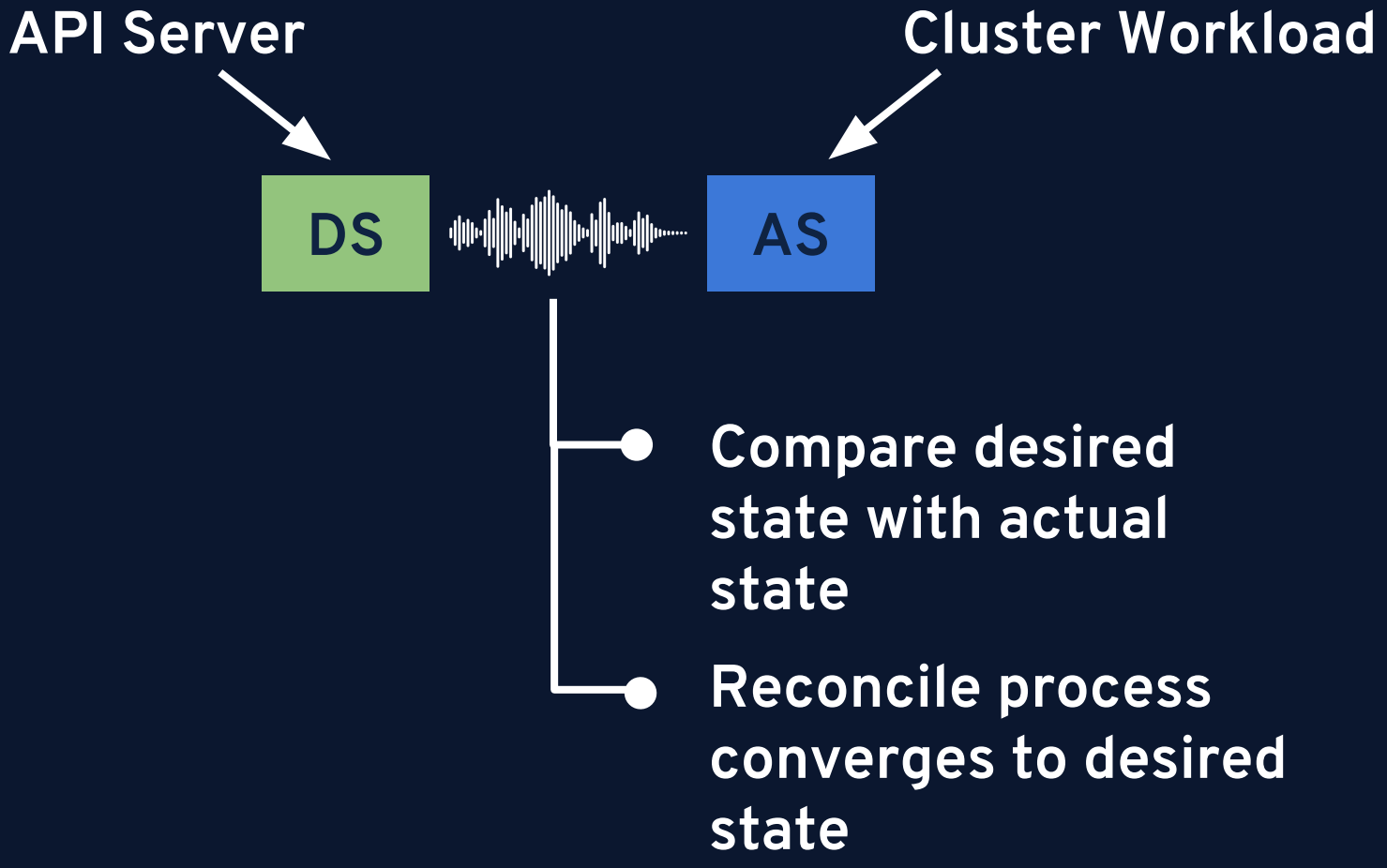
problem:

golang isn't going to fly

solution:

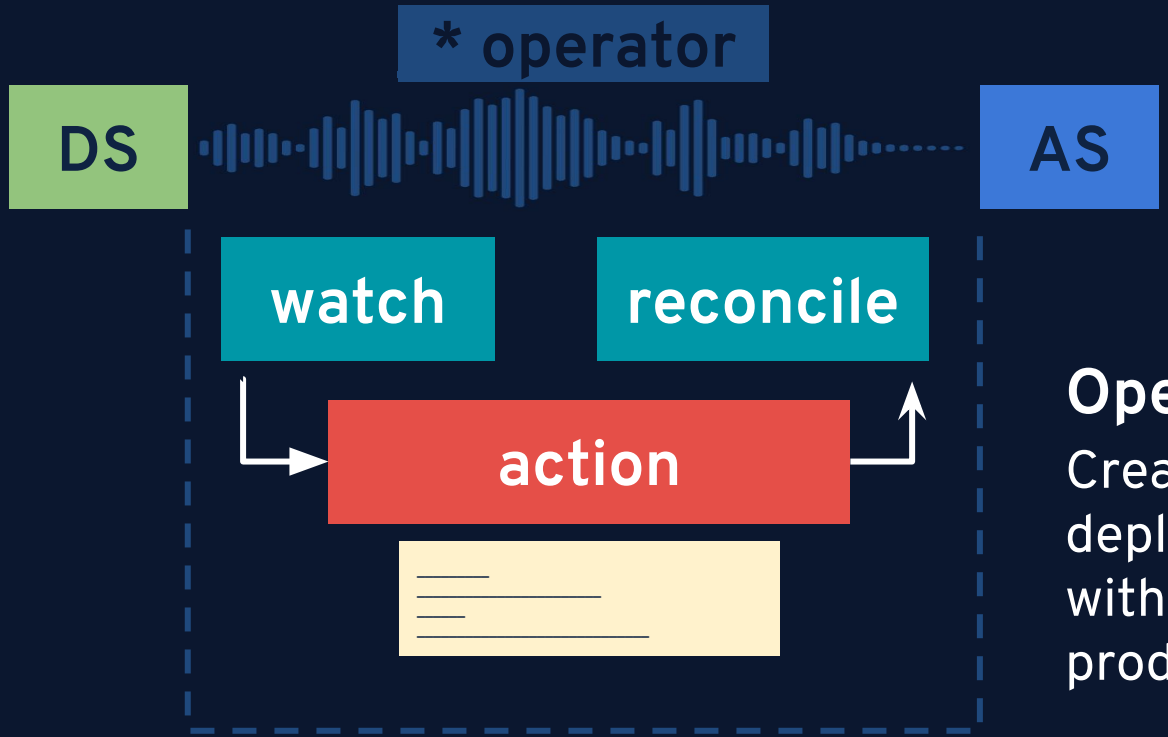
skip go, succeed with helm charts or ansible

EVERY PROBLEM BRINGS A SOLUTION


API Server
Cluster Workload
DS
AS
Compare desired state with actual state
Reconcile process converges to desired state

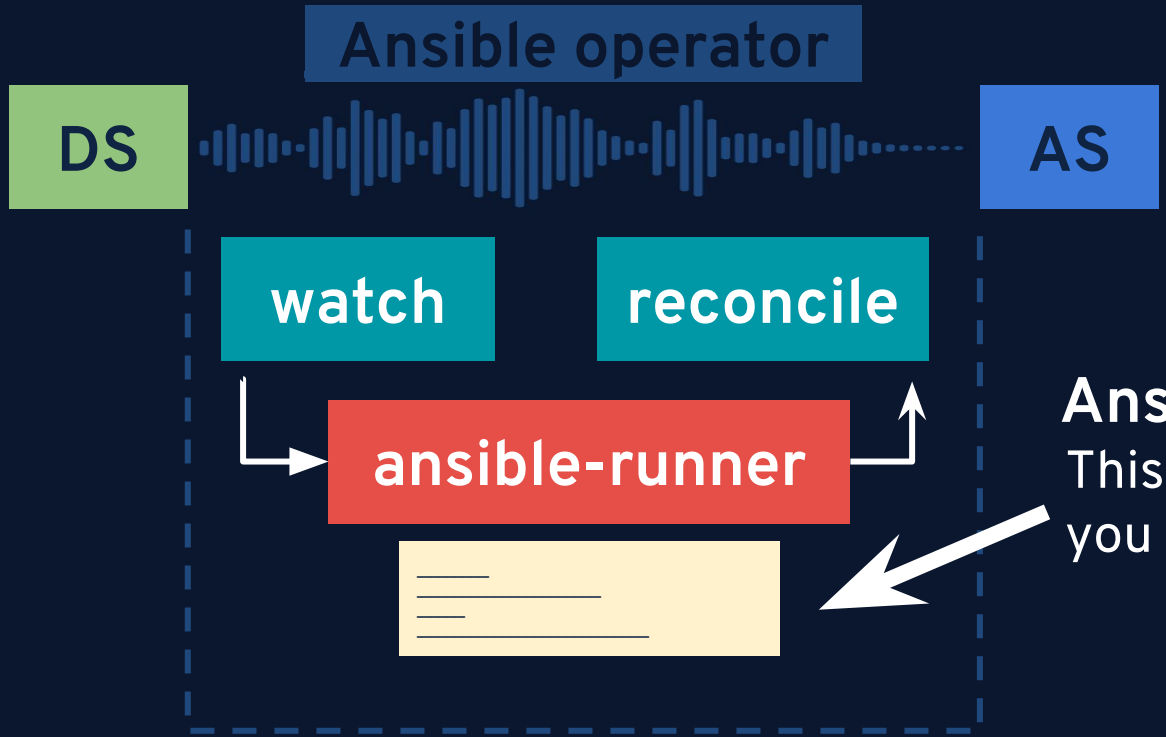

API Server
Cluster Workload
01010001
01010010
10101011
01011001
0101001
DS
AS
1x simpleapp
2x simpleapp


API Server
Cluster Workload
DS
AS
Native K8s objects like...
DeploymentConfig
Services
Routes
etc.

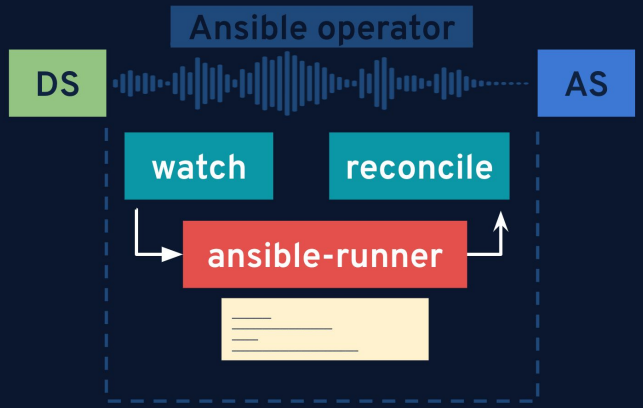


Operator as an Artifact

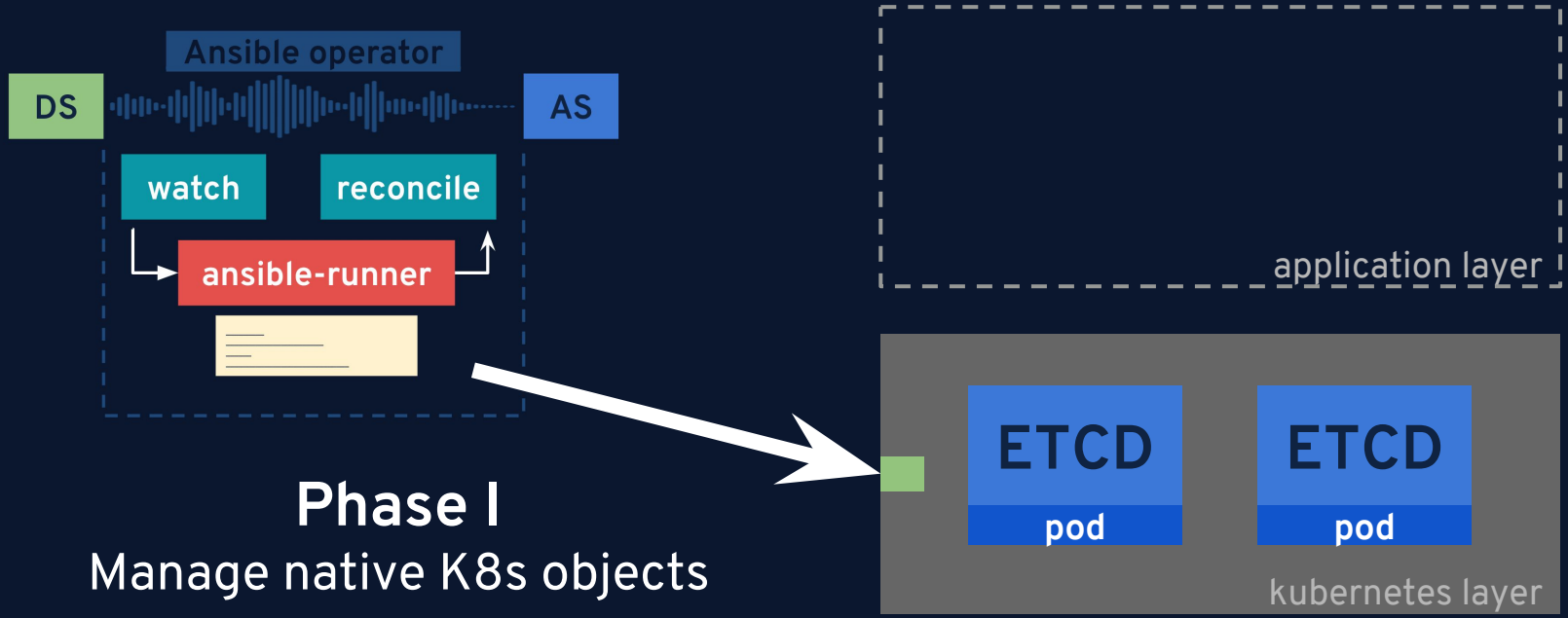
Create, version control, and deploy new versions to align with changes to underlying product versions.



Ansible playbook or role
This is the only component you need to worry about!


Ansible operator
DS
AS
watch
reconcile
ansible-runner


application layer
kubernetes layer



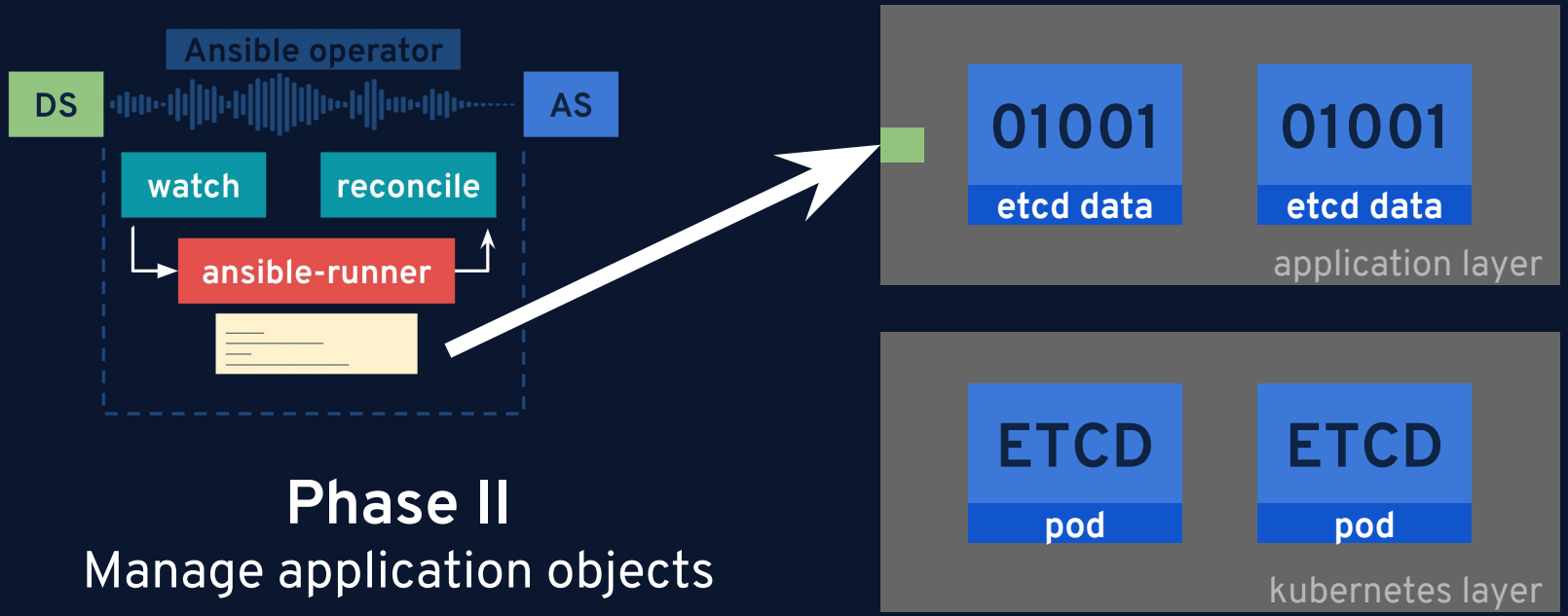
Phase I

Manage native K8s objects

```yaml
# tasks/main.yml
- name: Create k8s resources
  k8s:
    state: present
    definition: '{{ lookup("template", item) | from_yaml_all | list }}'
  with_items:
  - rbac.yaml
  - pod.yaml
  vars:
    ns: example-app
```

```yaml
# templates/pod.yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: example-app
  namespace: '{{ ns }}'
spec:
  serviceAccountName: example-app-cluster
  containers:
  - name: example-app
    image: busybox:latest
    command: ['sleep']
    args: ['3600']
```

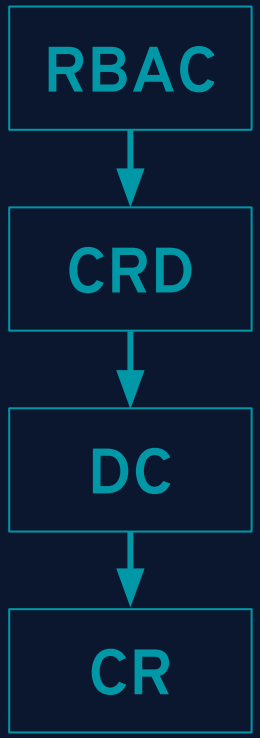
```yaml
# templates/rbac.yaml
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: example-app-cluster
  namespace: '{{ ns }}'
---
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: example-app-cluster
  namespace: '{{ ns }}'
rules:
- apiGroups: [""]
  resources: ["configmaps"]
  verbs: [ "get", "list", "watch", "create", "update", "delete" ]
---
# Allow the pods in this namespace to work with configmaps
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: example-app-cluster
  namespace: '{{ ns }}'
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: example-app-cluster
subjects:
- kind: ServiceAccount
  name: example-app-cluster
  namespace: '{{ ns }}'
```



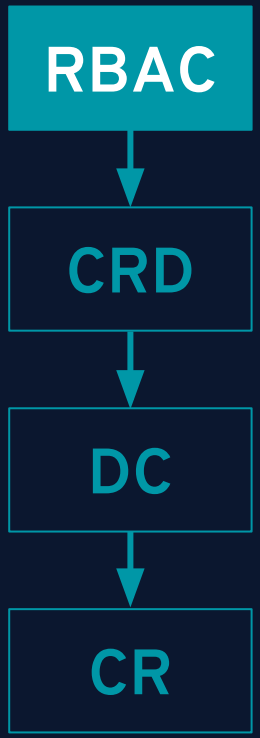
Phase II

Manage application objects

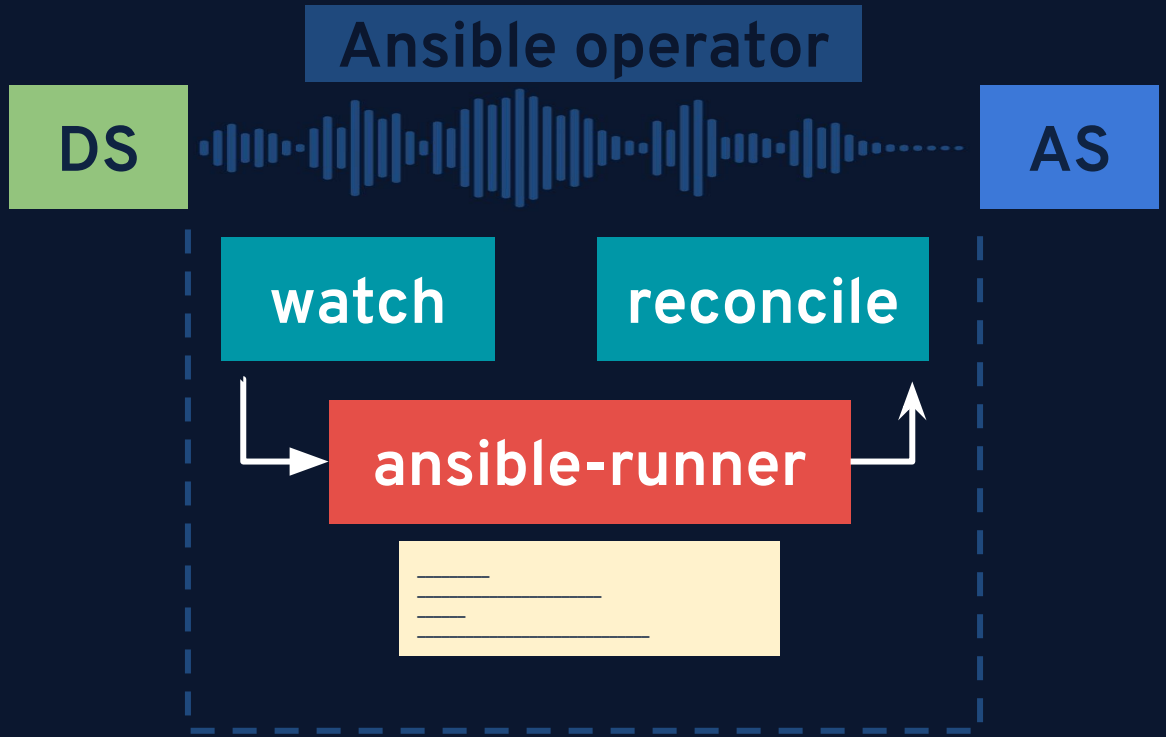
A GIFT OF THE DEMO TO YOU

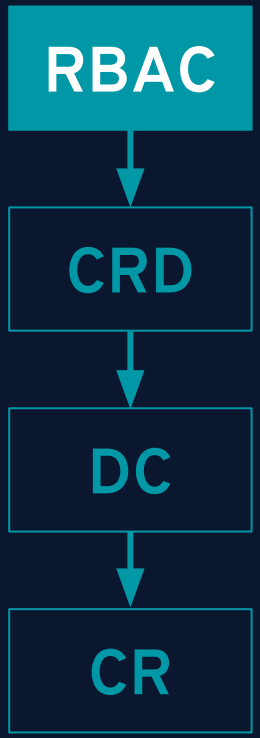


Demo Operator for data service SimpleDB, that manages instantiation and version upgrades.

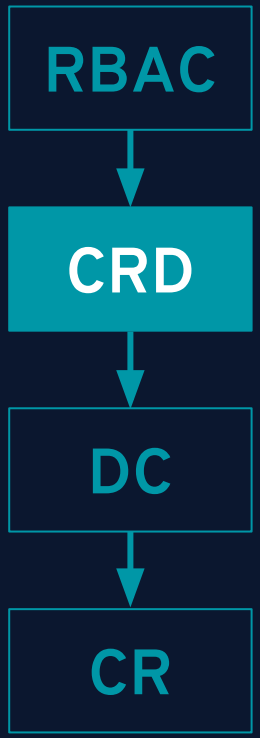


Create service account, role, and role binding. Our operator uses these to monitor events and reconcile desired and actual states.

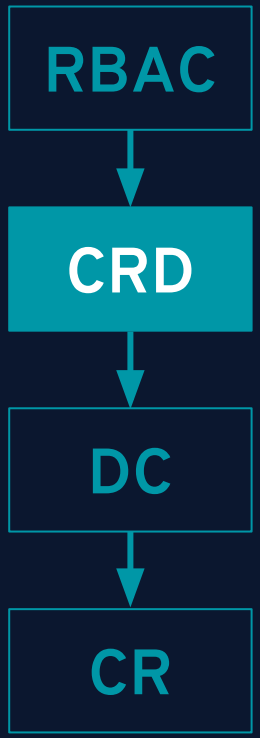

Ansible operator
DS
AS
watch
reconcile
ansible-runner



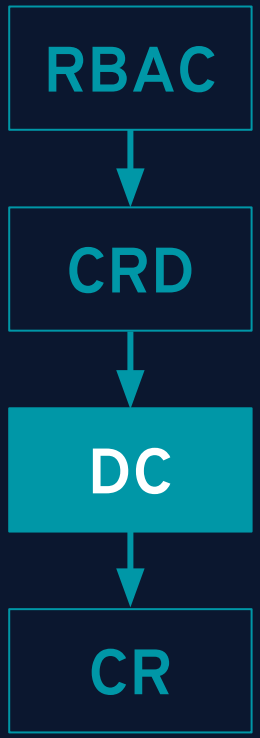

```yaml
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: simpledb
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: simpledb
rules:
  ...
---
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: simpledb
subjects:
- kind: ServiceAccount
  name: simpledb
roleRef:
  kind: Role
  name: simpledb
  apiGroup: rbac.authorization.k8s.io
```



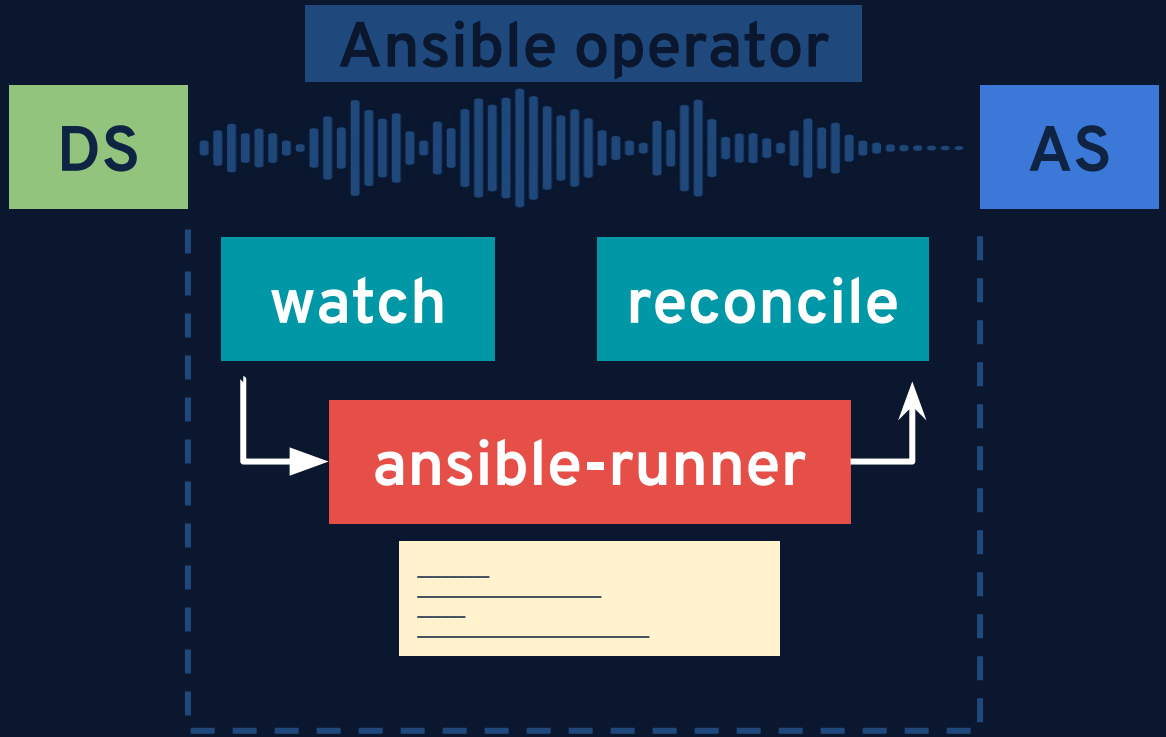
Define the custom resource SimpleDB. This extends what Kubernetes accepts, but doesn't actually change any behavior.

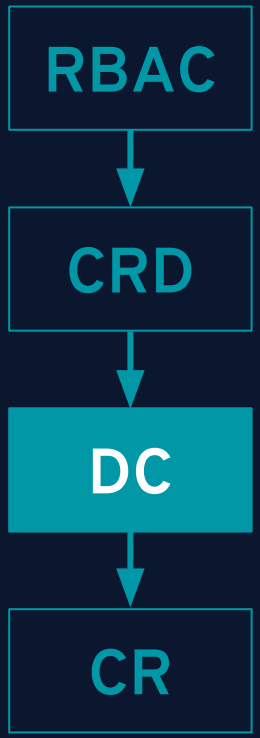



```yaml
---
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: simpledbs.example.com
spec:
  group: example.com
  names:
    kind: SimpleDB
    listKind: SimpleDBList
    plural: simpledbs
    singular: simpledb
  scope: Namespaced
  version: v1alpha1
```

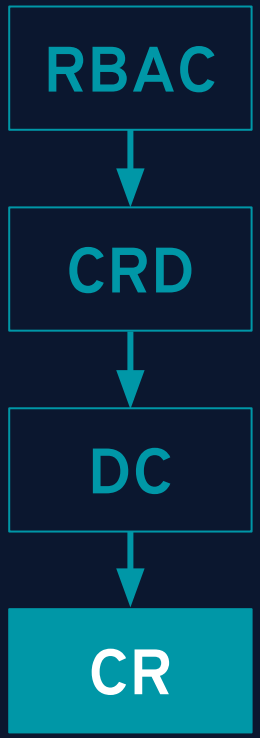


Define and deploy the Ansible Operator container which executes an ansible-runner process.

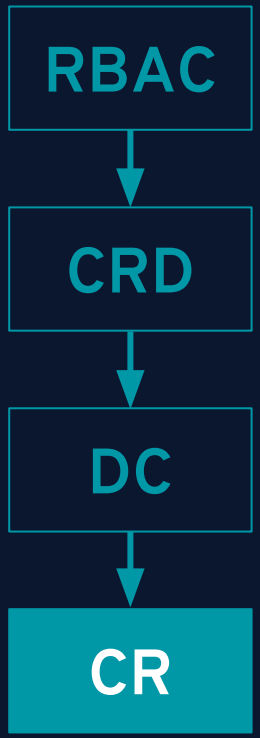

Ansible operator
DS
AS
watch
reconcile
ansible-runner




```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: simpledb
spec:
  template:
    spec:
      serviceAccountName: simpledb
      containers:
        - name: simpledb
          image: hk1232/operator-simpledb-runner:0.1
          env:
            - name: WATCH_NAMESPACE
              valueFrom:
                fieldRef:
                  fieldPath: metadata.namespace
            - name: OPERATOR_NAME
              value: "simpledb"
```



Instantiate our custom resource object. The operator is listening for any SimpleDB events in our namespace.




```yaml
---
apiVersion: example.com/v1alpha1
kind: SimpleDB
metadata:
  name: simpledb
spec:
  # Add fields here
  version: 1
```

GO FARTHER WITH THESE RESOURCES

- Introducing the operator framework
- water-hole's ansible-operator repo
- ansible-operator-demo repo
- Awesome operators in the wild

THANKS