RESILIENCE

REALIZED

KubeCon | CloudNativeCon

North America 2021

SIG Multicluster Intro & Deep Dive

Paul Morie (pmorie)
Jeremy Olmsted-Thompson (jeremyot)
Laura Lorenz (lauralorenz)

Hello!

We'll cover:

- What this SIG is about
- Current activity
  - ClusterSet / Namespace Sameness
  - Cluster ID / ClusterSet membership
  - Multicluster Services API / Multicluster DNS
  - And MORE!
- Deep(*-ish*) dive
  - MCS API
  - Cluster ID
  - Multicluster DNS
- How to contribute

What this SIG is about

- What should be the Kubernetes-native way to ….
  - expose workloads from multiple clusters to each other?
  - replicate workloads across clusters?
  - target deployments to specific clusters in a multicluster group?
- These pressing questions AND MORE are the purview of SIG-Multicluster!
- Touches many different functional areas, but we are still **working to identify the best, most durable primitives**
- We ~~want~~ need your input!
  - Real user stories and use cases are extremely valuable
  - Many projects are in alpha stage and still malleable
  - New tools expose new needs

Our approach

- Avoid premature standardization
- Avoid solving any optional problems
- Focus on specific functionality that we want to build
- Work backwards from specific problems into something bigger, *maybe*

Current activity

ClusterSet

- Note: this is a pre-API concept; it does not currently correspond to a resource - that's changing with Cluster ID
- ClusterSet represents a pattern of use from the field:
  - A group of clusters governed by a single authority
  - High degree of trust within the set
  - Namespace Sameness applies to clusters in the set
    - Permissions and characteristics are consistent across clusters for a given namespace
    - Namespaces don't have to exist in every cluster, but behave the same across those in which they do

Cluster ID

- KEP-2149
- Cluster scoped ClusterProperty CRD - name: value
- Discoverable within the cluster - self-awareness for the first time
  - id.k8s.io
- Allows a cluster to identify the ClusterSet to which it belongs
  - clusterset.k8s.io
- Uniquely identify clusters within a ClusterSet, for the lifetime of membership
- Provides a reference for multi-cluster tooling to build on within a cluster set (e.g. valid DNS label)
  - Disambiguate backends for headless services between clusters
  - A coordinate to use for scheduling work
  - An annotation for metrics and logs

MC Services API

- KEP-1645
- Services are a multi-cluster building block, solving a specific problem with wide appeal
- Builds on the concept of namespace sameness and allows a single service to span and/or be consumed by multiple clusters
- Focused only on the API and common behavior, leaving room for various implementations
  - Submariner, GKE, Istio
- Control plane can be centralized or decentralized but consumers only ever rely on local data
- ClusterIP and headless services just work as expected across clusters.
- Multicluster DNS

```yaml
apiVersion: v1
kind: Service
metadata:
  name: foo
  namespace: bar
spec:
  ports:
  - port: 80
  selector:
    app: foo
---
apiVersion: multicluster.x-k8s.io/v1alpha1
kind: ServiceExport
metadata:
  name: foo
  namespace: bar
```

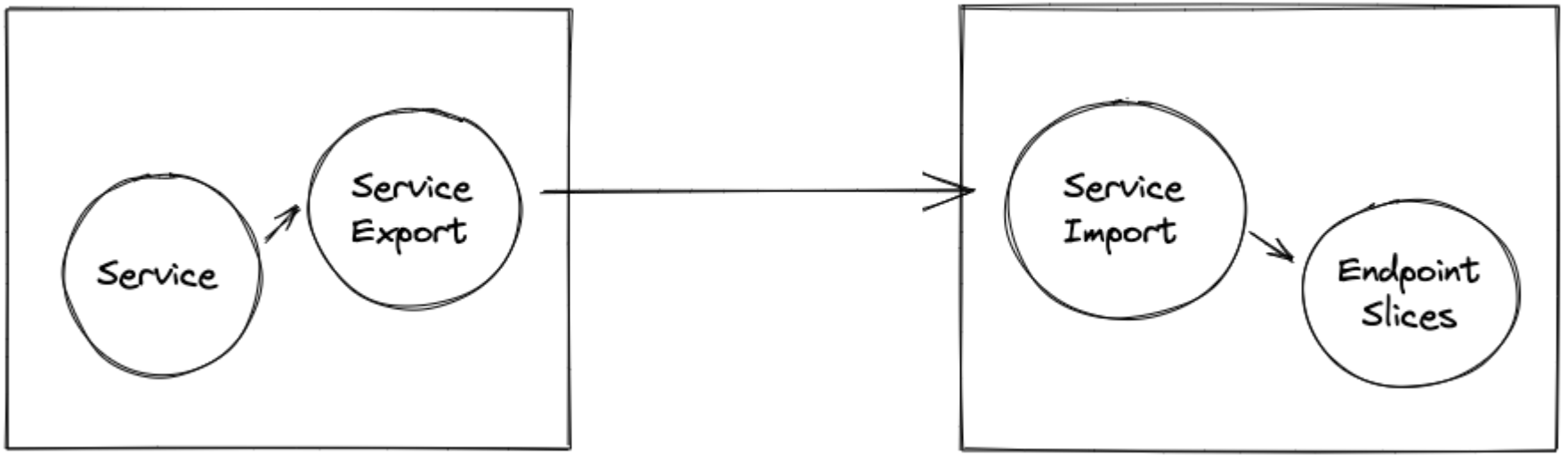
And MORE!

- Kubefed
  - Preparing for Beta
- Work API
  - Spreading groups of resources to different clusters
- Leader election
  - Currently
    - considering how this interacts with existing work to enhance leader election primitives in k8s
    - what SIG-MC should recommend or implement as a reference (*Join us!*)

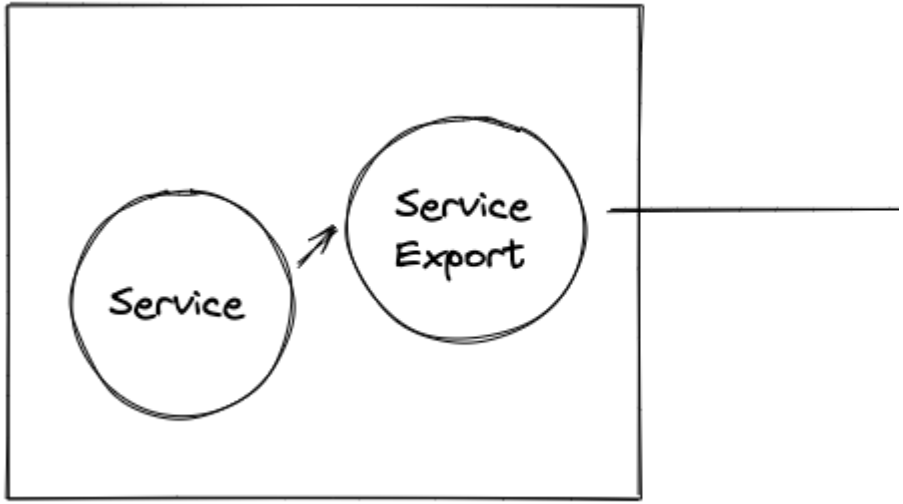
Deep (ok, kinda shallow) dives

MCS API and integrations
ClusterProperty
Multicluster DNS

MCS API



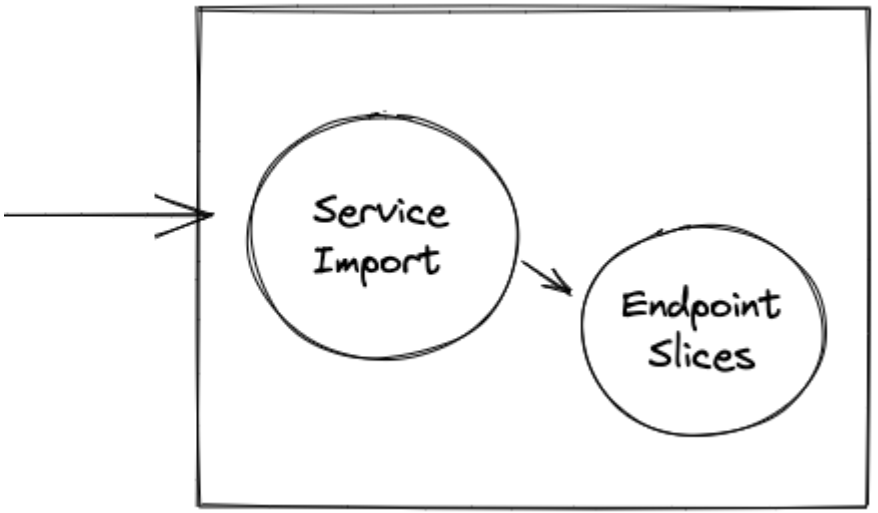
MCS API ServiceExport




```yaml
apiVersion: multicluster.x-k8s.io/v1alpha1
kind: ServiceExport
metadata:
  name: foo
  namespace: bar
```

MCS API ServiceImport


```yaml
apiVersion: multicluster.k8s.io/v1alpha1
kind: ServiceImport
metadata:
  name: my-svc
  namespace: my-ns
spec:
  ips:
  - 42.42.42.42
  type: "ClusterSetIP"
  ports:
  - name: http
    protocol: TCP
    port: 80
  sessionAffinity: None
status:
  clusters:
  - cluster: us-west2-a-my-cluster
```



MCS API integrations

- Istio
  - Multi phased approach (RFC) to integrate MCS, including implementing a full MCS controller in istiod
  - Standardizing on the MCS API benefits MCS users *AND* Istio users too
    - simplifies Istio's multicluster DNS infra
    - allows flexibility between service mesh and "cluster local"-like service discovery by using ServiceExports as their opt-in

- Gateway API
  - Supports referencing ServiceImports as backends for ingress traffic
    - Ex on GKE

```
1  kind: HTTPRoute
2  apiVersion: networking.x-k8s.io/v1alpha1
3  metadata:
4    name: sample-app-route
5    namespace: mcgi-bg
6    labels:
7      gateway: multi-cluster-gateway
8  spec:
9    rules:
10   - forwardTo:
11     - backendRef:
12         group: net.gke.io
13         kind: ServiceImport
14         name: sample-app-blue
15       port: 8080
16       weight: 50 # even 50/50 split
17     - backendRef:
18         group: net.gke.io
19         kind: ServiceImport
20         name: sample-app-green
21       port: 8080
22       weight: 50 # even 50/50 split
```

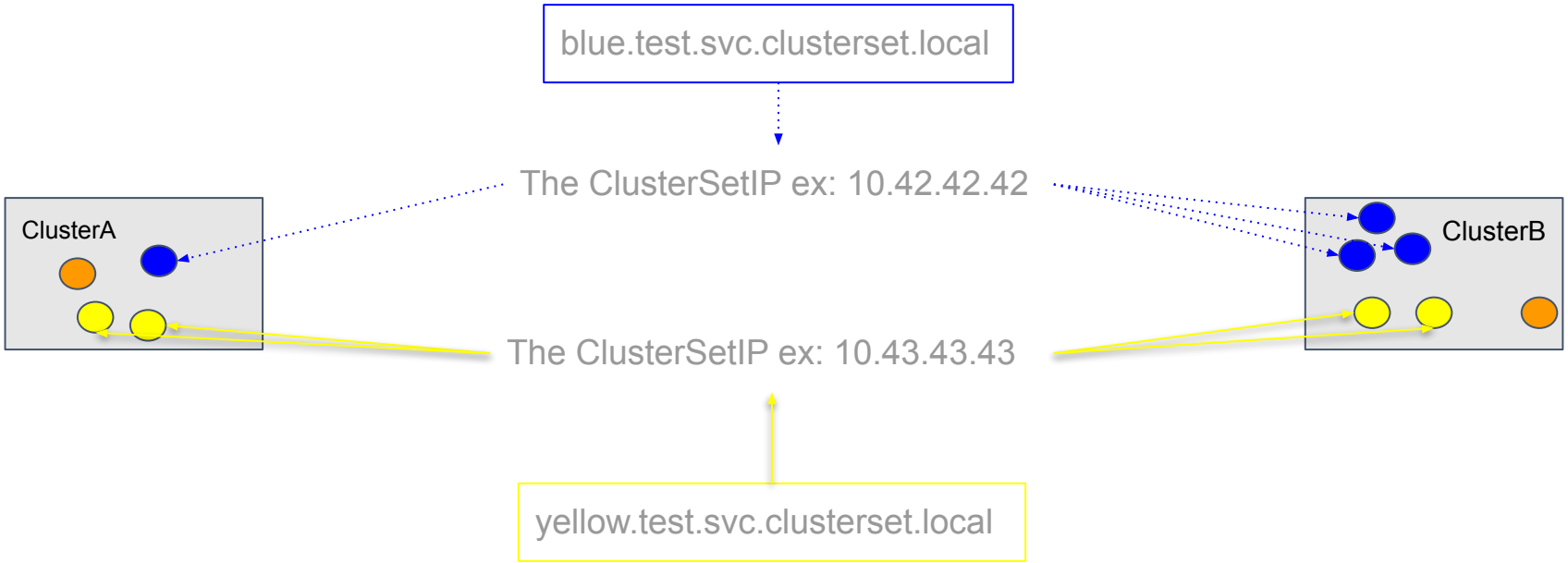
ClusterProperty CRD

- A very basic, general API to store one value per name
- Could be used to store any arbitrary properties about a cluster
- Two particular “well known properties” exist with certain restrictions for implementations following the MCS API standard:
  - `id.k8s.io`
    - The name of the cluster
  - `clusterset.k8s.io`
    - The name of the ClusterSet this cluster belongs to

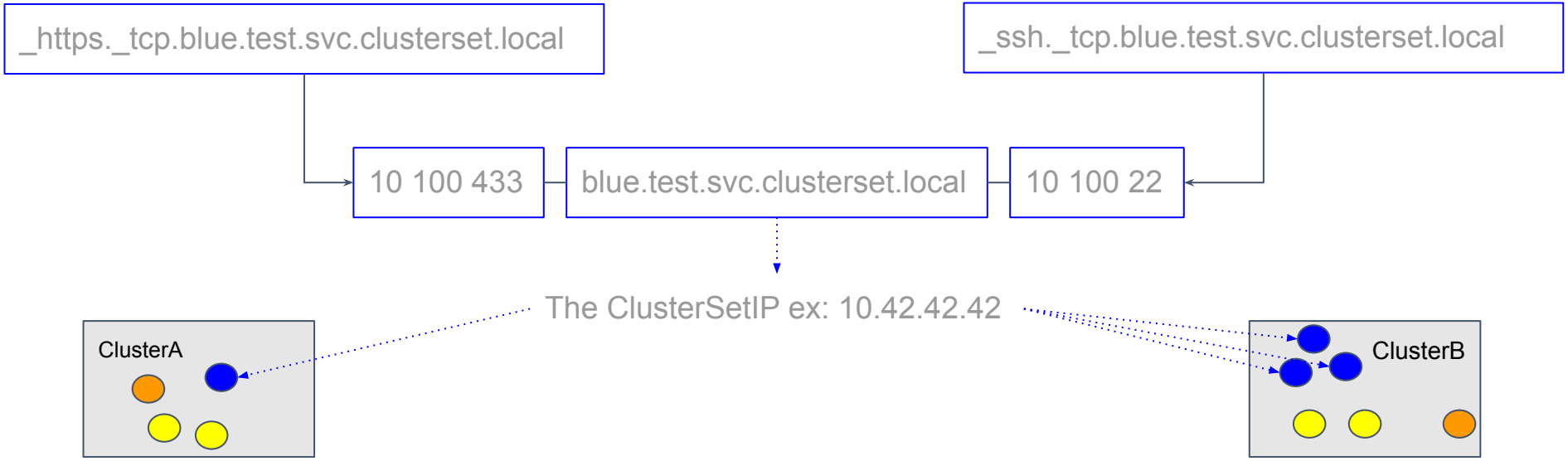
```yaml
# An example object of `id.k8s.io ClusterProperty`
# using a kube-system ns uuid as the id value (recommended):
apiVersion: clusterproperties.k8s.io/v1
kind: ClusterProperty
metadata:
  name: id.k8s.io
spec:
   value: 721ab723-13bc-11e5-aec2-42010af0021e
```

```yaml
# An example object of `clusterset.k8s.io ClusterProperty`:
apiVersion: clusterproperties.k8s.io/v1
kind: ClusterProperty
metadata:
  name: clusterset.k8s.io
spec:
   value: environ-1
```

ClusterSetIP A/AAAA records



ClusterSetIP SRV records




```
* Question Example:
   *   `_https._tcp.myservice.test.svc.clusterset.local. IN SRV`
*   Answer Example:
   *   `_https._tcp.myservice.test.svc.clusterset.local. 30 IN SRV 10 100 443 myservice.test.svc.clusterset.local.`
```

Multicluster Headless A/AAAA records

(and the same for yellow-1.clusterA.{...}, etc)

Multicluster Headless SRV records


```
*   Question Example:
    *    `_https._tcp.headless.test.svc.clusterset.local. IN SRV`
*   Answer Example:
    *   `_https._tcp.headless.test.svc.clusterset.local. 4 IN SRV 10 100 443 my-pet-1.clusterA.headless.test.svc.clusterset.local.`
    *   `_https._tcp.headless.test.svc.clusterset.local. 4 IN SRV 10 100 443 my-pet-1.clusterB.headless.test.svc.clusterset.local.`
    *   `_https._tcp.headless.test.svc.clusterset.local. 4 IN SRV 10 100 443 my-pet-2.clusterB.headless.test.svc.clusterset.local.`
    *   `_https._tcp.headless.test.svc.clusterset.local. 4 IN SRV 10 100 443 my-pet-3.clusterB.headless.test.svc.clusterset.local.`
```

Want to see demos?

- Laura Lorenz and Stephen Kitt of SIG-Multicluster have a talk in the Service Mesh track diving in even deeper
  - Here Be Services: Beyond the Cluster Boundary with Multicluster Services
- Includes demos of the MCS API and multicluster DNS on GKE and OpenShift


KubeCon | CloudNativeCon

North America 2021

RESILIENCE

REALIZED

Get involved

We need your input

Share your use cases, problems, and ideas

- **Home page:**

  https://github.com/kubernetes/community/tree/master/sig-multicluster

- **Slack channel:** https://kubernetes.slack.com/messages/sig-multicluster
- **List:** https://groups.google.com/forum/#!forum/kubernetes-sig-multicluster
- Meetings are biweekly Tuesdays, 12:30 eastern, 9:30 pacific, 16:30 UTC

RESILIENCE
REALIZED

KubeCon | CloudNativeCon
North America 2021

THANK YOU!