KubeCon | CloudNativeCon

North America 2019

Intro to the Kubernetes Working Group for Multi-tenancy

Tasha Drew, Co-Chair

What is multi-tenancy, really?

- Why do people want multi-tenancy?
- What is a tenant?
- Different Tenancy Models

Where can you find us?

- Slack
  - Kubernetes Slack, #wg-multitenancy
  - Join at https://slack.k8s.io/
- Google groups
  - https://groups.google.com/forum/#!forum/kubernetes-wg-multitenancy
  - Join the mailing list here for automatically receiving calendar invites and meeting notes
- Bi-weekly meeting (join google group for invite)
  - Tuesday 11am Pacific Time
  - Meetings are recorded and posted to YouTube
    - Playlist: https://www.youtube.com/playlist?list=PL69nYSiGNLP1tBA0W8zEe6UwPsabGQk-j
- Github https://github.com/kubernetes-sigs/multi-tenancy/

Who are we?

Chairs

- @tasha (me!)
    - Tasha Drew, Product Line Manager @ VMware
    - Work on Project Pacific and Tanzu / Kubernetes stuff on vSphere
- @srampal
    - Sanjeev Rampal, Principle Engineer @ Cisco

Project leads

- @Adrian Ludwin
    - Hierarchical Namespace Controller (“HNC,” sometimes sounds like “agency”)
    - Software Engineer @ Google
- @Fei Guo
    - Virtual Clusters, Tenant Controller
    - Software Engineer @ Alibaba
- @Jim Bugwadia
    - Multi-tenancy Benchmarks
    - Founder & CEO at Nirmata

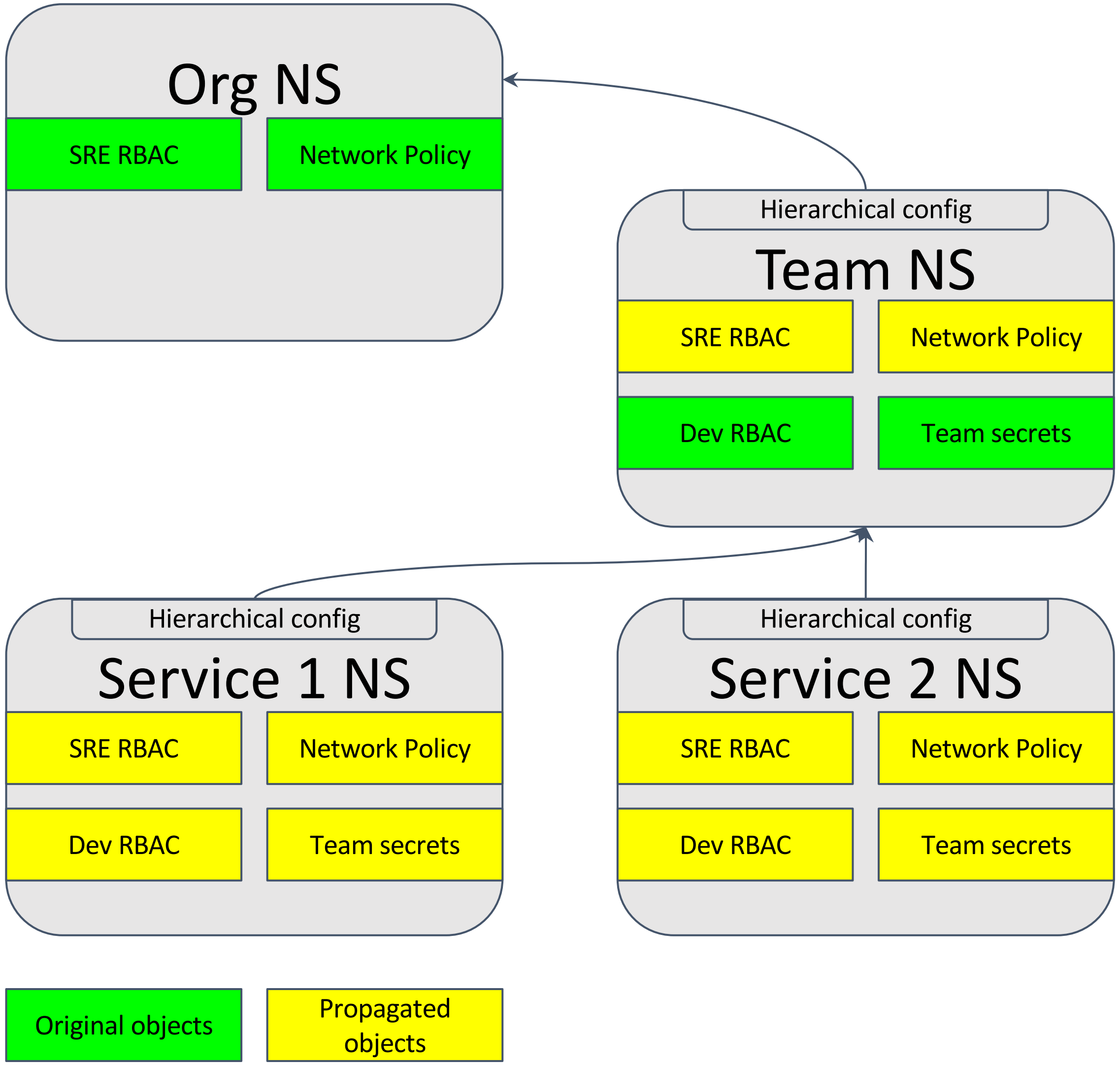
Hierarchical Namespace Controller

Design: http://bit.ly/k8s-hnc-design
Code: https://github.com/kubernetes-sigs/multi-tenancy/tree/master/incubator/hnc
Status: Active development

Goal: This is an early concept to allow namespaces to be linked to each other in parent-child relationships, and to allow certain objects from ancestors to be "visible" in their descendents, which is achieved by copying them. This is most useful for objects such as RBAC roles and bindings - a rolebinding made in a parent namespace will (under normal circumstances) also grant access in child namespaces as well, allowing for hierarchical administration.

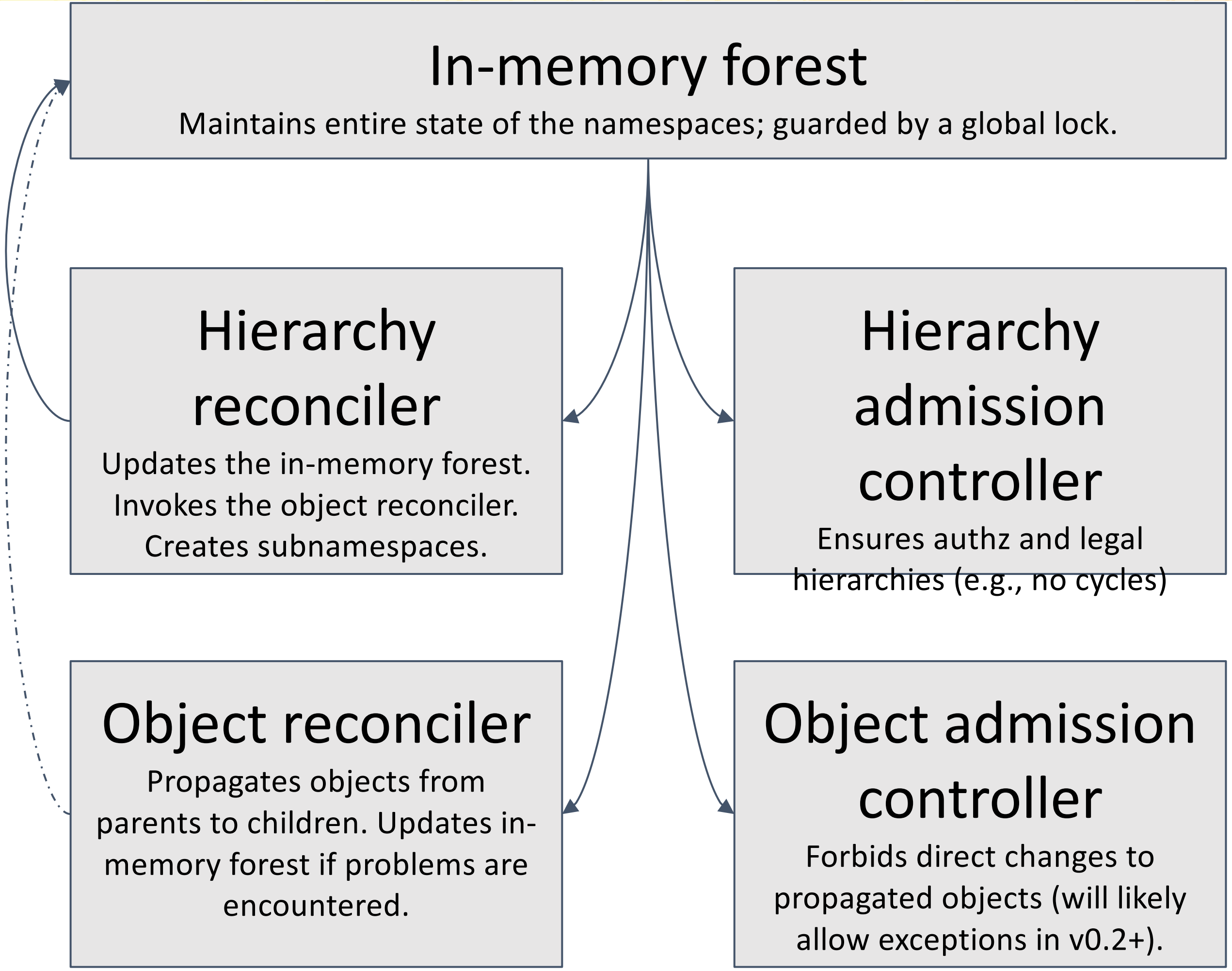
Hierarchical Namespace Controller

- Propagates policy objects from parents to children
  - Hardcoded list in v0.1 (Nov), aim to be configurable in v0.2 (eo2019)
- Self-service subnamespaces
  - No need for cluster-level privileges to create subnamespaces
- Hierarchical authz checks
  - “Subadmins” cannot deprive “superadmins” of access
- Integrations via K8s labels
  - Namespaces receive labels indicating the subtrees they’re in.



HNC Architecture

- Add-on controller based on kubebuilder, controller-runtime, etc
- Two major aspects:
  - Manage the hierarchy configurations
  - Manage the propagated objects
- Namespaces also updated
  - Labels applied to indicate the hierarchical structure.
  - Created (but never destroyed) for self-service subnamespaces on request.
- Problems reported via conditions
  - Admission controllers (webhooks) can fail so problems can creep in.
  - These are all reported via Conditions on the HierarchyConfiguration.



In-memory forest

Maintains entire state of the namespaces; guarded by a global lock.

Hierarchy reconciler

Updates the in-memory forest. Invokes the object reconciler. Creates subnamespaces.

Hierarchy admission controller

Ensures authz and legal hierarchies (e.g., no cycles)

Object reconciler

Propagates objects from parents to children. Updates in-memory forest if problems are encountered.

Object admission controller

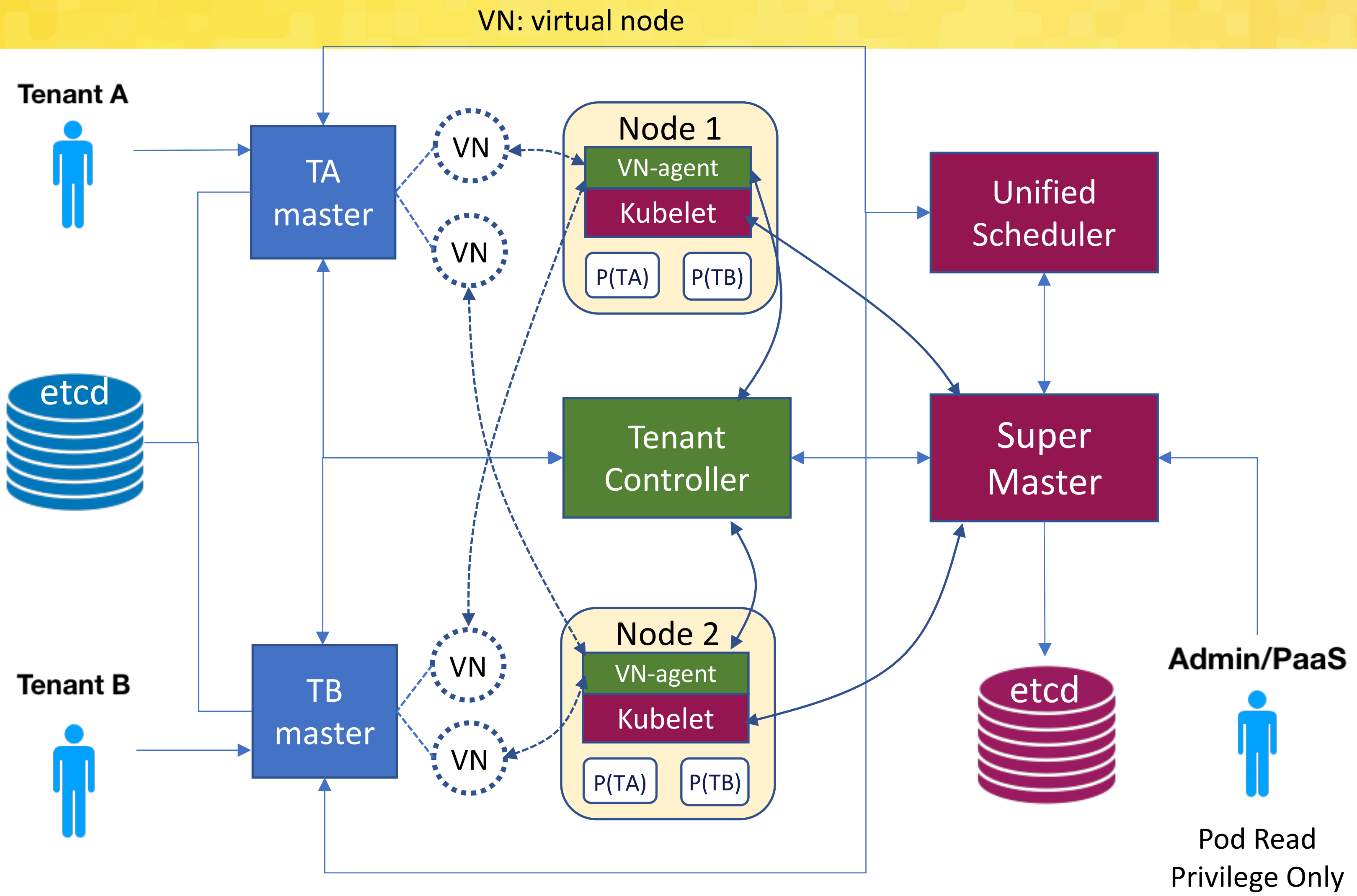
Forbids direct changes to propagated objects (will likely allow exceptions in v0.2+).

Virtual Clusters

Design:
https://docs.google.com/document/d/1EELeVaduYZ65j4AXg9bp3Kyn38GKDU5fAJ5LFcxt2ZU/edit#heading=h.7tna1yo4dzv
Code: https://github.com/kubernetes-sigs/multi-tenancy/tree/master/incubator/virtualcluster
Status: Active development

Goal: Virtual Cluster is a Kubernetes hard multi-tenancy solution for tenants who prefer a dedicated control plane. It provides API level isolation by deploying a tenant master for each tenant. It provides compute resource isolation by using Sandbox runtime like Kata container or gVisor. It provides network isolation using additional tenant network namespace. Overall, we think Virtual Cluster handles the Kubernetes multi-tenancy problem from a different angle and the community may be inspired by it to use Kubernetes in more use cases.

Virtual Clusters architecture



- Tenant master
  - upstream K8s master
- Tenant controller
  - Tenant lifecycle
  - Object-2-Tenant map
- VN-agent
  - A Kubelet proxy

Tenant Controller v2

Design: https://docs.google.com/document/d/1PkV7y_GHU_RfL2y8W-tLa98UnHuierkixXz-5uwKjMA/edit#
Code: https://github.com/kubernetes-sigs/multi-tenancy/tree/master/tenant
Status: Ideation

Goal: Have a controller that manages tenants in a cluster, and enforces policy over the tenant as a whole, who may have many namespaces.

A Tenant consists of a set of namespaces in which any account with sufficient permissions may create Kubernetes objects. The number of these objects and their resources consumption are totaled over all namespaces of a tenant.

For each metric (CPU, mem, # of objects, etc.) a limit may be set either per namespace or for the total per tenant.

Multi-tenancy Benchmarks

Design: https://docs.google.com/document/d/1O-G8jEpiJxOeYx9Pd2OuOSb8859dTRNmgBC5gJv0krE/edit#
Code: https://github.com/kubernetes-sigs/multi-tenancy/tree/master/benchmarks
Status: Active Development

Goal: Providing benchmarks that validate whether multi-tenancy has been achieved, independently of which tool or mechanism was used for multi-tenancy.

The motivation is to decouple how multi-tenancy is provisioned and managed, from the desired state. The proposal involves defining the desired state as a set of benchmarks organized by levels and providing a tool for validating that a set of desired states have been achieved.

Questions? & More multi-tenancy!

Sessions at Kubecon San Diego:

Tuesday, November 19, 4:25pm, Room 29ABCD
Panel: Control Plane vs Data Plane: Untangling the Tenents of Multi-tenancy

In this discussion, our panelists will share their proposals for the principles of multi-tenancy, according to both the type of concerns (control plane vs data plane) as well as the type of tenants (such as dev teams, production teams and third-party users).

Wednesday, November 20, 5:20pm, Room 1AB
Deep Dive: Kubernetes Working Group for Multi-tenancy
Sanjeev Rampal and Adrian Ludwin (and other guest stars!)

This deep dive of the working group for Multi-tenancy will include an in-depth technical exploration of multi-tenancy in core Kubernetes and the tooling and services the multi-tenancy working group has been developing to mainstream how users of Kubernetes can achieve multi-tenancy.

KubeCon | CloudNativeCon

North America 2019