Azure Meetup

Kubernetes Hands-On!

Microsoft Azure

VMWARE {code}

nirmata

Do this now!

- Signup for Azure

  https://azure.microsoft.com/en-us/free/

- Signup for Nirmata

  https://try.nirmata.io

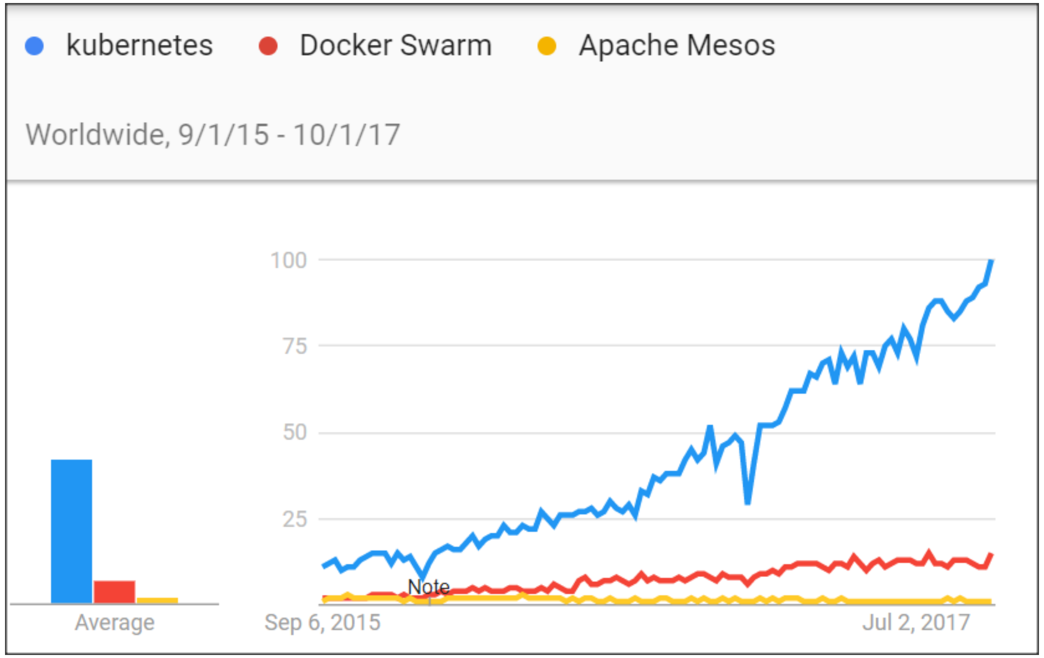
Agenda

- 6:30 – 6:45: Kubernetes Cluster Concepts
- *6:45 – 7:45: Setup your Azure K8s cluster*
- 7:45 – 8:00: Kubernetes Application Concepts
- *8:00 – 9:00: Manage K8s applications with Nirmata*

Why Kubernetes?

- Containers and Microservices enable DevOps
  - Agility, Cost-savings, Portability

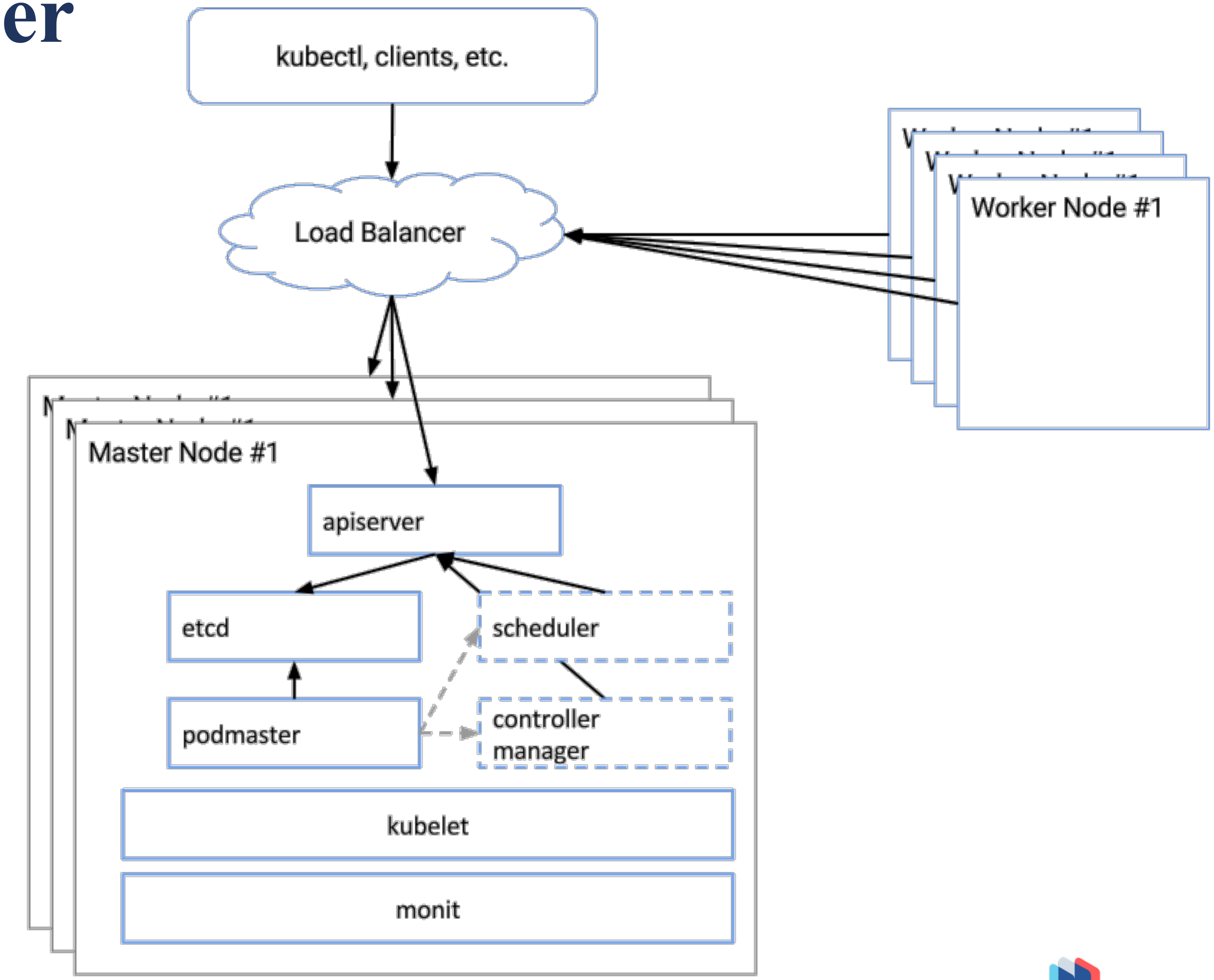
- Kubernetes has become the de-facto standard for container orchestration
  - Community-driven
  - Robust, scalable, and extensible



Kubernetes Cluster Concepts

Kubernetes Cluster

- Nodes
- Components
- Add-ons
- Cloud Provider
  - Networking
  - Storage



Master Nodes

- Master nodes run Kubernetes components
  - **kube-apiserver**: *front-end for the Kubernetes control-plane*
  - **etcd**: *datastore for the cluster*
  - **kube-controller-manager**: *controllers for routine cluster tasks*
  - **cloud-controller-manager**: *controllers specific to cloud providers*
  - **kube-scheduler**: *assigns Pods to nodes*

Each worker node runs

- Required:
  - kubelet: manages pods, executes liveness probes, reports pod and node status.
  - kube-proxy: network proxy; performs connection forwarding
  - docker / rkt / *containerd*: the container engine
- Optional
  - add-ons
  - supervisord
  - fluentd
- Your application pods

Common Add-ons

- DNS: serves DNS for Kubernetes components and containers. Consider as required.

- Heapster: provides container resource monitoring. Is used for Horizontal Pod Autoscaling.

- Web UI: dashboard to monitor and manage the cluster.

Networking

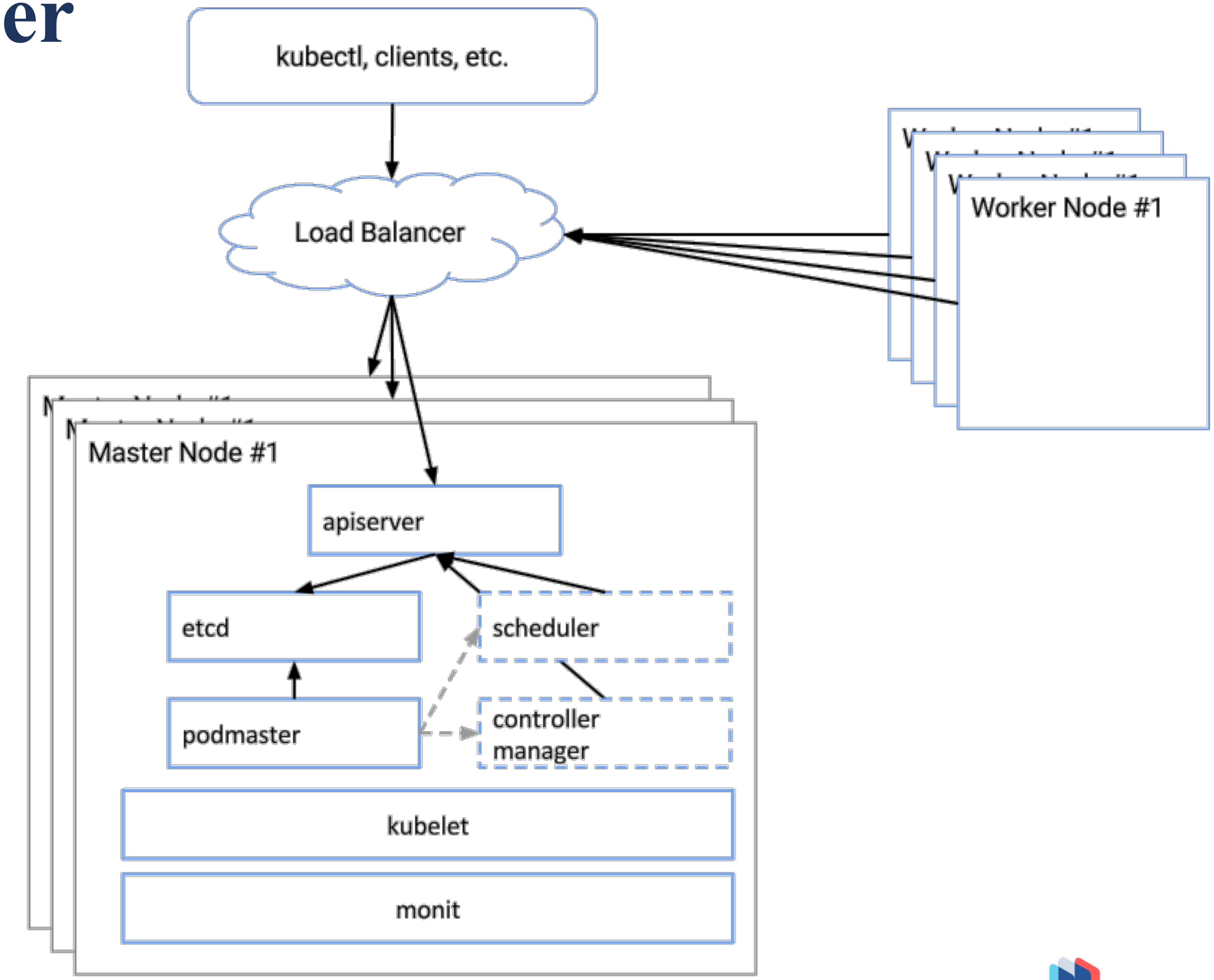
- K8s networking follows these principles:
  - All containers can communicate with all other containers without NAT
  - All nodes can communicate with all containers (and vice-versa) without NAT
  - The IP that a container sees itself as is the same IP that others see it as
- Each pod gets its own IP address
- CNI is the plugin model used by the Kubelet to invoke the networking implementation
- CNI plugins: Calico, Contiv, Flannel , GCE, …

Storage

- Pods can contain one or more **Volumes**
  - *Volume types: emptyDir, hostPath, persistentVolumeClaim, secret, awsElasticBlockStore, AzureDiskVolume, …*
- A **PersistentVolumeClaim** *requests a* **PersistentVolume** *that may be dynamically provisioned.*
  - Admins can configure **StorageClasses** for persistent volume claims like “bronze”, “silver”, or “gold”. A storage class has a **Provisioner**, like AzureDisk.

Kubernetes Cluster

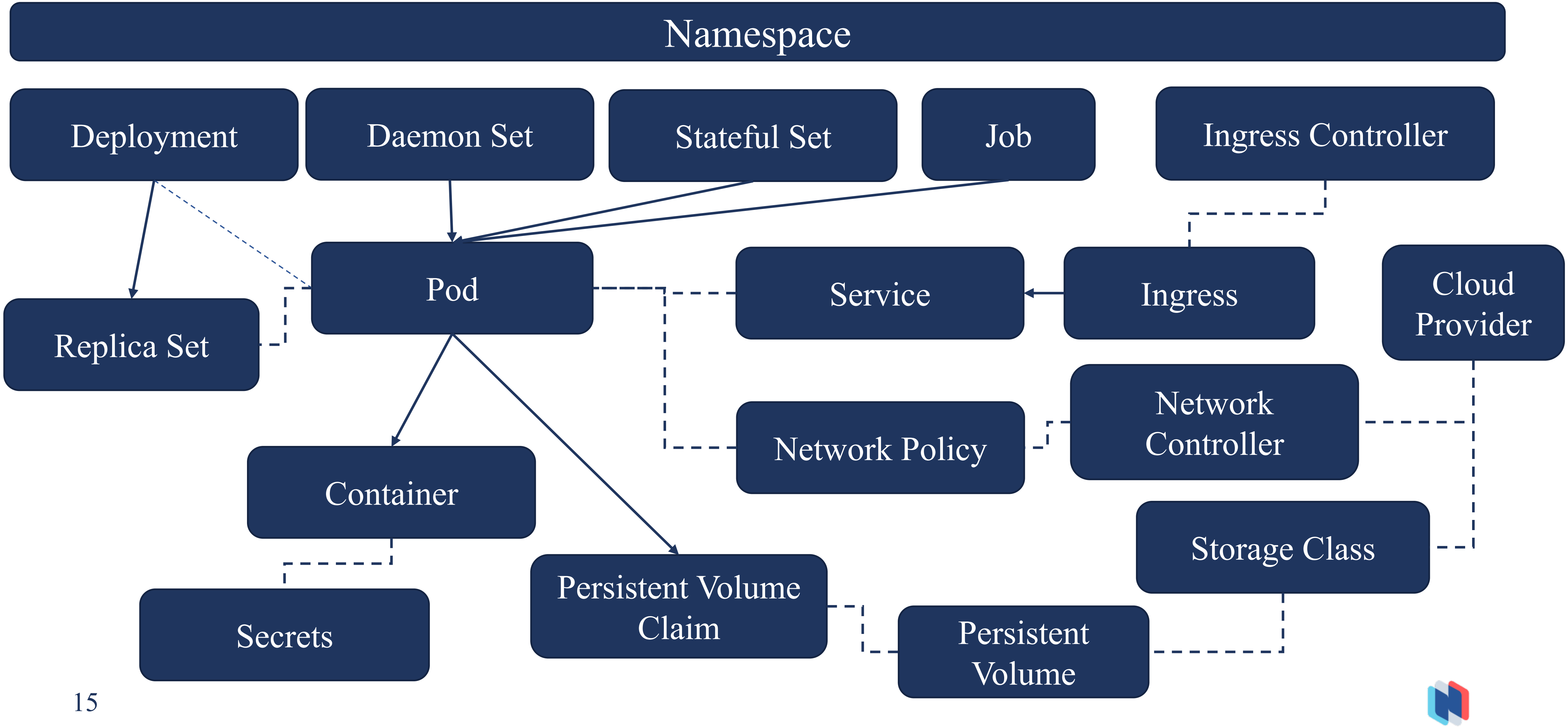
- Nodes
- Components
- Add-ons
- Cloud Provider
  - Networking
  - Storage



Setup your Azure Kubernetes Cluster

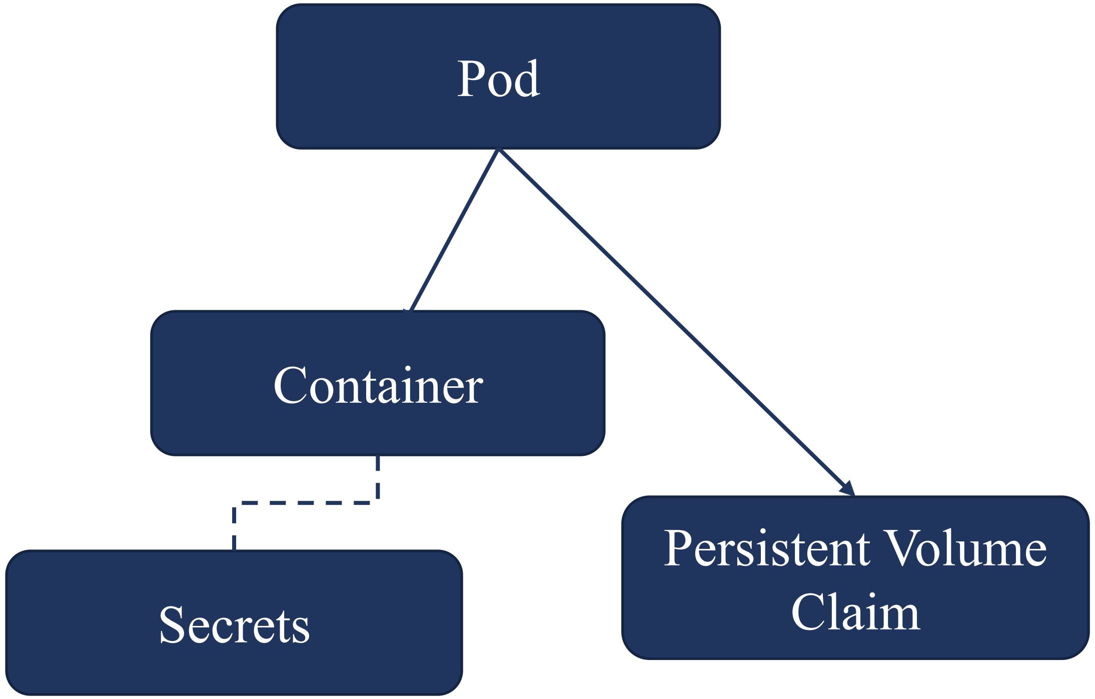
Kubernetes Application Concepts

Dude, where's my app?



Pods

- Basic unit of application deployment
- Contains
  - One or more Containers
  - One or more PVCs
- Other constructs
  - nodeSelector
  - affinity
  - serviceAccountName
  - secrets
  - initContainers

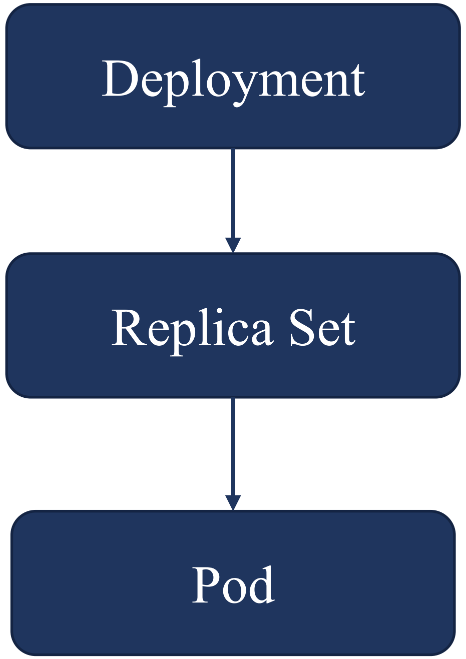


Managing Pods

- Pods can be managed individually, but *don't* do this!
- Pods lifecycles are best managed using one of:
  - Deployments
  - StatefulSets
  - DaemonSets
- Less often used:
  - ReplicaSets (Deployments manage ReplicaSets)
  - Jobs (short-lived run-to-completion tasks)

Deployment

- Deployments automatically create (*and delete*) ReplicaSets
- Rollout: a new ReplicaSet is created and scaled up. The existing ReplicaSet is scaled down.
- Rollback: only impacts the Pod template. Can rollback to a specific revision ID.
- Rolling upgrade strategy tunables:
  - maxUnavailable
  - maxSurge



StatefulSet

- Pods with stable identities
  - names, network, storage
- Ordered creation, updates, scaling, and deletion
  - Pods are created, and named, in order from {0…N-1}
- Use for clustered apps that use client-side identities
  - ZooKeeperAddresses: “zoo-1:2181, zoo-2:2181, zoo-3:2181”

DaemonSet

- Ensures that all Nodes run an instance of a Pod
- Useful for monitoring & security agents, log daemons, etc.
- A node selector can be used to target a subset of nodes

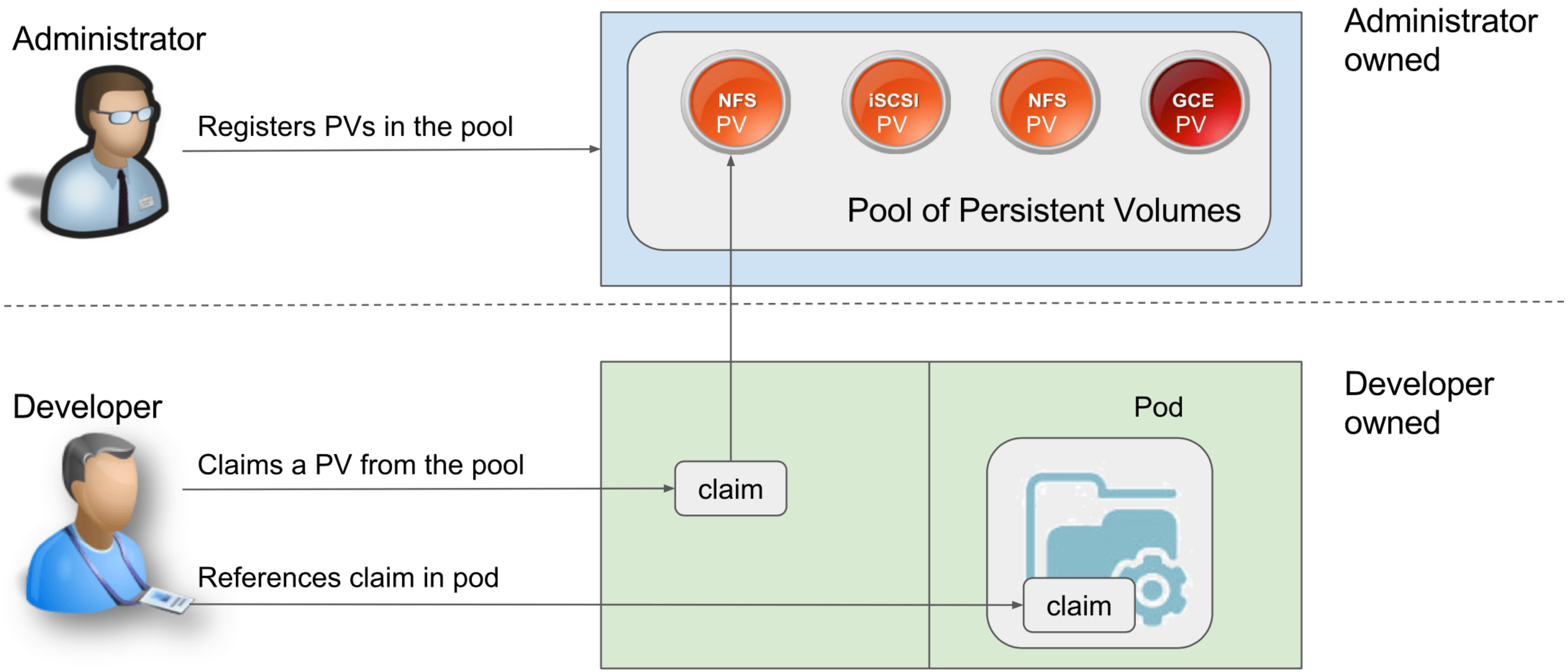
Networking your app

- Service: provides load-balancing. Addressed via IP (cluster IP) or a DNS name.

- Network Policy: manages routing rules across pods (east-west traffic.)

- Ingress: manages external routes to services (north-south traffic.) An Ingress Controller does the load-balancing. Ingress Resources specify the rules.

Configuring storage for your app



Source: Steve Watt, Red Hat

Summary

- Most apps will contain one or more services / tiers
- And each service will have:
  - Deployment → ReplicaSet → Pod → Container(s)
  - Service
  - Ingress *External facing services only*
  - Network Policy
  - Persistent Volume Claim(s)

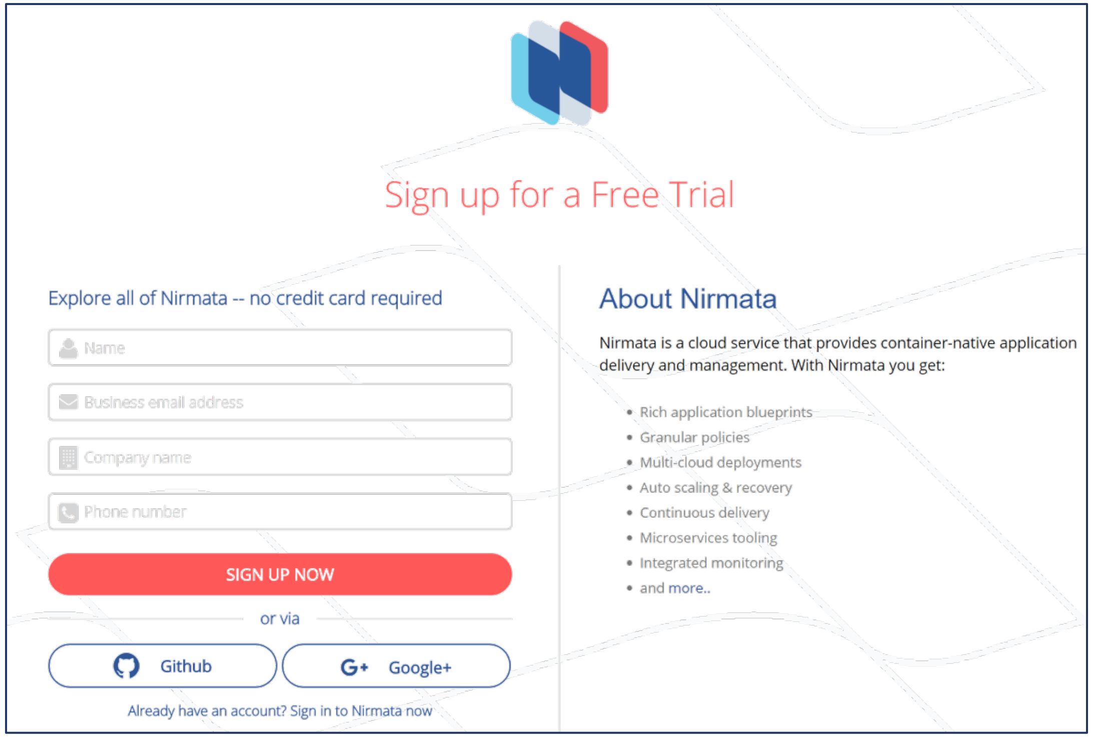
Manage your Kubernetes Applications

Signup for Nirmata

https://try.nirmata.io

Signup steps:

1. Enter your email
2. Nirmata will email you a secure access link
3. Click to setup your password and enter Nirmata!



Sign up for a Free Trial

Explore all of Nirmata -- no credit card required

About Nirmata

Nirmata is a cloud service that provides container-native application delivery and management. With Nirmata you get:

- Rich application blueprints
- Granular policies
- Multi-cloud deployments
- Auto scaling & recovery
- Continuous delivery
- Microservices tooling
- Integrated monitoring
- and more..

Already have an account? Sign in to Nirmata now

Discover your Azure Cluster

1. Go to “Clusters → Add Cluster”
2. Select Kubernetes
3. Select Option 1, “Yes…” as we
4. Provide a name and select Other as provider
5. Download and apply the Nirmata Controller YAML

Deploy an application

1. Import the blueprint from Catalog: shopme-k8s
2. Deploy an Environment
   - Select Container Manager: Kubernetes
   - Select Type: <Cluster name>
   - Select Application: shopmek8s
   - Click on Finish

Thank-you!

https://try.nirmata.com