LINUXCON containercon CHINA 中国 CLOUDOPEN

THINK OPEN

开放性思维

Not one size fits all, how to size Kubernetes clusters

Sahdev Zala / Guang Ya Liu

@sp_zala / @gyliu513

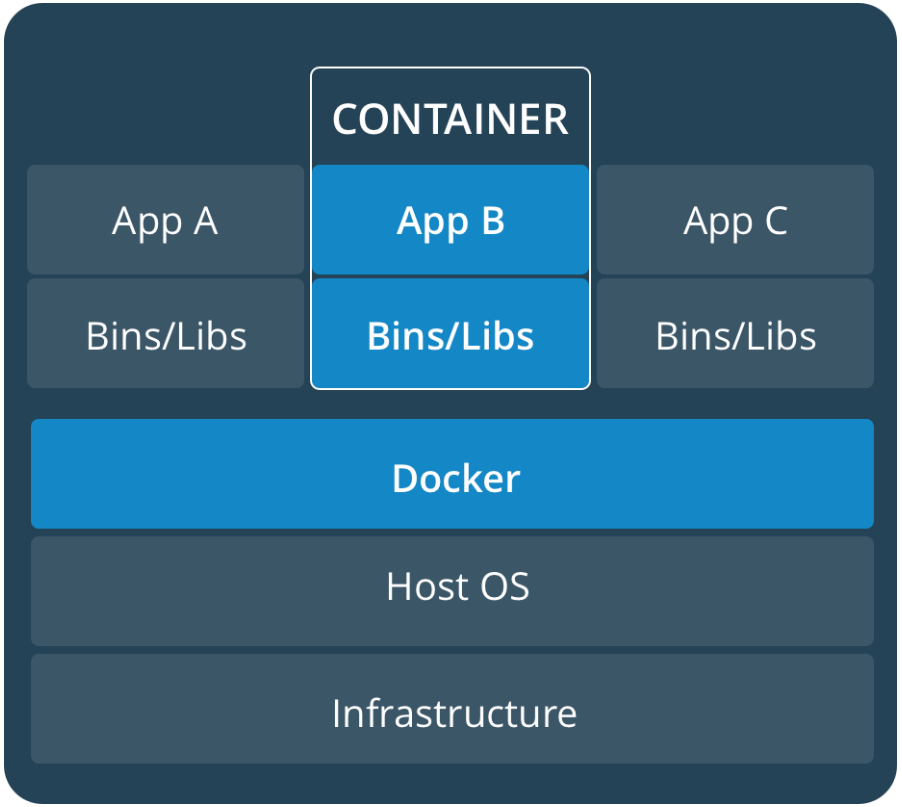
spzala@us.ibm.com / liugya@cn.ibm.com

Outline

- Some basics
  - Containers, Kubernetes
- Kubernetes cluster
  - What is it?
  - Design and sizing consideration
  - Optimization techniques
- Large scale enterprise cluster
  - How we created a 1000 node cluster
  - Lessons learned

Overview of Containers

- Abstraction at the app layer that packages code and dependencies together
- Multiple containers can run on the same machine and share the OS kernel with other containers, each running as isolated processes in user space

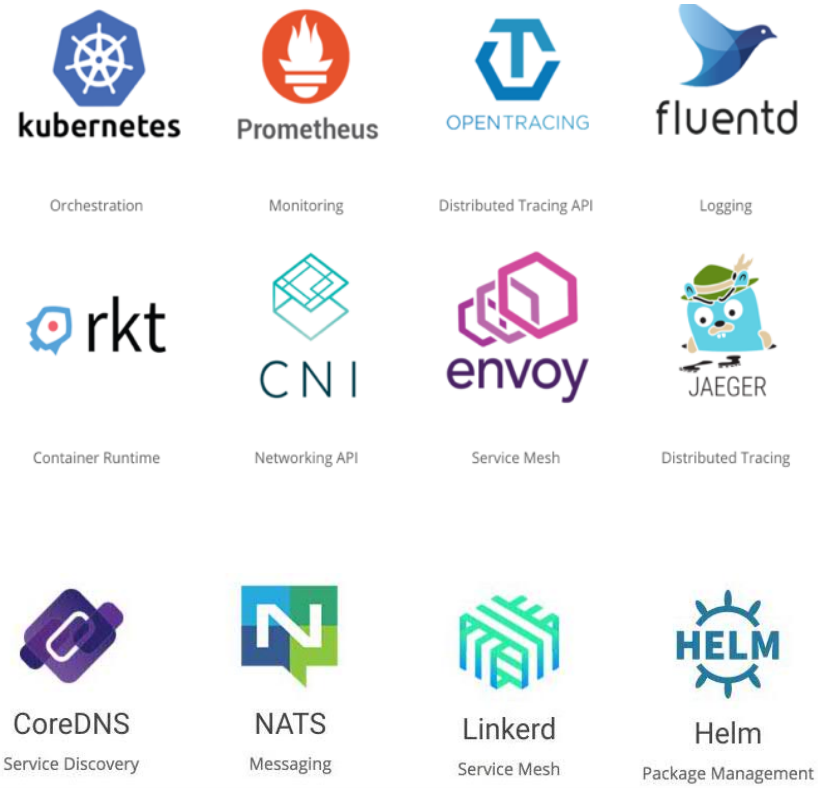


What is Kubernetes?

- Enterprise level container orchestration
- Provision, manage, scale applications (containers) across a cluster
- Manage infrastructure resources needed by applications
  - Compute
  - Volumes
  - Networks
  - And many many many more...
- Declarative model
  - Provide the "desired state" and Kubernetes will make it happen
- What's in a name?
  - Kubernetes (K8s/Kube): "Helmsman" in ancient Greek

Kubernetes Community Overview

- **C**loud **N**ative **C**omputing **F**oundation project
- Github Repositories
  - github.com/kubernetes/kubernetes
    - github.com/kubernetes/kubernetes/issues
    - github.com/kubernetes/kubernetes/pulls
  - github.com/kubernetes/website
  - github.com/kubernetes/community
- Special Interest Groups (SIGs)
- Slack channels
  - https://kubernetes.slack.com
- Mailing lists



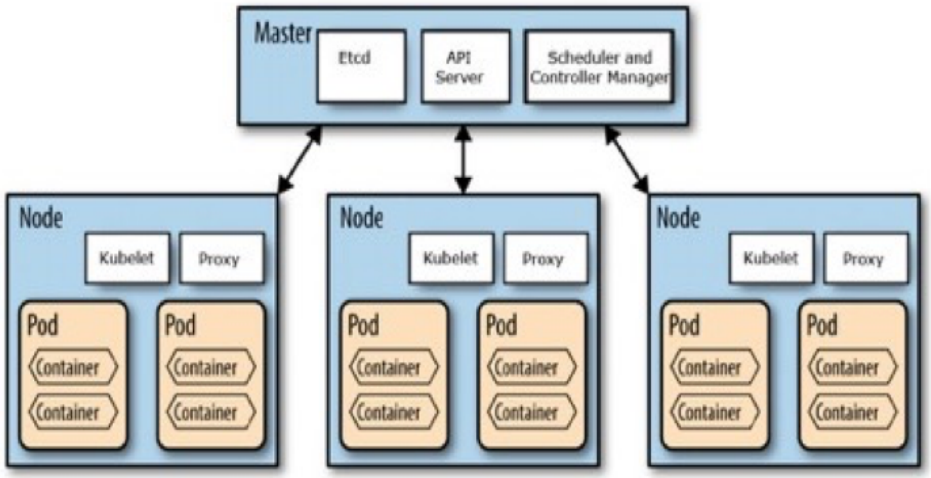
K8s – API vs Compute Resources

- Pod
- ReplicaSet
- Deployment
- Service
- ConfigMap
- Secrets
- Jobs
- …But how about your cluster and compute resources?
  - Node, CPU, Memory

Kubernetes Cluster

- A running Kubernetes cluster contains a cluster control plane (AKA *master*) and worker node(s), with cluster state backed by a distributed storage system(etcd). Cluster can be a single node to several nodes

- Kubernetes can run on various platforms – Laptop, VMs, Rack of bare metal servers. The effort required to set up a cluster varies from running a single command to crafting your own customized cluster



Kubernetes Cluster choices

- Local-machine Solutions
  - Minikube, DIND, Ubuntu on LXD
  - IBM Cloud Private Community Edition (CE) - https://hub.docker.com/r/ibmcom/cfc-installer/
  - Running Kubernetes locally has obvious development advantages, such as lower cost and faster iteration than constantly deploying and tearing down clusters on a public cloud.
- Cloud Provider Solutions
  - AKS, EKS, GKE, IKS..
    - Hosted/Managed cluster
- On-Premises Solutions
  - Allow you to create Kubernetes clusters on your internal, secure, cloud network with only a few commands
    - IBM Cloud Private, Kubermatics..
- Turnkey Solution, Custom Cluster Etc.

Factors Impacting Cluster Size

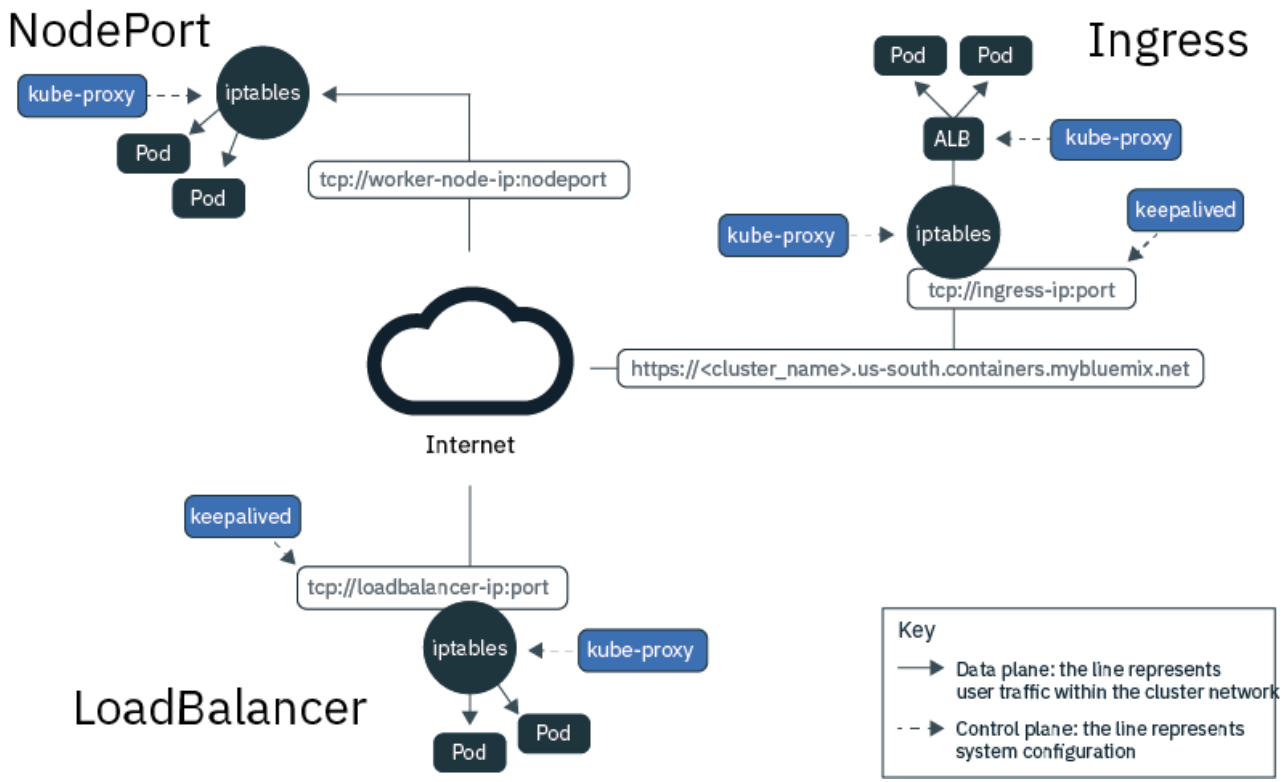
- Single node to several nodes - what's the right size for me?
- What you want to do with it?
  - Just kick off the tires – i.e. learn/play/development
  - Production level cluster
  - Managed by a cloud provider or you want to manage?
- What do you want to run on it?
  - One or few or many applications
  - Kind of applications
    - Big Data/Artificial Intelligence (AI) application
    - CPU vs Memory intensive
    - Stateless or Stateful
  - Scale vs Performance
  - Networking need
  - Monitoring, logging need
- What kind of traffic do you expect?
  - Steady heavy traffic
  - Burst traffic
- What is your your budget?
  - Hardware/Virtualization infrastructure set up
- Etc.

Scale vs Performance

- After reaching a recommended threshold, scale and performances are inversely proportional
  - Kubernetes has defined two service level objectives
    - Return 99% of all API calls in < 1sec
    - Start 99% of pods within < 5 sec
  - According to study, clusters with more than 5,000 nodes may not be able to achieve these service level objectives
  - Per what we learned, a single cluster with maximum 2500 nodes is good enough
    - Anything above, go for multi-cluster approach. This is not very stable yet but it's a WIP. Learn more here, https://github.com/kubernetes-sigs/federation-v2

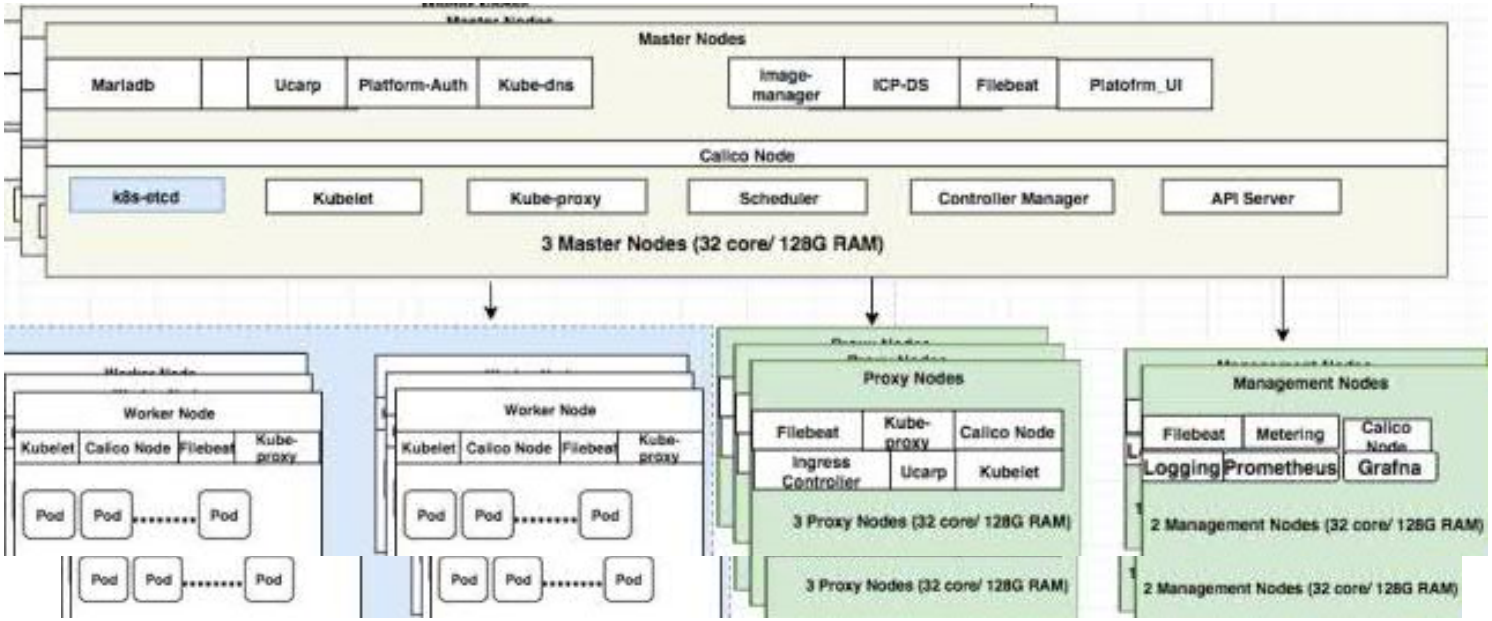
Networking Need

- NodePort, LoadBalancer, or Ingress services?
  - e.g. a Minikube cluster is not ideal if you want to expose your app with a LoadBalancer or Ingress services
- Also, what you using for networking – e.g. Calico, Flannel?



Single or Multiple Master Nodes

- A big cluster with single master may not be enough
  - You may need multiple master nodes and want to divide the load of master to multiple servers.
    - In our case,
      - We had 3 master nodes
      - Added management node for Monitoring, Logging
      - Multiple Proxy nodes



Requests and Limits

- Helps manage compute resources for containers
  - Specify how much CPU and memory (RAM) each Container needs by using requests and limits
    - ***Requests*** determines minimum cpu/memory required by container
    - ***Limits*** set the max cpu/memory allowed
- Improve scheduler efficiency
  - Allows Kubernetes to increases utilization, while at the same time maintaining resource guarantees for the containers that need guarantees

```
apiVersion: v1
kind: Pod
metadata:
  name: frontend
spec:
  containers:
  - name: db
    image: mysql
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
```

64 MiB

250 millicore/ millicpu

Node Selectors

- Provides control on how to assign a pod to nodes
- Simplest form of constrains
  - Constrain a pod to run on a specific nodes
- Useful in certain circumstances
  - Ensure that a pod ends up on a machine with an SSD attached to it

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    env: test
spec:
  containers:
  - name: nginx
    image: nginx
    imagePullPolicy: IfNotPresent
  nodeSelector:
    disktype: ssd
```

Node label

Node Affinity / Anti Affinity

- The node affinity expands the types of constraints in compare to Node Selector
  - Allows you to constrain which nodes your pod is eligible to be scheduled on, based on labels on the node
  - Two types
    - Hard – **requiredDuringSchedulingIgnoredDuringExecution**
      - *must* be met for a pod to be scheduled onto a nod
    - Soft – **preferredDuringSchedulingIgnoredDuringExecution**
      - scheduler will try to enforce but will not guarantee

```
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: kubernetes.io/e2e-az-name
            operator: In
            values:
            - e2e-az1
            - e2e-az2
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: another-node-label-key
            operator: In
            values:
            - another-node-label-value
  containers:
  - name: with-node-affinity
    image: k8s.gcr.io/pause:2.0
```

Inter-pod Affinity / Anti Affinity

- Allow you to constrain which nodes your pod is eligible to be scheduled based on labels on pods that are already running on the node rather than based on labels on nodes
- Inter-pod affinity and anti-affinity require substantial amount of processing which can slow down scheduling in large clusters significantly. We do not recommend using them in clusters larger than several hundred nodes

```
spec:
  affinity:
    podAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: security
            operator: In
            values:
            - S1
        topologyKey: failure-domain.beta.kubernetes.io/zone
    podAntiAffinity:
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 100
        podAffinityTerm:
          labelSelector:
            matchExpressions:
            - key: security
              operator: In
              values:
              - S2
          topologyKey: kubernetes.io/hostname
  containers:
  - name: with-pod-affinity
    image: k8s.gcr.io/pause:2.0
```

Taints and Tolerations

- Taints are the opposite to the Node Affinity. They are key-value pairs associated with an effect
- Together they ensure that pods are not scheduled onto inappropriate nodes
- Provides a flexible way to steer pods *away* from certain nodes

Adding taint to a node

```
kubectl taint nodes node1 key=value:NoSchedule
```

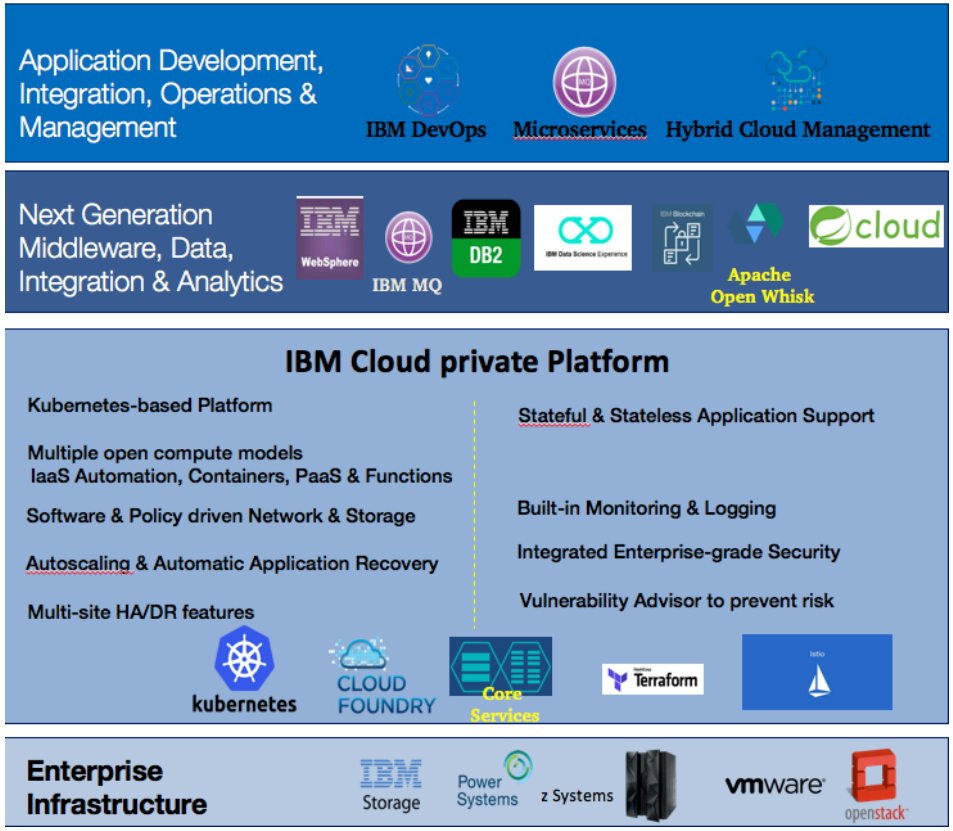
Corresponding pod

```
tolerations:
- key: "key"
  operator: "Equal"
  value: "value"
  effect: "NoSchedule"
```

Kubernetes Based IBM Cloud Private

- Kubernetes is not enough
- An enterprise Kubernetes distribution should also include some other core services for logging, monitoring etc

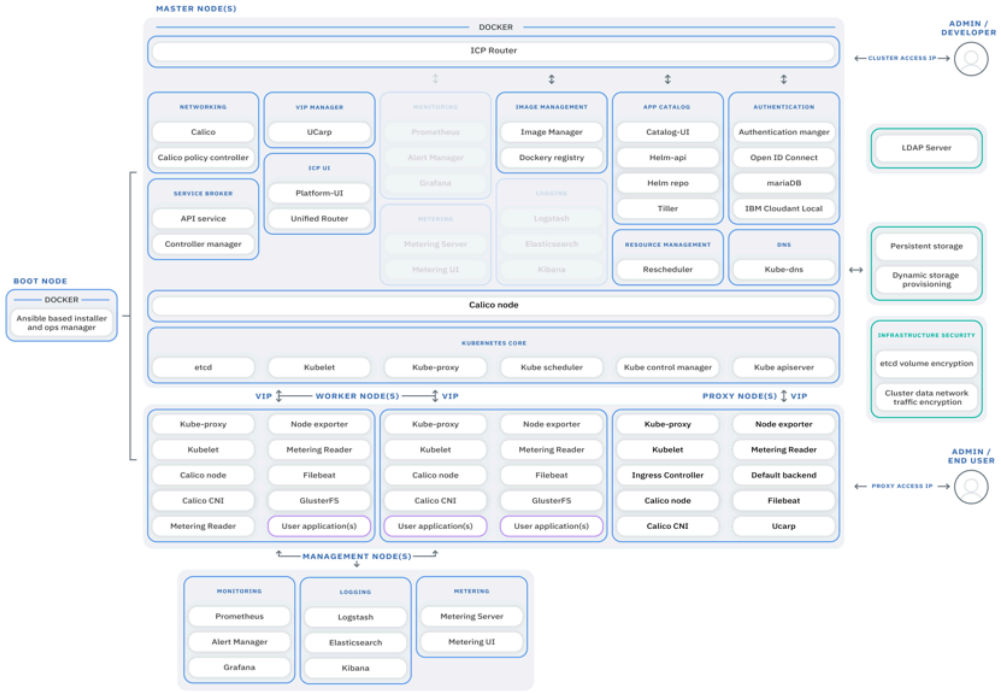
- Learn more about IBM Cloud Private at here https://www-03.ibm.com/support/knowledgecenter/SSBS6K_2.1.0.3/kc_welcome_containers.html



Deployment Topology

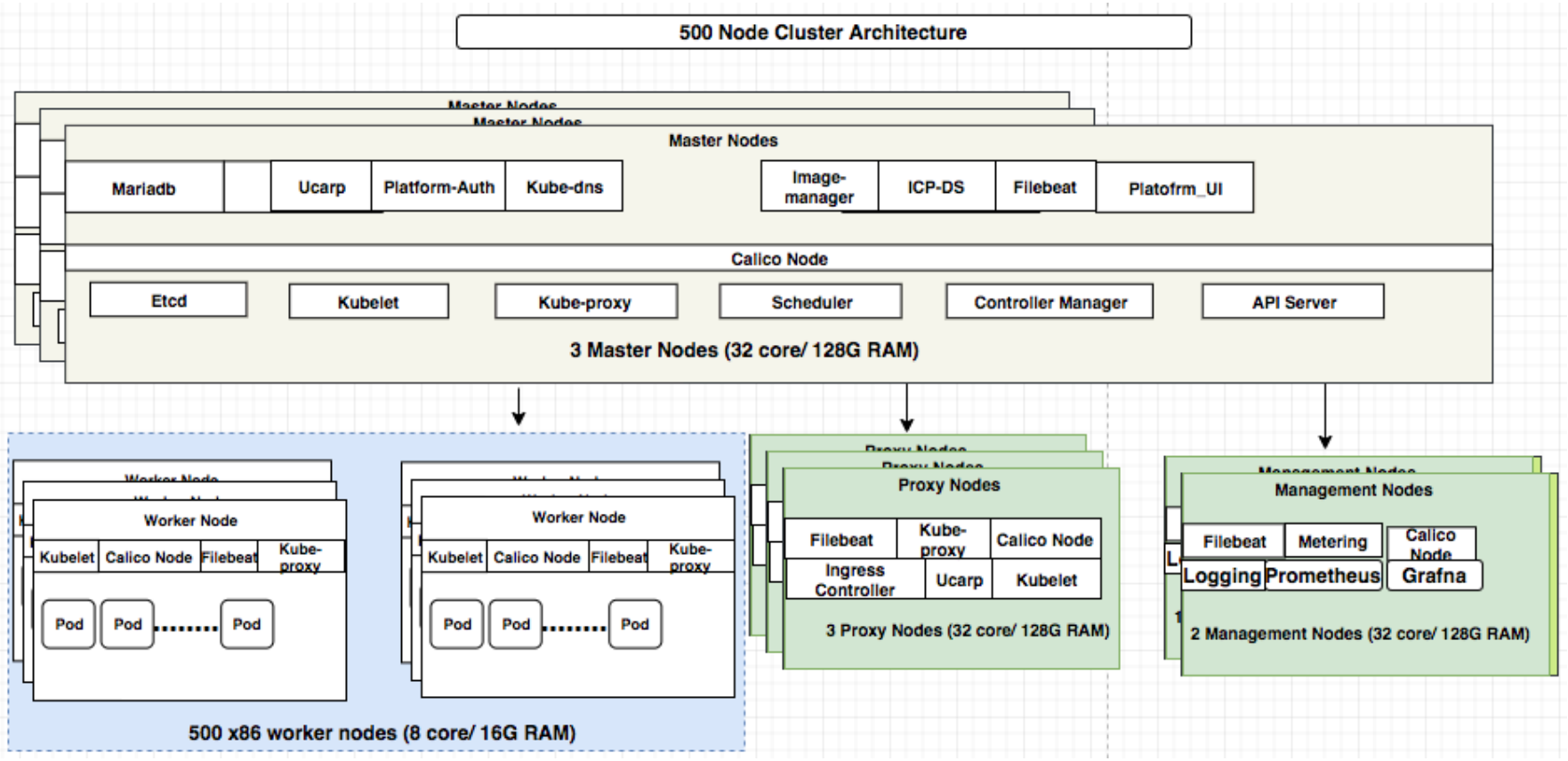
- Best Deployment
  - Master
  - Management
  - Worker
  - Proxy
  - Dynamic host group

```
[master]
9.111.255.77

[worker]
9.111.255.78
9.111.255.79

[proxy]
9.111.255.80

[management]
9.111.255.81
9.111.255.82

[hostgroup-db2]
9.111.255.83
9.111.255.84

[hostgroup-mysql]
9.111.255.85
9.111.255.86
```



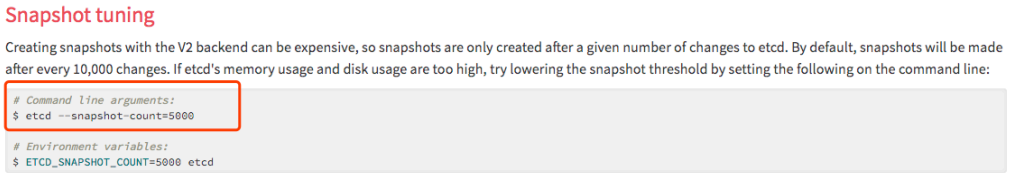
500 Nodes Deployment Arch

- IBM Cloud Private 2.1.0.2 which was released in 2018.3
- Calico V2 with Node to Node Mesh
- Sharing one etcd cluster between Kubernetes and Calicos



Network Impact for 500+ Nodes

- Kubernetes claim support 5000 nodes, why IBM Cloud Private cannot in 2.1.0.2?
  - IBM Cloud Private using calico as default network
  - IBM Cloud Private Calico using node-to-node mesh to configure peering between all calico nodes.
  - etcd load is very high when deploying 1000 node cluster, most load is from calico
  - Node-to-node mesh stops working if there are more than 700 nodes in the cluster.
  - Mesh number would be 1000! in a 1000 node cluster which is not acceptable!



Snapshot tuning

Creating snapshots with the V2 backend can be expensive, so snapshots are only created after a given number of changes to etcd. By default, snapshots will be made after every 10,000 changes. If etcd's memory usage and disk usage are too high, try lowering the snapshot threshold by setting the following on the command line:

```
# Command line arguments:
$ etcd --snapshot-count=5000

# Environment variables:
$ ETCD_SNAPSHOT_COUNT=5000 etcd
```

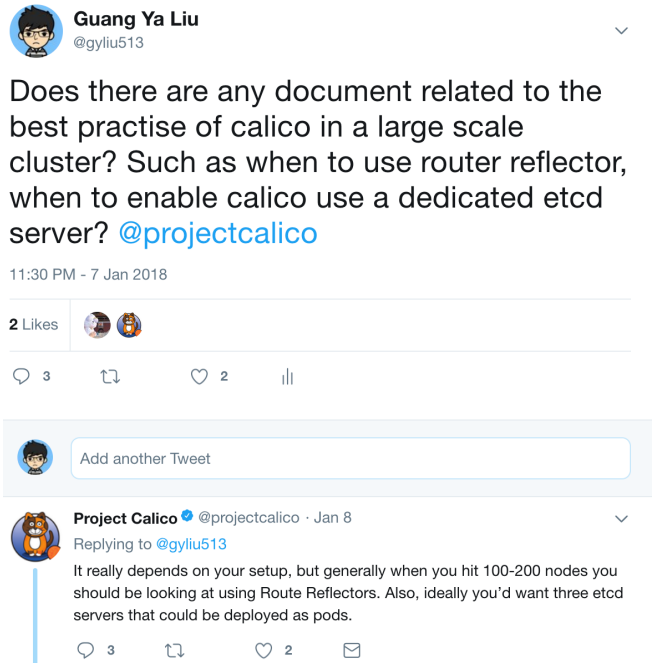
Requirements

- An existing Kubernetes cluster running Kubernetes >= v1.1. To use network policy, Kubernetes >= v1.3.0 is required.
- An `etcd` cluster accessible by all nodes in the Kubernetes cluster
  - Calico can share the etcd cluster used by Kubernetes, but in some cases it's recommended that a separate cluster is set up. A number of production users do share the etcd cluster between the two, but separating them gives better performance at high scale.

Calico

In the tested architecture Calico was configured without route reflectors for BIRD BGP daemons. Therefore, Calico established a full mesh connections between all nodes in the cluster. This operation took significant time during node startup.

It is recommended to configure route reflectors for BGP daemons in all cases at scale of 1000 nodes. This will reduce the number of BGP connections across the cluster and improve startup time for nodes.



Guang Ya Liu
@gyliu513

Does there are any document related to the best practise of calico in a large scale cluster? Such as when to use router reflector, when to enable calico use a dedicated etcd server? @projectcalico

11:30 PM - 7 Jan 2018

2 Likes

3 2

Add another Tweet

Project Calico @projectcalico · Jan 8

Replying to @gyliu513

It really depends on your setup, but generally when you hit 100-200 nodes you should be looking at using Route Reflectors. Also, ideally you'd want three etcd servers that could be deployed as pods.

3 2

https://docs.projectcalico.org/v2.6/getting-started/kubernetes/installation/integration#requirements
http://fuel-ccp.readthedocs.io/en/latest/design/k8s_1000_nodes_architecture.html
https://coreos.com/etcd/docs/latest/tuning.html

ETCD Benchmark Test

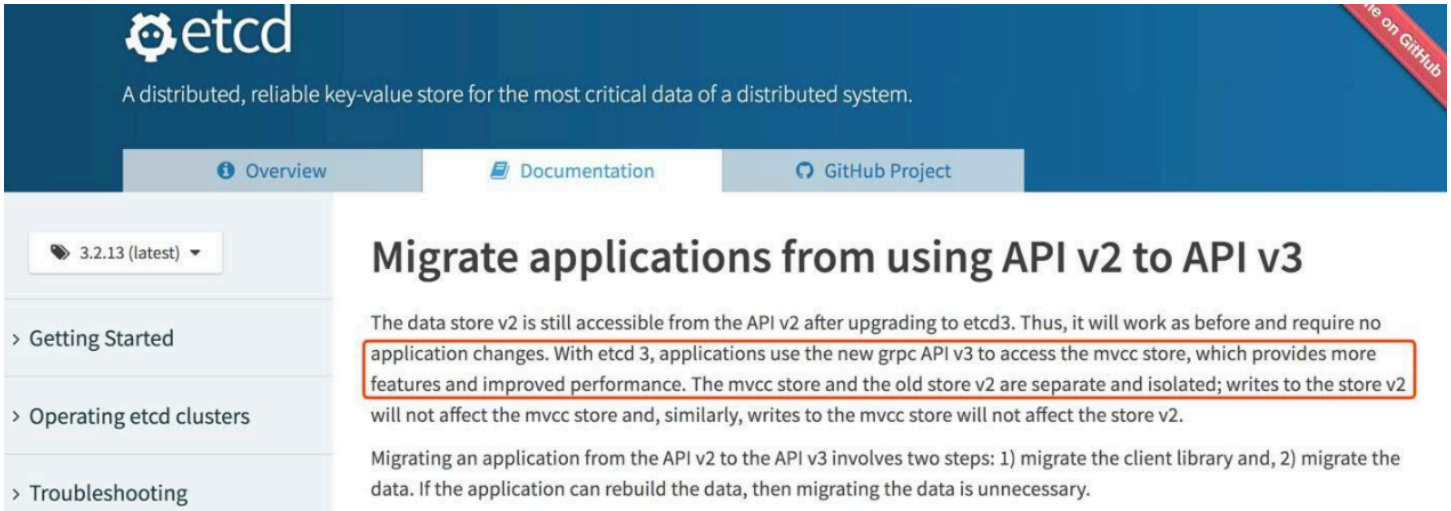
- ETCD Benchmark Comparison
  - Calico V2 with ETCD V2 API
  - Calico V3 with ETCD V3 API
- Conclusion
  - Migrate to Calico V3 and use ETCD V3 API for IBM Cloud Private

1. The etcd benchmark when calico v2 added

Number of Keys	Key size(byte)	Number of connections	Number of Clients	WRITE QPS	LATENCY per Request(ms)
10,000	8	1	100	53	18
100,000	8	100	1000	9234	101

2. The etcd benchmark when calico v3 added

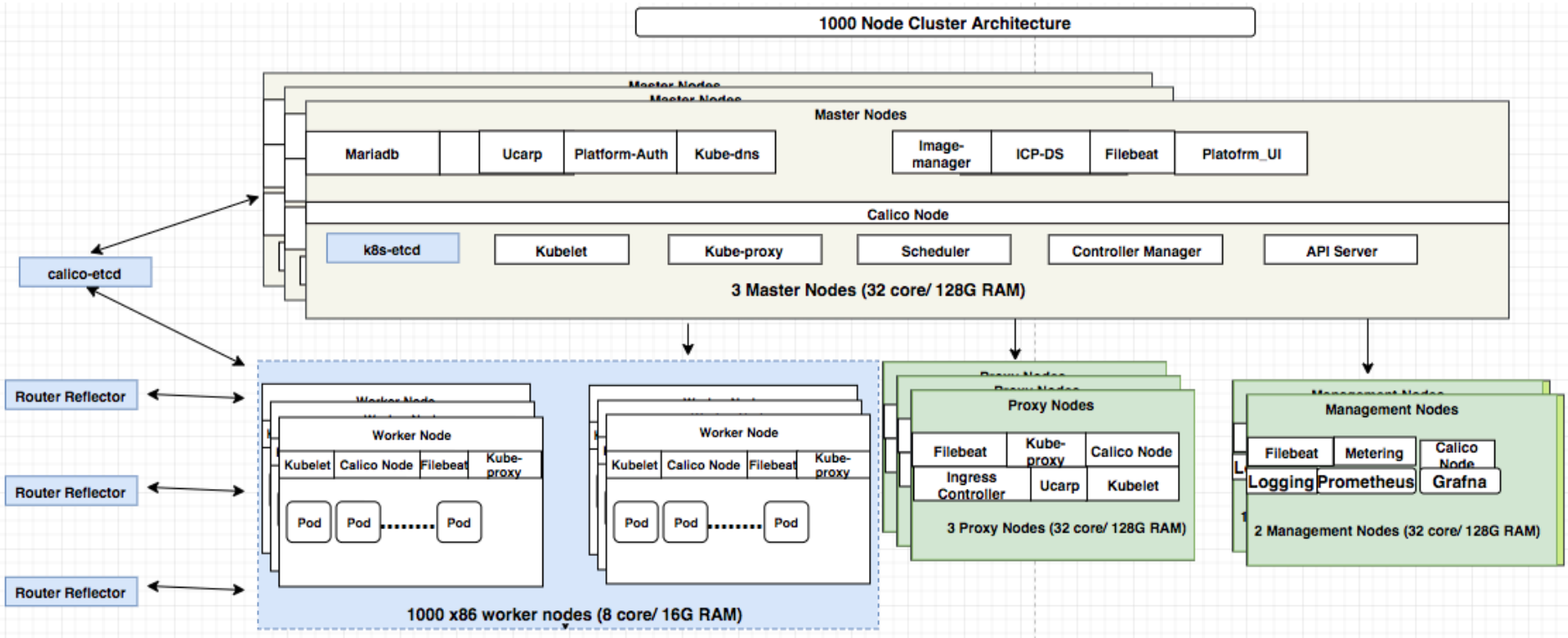
Number of Keys	Key size(byte)	Number of connections	Number of Clients	WRITE QPS	LATENCY per Request(ms)
10,000	8	1	100	190	5.2
100,000	8	100	1000	16020	64



https://coreos.com/etcd/docs/latest/op-guide/v2-migration.html

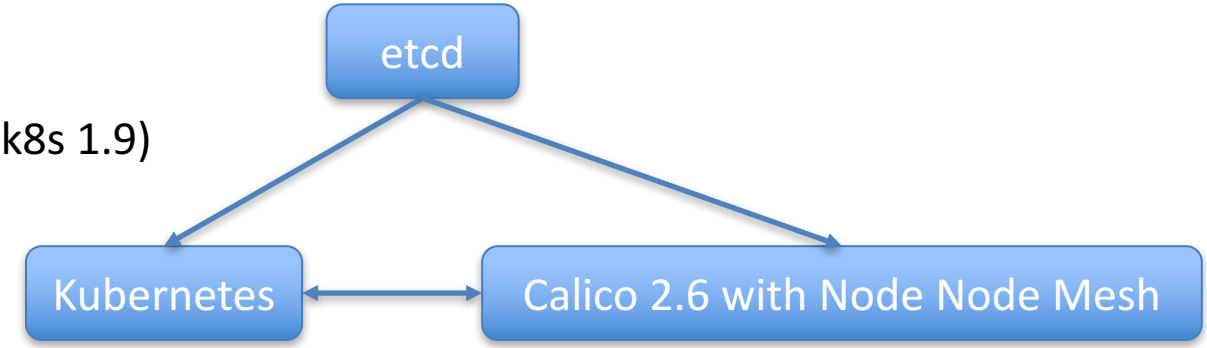
1000+ Nodes Deployment Arch

- IBM Cloud Private 2.1.0.3 which was released in 2018.5
- Calico V3 with Router Reflector
- ETCD V3

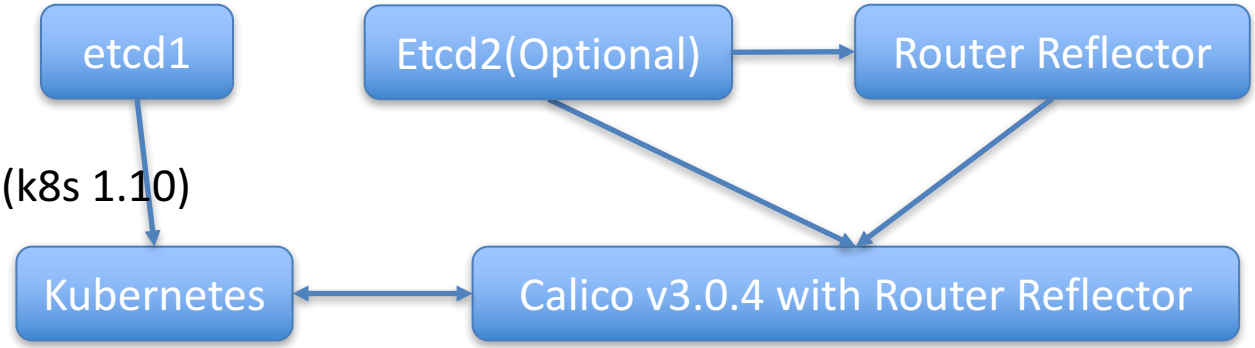


Deployment Topology Changes

IBM Cloud Private 2.1.0.2 (k8s 1.9)



IBM Cloud Private 2.1.0.3 (k8s 1.10)



Summary

- Sizing Kubernetes cluster can be challenging specially for large scale cluster
- Be benefitted from the experience of others
  - Do good research on what others recommend. Learn from already proven approaches.
- Understand scheduler optimization techniques in Kubernetes
- Etcd storage with SSD for better performance

Thank You!!

THANK YOU!!


LINUXCON
containercon
CLOUDOPEN
CHINA 中国
THINK OPEN
开放性思维
LF ASIA, LLC