KubeCon

North America 2017

Cost-effective Compute Clusters with Spot and Preemptible Instances

Presenters

Slides
http://bit.ly/cost-effective-k8s

Arun Sriraman
Software Engineer
Platform9 Systems

Bich Le
Co-founder & Chief Architect
Platform9 Systems

About Platform9 Systems

deploy, manage & maintain { kubernetes | openstack | fission } as a service

on the infrastructure of your choice

Agenda

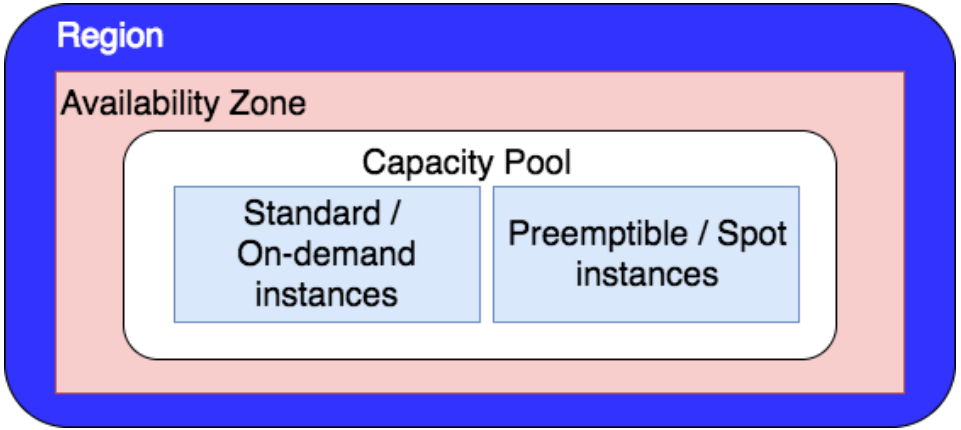
- Motivation
- Bidding Strategies
- Implementation strategies
- Supporting K8s mechanisms
- Application Scheduling considerations
- Case Study
- Demo

Introducing spot & preemptible instances

- Cheaper instances (60-80% savings)
- But with a catch: can be terminated any time
  - AWS: random
  - Google Cloud: within 24 hours
- Historically requires some skill
- But Kubernetes makes them easier to use and mainstream
  - Apps designed to tolerate node failure

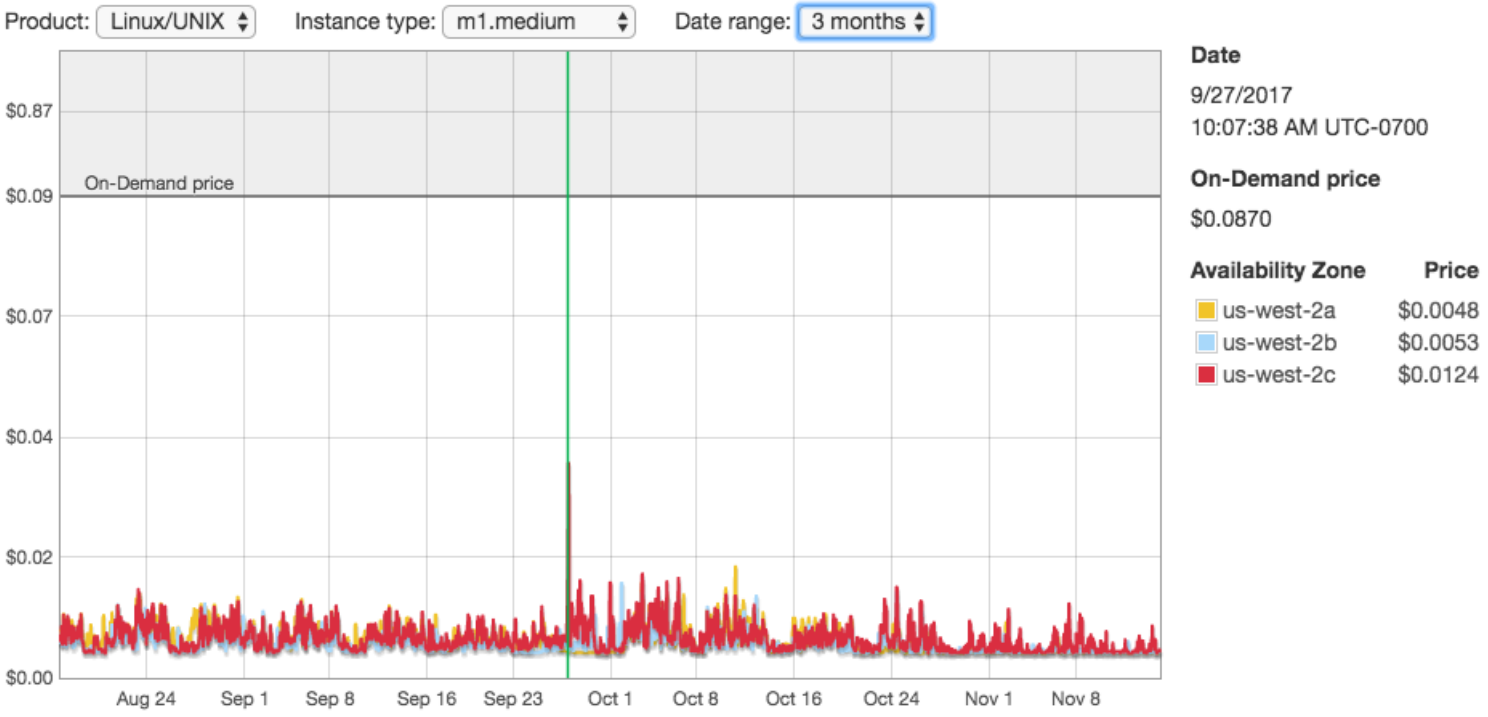
Bidding strategies

- Capacity pool (CP)
  - Logical container
  - Shares same AZ, region, OS and instance type
- Best practices
  - Build Price-Aware Applications
  - Check the Price History
  - Use Multiple Capacity Pools

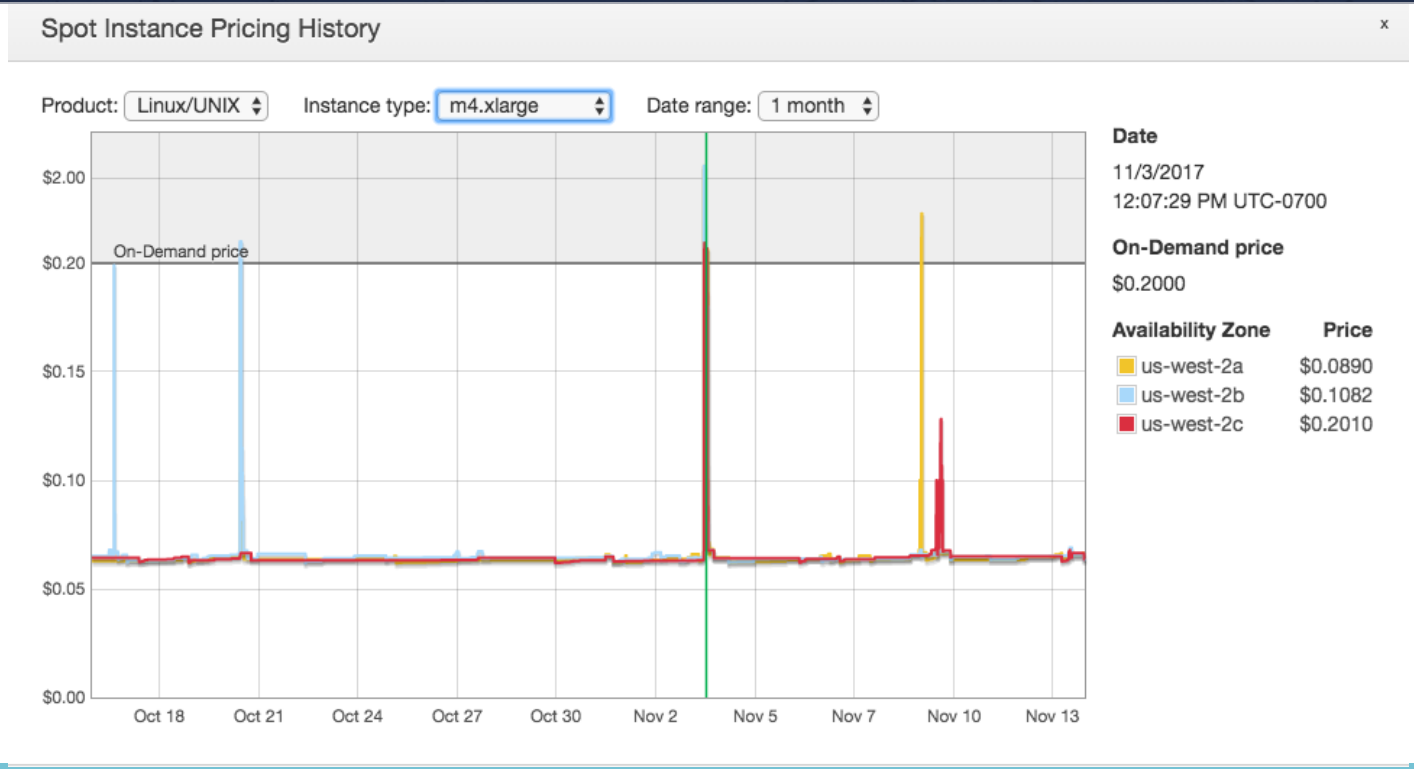


Source: https://aws.amazon.com/blogs/aws/focusing-on-spot-instances-lets-talk-about-best-practices/

Amazon EC2 Spot Instances



Amazon EC2 Spot Instances



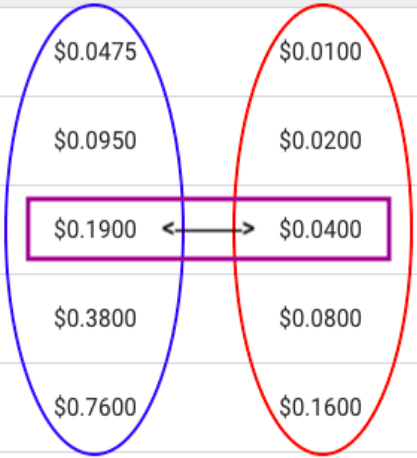
Amazon EC2 Spot Instance Pricing History

Region: Oregon (us-west-2)

Google Cloud

Oregon ▾ | Monthly ⟷ Hourly

Machine type	Virtual CPUs	Memory	Price (USD)	Preemptible price (USD)
n1-standard-1	1	3.75GB	$0.0475	$0.0100
n1-standard-2	2	7.5GB	$0.0950	$0.0200
n1-standard-4	4	15GB	$0.1900	$0.0400
n1-standard-8	8	30GB	$0.3800	$0.0800
n1-standard-16	16	60GB	$0.7600	$0.1600



~80% flat discount on list price

Excess/surplus capacity (not a secondary market)

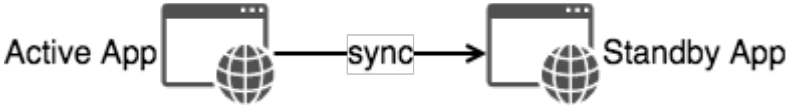
Benefits

Specific applications & use cases that benefit this scheme

- Elastic / bursting applications

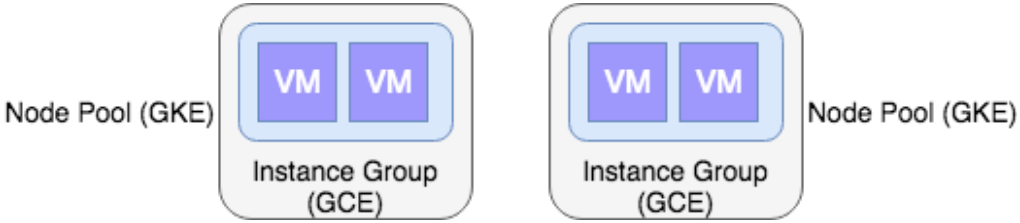
- Stateless compute intensive tasks (HPC workloads)

- Highly available clustered apps



- Horizontal node auto-scaling scenarios

Implementation Mechanisms - GKE



- Use multiple NodePools for a given cluster
- At least one node pool without preemptible instances
- NodePools can be added to a cluster and scaled dynamically
- Auto-upgrade and auto-repair support for ContainerOS image today.
- Future-proofing using 0 size node pool

Implementation Mechanisms - GKE

Two pools in the cluster - A fixed pool and a default-pool that has preemptible nodes enabled

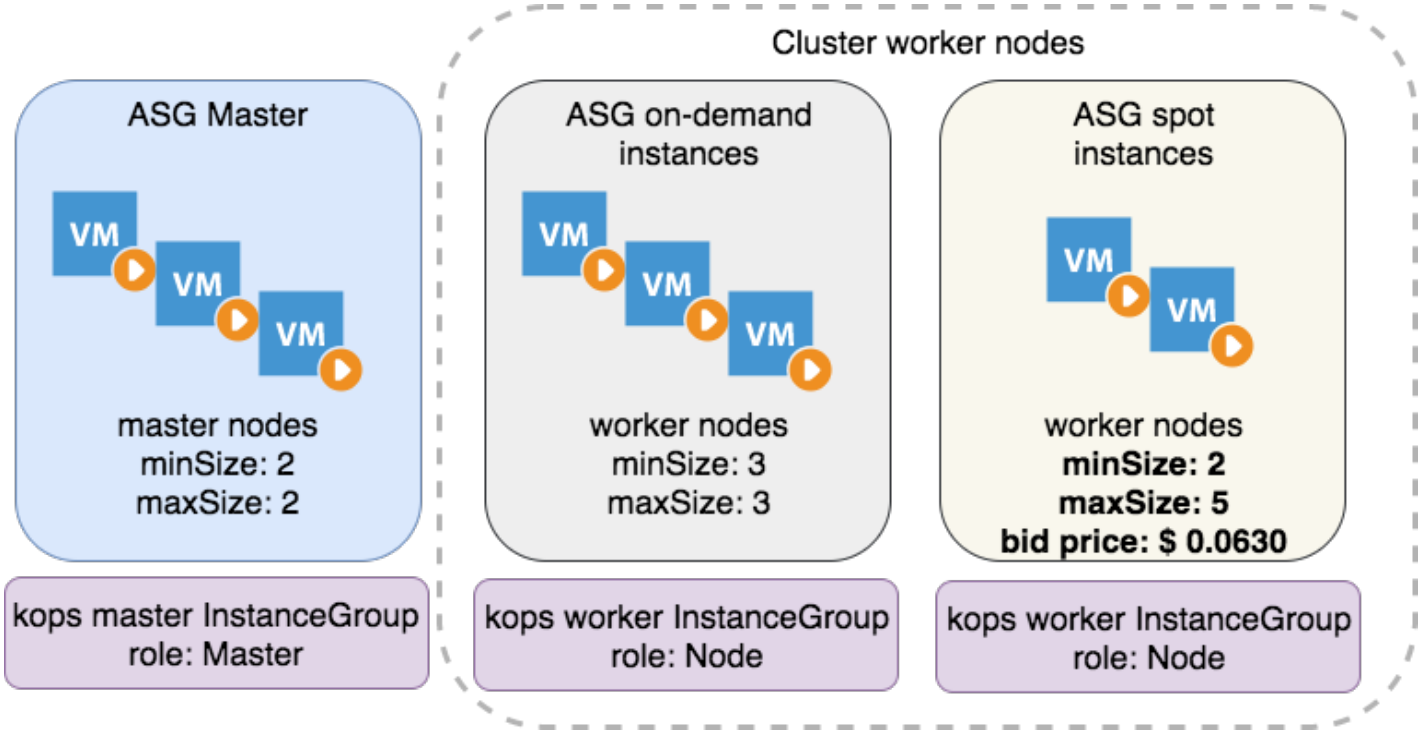
Name	fixed-pool	
Size	1	
Node version	1.8.3-gke.0	Change
Node image	Container-Optimized OS (cos)	Change
Machine type	n1-standard-1 (1 vCPU, 3.75 GB memory)	
Total cores	1 vCPU	
Total memory	3.75 GB	
Automatic node upgrades	Disabled	
Automatic node repair	Disabled	
Autoscaling	Off	
Preemptible nodes	Disabled	
Boot disk size in GB (per node)	10	
Local SSD disks (per node)	0	
Instance groups	gke-arun-kubecon-1-fixed-pool-d59c14c6-grp	

Name	default-pool	
Size	1	
Node version	1.8.3-gke.0	Change
Node image	Container-Optimized OS (cos)	Change
Machine type	custom (1 vCPU, 1 GB memory)	
Total cores	1 vCPU	
Total memory	1.00 GB	
Automatic node upgrades	Disabled	
Automatic node repair	Disabled	
Autoscaling	Off	
Preemptible nodes	Enabled	
Boot disk size in GB (per node)	10	
Local SSD disks (per node)	0	
Instance groups	gke-arun-kubecon-1-default-pool-a0176adb-grp	

Implementation Mechanisms - kops with AWS

- Open Source project - https://github.com/kubernetes/kops
- Supports deploying K8s clusters on AWS & GCE
- Supports Spot Instance for AWS
- Concept of Instance Groups (IG)
  - Master IG
  - Multiple node IG (workers)
- Each IG backed by an Auto Scaling Group (ASG)
- Ability to auto-scale and heal instance terminations

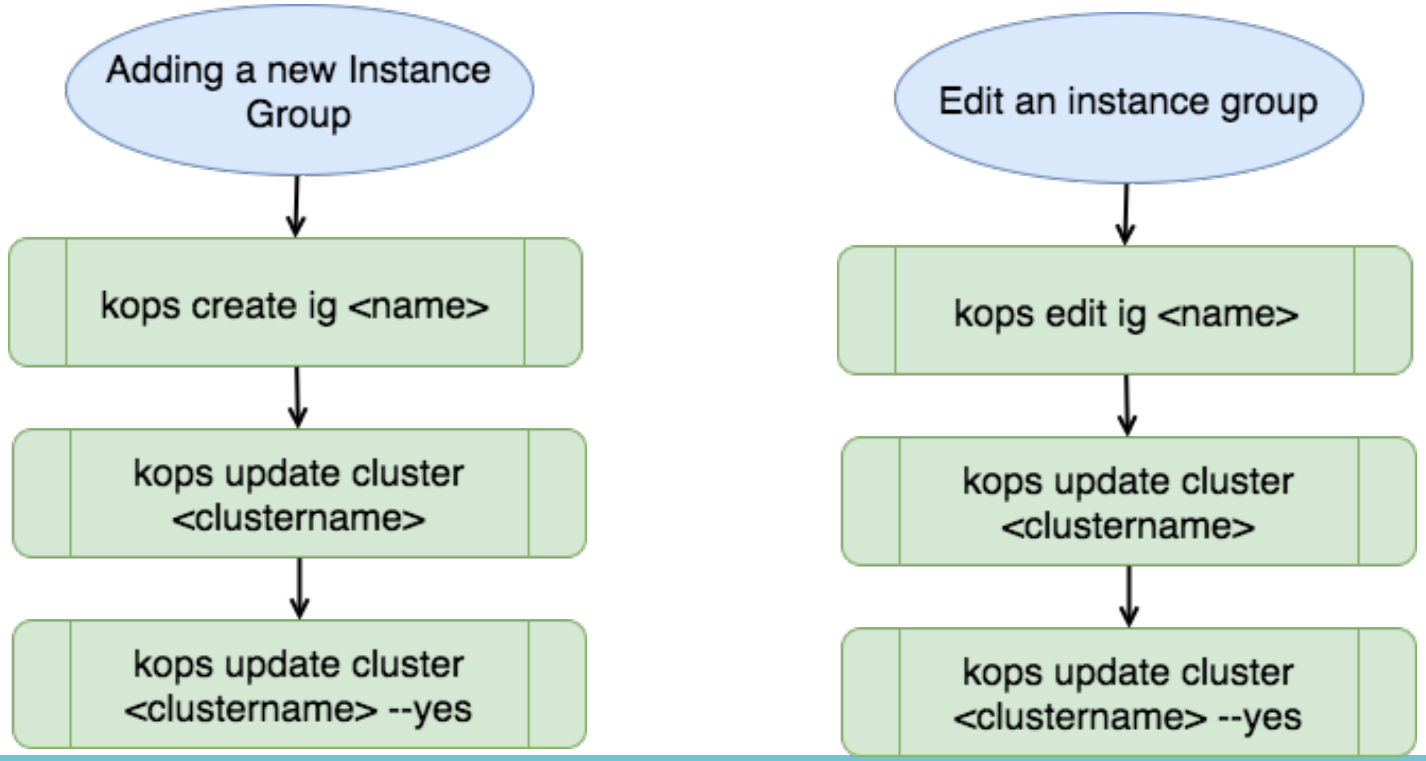
Implementation Mechanisms - kops with AWS



Implementation Mechanisms - kops with AWS

```
apiVersion: kops/v1alpha2
kind: InstanceGroup
metadata:
    labels:
    kops.k8s.io/cluster: arun-kops02.k8s.local
    name: nodes
spec:
  image: kope.io/k8s-1.7-debian-jessie-amd64-hvm-ebs-2017-07-28
  machineType: t2.medium
  maxPrice: "0.0630"
  maxSize: 2
  minSize: 2
  role: Node
  subnets:
   - us-west-2a
```

Implementation Mechanisms - kops with AWS



Demo

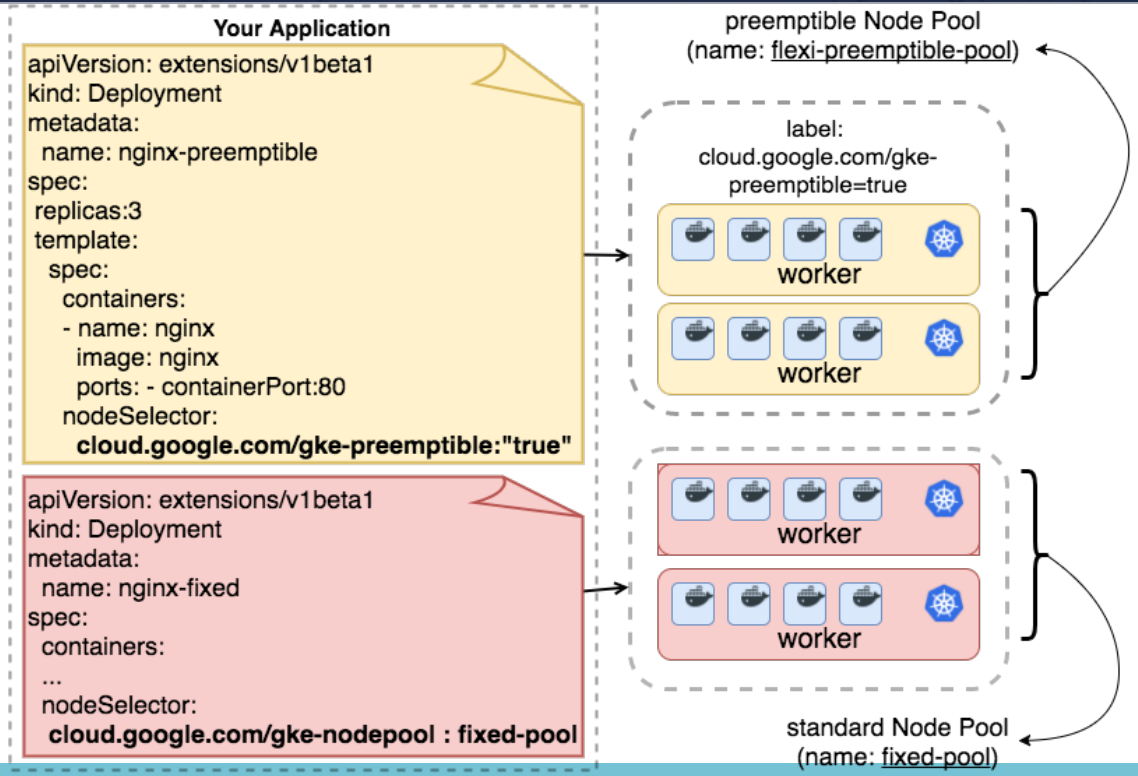
1. Cluster creation using kops on AWS

1. Horizontal pod auto scaling + Node autoscaling on GKE
   (Cloud bursting use-case)

Application scheduling considerations

- Stateless applications vs stateful applications
  - Minimum service availability
- Application replica distribution across nodes
- Node failure rescheduling considerations
  - Moving pods to same pool/different node pool
- Specific hardware requirements, eg. GPU processing, network

Supporting K8s mechanisms



NodeSelector

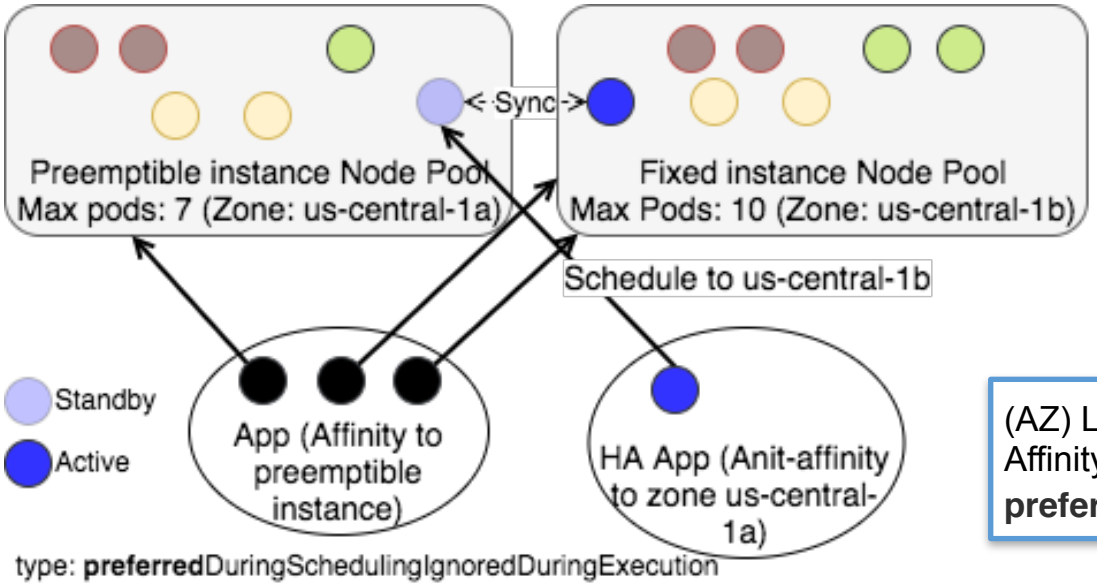
- Preemptible VMs come with a default label

cloud.google.com/gke-preemptible=true

- NodePools also have labels that can be used for scheduling decisions

Supporting K8s mechanisms

Using node affinity & anti-affinity with spot instances



- Affinity: Preferred preemptible resource for pod but not mandatory
- Anti-affinity: Preferred fixed node resource for pod but not mandatory

(AZ) Label: **failure-domain.beta.kubernetes.io/zone**
Affinity type:
preferredDuringSchedulingIgnoredDuringExecution

Demo

Application availability - nodeSelector and affinity using a GKE cluster

Case Study / Experiment

CloudProvider: GKE

Resource pool: 2 node cluster

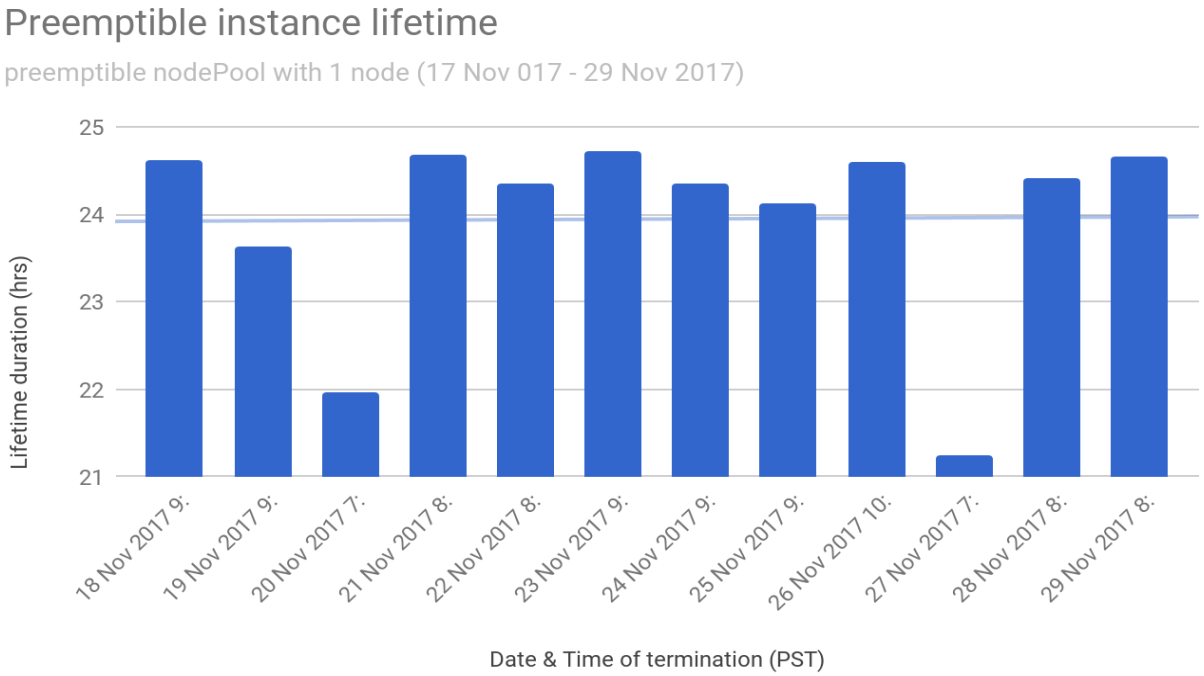
% split of preemptive and fixed nodes: 50%

K8s version: 1.8.3-gke.0

Duration: 12 days

Active workload: none

Case Study / Experiment



Observations

- Preemptible instance price fluctuation very low to none
- trend/average instance lifetime ~24 hrs
- Does not support shutdown hooks today
- Cannot turn off/on preemptive instances after Node Pool creation

Case Study / Experiment

Cost Analysis

Product	Resource	Usage	Amount
Compute Engine	Standard Intel N1 1 VCPU running in Americas	283.00 Hour	$13.44
Compute Engine	Preemptible Custom instance Core running in Americas	281.74 Hour	$1.97
Compute Engine	Storage PD Capacity	5,942.33 Gibibyte-hour	$0.33
Compute Engine	Preemptible Custom instance Ram running in Americas	281.74 Gibibyte-hour	$0.26
Compute Engine	Network Inter Zone Egress	1,887.05 Mebibyte	$0.02

Total Costs: $16.02

Case Study / Experiment

Total costs without preemptible instances: $24.65 (13.44 * 2 - 1.97 - .26)

Total Savings: $8.63 ($24.65 - $16.02)

Extrapolating to 100 node cluster, savings for a year would be: (100 / 2) * 8.63 * (365 / 12) = **$13,126.23**

Since our costing was for 2 nodes run for twelve days with 50% preemptive instances (50 fixed price nodes + 50 preemptible nodes)**

Thank You

- Slides
  - http://bit.ly/cost-effective-k8s
- For more info
  - www.platform9.com
- Please take a moment to provide your feedback
  - https://sayat.me/arunsriraman/