Scalability
&
Reliability

on Sept 7th 2019

bb bigbasket

India's largest online supermarket

- Nanda Kishore
- Sr. Principal Engineer, Platform Engineering
- https://twitter.com/none_daa
- https://www.linkedin.com/in/none-da/

Disclaimer

These are our learnings @ **BB**.

YMMV!

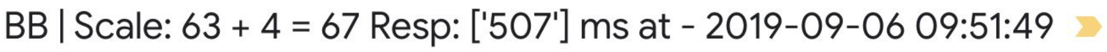
Scale

- 28 Cities.
- 12 M customers.
- 250K Orders / day.
- 40K+ Products.
- 3x by March 2020.

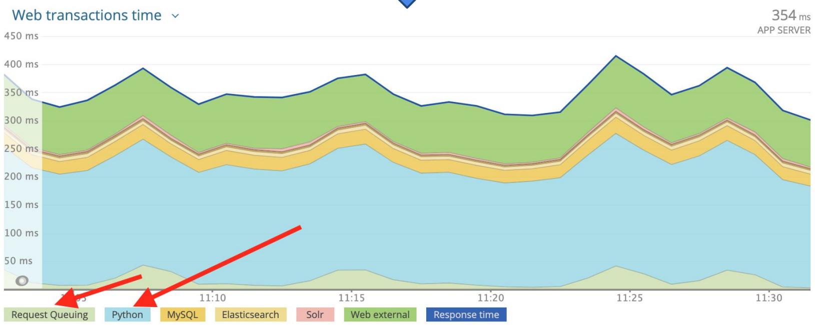
- 100 Million Edge hits per day.
- 30 Million origin hits per day.
- 120K Cache Reads per sec.
- 350+ servers on AWS.
- Multiple Services (Monolith, 20+ Microservices)
- Multiple Environments for every service.

Scalability Challenges

- Iteration 1: Scaling on CPU & Memory is not enough, nor possible always.
- Iteration 2: Scaling based on Avg response time (Newrelic)

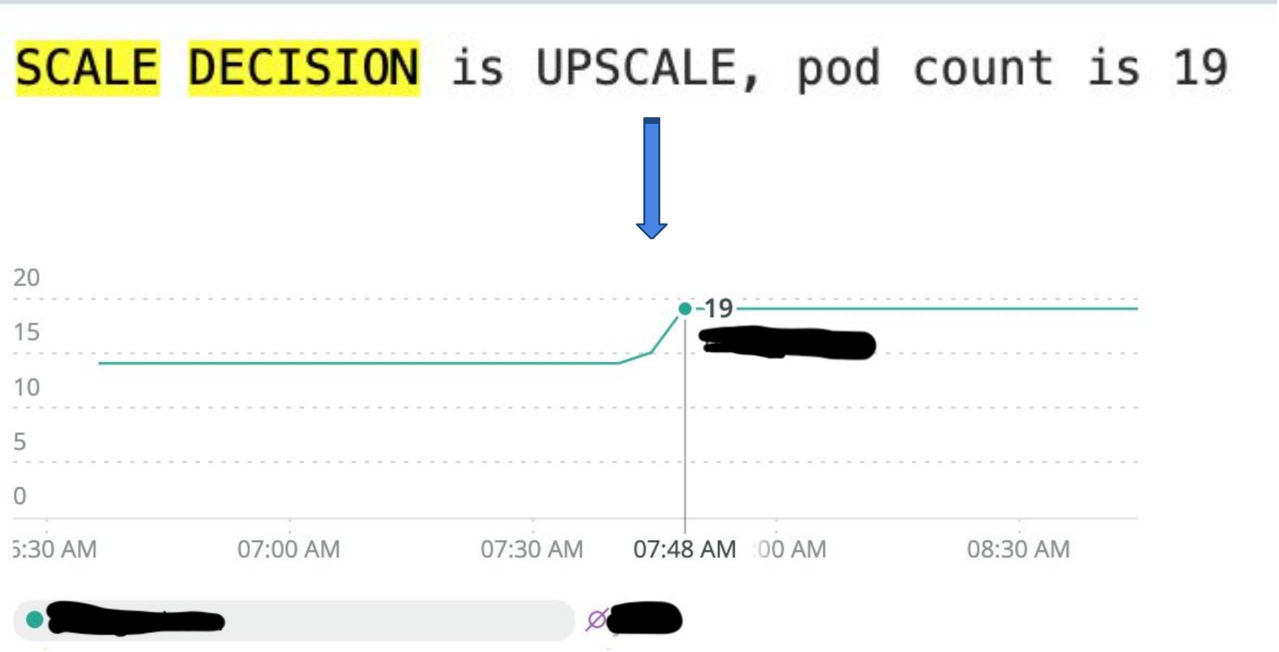


BB | Scale: 63 + 4 = 67 Resp: ['507'] ms at - 2019-09-06 09:51:49



Scalability Challenges

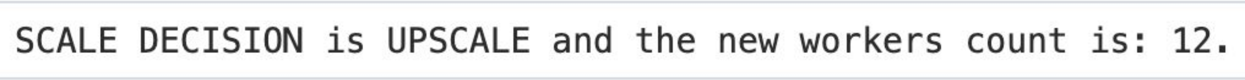
- Iteration 3: Scaling based on Requests count



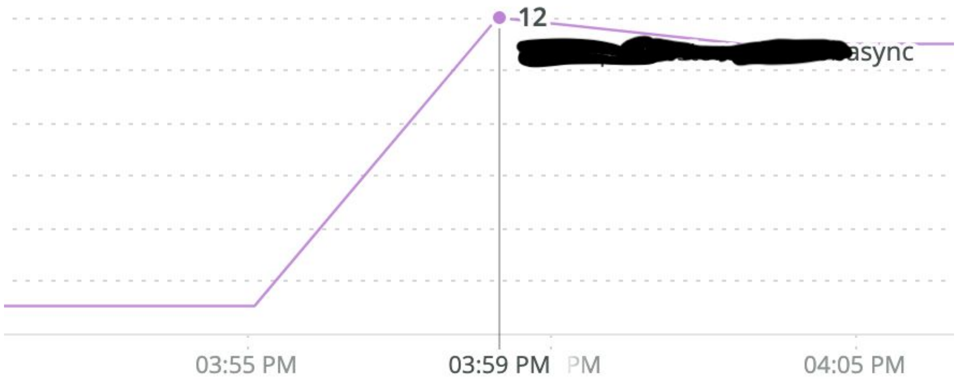
```
SCALE DECISION is UPSCALE, pod count is 19
```

Scalability Challenges

- Iteration 4: Scaling based on Queue Lag (For Kafka Consumers)



```
SCALE DECISION is UPSCALE and the new workers count is: 12.
```

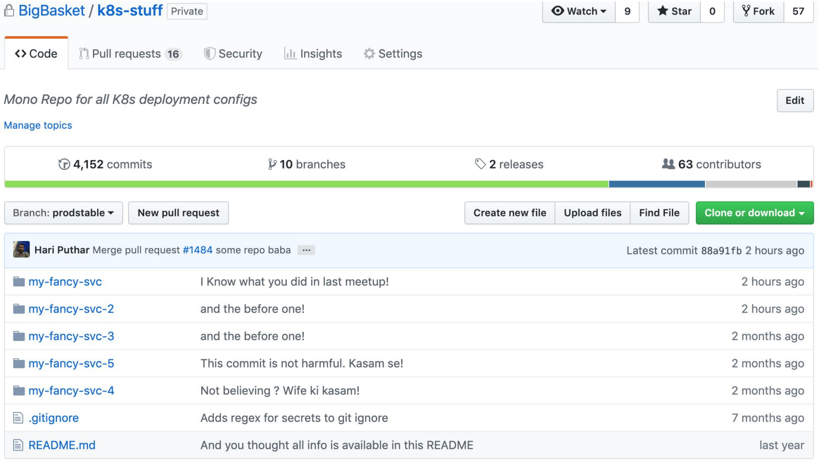


Reliability Challenges with K8s

- Monolith + 🐌 Traffic = 👍 🌈 😀
- Monolith + ⚡Traffic = 👎 💥 😫
- Enter…..µ▲ (s)…, Enter…. K8s
- But…. K8s is NOT a SILVER BULLET!!!
- Why ?

Reliability Challenges with K8s

- Challenge 1: Routine/scheduled deployments causing downtimes. 😱 🤦
- Note: All K8s deployments in BB are configuration driven.



BigBasket / k8s-stuff Private

Mono Repo for all K8s deployment configs

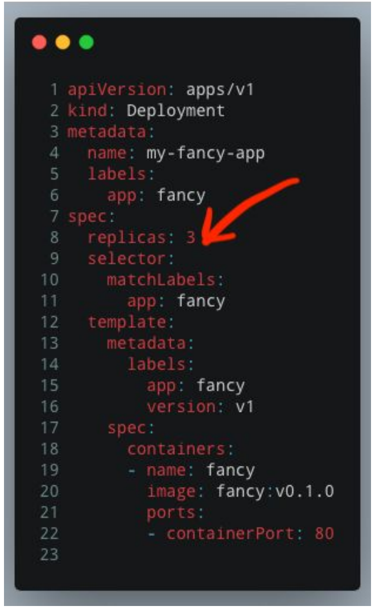
4,152 commits · 10 branches · 2 releases · 63 contributors

Hari Puthar Merge pull request #1484 some repo baba … Latest commit 88a91fb 2 hours ago

my-fancy-svc	I Know what you did in last meetup!	2 hours ago
my-fancy-svc-2	and the before one!	2 hours ago
my-fancy-svc-3	and the before one!	2 months ago
my-fancy-svc-5	This commit is not harmful. Kasam se!	2 months ago
my-fancy-svc-4	Not believing ? Wife ki kasam!	2 months ago
.gitignore	Adds regex for secrets to git ignore	7 months ago
README.md	And you thought all info is available in this README	last year

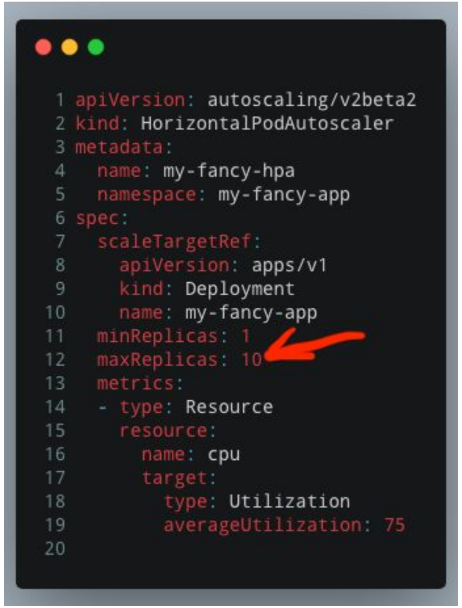
Reliability Challenges with K8s

- Usual deployment



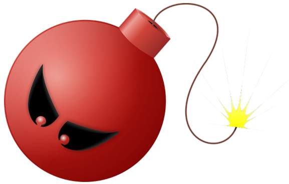
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-fancy-app
  labels:
    app: fancy
spec:
  replicas: 3
  selector:
    matchLabels:
      app: fancy
  template:
    metadata:
      labels:
        app: fancy
        version: v1
    spec:
      containers:
      - name: fancy
        image: fancy:v0.1.0
        ports:
        - containerPort: 80

```

+



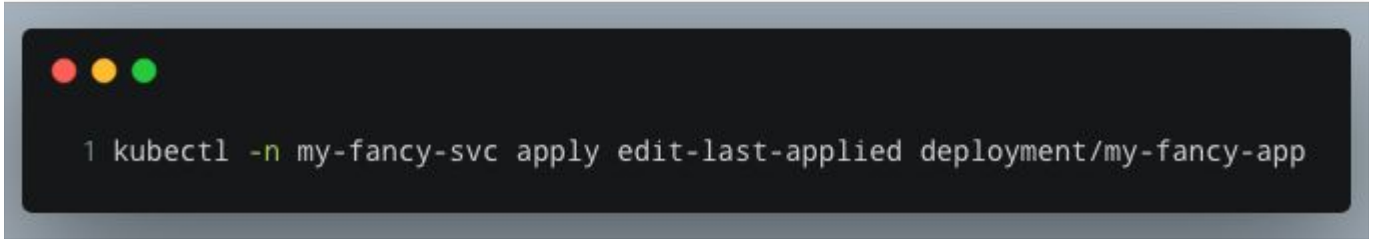
```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: my-fancy-hpa
  namespace: my-fancy-app
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: my-fancy-app
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 75

```

=



Reliability Challenges with K8s

1. Remove replicas: <n> setting from Deployment using :



```
kubectl -n my-fancy-svc apply edit-last-applied deployment/my-fancy-app
```

2. Update the config yaml (source code) accordingly.

3. Let HPA alone manage replicas count.

Reliability Challenges with K8s

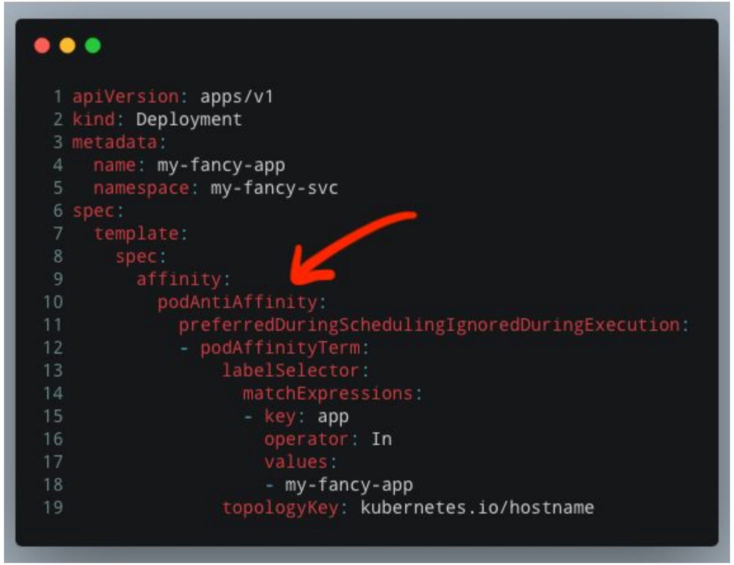
- Challenge 2: Not having a thorough understanding of maxSurge & maxUnavailable in a Deployment. (and Pdbs)
- 💡 maxSurge is rounded UP.
- 💡 maxUnavailable is rounded DOWN.
- And scaling up, scaling down happens in parallel.

Reliability Challenges with K8s

- If a deployment has 5 pods(version=v1), with a rolling deployment of :
  - maxSurge: 30%
  - maxUnavailable: 25%
- Immediately 4 v1 pods would receive **20% extra traffic**.

Reliability Challenges with K8s

- Challenge 3: Fair Distribution of Pods is not done by default
- Solution: Pod Anti-Affinity rules.



```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-fancy-app
  namespace: my-fancy-svc
spec:
  template:
    spec:
      affinity:
        podAntiAffinity:
          preferredDuringSchedulingIgnoredDuringExecution:
          - podAffinityTerm:
              labelSelector:
                matchExpressions:
                - key: app
                  operator: In
                  values:
                  - my-fancy-app
              topologyKey: kubernetes.io/hostname
```

Reliability Challenges with K8s

- Challenge 4: Fair Distribution of Pods based on instance life-cycle (On-demand vs Spot).
- Solution: WORK-in-Progress
- Many more reliability challenges...

Whats the future like ?

- Figuring out seamless upgrade strategies for fast paced projects like K8s.
- Figuring out unusual change in data patterns because of a small bug introduced by a microservice.
- Protecting services from cascade failures.
- Making critical systems like Kafka, more robust and self-healing as too many microservices evolve with light coupling architecture

Q & A

We are hiring...

nandakishore.b@bigbasket.com

We're

bigbasketeers

Are you?