

# Assignment 2 ACIT 3495 Kubernetes

Tamim Hemat and Daryush Balsara

# Agenda

- Overview
- NGINX
- AUTH
- UPLOAD
- MYSQL
- FILESYSTEM
- VIEWER
- Scalability



# Overview

- Our project is a video streaming platform
- We have 6 different clusters for each of our microservices.
- Each cluster has 1 pod that has been deployed
- Each of the clusters act as load balancers

Filter Enter property name or value

<input type="checkbox"/>	Status	Name ↑	Location	Number of nodes	Total vCPUs	Total memory	Notifications	Labels
<input type="checkbox"/>	✓	<a href="#">auth</a>	northamerica-northeast1-a	1	2	4 GB	—	⋮
<input type="checkbox"/>	✓	<a href="#">filesystem</a>	us-west1-b	2	4	8 GB	—	⋮
<input type="checkbox"/>	✓	<a href="#">mysql</a>	us-central1-a	1	2	4 GB	—	⋮
<input type="checkbox"/>	✓	<a href="#">nginx</a>	northamerica-northeast1-b	2	4	8 GB	—	⋮
<input type="checkbox"/>	✓	<a href="#">upload</a>	us-central1-b	2	4	8 GB	—	⋮
<input type="checkbox"/>	✓	<a href="#">viewer</a>	us-west1-a	2	4	8 GB	—	⋮

Filter **Is system object : False** Filter services and ingresses

<input type="checkbox"/>	Name ↑	Status	Type	Endpoints	Pods	Namespace	Clusters
<input type="checkbox"/>	<a href="#">acit3495-auth-service</a>	✓ OK	External load balancer	<a href="#">35.203.76.14:3000</a>	1/1	default	<a href="#">auth</a>
<input type="checkbox"/>	<a href="#">acit3495-filesystem-service</a>	✓ OK	External load balancer	<a href="#">34.168.79.180:22</a>	1/1	default	<a href="#">filesystem</a>
<input type="checkbox"/>	<a href="#">acit3495-mysql-service</a>	✓ OK	External load balancer	<a href="#">34.121.93.222:3306</a>	1/1	default	<a href="#">mysql</a>
<input type="checkbox"/>	<a href="#">acit3495-nginx-service</a>	✓ OK	External load balancer	<a href="#">35.234.245.231:80</a>	1/1	default	<a href="#">nginx</a>
<input type="checkbox"/>	<a href="#">acit3495-upload-service</a>	✓ OK	External load balancer	<a href="#">35.184.22.214:4000</a>	1/1	default	<a href="#">upload</a>
<input type="checkbox"/>	<a href="#">acit3495-viewer-service</a>	✓ OK	External load balancer	<a href="#">34.127.23.17:5000</a>	1/1	default	<a href="#">viewer</a>

# NGINX

- This reverse proxies to all our other services
- All IP for pods are hardcoded
- Implemented as a security feature
- If people go straight to the IP:PORT of each service they will just see "Cannot get /"
- Can only access other services through the reverse proxies
- Exposed on port 80

```
> kubectl get service acit3495-nginx-service -o yaml
Fetching cluster endpoint and auth data.
kubeconfig entry generated for nginx.
apiVersion: v1
kind: Service
metadata:
  annotations:
    cloud.google.com/neg: '{"ingress":true}'
  creationTimestamp: "2023-04-05T03:39:10Z"
  finalizers:
  - service.kubernetes.io/load-balancer-cleanup
  labels:
    app: acit3495-nginx
    name: acit3495-nginx-service
    namespace: default
    resourceVersion: "4814"
    uid: 2e6be423-b299-4765-aa0c-6efb2117754b
spec:
  allocateLoadBalancerNodePorts: true
  clusterIP: 10.88.1.28
  clusterIPs:
  - 10.88.1.28
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - nodePort: 30867
    port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: acit3495-nginx
  sessionAffinity: None
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
    - ip: 35.234.245.231

root@acit3495-nginx-7c77586788-v75s2:/# cat !$
cat /etc/nginx/sites-enabled/default
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    root /var/www/html;

    index index.html;

    server_name _;

    location / {
        try_files $uri $uri/ =404;
    }

    location /auth {
        proxy_pass http://35.203.76.14:3000;
    }

    location /upload {
        proxy_pass http://35.184.22.214:4000;
    }

    location /viewer {
        proxy_pass http://34.127.23.17:5000;
    }
}
```

# AUTH

- This is for authentication and signup for our services
- When signing up it stores all credentials to the user table

```
> kubectl get service acit3495-auth-service -o yaml
Fetching cluster endpoint and auth data.
kubeconfig entry generated for auth.
apiVersion: v1
kind: Service
metadata:
  annotations:
    cloud.google.com/neg: '{"ingress":true}'
    creationTimestamp: "2023-04-05T03:36:19Z"
  finalizers:
    - service.kubernetes.io/load-balancer-cleanup
  labels:
    app: acit3495-auth
  name: acit3495-auth-service
  namespace: default
  resourceVersion: "11282"
  uid: 11a916ea-621a-4391-99dc-dfb57204b5e3
spec:
  allocateLoadBalancerNodePorts: true
  clusterIP: 10.116.12.142
  clusterIPs:
    - 10.116.12.142
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
    - IPv4
  ipFamilyPolicy: SingleStack
  ports:
    - nodePort: 30971
      port: 3000
      protocol: TCP
      targetPort: 3000
  selector:
    app: acit3495-auth
  sessionAffinity: None
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
      - ip: 35.203.76.14
```

```
mysql> SELECT * FROM user;
```

user_id	username	password
1	tamim	tamim
2	chickenbingbong	password
3	bananafish	test

```
3 rows in set (0.00 sec)
```

```
mysql> █
```

# UPLOAD

- When you upload file is saves the filepath and name in the database
- The file itself will get stored in the filesystem pod

```
> kubectl get service acit3495-upload-service -o yaml
Fetching cluster endpoint and auth data.
kubeconfig entry generated for upload.
apiVersion: v1
kind: Service
metadata:
  annotations:
    cloud.google.com/neg: '{"ingress":true}'
  creationTimestamp: "2023-04-05T03:39:57Z"
  finalizers:
    - service.kubernetes.io/load-balancer-cleanup
  labels:
    app: acit3495-upload
    name: acit3495-upload-service
    namespace: default
    resourceVersion: "11025"
    uid: 114c5a16-a847-486c-b404-58ee6d3c06ff
spec:
  allocateLoadBalancerNodePorts: true
  clusterIP: 10.4.5.208
  clusterIPs:
    - 10.4.5.208
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
    - IPv4
  ipFamilyPolicy: SingleStack
  ports:
    - nodePort: 30359
      port: 4000
      protocol: TCP
      targetPort: 4000
  selector:
    app: acit3495-upload
  sessionAffinity: None
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
      - ip: 35.184.22.214
```

# MYSQL

- This pod stored the credentials to login to the app and the files after they have been uploaded
- Stores everything in 1 database called video streaming. This database has 2 tables for users and videos
- Its exposed on port 3306 internally
- Has a persistent volume claim on it which is where the database is stored

```
> kubectl get service acit3495-mysql-service -o yaml
Fetching cluster endpoint and auth data.
kubeconfig entry generated for mysql.
apiVersion: v1
kind: Service
metadata:
  annotations:
    cloud.google.com/neg: '{"ingress":true}'
    creationTimestamp: "2023-04-05T03:36:03Z"
  finalizers:
  - service.kubernetes.io/load-balancer-cleanup
  labels:
    app: acit3495-mysql
  name: acit3495-mysql-service
  namespace: default
  resourceVersion: "17179"
  uid: 87afb813-730b-4be8-90b0-f2226e11da04
spec:
  allocateLoadBalancerNodePorts: true
  clusterIP: 10.108.15.3
  clusterIPs:
  - 10.108.15.3
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - nodePort: 32496
    port: 3306
    protocol: TCP
    targetPort: 3306
  selector:
    app: acit3495-mysql
  sessionAffinity: None
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
    - ip: 34.121.93.222
```

```
> kubectl get persistentvolumeclaim mysql-pv-claim -o yaml
Fetching cluster endpoint and auth data.
kubeconfig entry generated for mysql.
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"PersistentVolumeClaim","metadata":{"annotations":{},"name":"mysql-pv-claim","namespace":"default"},"spec":{"accessModes":["ReadWriteOnce"],"resources":{"requests":{"storage":"2Gi"}}}}
  name: mysql-pv-claim
  namespace: default
  resourceVersion: "15045"
  uid: eb1dc8f5-4390-4356-9db4-69733ed62934
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  storageClassName: standard-rwo
  volumeMode: Filesystem
  volumeName: pvc-eb1dc8f5-4390-4356-9db4-69733ed62934
status:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 2Gi
  phase: Bound
```

# FILESYSTEM

- This service will store uploaded files
- This is not exposed on any port
- Every file is stored in /mnt/data
- This service has a persistent volume claim attached to it

```
chalsara2008@cloudshell: (assign2-acit3945) $ kubectl exec -it acit3495-filesystem-6cd4b79684-vn7jf bash
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl exec [POD] -- [COMMAND] instead.
root@acit3495-filesystem-6cd4b79684-vn7jf:/# ls
bin boot dev etc home lib lib32 lib64 libx32 media mnt opt proc root run sbin srv sys var
root@acit3495-filesystem-6cd4b79684-vn7jf:/# cd /mnt/data/
root@acit3495-filesystem-6cd4b79684-vn7jf:/mnt/data# ls
12-20_RAI-4469921373_01.webm  12-7_RAI-672680084_01.webm
12-20_RAI-4469921373_02.webm  'Hidden, Trump, and Obama Argue in a GTA Online Lobby.mp4'
                                'Raid Night in the White House.mp4'
                                'Tominie x FaoDnar - Seneca Overload.mp4'
root@acit3495-filesystem-6cd4b79684-vn7jf:/mnt/data#
```

```
Fetching cluster endpoint and auth data.
kubeconfig entry generated for filesystem.
apiVersion: v1
kind: Service
metadata:
  annotations:
    cloud.google.com/neg: '{"ingress":true}'
    creationTimestamp: "2023-04-05T03:43:12Z"
  finalizers:
    - service.kubernetes.io/load-balancer-cleanup
  labels:
    app: acit3495-filesystem
  name: acit3495-filesystem-service
  namespace: default
  resourceVersion: "8274"
  uid: 80610be7-ff1d-460a-bb3e-356ecf031af3
spec:
  allocateLoadBalancerNodePorts: true
  clusterIP: 10.56.11.93
  clusterIPs:
    - 10.56.11.93
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
    - IPv4
  ipFamilyPolicy: SingleStack
  ports:
    - nodePort: 30141
      port: 22
      protocol: TCP
      targetPort: 22
  selector:
    app: acit3495-filesystem
  sessionAffinity: None
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
      - ip: 34.168.79.180
```

```
> kubectl get persistentvolumeclaim filesystem-pv-claim -o yaml
kubectl get persistentvolumeclaim filesystem-pv-claim -o yaml
kubeconfig entry generated for filesystem.
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"PersistentVolumeClaim","metadata":{"annotations":{"kubernetes.io/last-applied-configuration":"\n      {\n        \"apiVersion\": \"v1\",
pv.kubernetes.io/bound-by-controller: "yes"
volume.beta.kubernetes.io/storage-provisioner: pd.csi.storage.gke.io
volume.kubernetes.io/selected-node: gke-filesystem-default-pool-02ff3cee-c023
volume.kubernetes.io/storage-provisioner: pd.csi.storage.gke.io
creationTimestamp: "2023-04-05T05:53:59Z"
finalizers:
  - kubernetes.io/pvc-protection
name: filesystem-pv-claim
namespace: default
resourceVersion: "72367"
uid: 335d73c2-1447-408d-980a-0954f43613f6
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: standard-rwo
  volumeMode: Filesystem
  volumeName: pvc-335d73c2-1447-408d-980a-0954f43613f6
```



# VIEWER

- This serves the uploaded files
- It makes a request to the database and the filesystem to pull the correct file based on the name of it

```
> kubectl get service acit3495-viewer-service -o yaml
Fetching cluster endpoint and auth data.
kubeconfig entry generated for viewer.
apiVersion: v1
kind: Service
metadata:
  annotations:
    cloud.google.com/neg: '{"ingress":true}'
  creationTimestamp: "2023-04-05T05:35:17Z"
  finalizers:
    - service.kubernetes.io/load-balancer-cleanup
  labels:
    app: acit3495-viewer
  name: acit3495-viewer-service
  namespace: default
  resourceVersion: "5467"
  uid: 244d0085-dc66-4727-9336-3d10349c50cc
spec:
  allocateLoadBalancerNodePorts: true
  clusterIP: 10.32.11.136
  clusterIPs:
    - 10.32.11.136
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
    - IPv4
  ipFamilyPolicy: SingleStack
  ports:
    - nodePort: 31961
      port: 5000
      protocol: TCP
      targetPort: 5000
  selector:
    app: acit3495-viewer
  sessionAffinity: None
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
      - ip: 34.127.23.17
```

# Scalability

- To test our horizontal scalability we simulated a large number of users/requests to see how well the system handled the load. We did this by performing a syn flood attack on our SEED VM.
- After deployment of one pod if we see increased traffic we can add more replicas or pods to our already existing deployment using the auto horizontal pod autoscaler.





DEMO



Thanks For Watching

Any Questions