**2. Implement Ingress Groups Demo with 3 Apps**

--- Reference - <https://github.com/stacksimplify/aws-eks-kubernetes-masterclass/tree/master/08-NEW-ELB-Application-LoadBalancers/08-12-IngressGroups>

**Introduction**

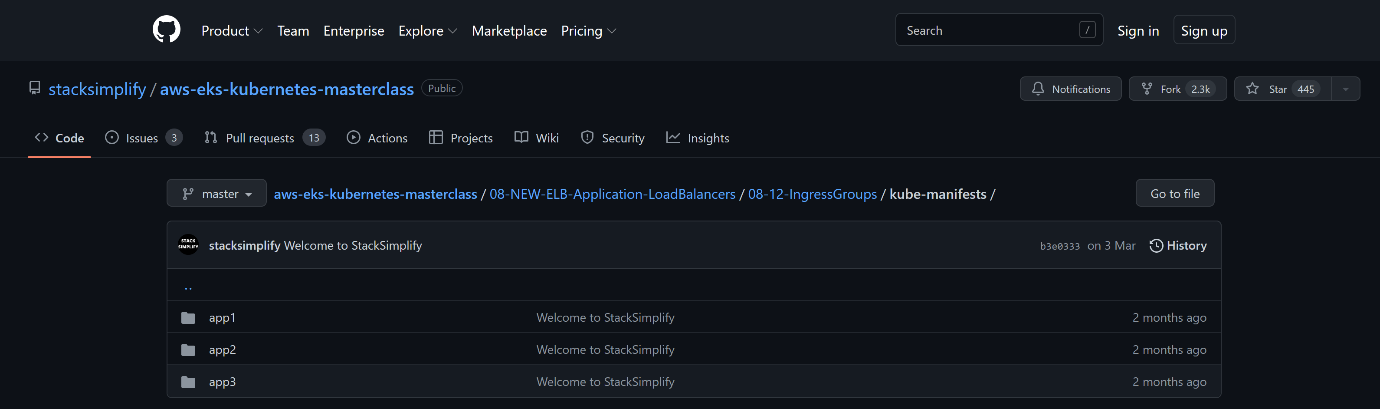
--- IngressGroup feature enables you to group multiple Ingress resources together.

--- The controller will automatically merge Ingress rules for all Ingresses within IngressGroup and support them with a single ALB.

--- In addition, most annotations defined on a Ingress only applies to the paths defined by that Ingress.

--- Demonstrate Ingress Groups concept with two Applications.

**Review App1 Ingress Manifest - Key Lines**



--- under kube-manifests, you will find these folders. Inside of these folders you will find manifests

--- File Name: **kube-manifests/app1/02-App1-Ingress.yml**

    # Ingress Groups

    alb.ingress.kubernetes.io/group.name: myapps.web

    alb.ingress.kubernetes.io/group.order: '10'

--- **01-Nginx-App1-Deployment-and-NodePortService.yml**

apiVersion: apps/v1

kind: Deployment

metadata:

  name: app1-nginx-deployment

  labels:

    app: app1-nginx

spec:

  replicas: 1

  selector:

    matchLabels:

      app: app1-nginx

  template:

    metadata:

      labels:

        app: app1-nginx

    spec:

      containers:

        - name: app1-nginx

          image: stacksimplify/kube-nginxapp1:1.0.0

          ports:

            - containerPort: 80

---

apiVersion: v1

kind: Service

metadata:

  name: app1-nginx-nodeport-service

  labels:

    app: app1-nginx

  annotations:

#Important Note:  Need to add health check path annotations in service level if we are planning to use multiple targets in a load balancer

    alb.ingress.kubernetes.io/healthcheck-path: /app1/index.html

spec:

  type: NodePort

  selector:

    app: app1-nginx

  ports:

    - port: 80

      targetPort: 80

--- **02-App1-Ingress.yml**

# Annotations Reference: https://kubernetes-sigs.github.io/aws-load-balancer-controller/latest/guide/ingress/annotations/

#apiVersion: extensions/v1beta1

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

  name: app1-ingress

  annotations:

    # Load Balancer Name

    alb.ingress.kubernetes.io/load-balancer-name: ingress-groups-demo

    # Ingress Core Settings

    #kubernetes.io/ingress.class: "alb" (OLD INGRESS CLASS NOTATION - STILL WORKS BUT RECOMMENDED TO USE IngressClass Resource)

    alb.ingress.kubernetes.io/scheme: internet-facing

    # Health Check Settings

    alb.ingress.kubernetes.io/healthcheck-protocol: HTTP

    alb.ingress.kubernetes.io/healthcheck-port: traffic-port

    #Important Note:  Need to add health check path annotations in service level if we are planning to use multiple targets in a load balancer

    alb.ingress.kubernetes.io/healthcheck-interval-seconds: '15'

    alb.ingress.kubernetes.io/healthcheck-timeout-seconds: '5'

    alb.ingress.kubernetes.io/success-codes: '200'

    alb.ingress.kubernetes.io/healthy-threshold-count: '2'

    alb.ingress.kubernetes.io/unhealthy-threshold-count: '2'

    ## SSL Settings

    alb.ingress.kubernetes.io/listen-ports: '[{"HTTPS":443}, {"HTTP":80}]'

    alb.ingress.kubernetes.io/certificate-arn: arn:aws:acm:us-east-1:180789647333:certificate/d86de939-8ffd-410f-adce-0ce1f5be6e0d

    #alb.ingress.kubernetes.io/ssl-policy: ELBSecurityPolicy-TLS-1-1-2017-01 #Optional (Picks default if not used)

    # SSL Redirect Setting

    alb.ingress.kubernetes.io/ssl-redirect: '443'

    # External DNS - For creating a Record Set in Route53

    external-dns.alpha.kubernetes.io/hostname: ingress-groups-demo601.stacksimplify.com

    # Ingress Groups

    alb.ingress.kubernetes.io/group.name: myapps.web

    alb.ingress.kubernetes.io/group.order: '10'

spec:

  ingressClassName: my-aws-ingress-class   # Ingress Class

  rules:

    - http:

        paths:

          - path: /app1

            pathType: Prefix

            backend:

              service:

                name: app1-nginx-nodeport-service

                port:

                  number: 80

**Review App2 Ingress Manifest - Key Lines**

--- **File Name: kube-manifests/app2/02-App2-Ingress.yml**

    # Ingress Groups

    alb.ingress.kubernetes.io/group.name: myapps.web

    alb.ingress.kubernetes.io/group.order: '20'

--- **01-Nginx-App2-Deployment-and-NodePortService.yml**

apiVersion: apps/v1

kind: Deployment

metadata:

  name: app2-nginx-deployment

  labels:

    app: app2-nginx

spec:

  replicas: 1

  selector:

    matchLabels:

      app: app2-nginx

  template:

    metadata:

      labels:

        app: app2-nginx

    spec:

      containers:

        - name: app2-nginx

          image: stacksimplify/kube-nginxapp2:1.0.0

          ports:

            - containerPort: 80

---

apiVersion: v1

kind: Service

metadata:

  name: app2-nginx-nodeport-service

  labels:

    app: app2-nginx

  annotations:

#Important Note:  Need to add health check path annotations in service level if we are planning to use multiple targets in a load balancer

    alb.ingress.kubernetes.io/healthcheck-path: /app2/index.html

spec:

  type: NodePort

  selector:

    app: app2-nginx

  ports:

    - port: 80

      targetPort: 80

--- **02-App2-Ingress.yml**

# Annotations Reference: https://kubernetes-sigs.github.io/aws-load-balancer-controller/latest/guide/ingress/annotations/

#apiVersion: extensions/v1beta1

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

  name: app2-ingress

  annotations:

    # Ingress Core Settings

    #kubernetes.io/ingress.class: "alb"

    alb.ingress.kubernetes.io/scheme: internet-facing

    # Load Balancer Name

    alb.ingress.kubernetes.io/load-balancer-name: ingress-groups-demo

    # Health Check Settings

    alb.ingress.kubernetes.io/healthcheck-protocol: HTTP

    alb.ingress.kubernetes.io/healthcheck-port: traffic-port

    #Important Note:  Need to add health check path annotations in service level if we are planning to use multiple targets in a load balancer

    #alb.ingress.kubernetes.io/healthcheck-path: /usermgmt/health-status

    alb.ingress.kubernetes.io/healthcheck-interval-seconds: '15'

    alb.ingress.kubernetes.io/healthcheck-timeout-seconds: '5'

    alb.ingress.kubernetes.io/success-codes: '200'

    alb.ingress.kubernetes.io/healthy-threshold-count: '2'

    alb.ingress.kubernetes.io/unhealthy-threshold-count: '2'

    ## SSL Settings

    alb.ingress.kubernetes.io/listen-ports: '[{"HTTPS":443}, {"HTTP":80}]'

    alb.ingress.kubernetes.io/certificate-arn: arn:aws:acm:us-east-1:180789647333:certificate/d86de939-8ffd-410f-adce-0ce1f5be6e0d

    #alb.ingress.kubernetes.io/ssl-policy: ELBSecurityPolicy-TLS-1-1-2017-01 #Optional (Picks default if not used)

    # SSL Redirect Setting

    alb.ingress.kubernetes.io/ssl-redirect: '443'

    # External DNS - For creating a Record Set in Route53

    external-dns.alpha.kubernetes.io/hostname: ingress-groups-demo601.stacksimplify.com

    # Ingress Groups

    alb.ingress.kubernetes.io/group.name: myapps.web

    alb.ingress.kubernetes.io/group.order: '20'

spec:

  ingressClassName: my-aws-ingress-class   # Ingress Class

  rules:

    - http:

        paths:

          - path: /app2

            pathType: Prefix

            backend:

              service:

                name: app2-nginx-nodeport-service

                port:

                  number: 80

**Review App3 Ingress Manifest - Key Lines**

    # Ingress Groups

    alb.ingress.kubernetes.io/group.name: myapps.web

    alb.ingress.kubernetes.io/group.order: '30'

--- **01-Nginx-App3-Deployment-and-NodePortService.yml**

apiVersion: apps/v1

kind: Deployment

metadata:

  name: app3-nginx-deployment

  labels:

    app: app3-nginx

spec:

  replicas: 1

  selector:

    matchLabels:

      app: app3-nginx

  template:

    metadata:

      labels:

        app: app3-nginx

    spec:

      containers:

        - name: app2-nginx

          image: stacksimplify/kubenginx:1.0.0

          ports:

            - containerPort: 80

---

apiVersion: v1

kind: Service

metadata:

  name: app3-nginx-nodeport-service

  labels:

    app: app3-nginx

  annotations:

#Important Note:  Need to add health check path annotations in service level if we are planning to use multiple targets in a load balancer

    alb.ingress.kubernetes.io/healthcheck-path: /index.html

spec:

  type: NodePort

  selector:

    app: app3-nginx

  ports:

    - port: 80

      targetPort: 80

--- **03-App3-Ingress-default-backend.yml**

# Annotations Reference: https://kubernetes-sigs.github.io/aws-load-balancer-controller/latest/guide/ingress/annotations/

#apiVersion: extensions/v1beta1

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

  name: app3-ingress

  annotations:

    # Ingress Core Settings

    #kubernetes.io/ingress.class: "alb"

    alb.ingress.kubernetes.io/scheme: internet-facing

    # Load Balancer Name

    alb.ingress.kubernetes.io/load-balancer-name: ingress-groups-demo

    # Health Check Settings

    alb.ingress.kubernetes.io/healthcheck-protocol: HTTP

    alb.ingress.kubernetes.io/healthcheck-port: traffic-port

    #Important Note:  Need to add health check path annotations in service level if we are planning to use multiple targets in a load balancer

    #alb.ingress.kubernetes.io/healthcheck-path: /usermgmt/health-status

    alb.ingress.kubernetes.io/healthcheck-interval-seconds: '15'

    alb.ingress.kubernetes.io/healthcheck-timeout-seconds: '5'

    alb.ingress.kubernetes.io/success-codes: '200'

    alb.ingress.kubernetes.io/healthy-threshold-count: '2'

    alb.ingress.kubernetes.io/unhealthy-threshold-count: '2'

    ## SSL Settings

    alb.ingress.kubernetes.io/listen-ports: '[{"HTTPS":443}, {"HTTP":80}]'

    alb.ingress.kubernetes.io/certificate-arn: arn:aws:acm:us-east-1:180789647333:certificate/d86de939-8ffd-410f-adce-0ce1f5be6e0d

    #alb.ingress.kubernetes.io/ssl-policy: ELBSecurityPolicy-TLS-1-1-2017-01 #Optional (Picks default if not used)

    # SSL Redirect Setting

    alb.ingress.kubernetes.io/ssl-redirect: '443'

    # External DNS - For creating a Record Set in Route53

    external-dns.alpha.kubernetes.io/hostname: ingress-groups-demo601.stacksimplify.com

    # Ingress Groups

    alb.ingress.kubernetes.io/group.name: myapps.web

    alb.ingress.kubernetes.io/group.order: '30'

spec:

  ingressClassName: my-aws-ingress-class   # Ingress Class

  defaultBackend:

    service:

      name: app3-nginx-nodeport-service

      port:

        number: 80

--- **note** - **external-dns.alpha.kubernetes.io/hostname: ingress-groups-demo601.stacksimplify.com** – this external dns is same for all three applications because this is context-based routing not host based routing.

**Deploy Apps with two Ingress Resources**

**# Deploy both Apps**

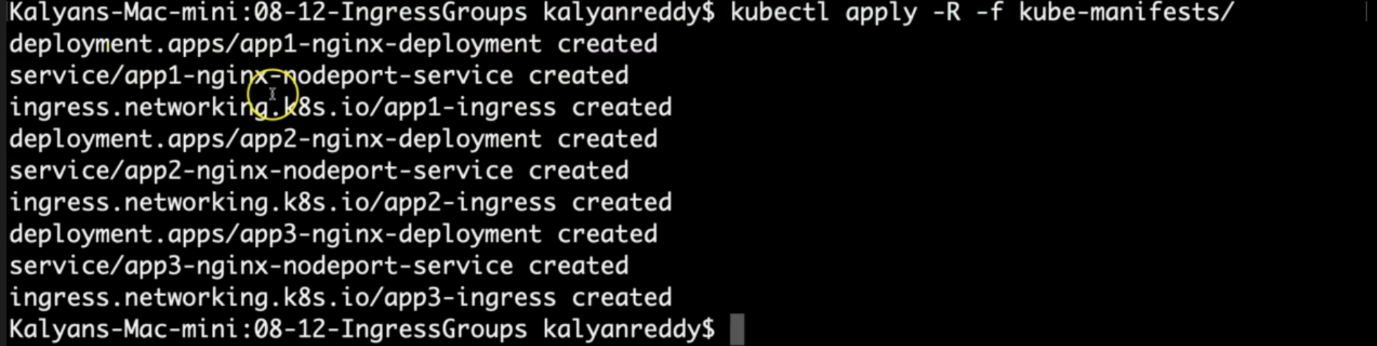
--- **kubectl apply -f kube-manifests/app1**

--- **kubectl apply -f kube-manifests/app3**

--- **kubectl apply -f kube-manifests/app3**

Or

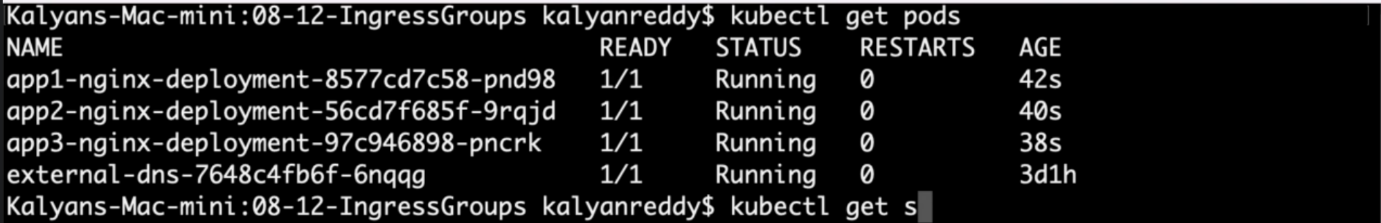
--- **kubectl apply -R -f kube-manifests** – you can apply for all or you can choose to apply for particular folder.



--- app1 deployment, nodeport service, ingress created. Some for others.

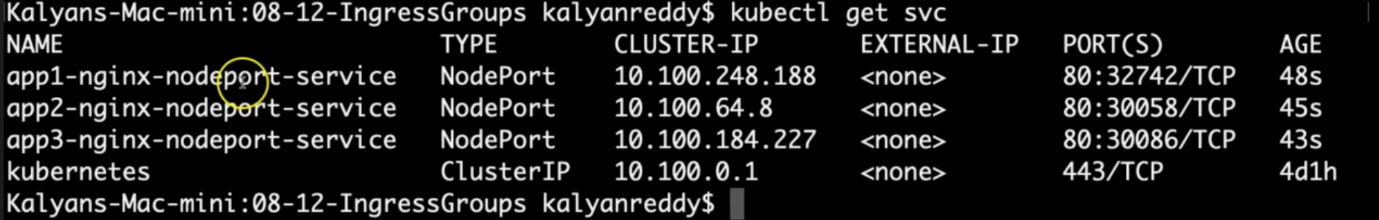
**# Verify Pods**

--- **kubectl get pods**



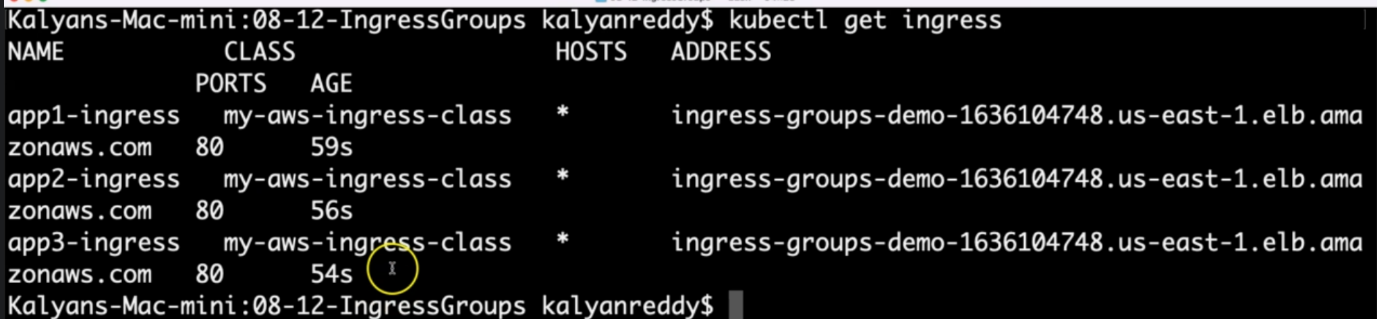
**# Verify services**

--- **kubectl get svc**



**# Verify Ingress**

--- **kubectl get ingress**

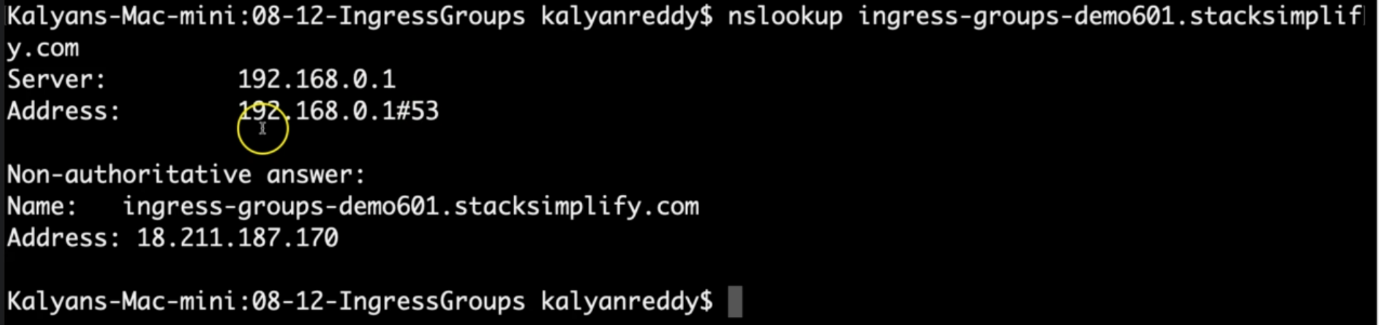


**Observation**:

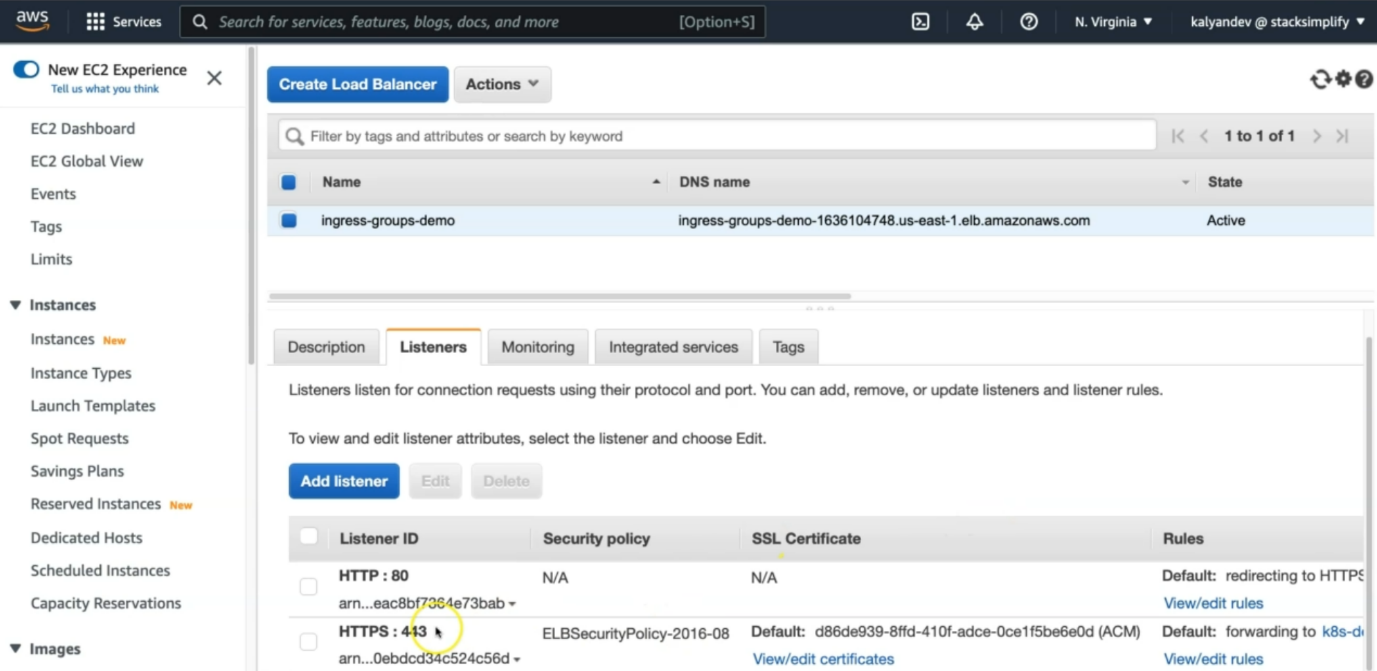
1. Three Ingress resources will be created with same ADDRESS value

2. Three Ingress Resources are merged to a single Application Load Balancer as those belong to same Ingress group "myapps.web"

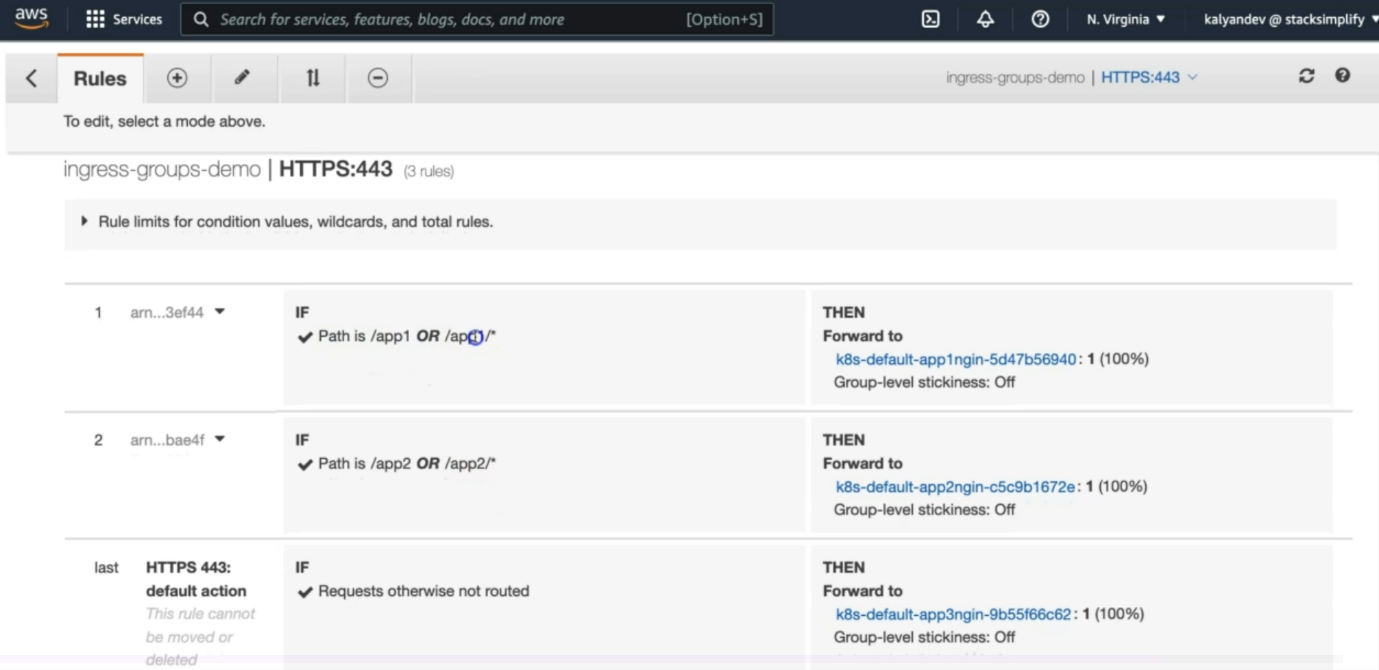
**# Nslookup**



**Load balancer**



--- click on HTTPS:443 **view/edit rules**.



--- for /app1 or app1/\*, it will forward the request to target group, form there it will routed to node port service, from there it will routed towards pod. Same for others.

**Verify on AWS Mgmt Console**

--- Go to Services -> EC2 -> Load Balancers

--- Verify Routing Rules for /app1 and /app2 and default backend

**Verify by accessing in browser**

**# Web URLs**

--- <http://ingress-groups-demo601.stacksimplify.com/app1/index.html>

--- <http://ingress-groups-demo601.stacksimplify.com/app2/index.html>

--- <http://ingress-groups-demo601.stacksimplify.com>

**Clean-Up**

**# Delete Apps from k8s cluster**

--- **kubectl delete -R -f kube-manifests/**

**## Verify Route53 Record Set to ensure our DNS records got deleted**

--- Go to Route53 -> Hosted Zones -> Records

--- The below records should be deleted automatically

1. ingress-groups-demo601.stacksimplify.com