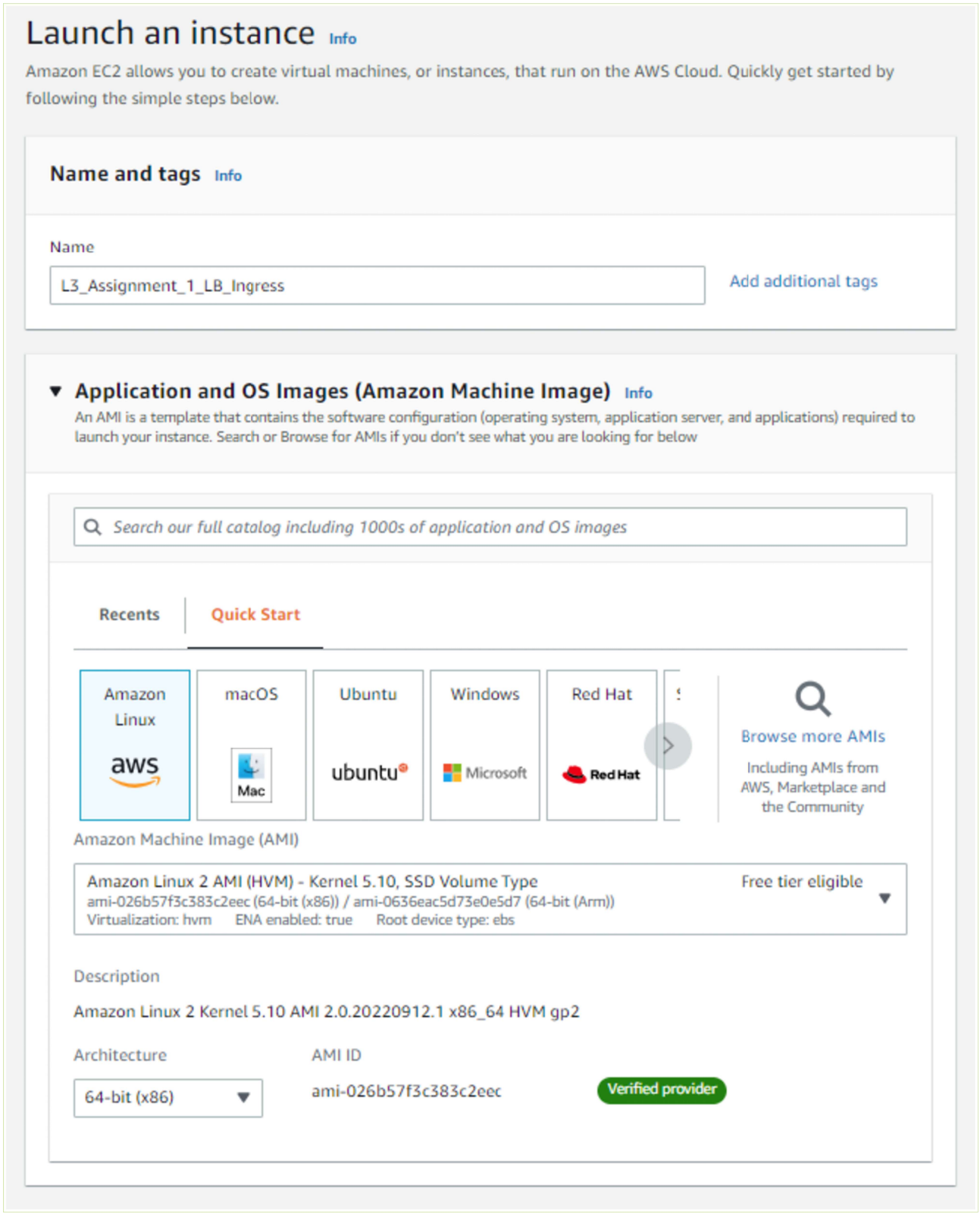
L3 Assignment 1 :: Configuring HTTP LoadBalancer for application using Ingress

Table of Contents

Step 1 :: Create an instance with t2.medium and storage 60 GB



Instance type Info

Instance type

t2.medium
Family: t2 2 vCPU 4 GiB Memory
On-Demand Linux pricing: 0.0464 USD per Hour
On-Demand Windows pricing: 0.0644 USD per Hour

Compare instance types

Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

28Sep2022

Create new key pair

Network settings Info

Edit

Network Info

vpc-389a5a45 | Default

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group

Select existing security group

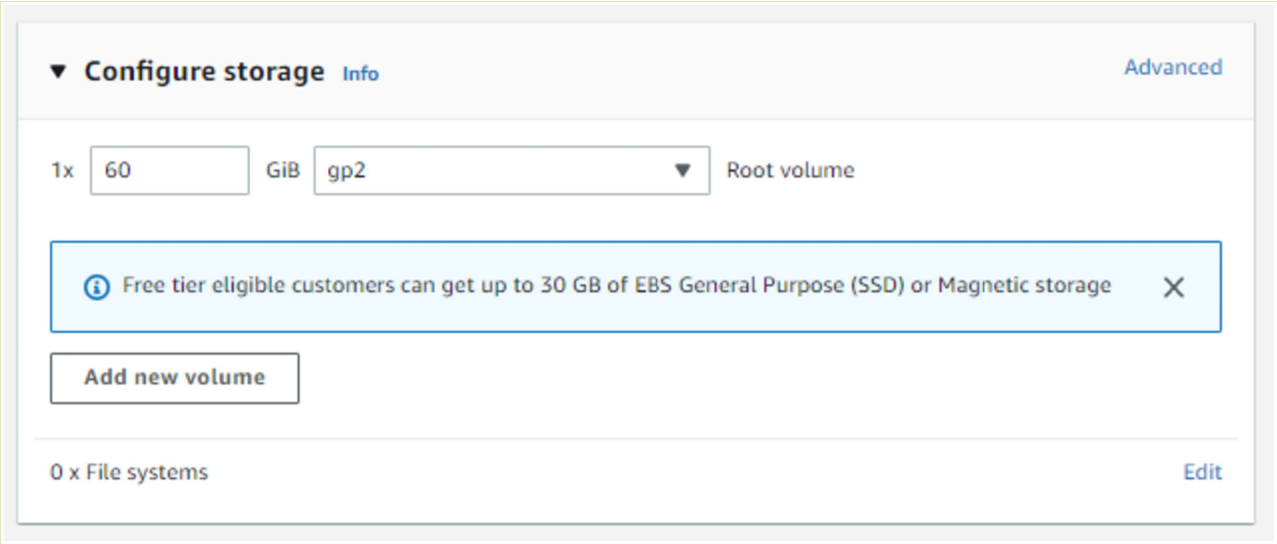
We'll create a new security group called '**launch-wizard-2**' with the following rules:

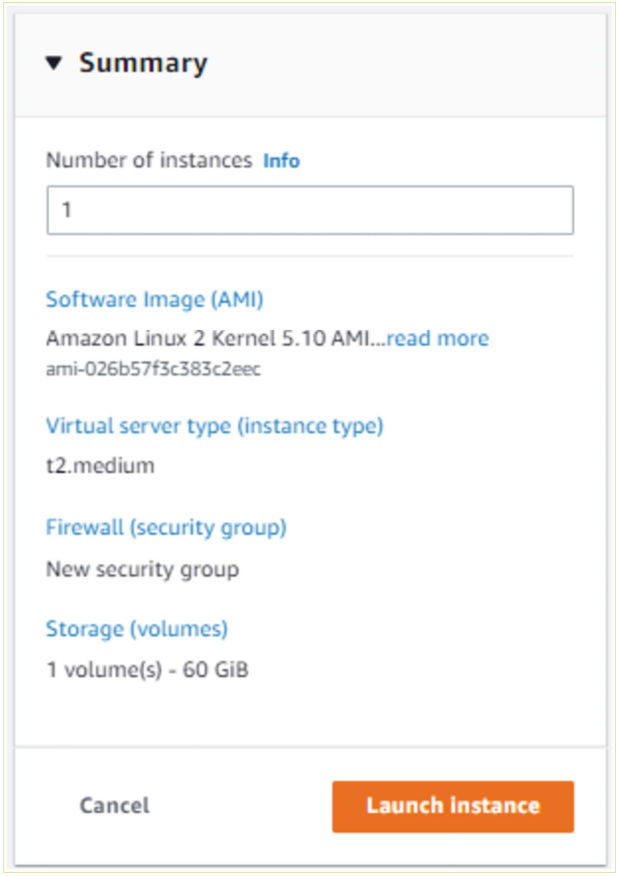
Allow SSH traffic from
Helps you connect to your instance

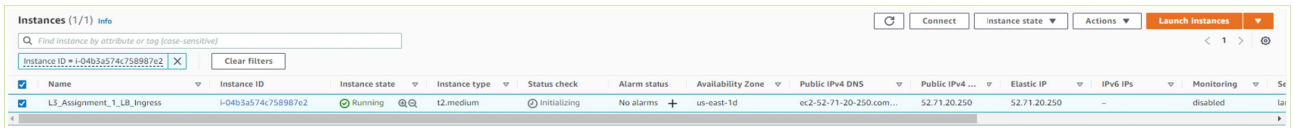
Anywhere
0.0.0.0/0

Allow HTTPs traffic from the internet
To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server


Configure storage Info
Advanced
1x
60
GiB
gp2
Root volume
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage
Add new volume
0 x File systems
Edit


Summary
Number of instances Info
1
Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...read more
ami-026b57f3c383c2eec
Virtual server type (instance type)
t2.medium
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 60 GiB
Cancel
Launch instance


Instances (1/1) Info
Connect
Instance state
Actions
Launch instances
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
Elastic IP
IPv6 IPs
Monitoring
L3_Assignment_1_LB_Ingress
i-04b3a574c758987e2
Running
t2.medium
Initializing
No alarms
us-east-1d
ec2-52-71-20-250.com...
52.71.20.250
52.71.20.250
disabled

Step 2 :: Install Docker

yum update
yum install -y docker
systemctl enable docker.service
systemctl start docker.service
systemctl status docker.service

```
[root@ip-172-31-25-117 ~]# systemctl status  docker.service
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2022-09-28 08:13:07 UTC; 5s ago
     Docs: https://docs.docker.com
  Process: 3602 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
  Process: 3600 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
 Main PID: 3605 (dockerd)
    Tasks: 8
   Memory: 22.3M
   CGroup: /system.slice/docker.service
           └─3605 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Sep 28 08:13:03 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:03.500859681Z" level=info msg="ClientConn swi...=grpc
Sep 28 08:13:06 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:06.840303343Z" level=warning msg="Your kernel...ight"
Sep 28 08:13:06 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:06.840341578Z" level=warning msg="Your kernel...vice"
Sep 28 08:13:06 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:06.840507532Z" level=info msg="Loading contai...art."
Sep 28 08:13:07 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:07.558830641Z" level=info msg="Default bridge...ress"
Sep 28 08:13:07 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:07.637722787Z" level=info msg="Loading contai...one."
Sep 28 08:13:07 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:07.650882200Z" level=info msg="Docker daemon"...10.17
Sep 28 08:13:07 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:07.650987356Z" level=info msg="Daemon has com...tion"
Sep 28 08:13:07 ip-172-31-25-117.ec2.internal systemd[1]: Started Docker Application Container Engine.
Sep 28 08:13:07 ip-172-31-25-117.ec2.internal dockerd[3605]: time="2022-09-28T08:13:07.676637253Z" level=info msg="API listen on ...sock"
Hint: Some lines were ellipsized, use -l to show in full.
[root@ip-172-31-25-117 ~]#
```

Step 3 :: Install Kubectl

curl -o kubectl https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.7/2022-06-29/bin/linux/amd64/kubectl
curl -o kubectl.sha256 https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.7/2022-06-29/bin/linux/amd64/kubectl.sha256
openssl sha1 -sha256 kubectl
chmod +x ./kubectl
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin
echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc
kubectl version --short --client

```
[root@ip-172-31-25-117 ~]# kubectl version --short --client
Client Version: v1.23.7-eks-4721010
[root@ip-172-31-25-117 ~]#
```

Step 4 :: Install Kind

curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.15.0/kind-linux-amd64
chmod +x ./kind
mv ./kind /usr/local/bin/kind
kind version

```
[root@ip-172-31-25-117 ~]# kind version
kind v0.15.0 go1.19 linux/amd64
[root@ip-172-31-25-117 ~]#
```

Step 5 :: Create a cluster using Kind

vi kind-config.yaml

kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
- role: control-plane
- role: worker
- role: worker
- role: worker

kind create cluster --config kind-config.yaml –name=assignment-1-lb-ingress

```
[root@ip-172-31-25-117 ~]# kind create cluster --config kind-config.yaml --name=assignment-1-lb-ingress
Creating cluster "assignment-1-lb-ingress" ...
 ✓ Ensuring node image (kindest/node:v1.25.0)
 ✓ Preparing nodes
 ✓ Writing configuration
 ✓ Starting control-plane
 ✓ Installing CNI
 ✓ Installing StorageClass
 ✓ Joining worker nodes
Set kubectl context to "kind-assignment-1-lb-ingress"
You can now use your cluster with:

kubectl cluster-info --context kind-assignment-1-lb-ingress

Thanks for using kind!
[root@ip-172-31-25-117 ~]#
```

kubectl get all
kubectl get node -o wide

```
[root@ip-172-31-25-117 ~]# kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   96s
[root@ip-172-31-25-117 ~]# kubectl get node -o wide
NAME                                    STATUS   ROLES           AGE    VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION                  CONTAINER-RUNTIME
assignment-1-lb-ingress-control-plane   Ready    control-plane   100s   v1.25.0   172.18.0.5    <none>        Ubuntu 22.04.1 LTS   5.10.135-122.509.amzn2.x86_64   containerd://1.6.7
assignment-1-lb-ingress-worker          Ready    <none>          64s    v1.25.0   172.18.0.3    <none>        Ubuntu 22.04.1 LTS   5.10.135-122.509.amzn2.x86_64   containerd://1.6.7
assignment-1-lb-ingress-worker2         Ready    <none>          65s    v1.25.0   172.18.0.4    <none>        Ubuntu 22.04.1 LTS   5.10.135-122.509.amzn2.x86_64   containerd://1.6.7
assignment-1-lb-ingress-worker3         Ready    <none>          65s    v1.25.0   172.18.0.2    <none>        Ubuntu 22.04.1 LTS   5.10.135-122.509.amzn2.x86_64   containerd://1.6.7
[root@ip-172-31-25-117 ~]#
```

docker ps -a

```
[root@ip-172-31-25-117 ~]# docker ps -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED         STATUS         PORTS                       NAMES
55017dc8c732   kindest/node:v1.25.0   "/usr/local/bin/entr…"   3 minutes ago   Up 2 minutes   127.0.0.1:33299->6443/tcp   assignment-1-lb-ingress-control-plane
7983de38f941   kindest/node:v1.25.0   "/usr/local/bin/entr…"   3 minutes ago   Up 2 minutes                               assignment-1-lb-ingress-worker3
9465cd909e32   kindest/node:v1.25.0   "/usr/local/bin/entr…"   3 minutes ago   Up 2 minutes                               assignment-1-lb-ingress-worker
b4137032bd18   kindest/node:v1.25.0   "/usr/local/bin/entr…"   3 minutes ago   Up 3 minutes                               assignment-1-lb-ingress-worker2
[root@ip-172-31-25-117 ~]#
```

Step 6 :: Create an Ingress-Controller

vi ingress-controller.yaml
kubectl apply -f ingress-controller.yaml
kubectl get all -n ingress-nginx

```
[root@ip-172-31-25-117 ~]# kubectl apply -f ingress-controller.yaml
namespace/ingress-nginx created
serviceaccount/ingress-nginx created
serviceaccount/ingress-nginx-admission created
role.rbac.authorization.k8s.io/ingress-nginx created
role.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrole.rbac.authorization.k8s.io/ingress-nginx created
clusterrole.rbac.authorization.k8s.io/ingress-nginx-admission created
rolebinding.rbac.authorization.k8s.io/ingress-nginx created
rolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
configmap/ingress-nginx-controller created
service/ingress-nginx-controller created
service/ingress-nginx-controller-admission created
deployment.apps/ingress-nginx-controller created
job.batch/ingress-nginx-admission-create created
job.batch/ingress-nginx-admission-patch created
ingressclass.networking.k8s.io/nginx created
validatingwebhookconfiguration.admissionregistration.k8s.io/ingress-nginx-admission created
[root@ip-172-31-25-117 ~]#
[root@ip-172-31-25-117 ~]#
[root@ip-172-31-25-117 ~]# kubectl get all -n ingress-nginx
NAME                                            READY   STATUS      RESTARTS   AGE
pod/ingress-nginx-admission-create-5k99v        0/1     Completed   0          33s
pod/ingress-nginx-admission-patch-hbnkw         0/1     Completed   1          33s
pod/ingress-nginx-controller-d7cc6c876-2lmkh    1/1     Running     0          33s

NAME                                         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)                      AGE
service/ingress-nginx-controller             NodePort    10.96.46.34    <none>        80:30126/TCP,443:31928/TCP   33s
service/ingress-nginx-controller-admission   ClusterIP   10.96.236.9    <none>        443/TCP                      33s

NAME                                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/ingress-nginx-controller   1/1     1            1           33s

NAME                                                  DESIRED   CURRENT   READY   AGE
replicaset.apps/ingress-nginx-controller-d7cc6c876    1         1         1       33s

NAME                                       COMPLETIONS   DURATION   AGE
job.batch/ingress-nginx-admission-create   1/1           8s         33s
job.batch/ingress-nginx-admission-patch    1/1           8s         33s
[root@ip-172-31-25-117 ~]#
```

kubectl get pods -n ingress-nginx

```
[root@ip-172-31-25-117 ~]# kubectl get pods -n ingress-nginx
NAME                                       READY   STATUS      RESTARTS   AGE
ingress-nginx-admission-create-5k99v       0/1     Completed   0          87s
ingress-nginx-admission-patch-hbnkw        0/1     Completed   1          87s
ingress-nginx-controller-d7cc6c876-2lmkh   1/1     Running     0          87s
[root@ip-172-31-25-117 ~]#
```

Step 7 :: Create two deployments in the cluster and expose them

kubectl create deployment web-1 --image=gcr.io/google-samples/hello-app:1.0
kubectl create deployment web-2 --image=gcr.io/google-samples/hello-app:2.0
kubectl get all

```
[root@ip-172-31-25-117 ~]# kubectl create deployment web-1 --image=gcr.io/google-samples/hello-app:1.0
deployment.apps/web-1 created
[root@ip-172-31-25-117 ~]# kubectl create deployment web-2 --image=gcr.io/google-samples/hello-app:2.0
deployment.apps/web-2 created
[root@ip-172-31-25-117 ~]#
[root@ip-172-31-25-117 ~]# kubectl get all
NAME                         READY   STATUS    RESTARTS   AGE
pod/web-1-64d7d68999-r9468   1/1     Running   0          50s
pod/web-2-9b89b499-mwpzl     1/1     Running   0          11s

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   12m

NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/web-1   1/1     1            1           50s
deployment.apps/web-2   1/1     1            1           11s

NAME                               DESIRED   CURRENT   READY   AGE
replicaset.apps/web-1-64d7d68999   1         1         1       50s
replicaset.apps/web-2-9b89b499     1         1         1       11s
[root@ip-172-31-25-117 ~]#
```

kubectl expose deployment web-1 --type=NodePort --port=8080
kubectl expose deployment web-2 --type=NodePort --port=8080
kubectl get all

```
[root@ip-172-31-25-117 ~]# kubectl expose deployment web-1 --type=NodePort --port=8080
service/web-1 exposed
[root@ip-172-31-25-117 ~]#
[root@ip-172-31-25-117 ~]# kubectl expose deployment web-2 --type=NodePort --port=8080
service/web-2 exposed
[root@ip-172-31-25-117 ~]#
[root@ip-172-31-25-117 ~]# kubectl get all
NAME                         READY   STATUS    RESTARTS   AGE
pod/web-1-64d7d68999-r9468   1/1     Running   0          3m37s
pod/web-2-9b89b499-mwpzl     1/1     Running   0          2m58s

NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service/kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP          15m
service/web-1        NodePort    10.96.149.216   <none>        8080:32029/TCP   17s
service/web-2        NodePort    10.96.38.150    <none>        8080:31632/TCP   10s

NAME                    READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/web-1   1/1     1            1           3m37s
deployment.apps/web-2   1/1     1            1           2m58s

NAME                               DESIRED   CURRENT   READY   AGE
replicaset.apps/web-1-64d7d68999   1         1         1       3m37s
replicaset.apps/web-2-9b89b499     1         1         1       2m58s
[root@ip-172-31-25-117 ~]#
```

Step 8 :: Create an Ingress

vi ingress.yaml
kubectl apply -f ingress.yaml

```
[root@ip-172-31-25-117 ~]# kubectl apply -f ingress.yaml
ingress.networking.k8s.io/lb-ingress created
[root@ip-172-31-25-117 ~]#
```

kubectl get ingress

```
[root@ip-172-31-25-117 ~]# kubectl get ingress
NAME         CLASS   HOSTS         ADDRESS      PORTS   AGE
lb-ingress   nginx   hello-world   172.18.0.4   80      3m12s
[root@ip-172-31-25-117 ~]#
```

Step 9 :: Update /etc/hosts to access the web

Copy the Address and HOSTS displayed in ingress (# kubectl get ingress)

vi /etc/hosts

```
[root@ip-172-31-25-117 ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost6 localhost6.localdomain6
172.18.0.4  hello-world
[root@ip-172-31-25-117 ~]#
```

Step 10 :: Result :: Access the web pages

curl hello-world/v1
curl hello-world/v2

```
[root@ip-172-31-25-117 ~]# curl hello-world/v1
Hello, world!
Version: 1.0.0
Hostname: web-1-64d7d68999-r9468
[root@ip-172-31-25-117 ~]#
[root@ip-172-31-25-117 ~]# curl hello-world/v2
Hello, world!
Version: 2.0.0
Hostname: web-2-9b89b499-mwpzl
[root@ip-172-31-25-117 ~]#
```