Ali Youssef Solh
DevOps bootcamp Capstone

DevOps Infra Optimization

Notes

All source files are found at https://github.com/AliYoussefSolh/Infra-Optimization

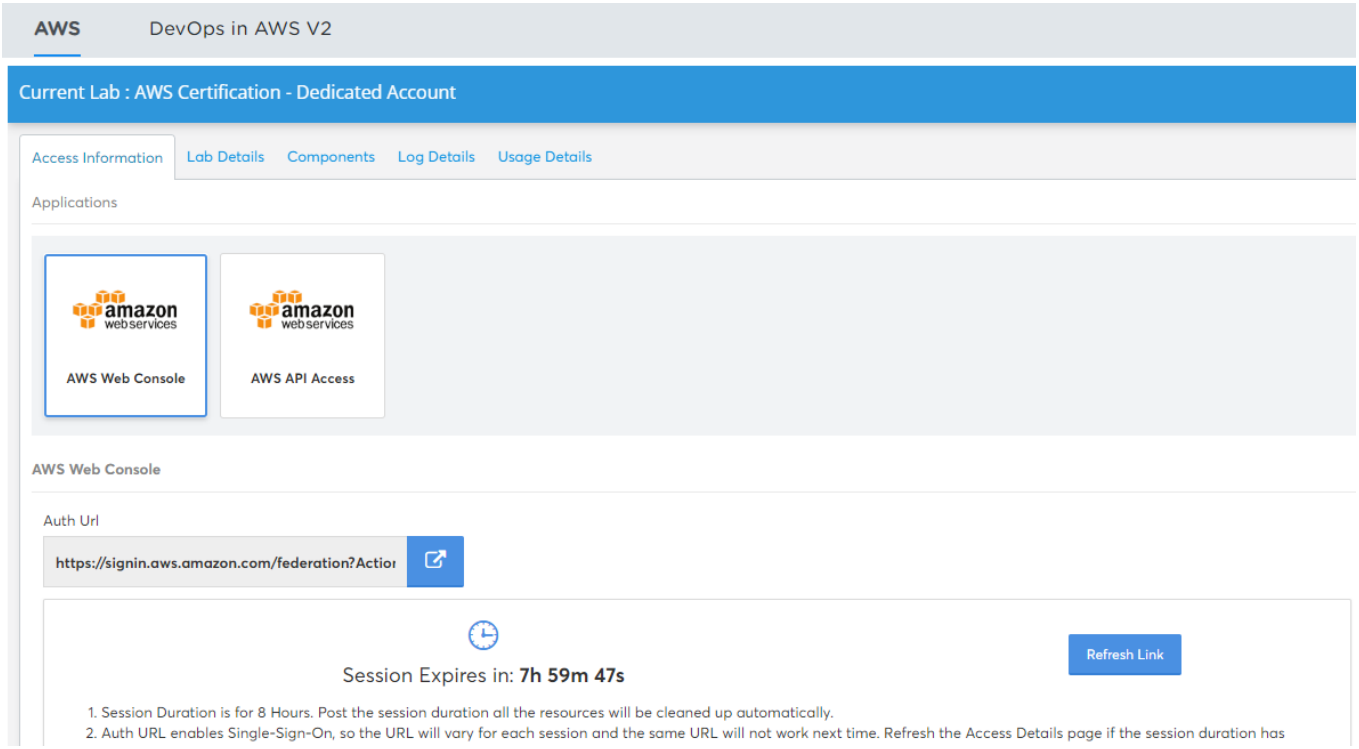
Because it makes advantage of the high availability of nodes provided by AWS EC2 instances, as well as the dependability and scalability of Kubernetes, this solution resolves the issue with the EasyPay application. This prevents downtime and allows the application to scale horizontally during high traffic times. The underlying configuration can be backed up using etcd snapshots. This will provide much better performance, and reliability, for the application than their current infrastructure.

Steps with Instructions

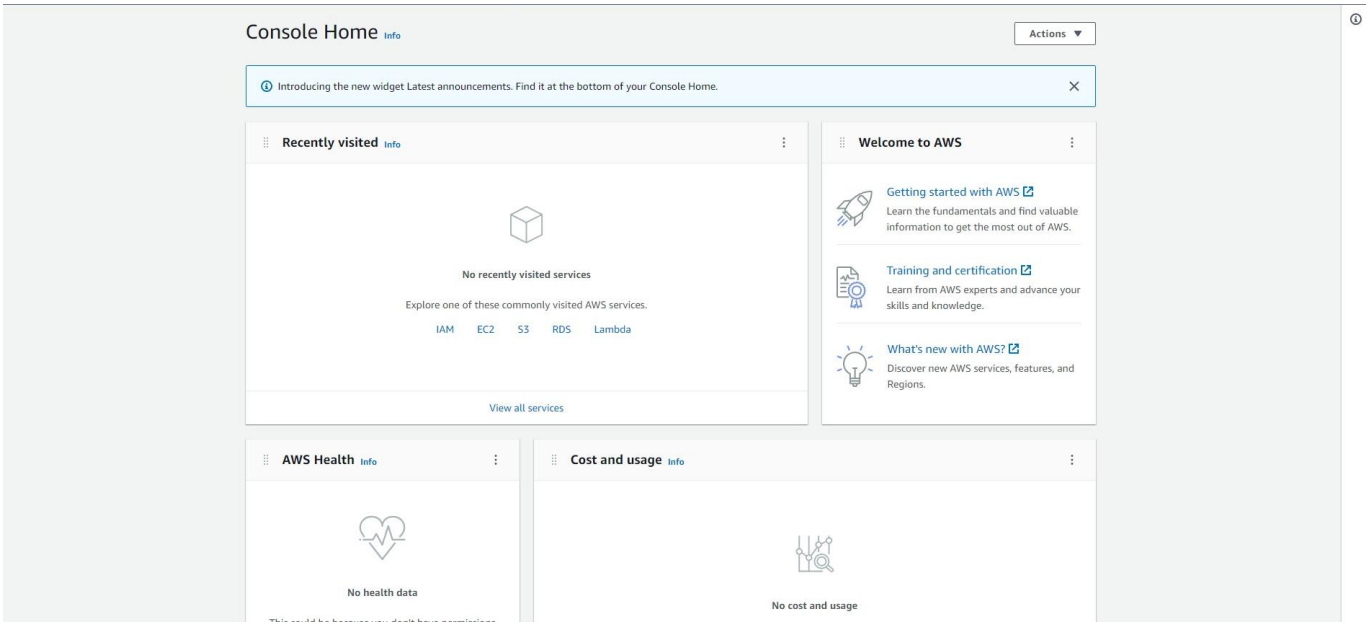
-Starting with the course lab environement

-We Log into the RDP session. This will be the Terraform controller

Start the AWS Lab environement



log into the AWS Console (will use this to manage and monitor our resources (EC2, EKS ….)



Create EC2 Instance using Terraform

From the RDP session:

- Verify installation of terraform

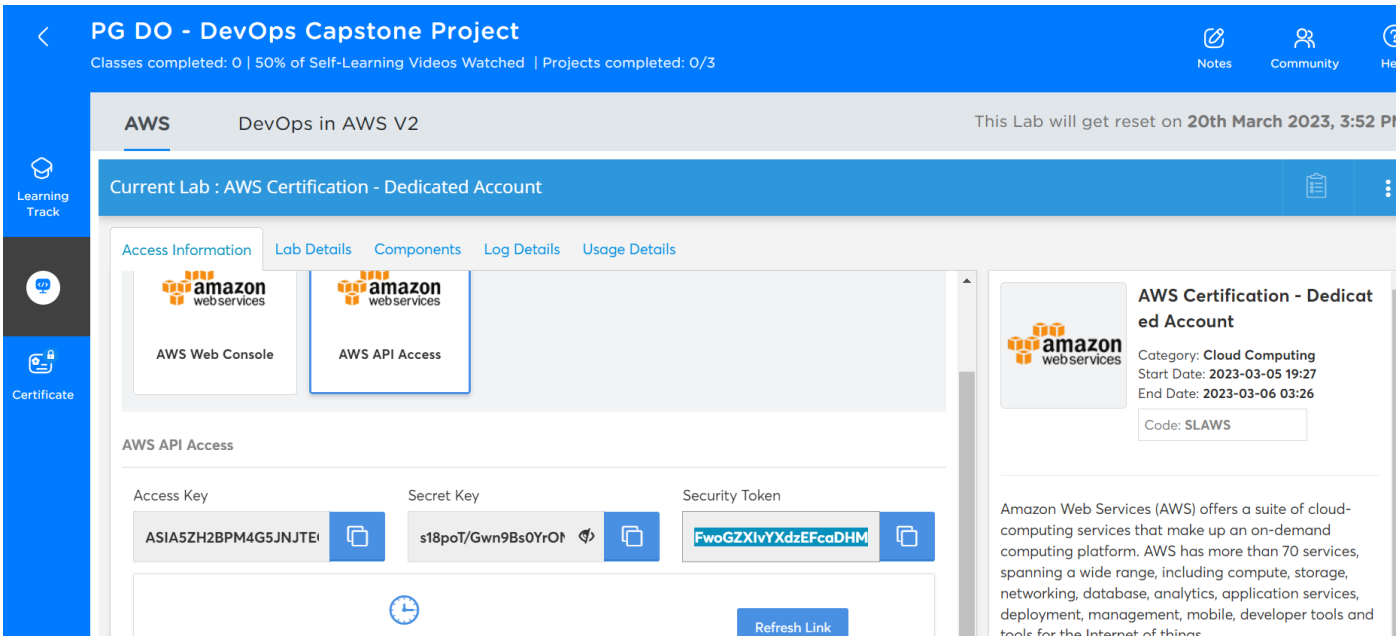
```
aliyoussefsolhl@ip-172-31-20-71:~$ terraform version

Terraform v1.1.6
on linux_amd64
```

```
Your version of Terraform is out of date! The latest version
is 1.2.3. You can update by downloading from
https://www.terraform.io/downloads.html
```

we export our AWS Key and ID for terraform to use

```
export AWS_ACCESS_KEY_ID="ASIA5ZH2BPM4G5JNJTEO"
export
AWS_SESSION_TOKEN="FwoGZXIvYXdzEFcaDHMHLMVfQUYDYrMSXiK4Acy
RuPPlKrmxMDtPZumQS0aRSEQSwydiUOrzliAPK1mp821umZCKlGLafpjJ8
0EbEenfuXmkHBNUN6qj79WzlImGiEZ6" export
AWS_SECRET_ACCESS_KEY="*****"
```

These values were pulled from the AWS API Access tab in the lab page



Create aws.tf

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ cat aws.tf

provider "aws" {
  region        = "us-east-1"
}
```

Initialization of Terraform project

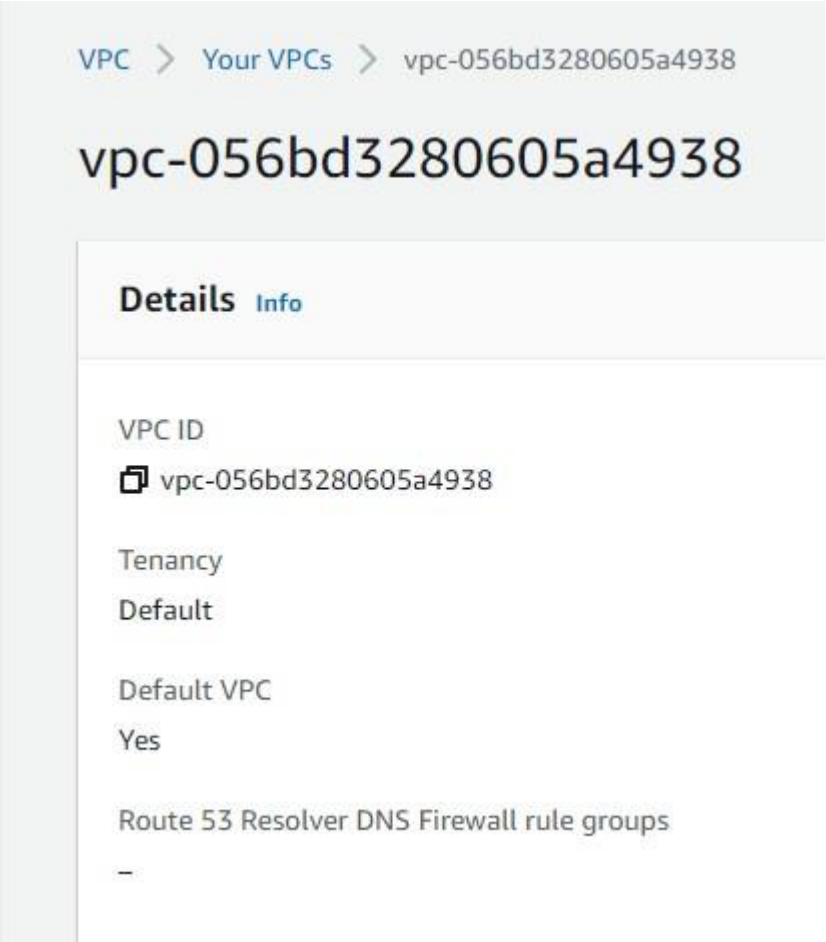
```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.20.0...
- Installed hashicorp/aws v4.20.0 (signed by HashiCorp)
```

```
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

verification that the terraform project initalization files are created

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ls -a
.  ..  .terraform  .terraform.lock.hcl  aws.tf
```

Obtain VPC ID `vpc-056bd3280605a4938`



Create main.tf that will define your resources

NOTE: The mentioned t2.micro and t3.micro do not have sufficient resources to satisfy the minimal system requirements due to kubeadm's requirements. t2.medium is therefore employed.

```
cat main.tf

resource "aws_instance" "ubuntu" {
  ami           = "ami-052efd3df9dad4825"
  instance_type = "t2.medium"
  key_name      = "${aws_key_pair.generated_key.key_name}"
  tags = {
    Name        = "terraform_instance"
  }
}

output "myEC2IP" {
  value = "${aws_instance.ubuntu.public_ip}"
}

resource "tls_private_key" "example" {
  algorithm = "RSA"
  rsa_bits  = 4096
}

resource "aws_key_pair" "generated_key" {
  key_name   = "mykey2"
  public_key = tls_private_key.example.public_key_openssh

provisioner "local-exec" { # Create "myKey.pem" to your computer!!
    command = "echo '${tls_private_key.example.private_key_pem}' > ./myKey.pem"
  }
}
```

Validate and deploy EC2 Instance

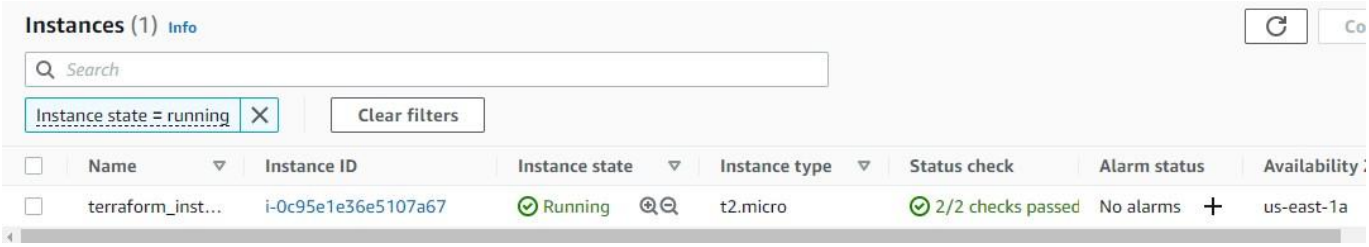
```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ terraform plan

...

aliyoussefsolhl@ip-172-31-20-71:~/capstone$ terraform apply

...

aws_instance.rhel: Creation complete after 42s [id=i-0c95e1e36e5107a67]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:
```

```
myEC2IP = "54.83.74.19"
```

We test this by ssh to the new EC2 instance (we change the permission of the key)

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ chmod 600 myKey.pem
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -i myKey.pem
ubuntu@54.83.74.19 [ec2-user@ip-172-31-20-71 ~]$ whoami && hostname
ec2-user
ip-172-31-20-71.ec2.internal
```

We can check the status from the AWS console

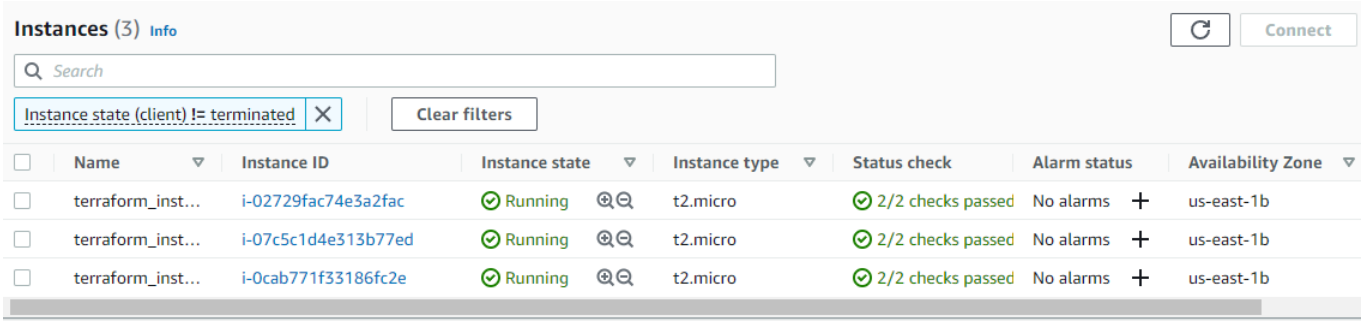


Now we will scale up by adding these 2 lines to main.tf

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ diff

  ami           = "ami-052efd3df9dad4825"
+ count = 3
  instance_type = "t2.medium"

...

  output "myEC2IP" {
- value = "${aws_instance.ubuntu.public_ip}"
+ value = "${aws_instance.ubuntu.*.public_ip}"
}
```

Now run apply to get the additional instances

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ terraform apply
```



From this point forward, I've added the following to my local "/etc/hosts" file to make hosts more readable.

```
...

54.226.15.188 master
54.167.63.97 node1
34.203.223.52 node2

...
```

Install and Configure Kubernetes

All the needed steps to install kubeadm and start our control plane (master) nodes are in the script `install_k8s.sh`

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ cat

install_k8s.sh #!/bin/bash
swapoff -a
curl -fsSL https://get.docker.com -o get-docker.sh
DRY_RUN=1 sudo sh ./get-docker.sh
sudo apt-get install -y apt-transport-https ca-certificates curl
sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg
https://packages.cloud.google.com/apt/doc/apt-key.gpg
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg]
https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee
/etc/apt/sources.list.d/kubernetes.list
sudo apt-get update
sudo apt-cache madison kubeadm
sudo apt-get install -y kubelet=1.23.6-00 kubeadm=1.23.6-00 kubectl=1.23.6-00
sudo hostnamectl set-hostname master.example.com
cat <<EOF | sudo tee /etc/docker/daemon.json
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
```

```
EOF

sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker

sudo kubeadm init
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
kubectl get nodes
```

Copy to master node

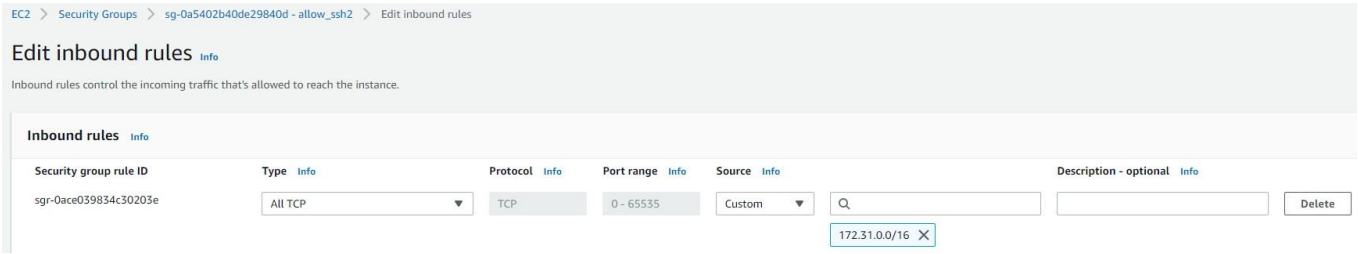
```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ chmod +x install_k8s.sh
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ scp -p -i myKey.pem
install_k8s.sh ubuntu@master:/home/ubuntu
install_k8s.sh
100% 1106     2.2MB/s   00:00
```

Execute the script on the remote system

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -i myKey.pem ubuntu@master
"./install_k8s.sh"

...

kubeadm join 172.31.16.119:6443 --token e8w6ko.4q91edczl30a6ar5 \
        --discovery-token-ca-cert-hash
sha256:1d7f217f7c8dc989b8328546559959298e959584d4fab71828c578b90533abdb
NAME                 STATUS     ROLES                  AGE    VERSION
master.example.com   NotReady   control-plane,master   3s     v1.23.6
```

Now we must set up our worker nodes First we must allow TCP traffic within subnet

Within `EC2 > Security Groups > sg-0a5402b40de29840d - allow_ssh2 > Edit inbound rules`




```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ scp -p -i myKey.pem node.sh
ubuntu@node2:~
```

```
node.sh                                                   100%  944     1.9MB/s
00:00

aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -t -i myKey.pem ubuntu@node2
"./node.sh"

...

aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -t -i myKey.pem ubuntu@node2 "sudo
kubeadm join 172.31.16.119:6443 --token e8w6ko.4q91edczl30a6ar5 --discovery-token-
ca-cert-hash
sha256:1d7f217f7c8dc989b8328546559959298e959584d4fab71828c578b90533abdb"

...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ scp -p -i myKey.pem
node.sh
node.sh                                                   100%  944     1.9MB/s
00:00

aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -t -i myKey.pem ubuntu@node1
"./node.sh"

...

aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -t -i myKey.pem ubuntu@node1 "sudo
kubeadm join 172.31.16.119:6443 --token e8w6ko.4q91edczl30a6ar5 --discovery-token-
ca-cert-hash
sha256:1d7f217f7c8dc989b8328546559959298e959584d4fab71828c578b90533abdb"

...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -i myKey.pem ubuntu@master "kubectl
get nodes"
```

```
NAME                 STATUS     ROLES                  AGE     VERSION
master.example.com   NotReady   control-plane,master   110m    v1.23.6
node1.example.com    NotReady   <none>                 8m32s   v1.23.6
node2.example.com    NotReady   <none>                 74s     v1.23.6
```

Setting up the overlay network

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -i myKey.pem ubuntu@master
ubuntu@node1:~$ kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-
version=$(kubectl version | base64 | tr -d '\n')"
```

```
aliyoussefsolhl@ip-172-31-20-71:~/capstone$ ssh -i myKey.pem ubuntu@master "kubectl
get nodes"

NAME                 STATUS   ROLES                  AGE     VERSION
master.example.com   Ready    control-plane,master   3h18m   v1.23.6
node1.example.com    Ready    <none>                 155m    v1.23.6
node2.example.com    Ready    <none>                 145m    v1.23.6
```

Deploy application

```
aliyoussefsolhl@ip-172-31-20-71 ssh -i myKey.pem
ubuntu@master ubuntu@master:~$ kubectl create -f mydb.yml
ubuntu@master:~$ kubectl expose deployment mydb --port=3306

ubuntu@master:~$ kubectl create -f wp.yml
ubuntu@master:~$ kubectl expose deployment wp --port=80 --type=NodePort
```

-Add security group policy for port

- Show access to WP config page

Apply network policy

```
kubectl apply -f np.yaml
```

Create Users and ACL

```
ubuntu@master:~$ kubectl create serviceaccount newroleadded
serviceaccount/newroleadded created

ubuntu@master:~$ kubectl create clusterrole newroleadded --verb=get --verb=list --
verb=create --verb=update --resource=pods
clusterrole.rbac.authorization.k8s.io/newroleadded created

ubuntu@master:~$ kubectl create clusterrolebinding newroleadded --
serviceaccount=default:newroleadded --clusterrole=newroleadded
clusterrolebinding.rbac.authorization.k8s.io/newroleadded created

ubuntu@master:~$ TOKEN=$(kubectl describe secrets "$(kubectl describe
serviceaccount newroleadded | grep -i Tokens | awk '{print $2}')" | grep token: |
awk '{print $2}')

ubuntu@master:~$ kubectl config set-credentials myuser1 --token=$TOKEN
User "myuser1" set.

ubuntu@master:~$ kubectl config set-context newcontextadded --cluster=kubernetes -
-user=myuser1
Context "newcontextadded" created.

ubuntu@master:~$ kubectl config use-context newcontextadded
Switched to context "newcontextadded".

ubuntu@master:~$ kubectl auth can-i get pods --all-namespaces
yes

ubuntu@master:~$ kubectl get all
NAME                     READY   STATUS    RESTARTS   AGE
mydb-659c7949cd-xrw25    1/1     Running   0          14m
wp-946c66d98-8csh4       1/1     Running   0          13m
Error from server (Forbidden): replicationcontrollers is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource
"replicationcontrollers" in API group "" in the namespace "default"
Error from server (Forbidden): services is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource "services" in
API group "" in the namespace "default"
Error from server (Forbidden): daemonsets.apps is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource "daemonsets" in
API group "apps" in the namespace "default"
Error from server (Forbidden): deployments.apps is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource "deployments" in
API group "apps" in the namespace "default"
Error from server (Forbidden): replicasets.apps is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource "replicasets" in
API group "apps" in the namespace "default"
Error from server (Forbidden): statefulsets.apps is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource "statefulsets"
in API group "apps" in the namespace "default"
Error from server (Forbidden): horizontalpodautoscalers.autoscaling is forbidden:
User "system:serviceaccount:default:newroleadded" cannot list resource
"horizontalpodautoscalers" in API group "autoscaling" in the namespace "default"
```

```
Error from server (Forbidden): cronjobs.batch is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource "cronjobs" in
API group "batch" in the namespace "default"
Error from server (Forbidden): jobs.batch is forbidden: User
"system:serviceaccount:default:newroleadded" cannot list resource "jobs" in API
group "batch" in the namespace "default"

ubuntu@master:~$ kubectl auth can-i get pods --all-namespaces
yes

ubuntu@master:~$ kubectl auth can-i get deployment --all-namespaces
no

ubuntu@master:~$  kubectl config use-context kubernetes-admin@kubernetes
Switched to context "kubernetes-admin@kubernetes".

ubuntu@master:~$  kubectl auth can-i get deployment --all-namespaces
yes
```

Take a snapshot of ETCD database

Install etcd client tools

```
sudo apt update -y
sudo apt install etcd-client
```

```
ubuntu@master:~$ hostname -I
172.31.16.119 172.17.0.1 10.32.0.1

ubuntu@master:~$ kubectl get nodes -o wide
NAME                 STATUS     ROLES                  AGE     VERSION   INTERNAL-
IP    EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION    CONTAINER-RUNTIME
master.example.com   Ready    control-plane,master   4h19m   v1.23.6
172.31.16.119   <none>        Ubuntu 22.04 LTS   5.15.0-1011-aws
docker://20.10.17
node1.example.com    Ready    <none>                 157m    v1.23.6
172.31.20.65    <none>        Ubuntu 22.04 LTS   5.15.0-1011-aws
docker://20.10.17
node2.example.com    Ready    <none>                 150m    v1.23.6   172.31.28.7
<none>        Ubuntu 22.04 LTS   5.15.0-1011-aws   docker://20.10.17


ubuntu@master:~$ export advertise_url="172.31.16.119:2379"

ubuntu@master:~$ echo $advertise_url
172.31.16.119:2379

ubuntu@master:~$ sudo ETCDCTL_API=3 etcdctl --endpoints $advertise_url --cacert
/etc/kubernetes/pki/etcd/ca.crt --key /etc/kubernetes/pki/etcd/server.key --cert
```

```
/etc/kubernetes/pki/etcd/server.crt snapshot save test1.db

2023-03-5 22:19:12.947059 I | clientv3: opened snapshot stream; downloading
2023-03-5 22:19:13.056969 I | clientv3: completed snapshot read; closing
Snapshot saved at test1.db

ubuntu@master:~$ du -h test1.db
7.7M    test1.db
```

Set criteria such that if the memory of CPU goes beyond 50%, environments automatically get scaled up and configured

---

Create Metrics server

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-
server/releases/latest/download/components.yaml

serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator
created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
```

Needs a patch

```
ubuntu@master:~$ kubectl get pods -n kube-system
NAME                                READY   STATUS    RESTARTS      AGE
coredns-64897985d-64mpp             1/1     Running   0             3h41m
coredns-64897985d-v5z2r             1/1     Running   0             3h41m
kube-apiserver-master.example.com   0/1     Pending   0             1s
kube-proxy-n99wn                    1/1     Running   0             112m
kube-proxy-p4swz                    1/1     Running   1             3h41m
kube-proxy-rp9nb                    1/1     Running   0             120m
metrics-server-847dcc659d-qtfqq     0/1     Running   0             48s
weave-net-2k5kl                     2/2     Running   2 (36m ago)   55m
weave-net-68qm9                     0/2     Pending   0             55m
weave-net-pl9j2                     2/2     Running   2 (36m ago)   55m
```

apply patch

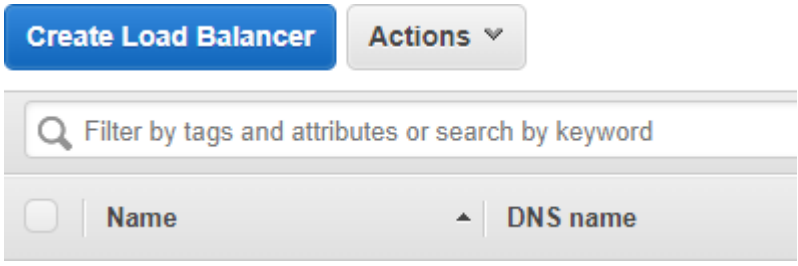
```
ubuntu@master:~$ wget -c
https://gist.githubusercontent.com/initcron/1a2bd25353e1faa22a0ad41ad1c01b62/raw/0
08e23f9fbf4d7e2cf79df1dd008de2f1db62a10/k8s-metrics-server.patch.yaml
--2022-06-24 22:30:42--
https://gist.githubusercontent.com/initcron/1a2bd25353e1faa22a0ad41ad1c01b62/raw/0
08e23f9fbf4d7e2cf79df1dd008de2f1db62a10/k8s-metrics-server.patch.yaml
Resolving gist.githubusercontent.com (gist.githubusercontent.com)...
185.199.109.133, 185.199.110.133, 185.199.111.133, ...
Connecting to gist.githubusercontent.com
(gist.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 205 [text/plain]
Saving to: ‘k8s-metrics-server.patch.yaml’

k8s-metrics-server.patch.yaml                      100%
[=================================================================================
=======================>]      205  --.-KB/s    in 0s

2023-03-5 22:30:42 (14.0 MB/s) - ‘k8s-metrics-server.patch.yaml’ saved [205/205]


ubuntu@master:~$ kubectl patch deploy metrics-server -p "$(cat k8s-metrics-
server.patch.yaml)" -n kube-system

deployment.apps/metrics-server patched
```

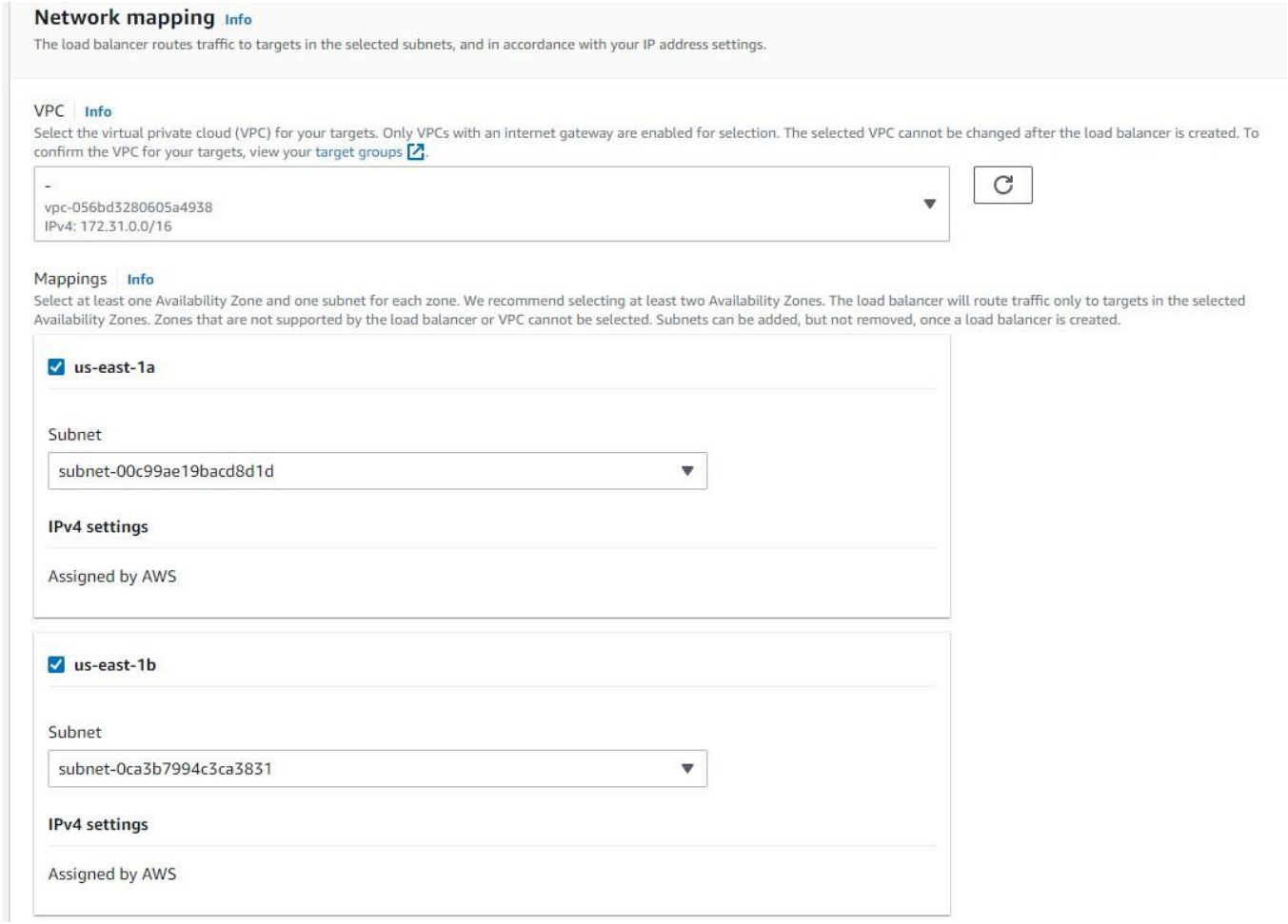
```
ubuntu@master:~$ kubectl get pods -n kube-system
NAME                                       READY   STATUS    RESTARTS         AGE
coredns-64897985d-64mpp                    1/1     Running   0
4h22m
coredns-64897985d-v5z2r                    1/1     Running   0
4h22m
etcd-master.example.com                    1/1     Running   1 (4h14m ago)    37m
kube-apiserver-master.example.com          1/1     Running   1 (4h14m ago)    37m
kube-controller-manager-master.example.com 1/1     Running   1 (4h14m ago)    37m
kube-proxy-n99wn                           1/1     Running   0
153m
kube-proxy-p4swz                           1/1     Running   1 (4h14m ago)
4h22m
kube-proxy-rp9nb                           1/1     Running   0
161m
kube-scheduler-master.example.com          1/1     Running   1 (4h14m ago)    36m
metrics-server-77b7f4f884-g5jl5            1/1     Running   0                57s
weave-net-2k5kl                            2/2     Running   2 (77m ago)      97m
weave-net-68qm9                            2/2     Running   0                97m
weave-net-pl9j2                            2/2     Running   2 (77m ago)      97m
```

Now that the metrics server is running we can apply out horizonal scaling policy

```
ubuntu@master:~$ kubectl apply -f hpa.yaml

ubuntu@master:~$ kubectl get horizontalpodautoscaler
NAME          REFERENCE        TARGETS          MINPODS    MAXPODS    REPLICAS    AGE
php-apache    Deployment/wp    0%/50%    1          10         1           23s
```

Create Load balancer





Basic configuration

Load balancer name
Name must be unique within your AWS account and cannot be changed after the load balancer is created.

devops-capstone

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme Info
Scheme cannot be changed after the load balancer is created.

- (•) Internet-facing
  An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. Learn more
- ( ) Internal
  An internal load balancer routes requests from clients to targets using private IP addresses.

IP address type Info
Select the type of IP addresses that your subnets use.

- (•) IPv4
  Recommended for internal load balancers.
- ( ) Dualstack
  Includes IPv4 and IPv6 addresses.

Network mapping Info

The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC Info
Select the virtual private cloud (VPC) for your targets. Only VPCs with an internet gateway are enabled for selection. The selected VPC cannot be changed after the load balancer is created. To confirm the VPC for your targets, view your target groups.

-
vpc-056bd3280605a4938
IPv4: 172.31.0.0/16

Mappings Info
Select at least one Availability Zone and one subnet for each zone. We recommend selecting at least two Availability Zones. The load balancer will route traffic only to targets in the selected Availability Zones. Zones that are not supported by the load balancer or VPC cannot be selected. Subnets can be added, but not removed, once a load balancer is created.

- [x] **us-east-1a**

Subnet

subnet-00c99ae19bacd8d1d

IPv4 settings

Assigned by AWS

Listeners and routing Info

A listener is a process that checks for connection requests, using the protocol and port you configure. Traffic received by the listener is then routed per your specification. You can specify multiple rules and multiple certificates per listener after the load balancer is created.

▼ Listener **HTTP:31478** Remove

Protocol: HTTP
Port: 31478
1-65535

Default action Info
Forward to: capstone-instances, HTTP
Target type: Instance, IPv4
Create target group

Add listener

Security groups Info

A security group is a set of firewall rules that control the traffic to your load balancer.

Security groups

Select up to 5 security groups

Create new security group

default sg-0c930c5d873c9b2a0
VPC: vpc-056bd3280605a4938

allow_ssh2 sg-0a5402b40de29840d
VPC: vpc-056bd3280605a4938

Basic configuration

Settings in this section cannot be changed after the target group is created.

Choose a target type

- (•) Instances
  - Supports load balancing to instances within a specific VPC.
  - Facilitates the use of Amazon EC2 Auto Scaling to manage and scale your EC2 capacity.
- ( ) IP addresses
  - Supports load balancing to VPC and on-premises resources.
  - Facilitates routing to multiple IP addresses and network interfaces on the same instance.
  - Offers flexibility with microservice based architectures, simplifying inter-application communication.
  - Supports IPv6 targets, enabling end-to-end IPv6 communication, and IPv4-to-IPv6 NAT.
- ( ) Lambda function
  - Facilitates routing to a single Lambda function.
  - Accessible to Application Load Balancers only.
- ( ) Application Load Balancer
  - Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
  - Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

capstone-instances

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol : Port

HTTP : 31406

VPC

Select the VPC with the instances that you want to include in the target group.

-
vpc-056bd3280605a4938
IPv4: 172.31.0.0/16

Protocol version

- (•) HTTP1
  Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.
- ( ) HTTP2
  Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.
- ( ) gRPC

Health checks

The associated load balancer periodically sends requests, per the settings below, to the registered targets to test their status.

Health check protocol

HTTP

Health check path

Use the default path of "/" to ping the root, or specify a custom path if preferred.

/healthz

Up to 1024 characters allowed.

▶ Advanced health check settings

Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (3/3)

Filter resources by property or value

Instance ID	Name	State	Security groups	Zone	Subnet ID
i-02da2d718bd3bcb4e	terraform_instance	running	allow_ssh2	us-east-1b	subnet-0ca3b7994c3ca3831
i-0f531cb0547fafbb5	terraform_instance	running	allow_ssh2	us-east-1b	subnet-0ca3b7994c3ca3831
i-009f0515f0b3468b1	terraform_instance	running	allow_ssh2	us-east-1b	subnet-0ca3b7994c3ca3831

3 selected

Ports for the selected instances

Ports for routing traffic to the selected instances.

31406

1-65535 (separate multiple ports with commas)

Include as pending below

Review targets

Targets (0)

All | Filter resources by property or value | Remove all pending

Remove	Health status	Instance ID	Name	Port	State	Security groups	Zone	Subnet ID

No instances added yet

Specify instances above, or leave the group empty if you prefer to add targets later.

0 pending

Cancel | Previous | Create target group

Summary

Review and confirm your configurations. Estimate cost

Basic configuration Edit

devops-capstone

- Internet-facing
- IPv4

Security groups Edit

- default
  sg-0c930c5d873c9b2a0
- allow_ssh2
  sg-0a5402b40de29840d

Network mapping Edit

VPC vpc-056bd3280605a4938

- us-east-1a
  subnet-00c99ae19bacd8d1d
- us-east-1b
  subnet-0ca3b7994c3ca3831

Listeners and routing Edit

- HTTP:31478 defaults to
  capstone-instances

Add-on services Edit

None

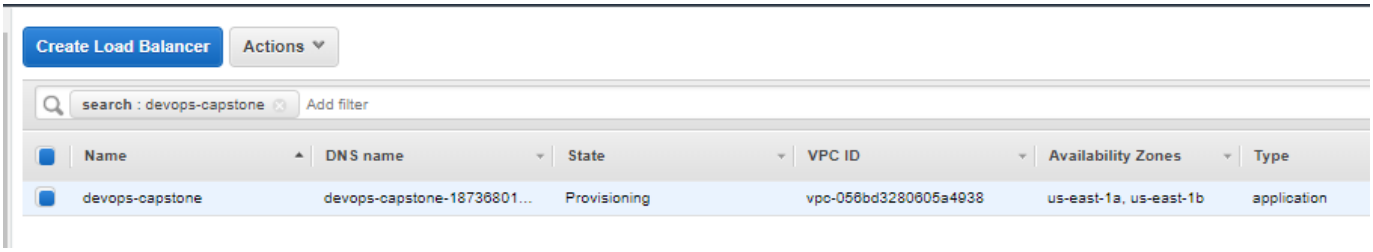
Tags Edit

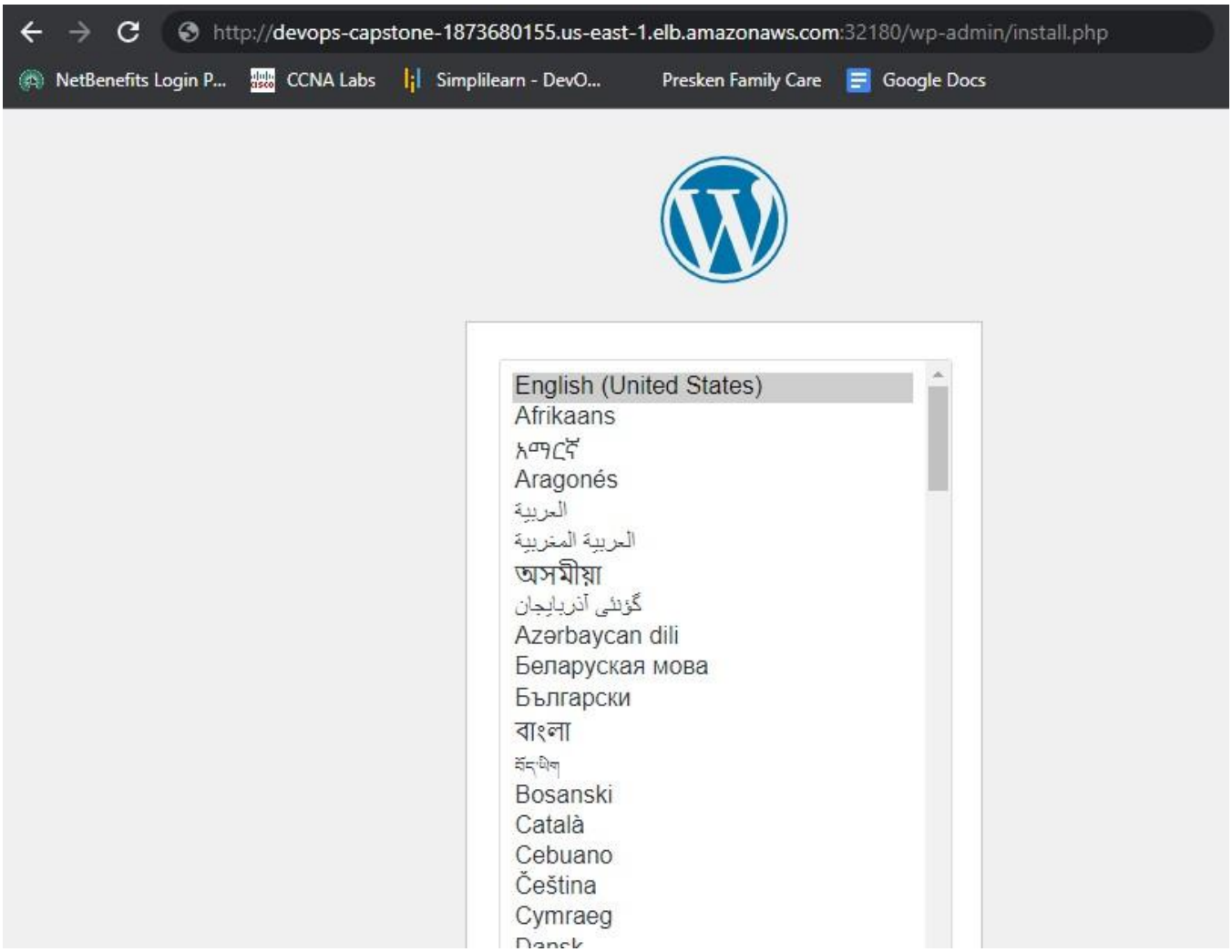
None

Attributes

Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.

Cancel | Create load balancer


Create Load Balancer
Actions
search : devops-capstone
Add filter
Name
DNS name
State
VPC ID
Availability Zones
Type
devops-capstone
devops-capstone-18736801...
Provisioning
vpc-056bd3280605a4938
us-east-1a, us-east-1b
application


http://devops-capstone-1873680155.us-east-1.elb.amazonaws.com:32180/wp-admin/install.php
NetBenefits Login P...
CCNA Labs
Simplilearn - DevO...
Presken Family Care
Google Docs
English (United States)
Afrikaans
አማርኛ
Aragonés
العربية
العربية المغربية
অসমীয়া
گؤنئی آذربایجان
Azərbaycan dili
Беларуская мова
Български
বাংলা
Bosanski
Català
Cebuano
Čeština
Cymraeg