Enable the AWS Load balancer controller

Enable the AWS Load balancer controller on the cluster

```
1
cd ~/environment/tfekscode/lb2
```

Initialize Terraform:

```
1
terraform init
```

Validate the Terraform code:

```
1
terraform validate
```

Plan the deployment:

```
1
terraform plan -out tfplan
An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_iam_policy.load-balancer-policy will be created
  + resource "aws_iam_policy" "load-balancer-policy" {
      + arn         = (known after apply)
      + description = "AWS LoadBalancer Controller IAM Policy"
      + id          = (known after apply)
      + name        = "AWSLoadBalancerControllerIAMPolicy"
      + path        = "/"
      + policy      = jsonencode(
          {
```

```
+ Statement = [
  + {
    + Action   = [
      + "iam:CreateServiceLinkedRole",
      + "ec2:DescribeAccountAttributes",
      + "ec2:DescribeAddresses",
      + "ec2:DescribeInternetGateways",
      + "ec2:DescribeVpcs",
      + "ec2:DescribeSubnets",
      + "ec2:DescribeSecurityGroups",
      + "ec2:DescribeInstances",
      + "ec2:DescribeNetworkInterfaces",
      + "ec2:DescribeTags",
      + "elasticloadbalancing:DescribeLoadBalancers",
      + "elasticloadbalancing:DescribeLoadBalancerAttributes",
      + "elasticloadbalancing:DescribeListeners",
      + "elasticloadbalancing:DescribeListenerCertificates",
      + "elasticloadbalancing:DescribeSSLPolicies",
      + "elasticloadbalancing:DescribeRules",
      + "elasticloadbalancing:DescribeTargetGroups",
      + "elasticloadbalancing:DescribeTargetGroupAttributes",
      + "elasticloadbalancing:DescribeTargetHealth",
      + "elasticloadbalancing:DescribeTags",
     ]
    + Effect   = "Allow"
    + Resource = "*"
   },
  + {
    + Action   = [
      + "cognito-idp:DescribeUserPoolClient",
      + "acm:ListCertificates",
      + "acm:DescribeCertificate",
      + "iam:ListServerCertificates",
      + "iam:GetServerCertificate",
      + "waf-regional:GetWebACL",
      + "waf-regional:GetWebACLForResource",
```

```
      + "waf-regional:AssociateWebACL",
      + "waf-regional:DisassociateWebACL",
      + "wafv2:GetWebACL",
      + "wafv2:GetWebACLForResource",
      + "wafv2:AssociateWebACL",
      + "wafv2:DisassociateWebACL",
      + "shield:GetSubscriptionState",
      + "shield:DescribeProtection",
      + "shield:CreateProtection",
      + "shield:DeleteProtection",
    ]
  + Effect   = "Allow"
  + Resource = "*"
 },
+ {
  + Action   = [
      + "ec2:AuthorizeSecurityGroupIngress",
      + "ec2:RevokeSecurityGroupIngress",
    ]
  + Effect   = "Allow"
  + Resource = "*"
 },
+ {
  + Action   = [
      + "ec2:CreateSecurityGroup",
    ]
  + Effect   = "Allow"
  + Resource = "*"
 },
+ {
  + Action    = [
      + "ec2:CreateTags",
    ]
  + Condition = {
      + Null       = {
          + aws:RequestTag/elbv2.k8s.aws/cluster = "false"
```

```
        }
      + StringEquals = {
        + ec2:CreateAction = "CreateSecurityGroup"
        }
    }
  + Effect   = "Allow"
  + Resource = "arn:aws:ec2:*:*:security-group/*"
 },
+ {
  + Action   = [
    + "ec2:CreateTags",
    + "ec2:DeleteTags",
   ]
  + Condition = {
    + Null = {
      + aws:RequestTag/elbv2.k8s.aws/cluster  = "true"
      + aws:ResourceTag/elbv2.k8s.aws/cluster = "false"
      }
    }
  + Effect   = "Allow"
  + Resource = "arn:aws:ec2:*:*:security-group/*"
 },
+ {
  + Action   = [
    + "ec2:AuthorizeSecurityGroupIngress",
    + "ec2:RevokeSecurityGroupIngress",
    + "ec2:DeleteSecurityGroup",
   ]
  + Condition = {
    + Null = {
      + aws:ResourceTag/elbv2.k8s.aws/cluster = "false"
      }
    }
  + Effect   = "Allow"
  + Resource = "*"
 },
```

```
+ {
  + Action   = [
    + "elasticloadbalancing:CreateLoadBalancer",
    + "elasticloadbalancing:CreateTargetGroup",
   ]
  + Condition = {
    + Null = {
      + aws:RequestTag/elbv2.k8s.aws/cluster = "false"
     }
   }
  + Effect   = "Allow"
  + Resource  = "*"
 },
+ {
  + Action   = [
    + "elasticloadbalancing:CreateListener",
    + "elasticloadbalancing:DeleteListener",
    + "elasticloadbalancing:CreateRule",
    + "elasticloadbalancing:DeleteRule",
   ]
  + Effect   = "Allow"
  + Resource = "*"
 },
+ {
  + Action   = [
    + "elasticloadbalancing:AddTags",
    + "elasticloadbalancing:RemoveTags",
   ]
  + Condition = {
    + Null = {
      + aws:RequestTag/elbv2.k8s.aws/cluster  = "true"
      + aws:ResourceTag/elbv2.k8s.aws/cluster = "false"
     }
   }
  + Effect   = "Allow"
  + Resource  = [
```

```
      + "arn:aws:elasticloadbalancing:*:*:targetgroup/*/*",
      + "arn:aws:elasticloadbalancing:*:*:loadbalancer/net/*/*",
      + "arn:aws:elasticloadbalancing:*:*:loadbalancer/app/*/*",
    ]
 },
+ {
  + Action    = [
      + "elasticloadbalancing:ModifyLoadBalancerAttributes",
      + "elasticloadbalancing:SetIpAddressType",
      + "elasticloadbalancing:SetSecurityGroups",
      + "elasticloadbalancing:SetSubnets",
      + "elasticloadbalancing:DeleteLoadBalancer",
      + "elasticloadbalancing:ModifyTargetGroup",
      + "elasticloadbalancing:ModifyTargetGroupAttributes",
      + "elasticloadbalancing:DeleteTargetGroup",
    ]
  + Condition = {
      + Null = {
          + aws:ResourceTag/elbv2.k8s.aws/cluster = "false"
        }
    }
  + Effect    = "Allow"
  + Resource  = "*"
 },
+ {
  + Action   = [
      + "elasticloadbalancing:RegisterTargets",
      + "elasticloadbalancing:DeregisterTargets",
    ]
  + Effect   = "Allow"
  + Resource = "arn:aws:elasticloadbalancing:*:*:targetgroup/*/*"
 },
+ {
  + Action   = [
      + "elasticloadbalancing:SetWebAcl",
      + "elasticloadbalancing:ModifyListener",
```

```
            + "elasticloadbalancing:AddListenerCertificates",
            + "elasticloadbalancing:RemoveListenerCertificates",
            + "elasticloadbalancing:ModifyRule",
          ]
        + Effect   = "Allow"
        + Resource = "*"
      },
    ]
  + Version   = "2012-10-17"
 }
 )
}

# helm_release.aws-load-balancer-controller will be created
+ resource "helm_release" "aws-load-balancer-controller" {
  + atomic                     = false
  + chart                      = "aws-load-balancer-controller"
  + cleanup_on_fail            = false
  + create_namespace           = false
  + dependency_update          = false
  + disable_crd_hooks          = false
  + disable_openapi_validation = false
  + disable_webhooks           = false
  + force_update               = false
  + id                         = (known after apply)
  + lint                       = false
  + max_history                = 0
  + metadata                   = (known after apply)
  + name                       = "aws-load-balancer-controller"
  + namespace                  = "kube-system"
  + recreate_pods              = false
  + render_subchart_notes      = true
  + replace                    = false
  + repository                 = "https://aws.github.io/eks-charts"
  + reset_values               = false
  + reuse_values               = false
```

```
  + skip_crds               = false
  + status                  = "deployed"
  + timeout                 = 300
  + verify                  = false
  + version                 = "1.1.2"
  + wait                    = true

  + set {
      + name  = "clusterName"
      + value = "mycluster1"
    }
  + set {
      + name  = "image.repository"
      + value = "602401143452.dkr.ecr.eu-west-1.amazonaws.com/amazon/aws-load-balancer-controller"
    }
  + set {
      + name  = "image.tag"
      + value = "v2.1.0"
    }
  + set {
      + name  = "serviceAccount.name"
      + value = "aws-load-balancer-controller"
    }
}

# null_resource.destroy will be created
+ resource "null_resource" "destroy" {
  + id      = (known after apply)
  + triggers = (known after apply)
}

# null_resource.policy will be created
+ resource "null_resource" "policy" {
  + id      = (known after apply)
  + triggers = (known after apply)
}
```

```
 # null_resource.post-policy will be created
 + resource "null_resource" "post-policy" {
    + id       = (known after apply)
    + triggers = (known after apply)
  }

Plan: 5 to add, 0 to change, 0 to destroy.

------------------------------------------------------------------------

This plan was saved to: tfplan

To perform exactly these actions, run the following command to apply:
    terraform apply "tfplan"
```

You can see from the plan the following resources will be created

- A Load Balancer policy.
- A null provider "policy" to create the policy.
- A null provider "post_policy" to implement the CRD.
- Terraform installs the helm chart for the Load Balancer Controller.

Build the environment:

```
1
terraform apply tfplan
null_resource.policy: Creating...
null_resource.policy: Provisioning with 'local-exec'...
null_resource.policy (local-exec): Executing: ["/bin/bash" "-c" "          curl -o iam-policy.json
https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/main/docs/install/iam_policy.json\n"]
null_resource.policy (local-exec):   % Total    % Received % Xferd  Average Speed   Time    Time     Time
Current
null_resource.policy (local-exec):                                  Dload  Upload   Total   Spent    Left  Speed
null_resource.policy (local-exec):   0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
```

null_resource.policy (local-exec): 100 6620 100 6620   0   0 32292   0 --:--:-- --:--:-- --:--:-- 32292

null_resource.policy: Creation complete after 1s [id=4770428406038776500]

aws_iam_policy.load-balancer-policy: Creating...

aws_iam_policy.load-balancer-policy: Creation complete after 1s [id=arn:aws:iam::984587260948:policy/AWSLoadBalancerControllerIAMPolicy]

null_resource.destroy: Creating...

null_resource.post-policy: Creating...

null_resource.destroy: Creation complete after 0s [id=381548657479034480]

null_resource.post-policy: Provisioning with 'local-exec'...

null_resource.post-policy (local-exec): Executing: ["/bin/bash" "-c" "   reg=$(echo arn:aws:eks:eu-west-1:984587260948:cluster/mycluster1 | cut -f4 -d':')\n   acc=$(echo arn:aws:eks:eu-west-1:984587260948:cluster/mycluster1 | cut -f5 -d':')\n   cn=$(echo mycluster1)\n   echo \"$reg $cn $acc\"\n   ./post-policy.sh $reg $cn $acc\n   echo \"reannotate nodes\"\n   cd ../eks-cidr\n ./reannotate-nodes.sh\n   echo \"done\"\n"]

null_resource.post-policy (local-exec): eu-west-1 mycluster1 984587260948

null_resource.post-policy (local-exec): REGION is eu-west-1

null_resource.post-policy (local-exec): CLUSTER is mycluster1

null_resource.post-policy (local-exec): ACCOUNT is 984587260948

null_resource.post-policy (local-exec): --2021-01-10 14:16:30-- https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-controller/crds/crds.yaml

null_resource.post-policy (local-exec): Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 199.232.24.133

null_resource.post-policy (local-exec): Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|199.232.24.133|:443... connected.

null_resource.post-policy (local-exec): HTTP request sent, awaiting response...

null_resource.post-policy (local-exec): 200 OK

null_resource.post-policy (local-exec): Length: 7518 (7.3K) [text/plain]

null_resource.post-policy (local-exec): Saving to: ‘crds.yaml’

null_resource.post-policy (local-exec):   0K ....   100% 854K=0.009s

null_resource.post-policy (local-exec): 2021-01-10 14:16:30 (854 KB/s) - ‘crds.yaml’ saved [7518/7518]

null_resource.post-policy (local-exec): customresourcedefinition.apiextensions.k8s.io/targetgroupbindings.elbv2.k8s.aws created

null_resource.post-policy (local-exec): reannotate nodes

null_resource.post-policy (local-exec): ip-10-0-1-117.eu-west-1.compute.internal eu-west-1a

null_resource.post-policy (local-exec): kubectl annotate node ip-10-0-1-117.eu-west-1.compute.internal k8s.amazonaws.com/eniConfig=eu-west-1a-pod-netconfig

```
null_resource.post-policy (local-exec): error: --overwrite is false but found the following declared annotation(s): 'k8s.amazonaws.com/eniConfig' already has a value (eu-west-1a-pod-netconfig)

null_resource.post-policy (local-exec): ip-10-0-3-121.eu-west-1.compute.internal eu-west-1c

null_resource.post-policy (local-exec): kubectl annotate node ip-10-0-3-121.eu-west-1.compute.internal k8s.amazonaws.com/eniConfig=eu-west-1c-pod-netconfig

null_resource.post-policy (local-exec): error: --overwrite is false but found the following declared annotation(s): 'k8s.amazonaws.com/eniConfig' already has a value (eu-west-1c-pod-netconfig)

null_resource.post-policy (local-exec): done

null_resource.post-policy: Creation complete after 4s [id=7544399628196147368]

helm_release.aws-load-balancer-controller: Creating...

helm_release.aws-load-balancer-controller: Creation complete after 3s [id=aws-load-balancer-controller]


Apply complete! Resources: 5 added, 0 changed, 0 destroyed.


The state of your infrastructure has been saved to the path
below. This state is required to modify and destroy your
infrastructure, so keep it safe. To inspect the complete state
use the `terraform show` command.


State path: terraform.tfstate
```

The above has:

- Downloaded the policy definition file.
- Created a Load Balancer policy using the file.
- Started the post-policy.sh shell script which:
  - Downloads and creates the Custom Resource Definition extension.
- Installs the aws-load-balancer-controller helm chart.

Confirm the controller is operational with the command below and look for "Running" in the output:

```
1
kubectl get pods -A | grep aws-load-balancer-controller
kube-system   aws-load-balancer-controller-67bc87c6bf-fzd6x  1/1    Running  0       3m41s
```

you can also look at the helm output with:

```
1
helm ls -n kube-system
```