Annotate the worker nodes

We will use Terraform to start a scripts via a null_resource definition to configure our nodegroup to use the secondary CIDR for POD networking:

```
cd ~/environment/tfekscode/extra/eks-cidr2
```

```
terraform init
```

```
Initializing the backend...

Initializing provider plugins...
- terraform.io/builtin/terraform is built in to Terraform
- Finding hashicorp/external versions matching "~> 2.0"...
- Finding hashicorp/aws versions matching "~> 3.22"...
- Finding hashicorp/null versions matching "~> 3.0"...
- Installing hashicorp/external v2.0.0...
- Installed hashicorp/external v2.0.0 (signed by HashiCorp)
- Installing hashicorp/aws v3.23.0...
- Installed hashicorp/aws v3.23.0 (signed by HashiCorp)
- Installing hashicorp/null v3.0.0...
- Installed hashicorp/null v3.0.0 (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

Validate the Terraform code

```
1
terraform validate
Success! The configuration is valid.
```

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:

  # null_resource.cidr2 will be created
  + resource "null_resource" "cidr2" {
      + id       = (known after apply)
      + triggers = (known after apply)
    }

Plan: 1 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"
```

This step will take several minutes - the null_resource starts a script:

annotate-nodes.sh

This script generates and applies a per zone configuration for the ENIConfig Custom Resource Definition and then annotates the appropriate node in the nodegroup ng1-mycluster1.

Build the environment:

```
1
terraform apply tfplan
null_resource.cidr2: Creating...
null_resource.cidr2: Provisioning with 'local-exec'...
null_resource.cidr2 (local-exec): Executing: ["/bin/bash" "-c" "      sleep 5 #\u00a0let nodes settle\n      az1=$(echo eu-west-1a)\n      az2=$(echo eu-west-1b)\n      az3=$(echo eu-west-1c)\n      sub1=$(echo subnet-0c9a98e1eb7e55690)\n      sub2=$(echo subnet-08902f03323062448)\n      sub3=$(echo subnet-05af11eb3f3842939)\n      cn=$(echo mycluster1)\n      echo $az1 $az2 $az3 $sub1 $sub2 $sub3 $cn\n      #echo -e \"\\x1B[35mCycle nodes for custom CNI setting (takes a few minutes) ......\\x1B[0m\"\n      #./cni-cycle-nodes2.sh $cn\n      echo -e \"\\x1B[33mAnnotate nodes ......\\x1B[0m\"\n      ./annotate-nodes2.sh $az1 $az2 $az3 $sub1 $sub2 $sub3 $cn\n      echo -e \"\\x1B[32mShould see coredns on 100.64.x.y addresses ......\\x1B[0m\"\n      echo -e \"\\x1B[32mkubectl get pods -A -o wide | grep coredns\\x1B[0m\"\n"]
null_resource.cidr2 (local-exec): eu-west-1a eu-west-1b eu-west-1c subnet-0c9a98e1eb7e55690 subnet-08902f03323062448 subnet-05af11eb3f3842939 mycluster1
null_resource.cidr2 (local-exec): Annotate nodes ......
null_resource.cidr2 (local-exec): CLUSTER is mycluster1
null_resource.cidr2 (local-exec): NAME                                         CREATED AT
null_resource.cidr2 (local-exec): eniconfigs.crd.k8s.amazonaws.com             2023-04-16T13:59:40Z
null_resource.cidr2 (local-exec): ingressclassparams.elbv2.k8s.aws             2023-04-16T15:05:11Z
null_resource.cidr2 (local-exec): securitygrouppolicies.vpcresources.k8s.aws   2023-04-16T13:59:42Z
null_resource.cidr2 (local-exec): targetgroupbindings.elbv2.k8s.aws            2023-04-16T15:05:11Z
null_resource.cidr2 (local-exec): Descr EC2 instance i-0d56ae1b7c4cfe405 ...
null_resource.cidr2 (local-exec): subnet subnet-0c9a98e1eb7e55690 zone eu-west-1a
null_resource.cidr2 (local-exec): subnet subnet-08902f03323062448 zone eu-west-1b
null_resource.cidr2 (local-exec): subnet subnet-05af11eb3f3842939 zone eu-west-1c
null_resource.cidr2 (local-exec): eu-west-1a
null_resource.cidr2 (local-exec): created eu-west-1a-pod-netconfig2.yaml
null_resource.cidr2 (local-exec): apiVersion: crd.k8s.amazonaws.com/v1alpha1
null_resource.cidr2 (local-exec): kind: ENIConfig
null_resource.cidr2 (local-exec): metadata:
null_resource.cidr2 (local-exec):  name: eu-west-1a-pod-netconfig2
null_resource.cidr2 (local-exec): spec:
null_resource.cidr2 (local-exec):  subnet: subnet-0c9a98e1eb7e55690
null_resource.cidr2 (local-exec):  securityGroups:
null_resource.cidr2 (local-exec):  - sg-0cfcc654ed78031aa
null_resource.cidr2 (local-exec): eu-west-1b
null_resource.cidr2 (local-exec): created eu-west-1b-pod-netconfig2.yaml
null_resource.cidr2 (local-exec): eu-west-1c
null_resource.cidr2 (local-exec): created eu-west-1c-pod-netconfig2.yaml
null_resource.cidr2 (local-exec): apply the CRD eu-west-1a
null_resource.cidr2: Still creating... [10s elapsed]
null_resource.cidr2 (local-exec): eniconfig.crd.k8s.amazonaws.com/eu-west-1a-pod-netconfig2 created
null_resource.cidr2 (local-exec): apply the CRD eu-west-1b
null_resource.cidr2 (local-exec): eniconfig.crd.k8s.amazonaws.com/eu-west-1b-pod-netconfig2 created
null_resource.cidr2 (local-exec): apply the CRD eu-west-1c
null_resource.cidr2 (local-exec): eniconfig.crd.k8s.amazonaws.com/eu-west-1c-pod-netconfig2 created
null_resource.cidr2 (local-exec): pause 10s before annotate
null_resource.cidr2: Still creating... [20s elapsed]
null_resource.cidr2 (local-exec): Found 2 nodes to annotate of 4
null_resource.cidr2 (local-exec): ip-10-0-1-248.eu-west-1.compute.internal eu-west-1a
null_resource.cidr2 (local-exec): kubectl annotate node ip-10-0-1-248.eu-west-1.compute.internal
k8s.amazonaws.com/eniConfig=eu-west-1a-pod-netconfig2
null_resource.cidr2 (local-exec): node/ip-10-0-1-248.eu-west-1.compute.internal annotated
null_resource.cidr2 (local-exec): ip-10-0-2-235.eu-west-1.compute.internal eu-west-1b
null_resource.cidr2 (local-exec): kubectl annotate node ip-10-0-2-235.eu-west-1.compute.internal
k8s.amazonaws.com/eniConfig=eu-west-1b-pod-netconfig2
null_resource.cidr2 (local-exec): node/ip-10-0-2-235.eu-west-1.compute.internal annotated
null_resource.cidr2 (local-exec): Background reannotate
null_resource.cidr2 (local-exec): Should see coredns on 100.64.x.y addresses ......
null_resource.cidr2 (local-exec): kubectl get pods -A -o wide | grep coredns
null_resource.cidr2: Still creating... [30s elapsed]
....
null_resource.cidr2: Still creating... [1m30s elapsed]
null_resource.cidr2: Creation complete after 1m33s [id=2148894349933258708]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Note: there is no need to cycle the nodes (terminate them) in this case as when the node group was created the CNI extension was already in place - so we only need to annotate the nodes.