**Configure ENICONFIG to assign secondary CIDR to pods**

# **Enable the VPC CNI**

* Validate the Latest Plugins must be installed

# kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/" -f 2

amazon-k8s-cni-init:v1.18.5-eksbuild.1

amazon-k8s-cni:v1.18.5-eksbuild.1

amazon

* Set the AWS\_VPC\_K8S\_CNI\_CUSTOM\_NETWORK\_CFG environment variable to true in the aws-node[.noloc]`DaemonSet.

# kubectl set env daemonset aws-node -n kube-system AWS\_VPC\_K8S\_CNI\_CUSTOM\_NETWORK\_CFG=true

* Set the AWS\_VPC\_K8S\_CNI\_EXTERNALSNAT environment variable to false to allow outside traffic

# kubectl set env daemonset aws-node -n kube-system AWS\_VPC\_K8S\_CNI\_EXTERNALSNAT=false

* Validation

# kubectl describe daemonset aws-node --namespace kube-system |grep AWS\_VPC\_K8S

AWS\_VPC\_K8S\_CNI\_CUSTOM\_NETWORK\_CFG: true

AWS\_VPC\_K8S\_CNI\_EXTERNALSNAT: false

AWS\_VPC\_K8S\_CNI\_LOGLEVEL: DEBUG

AWS\_VPC\_K8S\_CNI\_LOG\_FILE: /host/var/log/aws-routed-eni/ipamd.log

AWS\_VPC\_K8S\_CNI\_RANDOMIZESNAT: prng

AWS\_VPC\_K8S\_CNI\_VETHPREFIX: eni

AWS\_VPC\_K8S\_PLUGIN\_LOG\_FILE: /var/log/aws-routed-eni/plugin.log

AWS\_VPC\_K8S\_PLUGIN\_LOG\_LEVEL: DEBUG

* Set the ENI\_CONFIG\_LABEL\_DEF environment variable to topology.kubernetes.io/zone to allow outside traffic

# kubectl set env daemonset aws-node -n kube-system ENI\_CONFIG\_LABEL\_DEF=topology.kubernetes.io/zone

* Validation

# kubectl describe daemonset aws-node --namespace kube-system |grep ENI\_CONFIG

ENI\_CONFIG\_LABEL\_DEF: topology.kubernetes.io/zone

# **Configure the ENI\_Config**

* Create the yaml configuration file as below

# vi eniconfig.yaml

apiVersion: crd.k8s.amazonaws.com/v1alpha1

kind: ENIConfig

metadata:

name: ap-south-1a

spec:

securityGroups:

- sg-0c4873f15f430d033

subnet: subnet-01199e4502a7cca61

---

apiVersion: crd.k8s.amazonaws.com/v1alpha1

kind: ENIConfig

metadata:

name: ap-south-1b

spec:

securityGroups:

- sg-0c4873f15f430d033

subnet: subnet-067ef3eb4bdc641a6

---

apiVersion: crd.k8s.amazonaws.com/v1alpha1

kind: ENIConfig

metadata:

name: ap-south-1c

spec:

securityGroups:

- sg-0c4873f15f430d033

subnet: subnet-03dfed99a32088988

* Apply the configuration

# kubectl apply -f eniconfig.yaml

eniconfig.crd.k8s.amazonaws.com/ap-south-1a created

eniconfig.crd.k8s.amazonaws.com/ap-south-1b created

eniconfig.crd.k8s.amazonaws.com/ap-south-1c created

* Validate the configuration

# kubectl get eniconfig

NAME AGE

ap-south-1a 81s

ap-south-1b 81s

ap-south-1c 81s

* Finally, we need to annotate nodes with custom network configs

# kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-10-199-106-102.ap-south-1.compute.internal Ready <none> 2d23h v1.30.2-eks-1552ad0

ip-10-199-106-156.ap-south-1.compute.internal Ready <none> 2d23h v1.30.2-eks-1552ad0

ip-10-199-107-120.ap-south-1.compute.internal Ready <none> 3d15h v1.30.2-eks-1552ad0

# kubectl annotate node ip-10-199-106-102.ap-south-1.compute.internal k8s.amazonaws.com/eniConfig=ap-south-1a

# kubectl annotate node ip-10-199-106-156.ap-south-1.compute.internal k8s.amazonaws.com/eniConfig=ap-south-1b

# kubectl annotate node ip-10-199-107-120.ap-south-1.compute.internal k8s.amazonaws.com/eniConfig=ap-south-1c

* Validation

# kubectl describe nodes | grep 'failure-domain.beta.kubernetes.io/zone'

failure-domain.beta.kubernetes.io/zone=ap-south-1a

failure-domain.beta.kubernetes.io/zone=ap-south-1b

failure-domain.beta.kubernetes.io/zone=ap-south-1c

* enable Kubernetes to automatically apply the corresponding ENIConfig for the node’s Availability Zone

# kubectl set env daemonset aws-node -n kube-system ENI\_CONFIG\_LABEL\_DEF=failure-domain.beta.kubernetes.io/zone

daemonset.apps/aws-node env updated

AND

# kubectl set env daemonset aws-node -n kube-system ENI\_CONFIG\_LABEL\_DEF=topology.kubernetes.io/zone

**Note** : Kubernetes will now automatically apply the corresponding ENIConfig matching the nodes AZ, and no need to manually annotate the new EC2 instance with ENIConfig

* Validation

# kubectl describe daemonset aws-node --namespace kube-system |grep aws-node

* Terminate the EKS-node from console and rollout and restart with below command.

# kubectl rollout restart ds aws-node -n kube-system

daemonset.apps/aws-node restarted