**Attach eni to instance**

Before we are able to perform this we need to create the management and backup ip. The rationale behind the management ip and attaching it to the instance is to allow accessing the server from **cyberark**. Next, we need to attach the backup ip to the instance which act as a **BAAS** which we will be accessing via **ITSM.**

**Creation of backup and management eni.**

Before we are able to create a eni, we need to create the security group.

**backup eni security group**

terraform {

    source = "git::git@sgts.gitlab-dedicated.com:wog/mha/ica-e-services/ica\_common\_services/app/gccplus\_modules.git//tg-modules//secgrp"

}

include "root" {

    path = find\_in\_parent\_folders()

}

locals {

    zone\_vars        = read\_terragrunt\_config(find\_in\_parent\_folders("zone.hcl"))

    regzone\_vars     = read\_terragrunt\_config(find\_in\_parent\_folders("regzone.hcl"))

    dlzzone\_vars     = read\_terragrunt\_config(find\_in\_parent\_folders("dlzzone.hcl"))

    proj\_vars        = read\_terragrunt\_config(find\_in\_parent\_folders("common.hcl"))

    app\_vars         = read\_terragrunt\_config(find\_in\_parent\_folders("app.hcl"))

    proj\_code        = local.proj\_vars.locals.proj\_code

    app\_name         = local.app\_vars.locals.app\_name

    vpc\_id           = local.zone\_vars.locals.vpc\_id

    s3\_prefix\_list   = local.zone\_vars.locals.s3\_prefix\_list

    cidr\_block       = local.regzone\_vars.locals.cidr\_block

    vpc\_subnets\_cidr = local.dlzzone\_vars.locals.vpc\_subnets\_cidr

    vpc\_subnets\_cidr\_reg = local.regzone\_vars.locals.vpc\_subnets\_cidr

    patchrepo        = local.zone\_vars.locals.patchrepo

    bckp\_mediasvr    = local.zone\_vars.locals.bckp\_mediasvr

    bckp\_mstrsvr     = local.zone\_vars.locals.bckp\_mstrsvr

    eps\_cidr         = local.zone\_vars.locals.eps\_cidr

    nessus\_cidr      = local.zone\_vars.locals.nessus\_cidr

    custom\_name         = local.app\_vars.locals.custom\_name

    inbound\_rules = concat(

      [for cidr in local.patchrepo : [cidr, "443", "443", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.patchrepo : [cidr, "4120", "4120", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.eps\_cidr : [cidr, "443", "443", "tcp", "Traffic from eps "]],

      [for cidr in local.eps\_cidr : [cidr, "4118", "4122", "tcp", "Traffic from eps"]],

    )

    outbound\_rules = concat(

      [for cidr in local.patchrepo : [cidr, "443", "443", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.patchrepo : [cidr, "4120", "4120", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.eps\_cidr : [cidr, "443", "443", "tcp", "Traffic from eps "]],

      [for cidr in local.eps\_cidr : [cidr, "4118", "4122", "tcp", "Traffic from eps"]],

      [[local.nessus\_cidr, "8834", "8834", "tcp", "nessus"]]

    )

}

inputs = {

  vpc\_id = local.vpc\_id

  app\_name = "${local.app\_name}-${local.custom\_name}"

  sg\_description = "Security group for ${local.app\_name}"

  inbound\_rules  = zipmap(range(length(local.inbound\_rules)), local.inbound\_rules)

  outbound\_rules = zipmap(range(length(local.outbound\_rules)), local.outbound\_rules)

  pf\_inbound\_rules = {

        "0" = [local.s3\_prefix\_list, "443", "443", "TCP"]

    }

  pf\_outbound\_rules = {

        "0" = [local.s3\_prefix\_list, "443", "443", "TCP"]

    }

}

backup eni

terraform {

    source = "git::git@sgts.gitlab-dedicated.com:wog/mha/ica-e-services/ica\_common\_services/app/gccplus\_modules.git//tg-modules//eni"

}

include "root" {

    path = find\_in\_parent\_folders()

}

dependency secgrp {

    config\_path = "../secgrp"

}

locals{

    zone\_vars        = read\_terragrunt\_config(find\_in\_parent\_folders("zone.hcl"))

    bckp\_subnet\_cidr = local.zone\_vars.locals.bckp\_subnet\_cidr

    bckp\_subnet\_id   = local.zone\_vars.locals.bckp\_subnet\_id

}

inputs = {

    eni = {

        0 = {

            subnet\_id       = local.bckp\_subnet\_id[0]

            private\_ips     = [cidrhost(local.bckp\_subnet\_cidr[0],13)],

            security\_groups = [dependency.secgrp.outputs.sg\_id],

        }

        1 = {

            subnet\_id       = local.bckp\_subnet\_id[1]

            private\_ips     = [cidrhost(local.bckp\_subnet\_cidr[1],13)]

            security\_groups = [dependency.secgrp.outputs.sg\_id]

        }

    }

}

# management eni security group

terraform {

    source = "git::git@sgts.gitlab-dedicated.com:wog/mha/ica-e-services/ica\_common\_services/app/gccplus\_modules.git//tg-modules//secgrp"

}

include "root" {

    path = find\_in\_parent\_folders()

}

locals {

    zone\_vars        = read\_terragrunt\_config(find\_in\_parent\_folders("zone.hcl"))

    regzone\_vars     = read\_terragrunt\_config(find\_in\_parent\_folders("regzone.hcl"))

    dlzzone\_vars     = read\_terragrunt\_config(find\_in\_parent\_folders("dlzzone.hcl"))

    proj\_vars        = read\_terragrunt\_config(find\_in\_parent\_folders("common.hcl"))

    app\_vars         = read\_terragrunt\_config(find\_in\_parent\_folders("app.hcl"))

    proj\_code        = local.proj\_vars.locals.proj\_code

    app\_name         = local.app\_vars.locals.app\_name

    vpc\_id           = local.zone\_vars.locals.vpc\_id

    s3\_prefix\_list   = local.zone\_vars.locals.s3\_prefix\_list

    cidr\_block       = local.regzone\_vars.locals.cidr\_block

    vpc\_subnets\_cidr = local.dlzzone\_vars.locals.vpc\_subnets\_cidr

    vpc\_subnets\_cidr\_reg = local.regzone\_vars.locals.vpc\_subnets\_cidr

    patchrepo        = local.zone\_vars.locals.patchrepo

    bckp\_mediasvr    = local.zone\_vars.locals.bckp\_mediasvr

    bckp\_mstrsvr     = local.zone\_vars.locals.bckp\_mstrsvr

    eps\_cidr         = local.zone\_vars.locals.eps\_cidr

    nessus\_cidr      = local.zone\_vars.locals.nessus\_cidr

    custom\_name      = local.app\_vars.locals.custom\_name

    gdc\_cidr         = local.zone\_vars.locals.gdc\_cidr

    inbound\_rules = concat(

      [for cidr in local.patchrepo : [cidr, "443", "443", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.patchrepo : [cidr, "4120", "4120", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.eps\_cidr : [cidr, "443", "443", "tcp", "Traffic from eps "]],

      [for cidr in local.eps\_cidr : [cidr, "4118", "4122", "tcp", "Traffic from eps"]],

      [for cidr in local.gdc\_cidr : [cidr, "22", "22", "tcp", "Traffic from bedok"]],

      [for cidr in local.gdc\_cidr : [cidr, "135", "135", "tcp", "Traffic from bedok"]],

      [for cidr in local.gdc\_cidr : [cidr, "139", "139", "tcp", "Traffic from bedok"]],

      [for cidr in local.gdc\_cidr : [cidr, "443", "443", "tcp", "Traffic from bedok"]],

      [for cidr in local.gdc\_cidr : [cidr, "445", "445", "tcp", "Traffic from bedok"]],

      [for cidr in local.gdc\_cidr : [cidr, "3389", "3389", "tcp", "Traffic from bedok"]],

    )

    outbound\_rules = concat(

      [for cidr in local.patchrepo : [cidr, "443", "443", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.patchrepo : [cidr, "4120", "4120", "tcp", "Traffic from BaaS Patch Repo"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mediasvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "1556", "1556", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13724", "13724", "tcp", "Traffic from BaaS"]],

      [for cidr in local.bckp\_mstrsvr : [cidr, "13782", "13782", "tcp", "Traffic from BaaS"]],

      [for cidr in local.eps\_cidr : [cidr, "443", "443", "tcp", "Traffic from eps "]],

      [for cidr in local.eps\_cidr : [cidr, "4118", "4122", "tcp", "Traffic from eps"]],

      [[local.nessus\_cidr, "8834", "8834", "tcp", "nessus"]]

    )

}

inputs = {

  vpc\_id = local.vpc\_id

  app\_name = "${local.app\_name}-${local.custom\_name}"

  sg\_description = "Security group for ${local.app\_name}"

  inbound\_rules  = zipmap(range(length(local.inbound\_rules)), local.inbound\_rules)

  outbound\_rules = zipmap(range(length(local.outbound\_rules)), local.outbound\_rules)

  pf\_inbound\_rules = {

        "0" = [local.s3\_prefix\_list, "443", "443", "TCP"]

    }

  pf\_outbound\_rules = {

        "0" = [local.s3\_prefix\_list, "443", "443", "TCP"]

    }

}

management eni

terraform {

    source = "git::git@sgts.gitlab-dedicated.com:wog/mha/ica-e-services/ica\_common\_services/app/gccplus\_modules.git//tg-modules//eni"

}

include "root" {

    path = find\_in\_parent\_folders()

}

dependency secgrp {

    config\_path = "../secgrp"

}

locals{

    zone\_vars         = read\_terragrunt\_config(find\_in\_parent\_folders("zone.hcl"))

    mgmt\_subnet\_cidr  = local.zone\_vars.locals.mgmt\_subnet\_cidr

    mgmt\_subnet\_id    = local.zone\_vars.locals.mgmt\_subnet\_id

}

inputs = {

    eni = {

        0 = {

            subnet\_id       = local.mgmt\_subnet\_id[0]

            private\_ips     = [cidrhost(local.mgmt\_subnet\_cidr[0],13)],

            security\_groups = [dependency.secgrp.outputs.sg\_id],

        }

        1 = {

            subnet\_id       = local.mgmt\_subnet\_id[1]

            private\_ips     = [cidrhost(local.mgmt\_subnet\_cidr[1],13)]

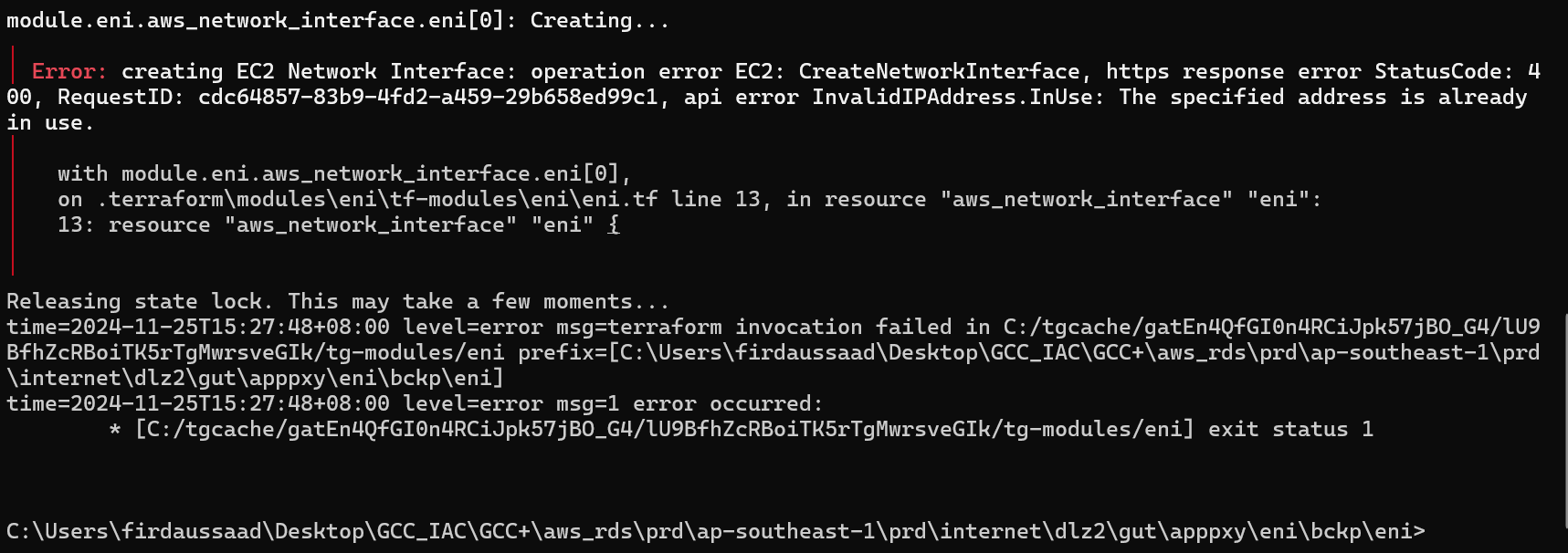
            security\_groups = [dependency.secgrp.outputs.sg\_id]

        }

    }

}

Once we have created the backup and management eni, we will attached it via script.



If we got this error, this shows the eni has been taken and created. To resolved this we just need to use a higher number.

eni\_mgmt=("<eni\_mgmt\_ip")

                       eni\_bckp=("eni\_bckp\_ip")

                       TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"`

                       INSTANCE\_ID=$(curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-data/instance-id)

                       EC2\_AVAIL\_ZONE=$(curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-data/placement/availability-zone |  awk '{print $1}')

                       echo $INSTANCE\_ID

                       echo $EC2\_AVAIL\_ZONE

                       aws ec2 attach-network-interface --region ap-southeast-1 --network-interface-id $eni\_mgmt  --instance-id $INSTANCE\_ID --device-index 1

                       aws ec2 attach-network-interface --region ap-southeast-1 --network-interface-id $eni\_bckp  --instance-id $INSTANCE\_ID --device-index 2

This is the [eni.sh](http://eni.sh) script. First we need to change eni\_mgmt\_ip and eni\_bckp\_ip. Next we shall also add the [eni.sh](http://eni.sh) to the userdata script.

Once we have made changes to [eni.sh](http://eni.sh), lets check the userdata script if it contains this set of commands as we aim to automate the process.

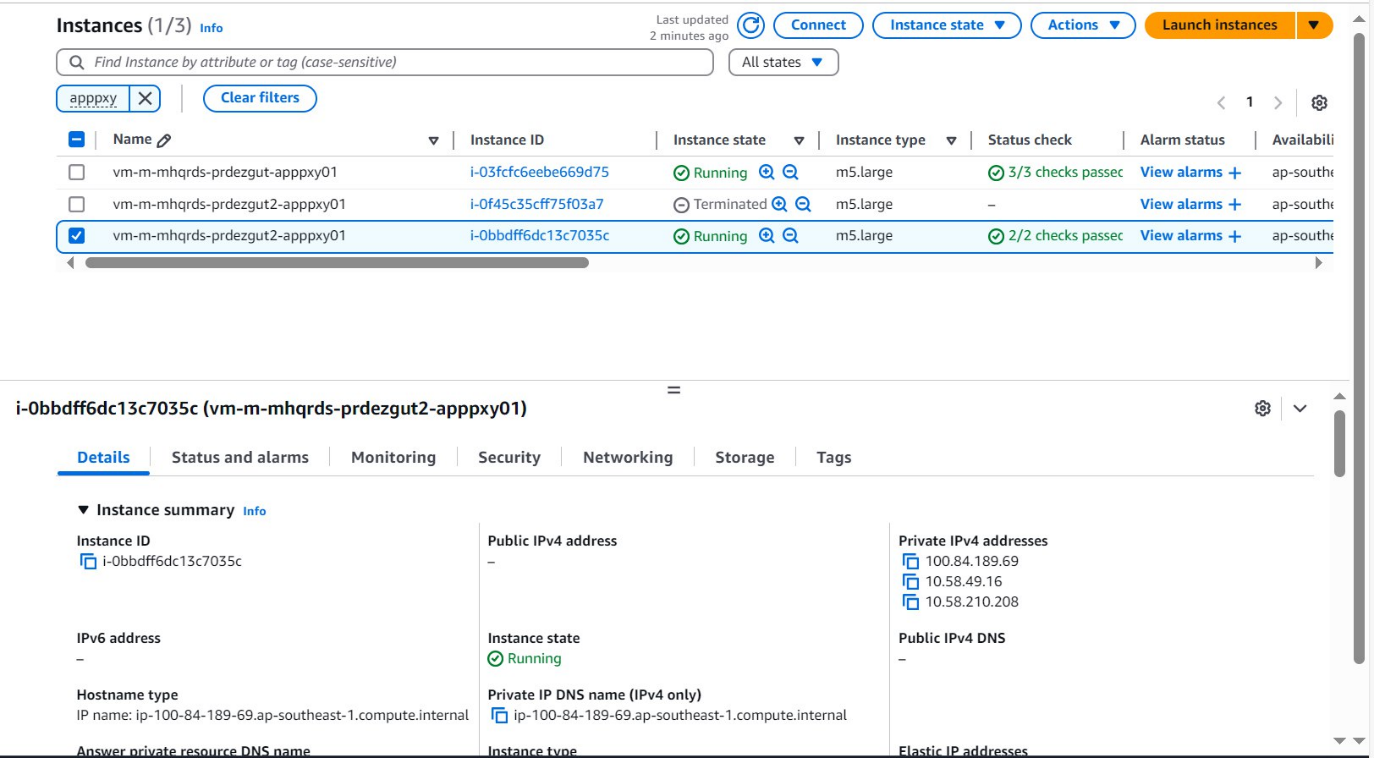
# Attach Backup & Mgmt IP

aws s3 --region ap-southeast-1 cp s3://${s3}/eni.sh /home/ec2-user/eni.sh

sudo dos2unix /home/ec2-user/eni01.sh

sudo chmod 755 /home/ec2-user/eni01.sh

/home/ec2-user/eni01.sh

Once we have done, lets perform a instance refresh.

Ensure that we have open the ports we are required. To know we can run this command in the instance we want to attached the eni. This shows we are about to connect the ec2 endpoint.

curl -v https://ec2.ap-southeast-1.amazonaws.com

