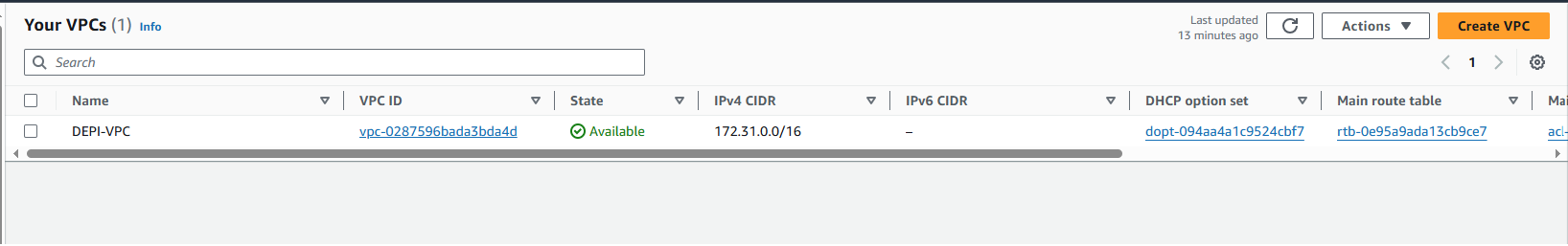
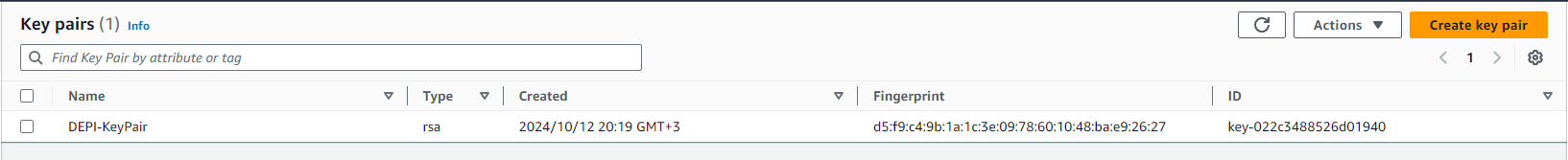
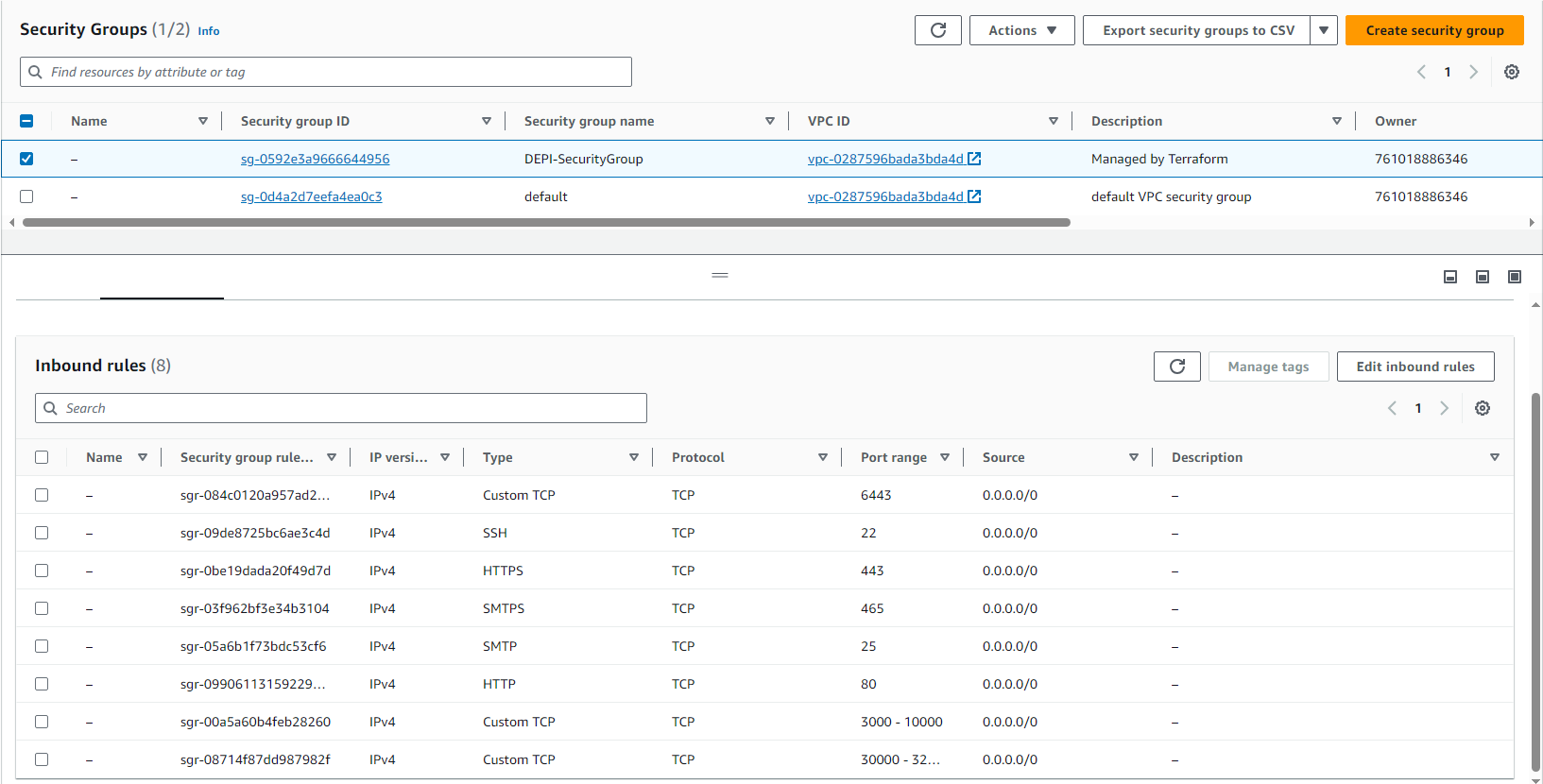
### 1. Use the default VPC



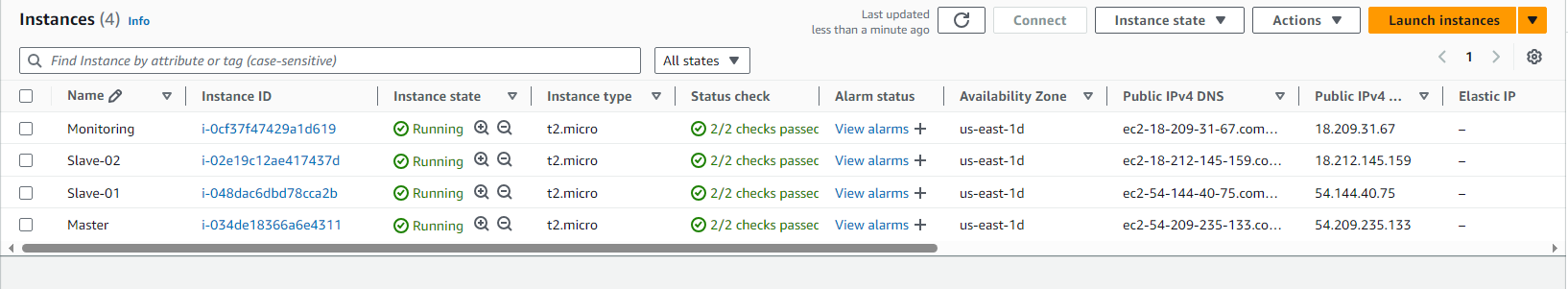
### 2. Key Pair



### 3. Security Group

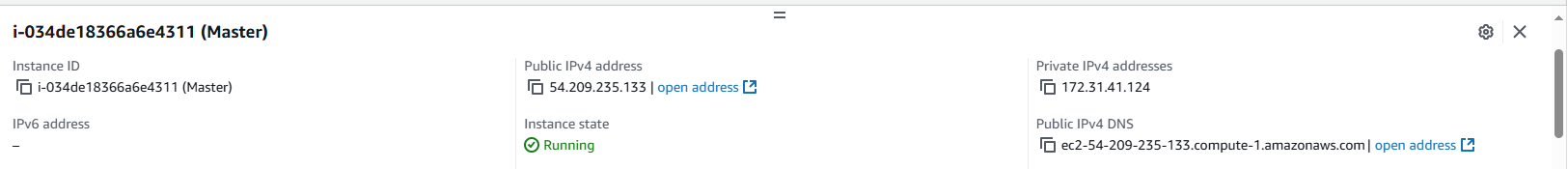


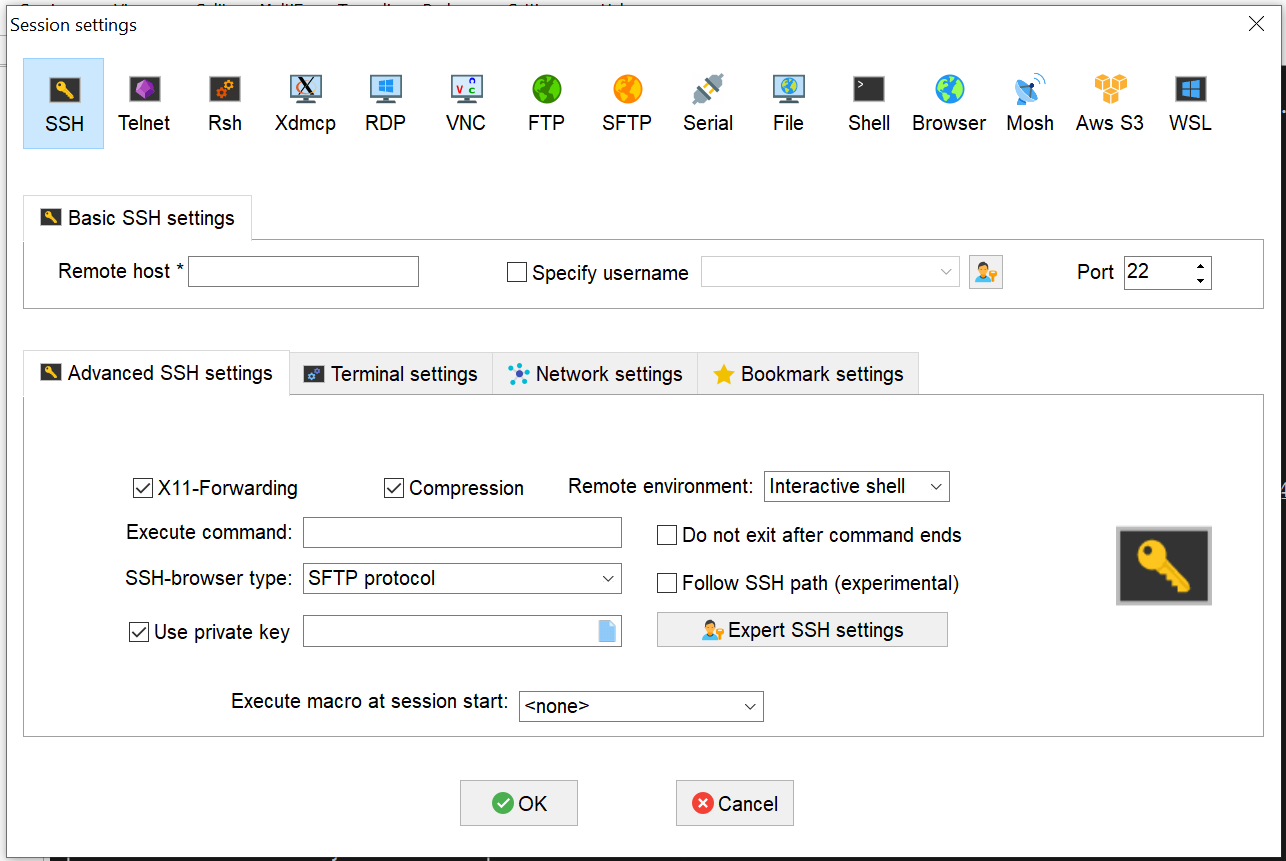
### 4. EC2 instances created using Terraform



## Setup the Kubernetes Cluster

### 1. Access the instances using MobaXterm application.

1. Create a new session.
2. Get the public IP address for each instance from AWS.
   1. 
3. Copy the public IP address for each instance to the **Remote host**.
4. Check the **Specify username** box and enter “ubuntu” as the username.
5. In the **Advanced SSH settings,** check the **Use private key** box and place the .pem file.
6. Duplicate the session to create the 2 worker nodes and the Monitoring sessions as well by replacing the Remote host with each IP address.



### 2. Setup the Master and Worker Nodes

    1. Run the below command to change to root [On Master & Worker Node]

        • sudo su

    2. Create an executable file and place the following commands then run the script [On Master & Worker Node]

        #   Update System Packages

            •   sudo apt-get update

        #   Install Docker

            •   sudo apt install docker.io -y

            •   sudo chmod 666 /var/run/docker.sock

        #   Install Required Dependencies for Kubernetes

            •   sudo apt-get install -y apt-transport-https ca-certificates curl gnupg

            •   sudo mkdir -p -m 755 /etc/apt/keyrings

        #   Add Kubernetes Repository and GPG Key

            •   Curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg -- dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

            •   echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

        #   Update Package List

            •   sudo apt update

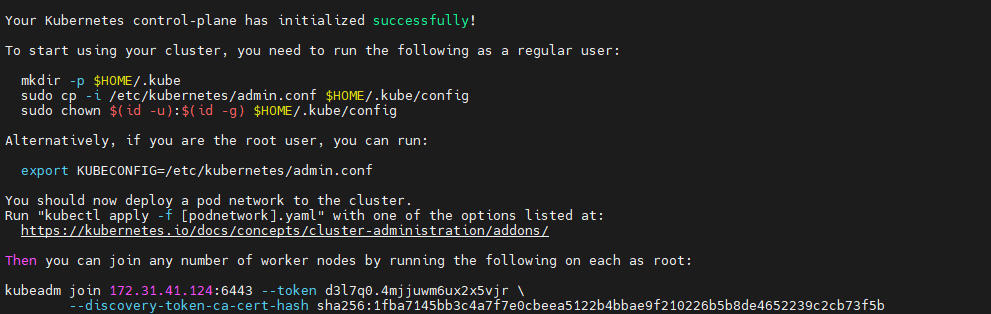
        #   Install Kubernetes Components

            •   sudo apt install -y kubeadm=1.28.1-1.1 kubelet=1.28.1-1.1 kubectl=1.28.1-1.1

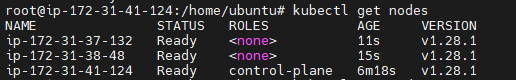
    3. Run the following commands on the Master node only

        #   Initialize Kubernetes Master Node

•   sudo kubeadm init--pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all



        #   After running the above command then our vm will acts as master node and it will generate token to connect this with slave node-copy the token and run the command in slave machines 1 & 2



        #   Configure Kubernetes Cluster

            •   mkdir -p $HOME/.kube

            •   sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

            •   sudo chown $(id -u):$(id -g) $HOME/.kube/config

        #   Deploy Networking Solution (Calico)

            •   kubectl apply -f https://docs.projectcalico.org/v3.20/manifests/calico.yaml

        #   Deploy Ingress Controller (NGINX)

            •   kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v0.49.0/deploy/static/provider/baremetal/deploy.yaml

    4.  We'll Scan Kubernetes Cluster For Any Kind Of Issues Using Cube Audit

        # Go To The Website & Copy The Linux\_amd\_64 Link

            •   https://github.com/shopify/kubeaudit/releases

        # Paste It Using wget Command

        # Now Untar The File Using tar-xvf File Name

        # sudo mv kubeaudit /usr/local/bin/->kubeaudit all

