RKE and kOps

Tools to speed-up the deployment of K8s clusters

RKE - Rancher Kubernetes Engine

- Open-Source
- Command-line executable
- Rancher
  - Kubernetes-as-a-Service products
- Features:
  - Declarative deployment
  - Local deployment
  - Multi-cloud deployment
  - Hybrid deployment (cloud + bare metal machines)
  - In-built services:
    - Nginx Ingress
    - Metrics-server
    - etc.

Documentation: https://rancher.com/docs/rke/latest/en/

RKE - Requirements

- The admin nodes must have “kubectl” installed
- Every node must have:
  - Dedicated User
    - Password-less SSH login
    - Member of the “docker” group
  - Docker
  - OpenSSH 7.0 or higher
  - 1+ GB RAM
  - 1+ CPU cores
  - Custom port-forwarding rules as discussed here: https://rancher.com/docs/rke/latest/en/os/#ports
- Every node inside the cluster must be reachable by all the other nodes

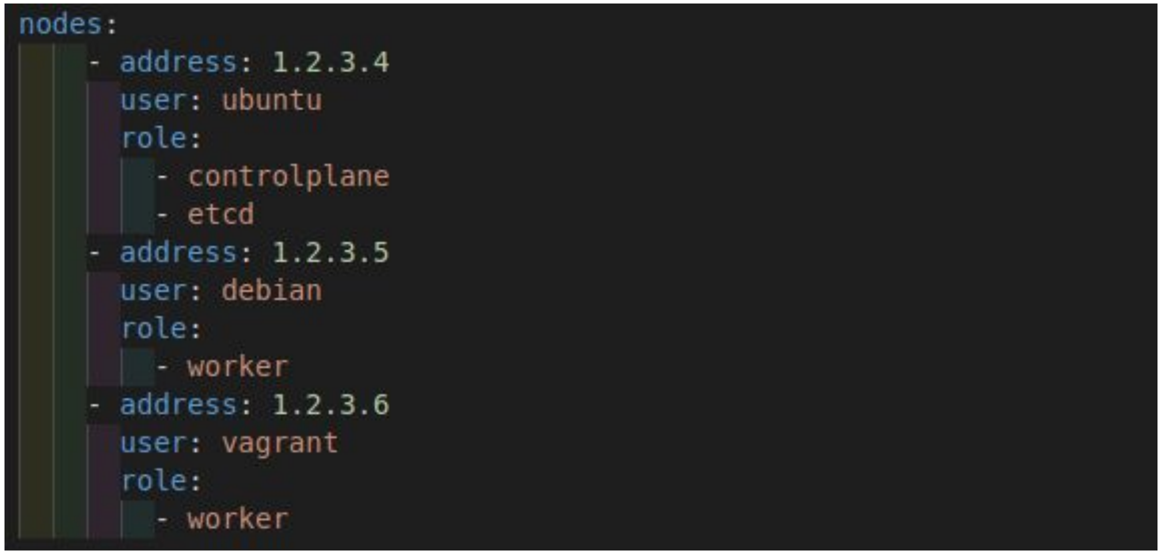
More at: https://rancher.com/docs/rke/latest/en/os/

RKE - Installation

- Explained here: https://rancher.com/docs/rke/latest/en/installation/
- Steps:
    - Download the latest RKE Release for your OS here: https://github.com/rancher/rke/#latest-release
    - Move the binary inside your $PATH and rename it “rke” (or “rke.exe” for Windows)
    - Make the binary executable: chmod +x rke
    - Test if it works: rke --version

RKE - Cluster.yml file

- Configuration file with nodes roles and information
    - Roles (master/controlplane, etcd, worker)
    - SSH information
    - more...
- Launched and maintained from the admin nodes
- Example:



```yaml
nodes:
    - address: 1.2.3.4
      user: ubuntu
      role:
        - controlplane
        - etcd
    - address: 1.2.3.5
      user: debian
      role:
        - worker
    - address: 1.2.3.6
      user: vagrant
      role:
        - worker
```

RKE - First deployment

- Generate a Cluster.yml file:
  - rke config --name cluster.yml
- Create the cluster:
  - rke up
- Copy "kube_config_cluster.yml" inside .kube folder (generally in your $HOME) and rename it as "config"
- Notes:
  - Make a copy of the following files:
    - cluster.yml
    - kube_config_cluster.yml
    - "cluster.rkestate
  - If you want to update the cluster (f.i. add a worker node):
    - Update the "cluster.yml" file with the newest configurations
    - Run: rke up --update-only
  - Delete a cluster:
    - rke remove

kOps - Kubernets Operations

- Open-Source
- Command-line executable
- Currently (as of 05/2022) is more AWS oriented
  - Digital Ocean/OpenStack in beta support
  - Azure/GCE in alpha support
- Features:
  - Declarative deployment
  - Highly customizable
  - No real hard requirements for the admin nodes
  - No real hard requirements for the machines needed to build the cluster
  - Highly compatible with AWS

Website: https://kops.sigs.k8s.io/

kOps - Installation

- Requirements for admin machines:
  - kubectl installed
- Linux/macOS:
  - curl -Lo kops https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag_name | cut -d '"' -f 4)/kops-linux-amd64
  - chmod +x kops
  - sudo mv kops /usr/local/bin/kops
- Windows:
  - Get kops-windows-amd64 from their releases: https://github.com/kubernetes/kops/releases/latest
  - Rename kops-windows-amd64 to kops.exe and store it in a preferred path
  - Make sure the path you chose is added to your Path environment variable
- More about the installation here https://kops.sigs.k8s.io/getting_started/install/

kOps - First Deployment AWS (1)

- Create a IAM user with the following policies
  https://kops.sigs.k8s.io/getting_started/aws/#setup-iam-user
- Create a S3 bucket to store the State/Versioning files regarding your cluster
  - aws s3api create-bucket \
     --bucket **rke-test-s3** \
     --region **eu-central-1**
  - aws s3api put-bucket-versioning --bucket **rke-test-s3** --versioning-configuration Status=Enabled
  - aws s3api put-bucket-encryption --bucket **rke-test-s3** --server-side-encryption-configuration '{"Rules":[{"ApplyServerSideEncryptionByDefault":{"SSEAlgorithm":"AES256"}}]}'

kOps - First Deployment (2)

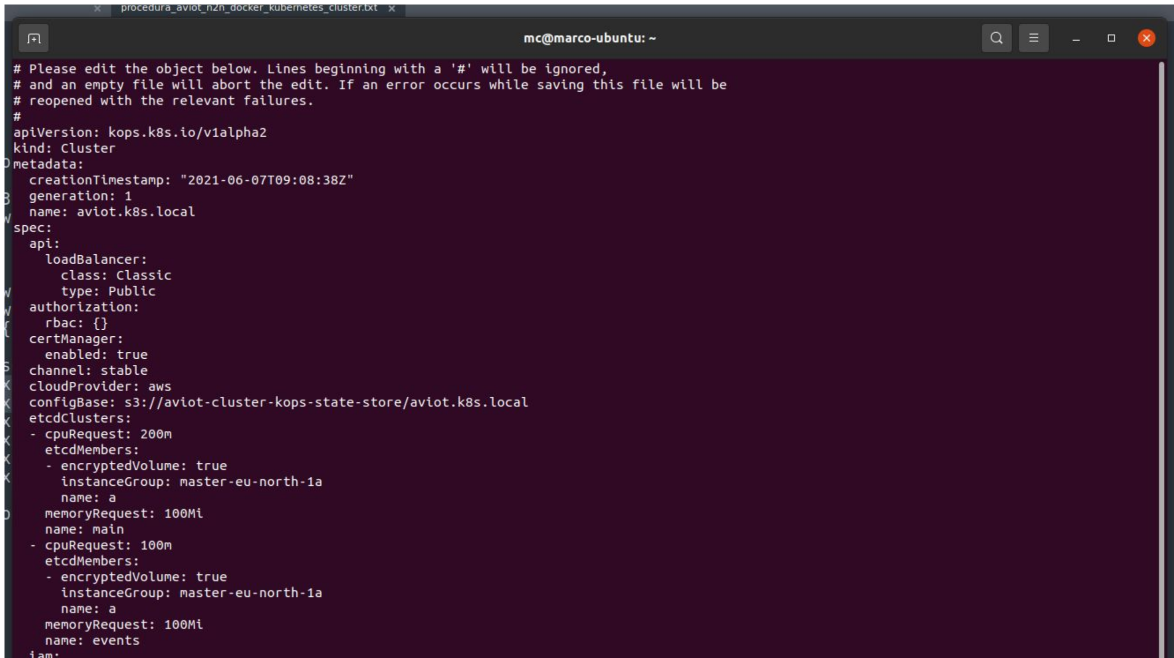
- Create the cluster:

    -
      ```
      export CLUSTER_NAME=rke-test.k8s.local
      export KOPS_STATE_STORE=s3://rke-test-s3
      export MASTER_SIZE="t3a.small"
      export NODE_SIZE="t3a.small"
      export ZONES="eu-central-1a,eu-central-1b,eu-central-1c"
      export MASTER_ZONES="eu-central-1a,eu-central-1b,eu-central-1c"
      export SSH_KEY="~/.ssh/id_rsa.pub"
      ```

    -
      ```
      kops create cluster \
        --state=$KOPS_STATE_STORE \
        --zones=$ZONES \
        --node-size=$NODE_SIZE \
        --node-count=2 \
        --node-volume-size=128 \ #GB
        --master-size=$MASTER_SIZE \
        --master-count=1 \
        --master-volume-size=128 \ #GB
        --ssh-public-key $SSH_KEY \
        ${CLUSTER_NAME};
      ```

kOps - First Deployment (3) - Optional

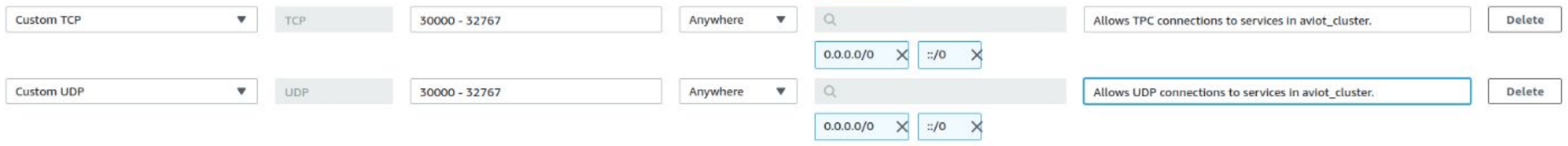
- kOps Addons
  - Metrics-server, Cert-Manager and more here: https://kops.sigs.k8s.io/addons/
- export EDITOR=nano
  kops edit cluster ${CLUSTER_NAME}



```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: kops.k8s.io/v1alpha2
kind: Cluster
metadata:
  creationTimestamp: "2021-06-07T09:08:38Z"
  generation: 1
  name: aviot.k8s.local
spec:
  api:
    loadBalancer:
      class: Classic
      type: Public
  authorization:
    rbac: {}
  certManager:
    enabled: true
  channel: stable
  cloudProvider: aws
  configBase: s3://aviot-cluster-kops-state-store/aviot.k8s.local
  etcdClusters:
  - cpuRequest: 200m
    etcdMembers:
    - encryptedVolume: true
      instanceGroup: master-eu-north-1a
      name: a
    memoryRequest: 100Mi
    name: main
  - cpuRequest: 100m
    etcdMembers:
    - encryptedVolume: true
      instanceGroup: master-eu-north-1a
      name: a
    memoryRequest: 100Mi
    name: events
  iam:
```

kOps - First Deployment (4)

- Confirm the addons/creation commands:
  kops update cluster --name ${CLUSTER_NAME} --yes
- Export a config (be careful: OVERWRITE IF EXISTS) file with 1 year expiration time (maximum value)
  - kops export kubecfg --name ${CLUSTER_NAME} --admin=8670h0m0s
- Start validation:
  - kops validate cluster --wait 10m
- Security group port-forwarding for NodePorts:
  - Log-in in AWS web console
  - Reach the EC2 service in the AWS Region where you've deployed your cluster
  - Reach the security group of one of the worker nodes (they share the same one)
  - Add an in-bound rule as the following (Custom TCP and UDP, range 30000-32767, Anywhere):



Custom TCP	TCP	30000 - 32767	Anywhere	0.0.0.0/0	::/0	Allows TPC connections to services in aviot_cluster.	Delete
Custom UDP	UDP	30000 - 32767	Anywhere	0.0.0.0/0	::/0	Allows UDP connections to services in aviot_cluster.	Delete

  - Click on Save rules

kOps - First Deployment (5)

- Update the Cluster configuration:

  -
    ```
    export EDITOR=nano
    export CLUSTER_NAME=rke-test.k8s.local
    export KOPS_STATE_STORE=s3://rke-test-s3
    kops edit cluster
    ```
- Destroy cluster:

  -
    ```
    export CLUSTER_NAME=rke-test
    export KOPS_STATE_STORE=s3://rke-test-s3
    kops delete cluster --name ${CLUSTER_NAME} --yes
    ```