Copy of Kubernetes-Dashboard with Istio-Gateway-&-Virtual-Service

NOTE: To setup the k8s-dashboard with Istio is complicated and official given method `PASSTHROUGH` somehow not working. In this Doc Istio Installation is not covered.

To make it work we have done some changes on kubernetes.yaml file which we have got from official link.

https://kubernetes.io/docs/tasks/access-application-cluster/web-ui-dashboard/

STEP-1:

First wget or copy the latest dashboard yaml on local editor and do changes as below or directly use below file.

```
https://raw.githubusercontent.com/kubernetes/dashboard/v2.5.0/aio/deploy
/recommended.yaml
```

REQUIRED-CHANGES

We need to update the Service and the Deployment configuration on k8s-dashboard.yaml. Starting with the service, we just need to change the ports:

```
kind: Service
apiVersion: v1
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kube-system
spec:
  ports:
    - port: 80
      targetPort: 9090
  selector:
    k8s-app: kubernetes-dashboard
```

We then need to update the container configuration in the deployment manifest:

```
containers:
  - name: kubernetes-dashboard
    image: k8s.gcr.io/kubernetes-dashboard-amd64:v1.10.1
    ports:
      - containerPort: 9090
        protocol: TCP
    args:
      - --token-ttl=6000
      - --enable-insecure-login
      - --insecure-bind-address=0.0.0.0
      - --insecure-port=9090
```

Make sure you delete this argument, otherwise the other settings will be ignored:

```
--auto-generate-certificates
```

Also change the liveness probe like this below.

```
livenessProbe:
            httpGet:
              scheme: HTTP
              path: /
              port: 9090
```

MY-OWN COMPLETE FILE WITH CHANGES.

```
# Copyright 2017 The Kubernetes Authors.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
# See the License for the specific language governing permissions and
# limitations under the License.

apiVersion: v1
kind: Namespace
metadata:
  name: kubernetes-dashboard

---

apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard

---

kind: Service
```

```
apiVersion: v1
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
spec:
  ports:
    - port: 80
      targetPort: 9090
  selector:
    k8s-app: kubernetes-dashboard

---

apiVersion: v1
kind: Secret
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard-certs
  namespace: kubernetes-dashboard
type: Opaque

---

apiVersion: v1
kind: Secret
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard-csrf
  namespace: kubernetes-dashboard
type: Opaque
data:
  csrf: ""

---

apiVersion: v1
kind: Secret
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard-key-holder
  namespace: kubernetes-dashboard
type: Opaque

---
```

```
kind: ConfigMap
apiVersion: v1
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard-settings
  namespace: kubernetes-dashboard

---

kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
rules:
  # Allow Dashboard to get, update and delete Dashboard exclusive
secrets.
  - apiGroups: [""]
    resources: ["secrets"]
    resourceNames: ["kubernetes-dashboard-key-holder", "kubernetes-
dashboard-certs", "kubernetes-dashboard-csrf"]
    verbs: ["get", "update", "delete"]
    # Allow Dashboard to get and update 'kubernetes-dashboard-settings'
config map.
  - apiGroups: [""]
    resources: ["configmaps"]
    resourceNames: ["kubernetes-dashboard-settings"]
    verbs: ["get", "update"]
    # Allow Dashboard to get metrics.
  - apiGroups: [""]
    resources: ["services"]
    resourceNames: ["heapster", "dashboard-metrics-scraper"]
    verbs: ["proxy"]
  - apiGroups: [""]
    resources: ["services/proxy"]
    resourceNames: ["heapster", "http:heapster:", "https:heapster:",
"dashboard-metrics-scraper", "http:dashboard-metrics-scraper"]
    verbs: ["get"]

---

kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
```

```yaml
rules:
  # Allow Metrics Scraper to get metrics from the Metrics server
  - apiGroups: ["metrics.k8s.io"]
    resources: ["pods", "nodes"]
    verbs: ["get", "list", "watch"]

---

apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: kubernetes-dashboard
subjects:
  - kind: ServiceAccount
    name: kubernetes-dashboard
    namespace: kubernetes-dashboard

---

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: kubernetes-dashboard
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: kubernetes-dashboard
subjects:
  - kind: ServiceAccount
    name: kubernetes-dashboard
    namespace: kubernetes-dashboard

---

kind: Deployment
apiVersion: apps/v1
metadata:
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
spec:
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      k8s-app: kubernetes-dashboard
  template:
    metadata:
      labels:
        k8s-app: kubernetes-dashboard
    spec:
      securityContext:
        seccompProfile:
          type: RuntimeDefault
      containers:
        - name: kubernetes-dashboard
          image: kubernetesui/dashboard:v2.5.0
          imagePullPolicy: Always
          ports:
            - containerPort: 9090
              protocol: TCP
          args:
            - --namespace=kubernetes-dashboard
            - --token-ttl=6000
            - --enable-insecure-login
            - --insecure-bind-address=0.0.0.0
            - --insecure-port=9090
            # Uncomment the following line to manually specify
Kubernetes API server Host
            # If not specified, Dashboard will attempt to auto discover
the API server and connect
            # to it. Uncomment only if the default does not work.
            # - --apiserver-host=http://my-address:port
          volumeMounts:
            - name: kubernetes-dashboard-certs
              mountPath: /certs
              # Create on-disk volume to store exec logs
            - mountPath: /tmp
              name: tmp-volume
          livenessProbe:
            httpGet:
              scheme: HTTP
              path: /
              port: 9090
            initialDelaySeconds: 30
            timeoutSeconds: 30
          securityContext:
            allowPrivilegeEscalation: false
            readOnlyRootFilesystem: true
            runAsUser: 1001
            runAsGroup: 2001
      volumes:
        - name: kubernetes-dashboard-certs
          secret:
            secretName: kubernetes-dashboard-certs
        - name: tmp-volume
          emptyDir: {}
      serviceAccountName: kubernetes-dashboard
      nodeSelector:
        "kubernetes.io/os": linux
      # Comment the following tolerations if Dashboard must not be
deployed on master
      tolerations:
        - key: node-role.kubernetes.io/master
          effect: NoSchedule

---

kind: Service
apiVersion: v1
metadata:
  labels:
    k8s-app: dashboard-metrics-scraper
  name: dashboard-metrics-scraper
  namespace: kubernetes-dashboard
spec:
  ports:
    - port: 8000
      targetPort: 8000
  selector:
    k8s-app: dashboard-metrics-scraper

---

kind: Deployment
apiVersion: apps/v1
metadata:
  labels:
    k8s-app: dashboard-metrics-scraper
  name: dashboard-metrics-scraper
  namespace: kubernetes-dashboard
spec:
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      k8s-app: dashboard-metrics-scraper
  template:
    metadata:
      labels:
        k8s-app: dashboard-metrics-scraper
    spec:
      securityContext:
        seccompProfile:
          type: RuntimeDefault
      containers:
        - name: dashboard-metrics-scraper
          image: kubernetesui/metrics-scraper:v1.0.7
          ports:
            - containerPort: 8000
              protocol: TCP
          livenessProbe:
            httpGet:
              scheme: HTTP
              path: /
              port: 8000
            initialDelaySeconds: 30
            timeoutSeconds: 30
          volumeMounts:
          - mountPath: /tmp
            name: tmp-volume
          securityContext:
            allowPrivilegeEscalation: false
            readOnlyRootFilesystem: true
            runAsUser: 1001
            runAsGroup: 2001
      serviceAccountName: kubernetes-dashboard
      nodeSelector:
        "kubernetes.io/os": linux
      # Comment the following tolerations if Dashboard must not be
deployed on master
      tolerations:
        - key: node-role.kubernetes.io/master
          effect: NoSchedule
      volumes:
        - name: tmp-volume
          emptyDir: {
```

STEP-2:

Run command: kubectl apply -f k8s-dashboard.yaml [whatever you have given name]

It will setup the namespaces and clusterrole , clusterrolebinding etc.

STEP-3:

Now create Gateway and virtual service.

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: gateway-k8s-dash-dev-qa-gw
  namespace: kubernetes-dashboard
spec:
  selector:
    istio: ingressgateway
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - yourdomain.com

---

apiVersion: networking.istio.io/v1beta1
kind: VirtualService
metadata:
  name: gateway-k8s-dash-dev-qa-vs
  namespace: kubernetes-dashboard
spec:
  hosts:
  - yourdomain.com
  gateways:
  - gateway-k8s-dash-dev-qa-gw
  http:
  - route:
    - destination:
        host: kubernetes-dashboard
        port:
          number: 80
```

Command: **kubectl apply -f gateway-vs.yaml**

This will create gateway and virtual service on kubernetes-dashboard namespace. On AWS End please create CNAME Record for domain with Istio-ALB Endpoint.

Kubernetes Dashboard Authentication

Create the dashboard service account

```
kubectl create serviceaccount dashboard-admin-sa
```

This will create a service account named dashboard-admin-sa in the default namespace.

Next bind the dashboard-admin-service-account service account to the cluster-admin role

```
kubectl create clusterrolebinding dashboard-admin-sa
--clusterrole=cluster-admin --serviceaccount=default:dashboard-admin-sa
```

When we created the dashboard-admin-sa service account Kubernetes also created a secret for it.

List secrets using:

```
kubectl get secrets
```

Use kubectl describe to get the access token:

```
kubectl describe secret dashboard-admin-sa-token-kw7vn
```

Kubernetes Dashboard Authentication with Read-only-Access

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: read-only-user
  namespace: kubernetes-dashboard
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  labels:
  name: read-only-clusterrole
  namespace: default
rules:
- apiGroups:
  - ""
  resources: ["*"]
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - extensions
  resources: ["*"]
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - apps
  resources: ["*"]
  verbs:
  - get
  - list
  - watch
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: read-only-binding
roleRef:
  kind: ClusterRole
  name: read-only-clusterrole
  apiGroup: rbac.authorization.k8s.io
subjects:
- kind: ServiceAccount
  name: read-only-user
  namespace: kubernetes-dashboard
```

This will create read-only-user secret on Kubernetes dashboard. Use this secret token to share with dev-team.

ADD-ON

Below yaml is same k8s-dashboard access as a read user. difference is that it does not list secrets, config-map and other thing.

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: read-only-user
  namespace: kubernetes-dashboard
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  labels:
  name: read-only-clusterrole
  namespace: default
rules:
- apiGroups:
  - ""
  resources:
  - configmaps
  - endpoints
  - persistentvolumeclaims
  - pods
  - replicationcontrollers
  - replicationcontrollers/scale
  - serviceaccounts
  - services
  - nodes
  - persistentvolumeclaims
  - persistentvolumes
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - bindings
  - events
  - limitranges
  - namespaces/status
  - pods/log
  - pods/status
  - replicationcontrollers/status
  - resourcequotas
  - resourcequotas/status
```

```
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - ""
    resources:
    - namespaces
    verbs:
    - get
    - list
    - watch
  - apps
    resources:
    - daemonsets
    - deployments
    - deployments/scale
    - replicasets
    - replicasets/scale
    - statefulsets
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - autoscaling
    resources:
    - horizontalpodautoscalers
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - batch
    resources:
    - cronjobs
    - jobs
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - extensions
    resources:
    - daemonsets
    - deployments
    - deployments/scale
    - ingresses
    - networkpolicies
    - replicasets
    - replicasets/scale
    - replicationcontrollers/scale
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - policy
    resources:
    - poddisruptionbudgets
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - networking.k8s.io
    resources:
    - networkpolicies
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - storage.k8s.io
    resources:
    - storageclasses
    - volumeattachments
    verbs:
    - get
    - list
    - watch
  - apiGroups:
    - rbac.authorization.k8s.io
    resources:
    - clusterrolebindings
    - clusterroles
    - roles
    - rolebindings
    verbs:
    - get
    - list
    - watch
  ---
  apiVersion: rbac.authorization.k8s.io/v1
  kind: ClusterRoleBinding
  metadata:
    name: read-only-binding
  roleRef:
    kind: ClusterRole
    name: read-only-clusterrole
    apiGroup: rbac.authorization.k8s.io
  subjects:
  - kind: ServiceAccount
    name: read-only-user
    namespace: kubernetes-dashboard
```