Configuring a Workload Identity on your deployment

Required:

serviceAccountName

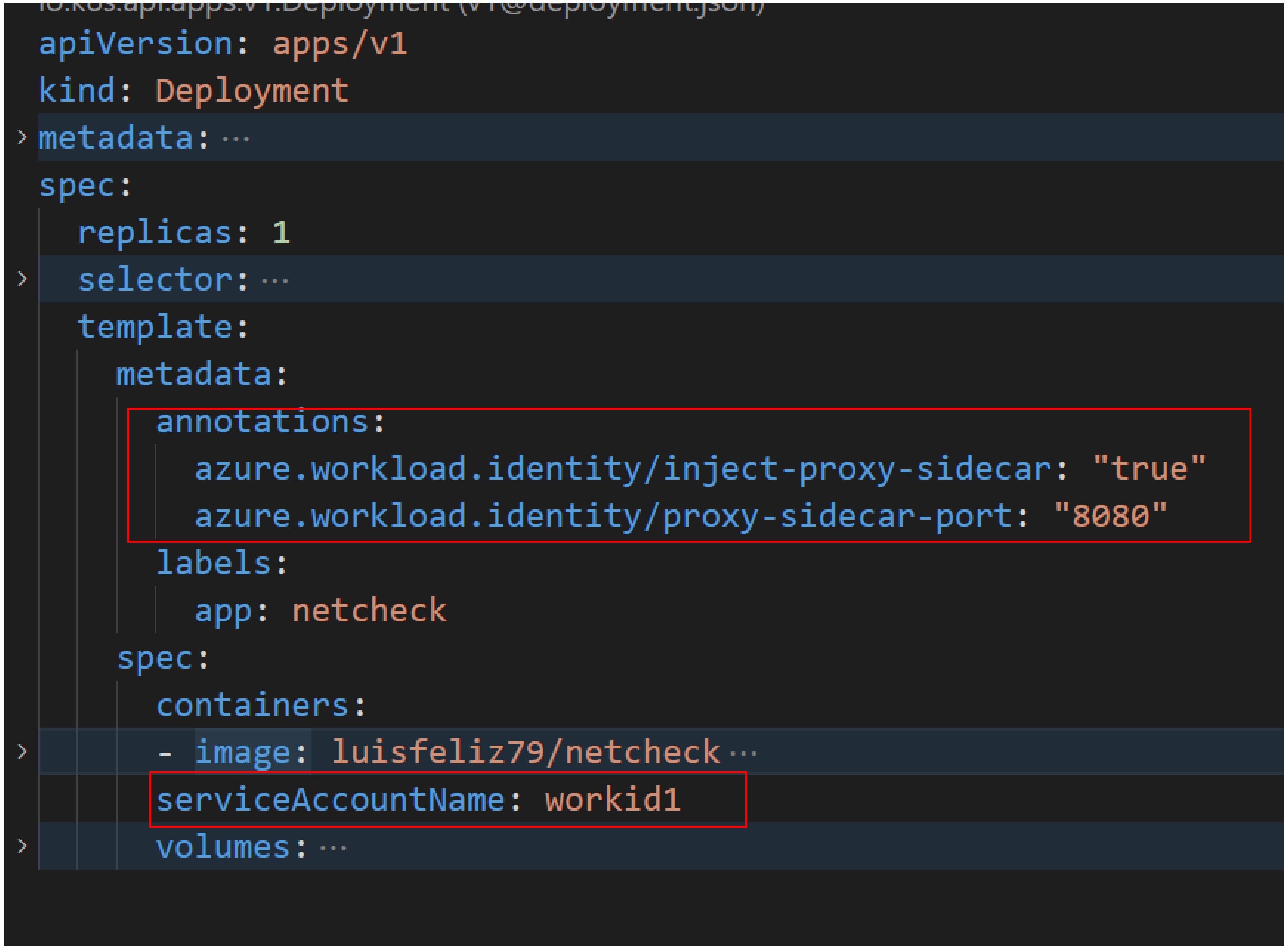
- Assigns the MI to this deployment/pods
- This should be the name of the serviceAccount, which should be in the **same namespace** as the deployment/pod

Optional:

Inject-proxy-sidecar

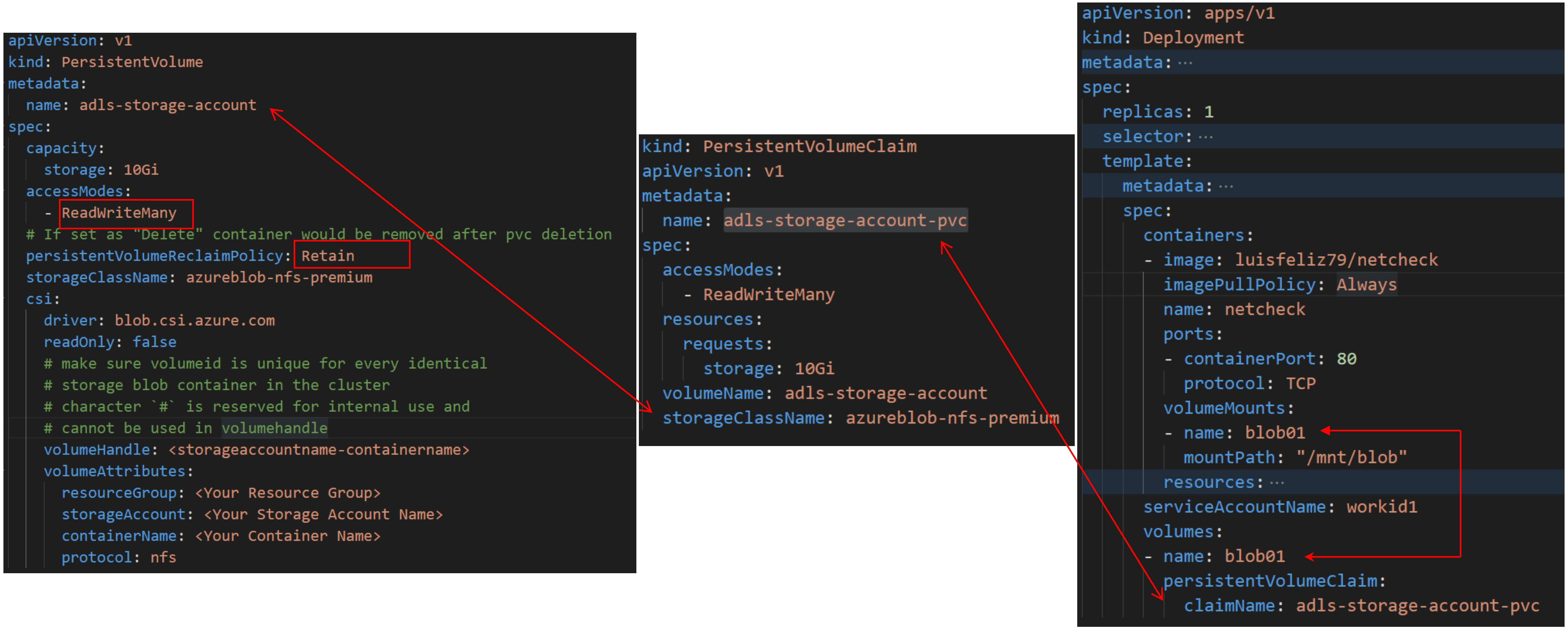
- Enables the Managed Identity endpoint http://169.254.169.254/metadata/identity/oauth2/token
- This is a backwards compatibility feature for workloads that rely on the Pod Identity feature. This feature injects an additional container in your pod which intercepts authentication requests.
- **Use this when your app does not yet** use the latest Azure SDK Libraries or if you have written custom rest calls to get an access token via the Managed identity endpoint

https://learn.microsoft.com/en-us/azure/aks/workload-identity-overview



```yaml
io.k8s.api.apps.v1.Deployment (v1@deployment.json)
apiVersion: apps/v1
kind: Deployment
metadata: ···
spec:
  replicas: 1
  selector: ···
  template:
    metadata:
      annotations:
        azure.workload.identity/inject-proxy-sidecar: "true"
        azure.workload.identity/proxy-sidecar-port: "8080"
      labels:
        app: netcheck
    spec:
      containers:
      - image: luisfeliz79/netcheck ···
      serviceAccountName: workid1
      volumes: ···
```

Mounting Storage with Azure Blob CSI - NFS

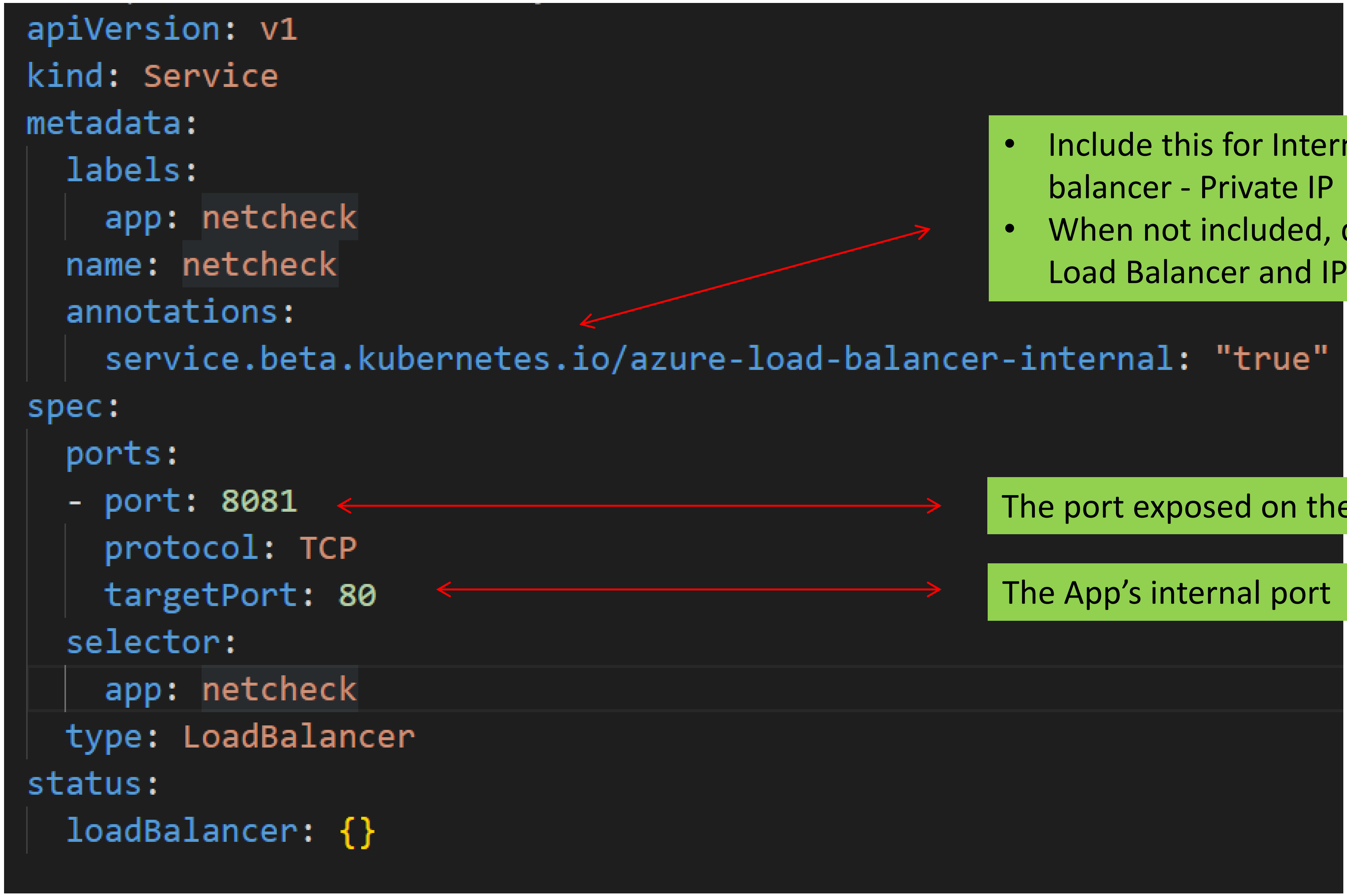


```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: adls-storage-account
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteMany
  # If set as "Delete" container would be removed after pvc deletion
  persistentVolumeReclaimPolicy: Retain
  storageClassName: azureblob-nfs-premium
  csi:
    driver: blob.csi.azure.com
    readOnly: false
    # make sure volumeid is unique for every identical
    # storage blob container in the cluster
    # character `#` is reserved for internal use and
    # cannot be used in volumehandle
    volumeHandle: <storageaccountname-containername>
    volumeAttributes:
      resourceGroup: <Your Resource Group>
      storageAccount: <Your Storage Account Name>
      containerName: <Your Container Name>
      protocol: nfs
```

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: adls-storage-account-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
  volumeName: adls-storage-account
  storageClassName: azureblob-nfs-premium
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata: ···
spec:
  replicas: 1
  selector: ···
  template:
    metadata: ···
    spec:
      containers:
      - image: luisfeliz79/netcheck
        imagePullPolicy: Always
        name: netcheck
        ports:
        - containerPort: 80
          protocol: TCP
        volumeMounts:
        - name: blob01
          mountPath: "/mnt/blob"
        resources: ···
      serviceAccountName: workid1
      volumes:
      - name: blob01
        persistentVolumeClaim:
          claimName: adls-storage-account-pvc
```

Configuring a Load balancer and exposing apps outside of the AKS Cluster



```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: netcheck
  name: netcheck
  annotations:
    service.beta.kubernetes.io/azure-load-balancer-internal: "true"
spec:
  ports:
  - port: 8081
    protocol: TCP
    targetPort: 80
  selector:
    app: netcheck
  type: LoadBalancer
status:
  loadBalancer: {}
```

- Include this for Internal Load balancer - Private IP
- When not included, default is Public Load Balancer and IP

The port exposed on the Load Balancer

The App’s internal port