Домашнее задание для к уроку 7 - Продвинутые абстракции Kubernetes

! Задание нужно выполнять в нэймспэйсе default
Разверните в кластере сервер систему мониторинга Prometheus.

- Создайте в кластере ConfigMap со следующим содержимым:

```
prometheus.yml: |
  global:
    scrape_interval: 30s

  scrape_configs:
    - job_name: 'prometheus'
      static_configs:
      - targets: ['localhost:9090']
    - job_name: 'kubernetes-nodes'
      kubernetes_sd_configs:
      - role: node
      relabel_configs:
      - source_labels: [__address__]
        regex: (.+):(.+)
        target_label: __address__
        replacement: ${1}:9101
```

Создайте объекты для авторизации Prometheus сервера в Kubernetes-API.

```
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: prometheus
  namespace: default
---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRole
metadata:
  name: prometheus
rules:
- apiGroups: [""]
  resources:
  - nodes
  verbs: ["get", "list", "watch"]
---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRoleBinding
metadata:
  name: prometheus
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: prometheus
subjects:
- kind: ServiceAccount
  name: prometheus
  namespace: default
```

- Создайте StatefulSet для Prometheus сервера из образа prom/prometheus:v2.19.2 с одной репликой

В нем должнен быть описан порт 9090 TCP
volumeClaimTemplate - ReadWriteOnce, 5Gi, подключенный по пути /prometheus
Подключение конфигмапа с настройками выше по пути /etc/prometheus

Так же в этом стейтфулсете нужно объявить initContainer для изменения прав на вольюм data на 777.
См пример из лекции 4: practice/4.resources-and-persistence/persistence/deployment.yaml

> Не забудьте указать обязательное поле serviceName
> Так же укажите поле serviceAccount: prometheus на одном уровне с containers, initContainers, volumes
> См пример с rabbitmq из материалов лекции.

- Создайте service и ingress для этого стейтфулсета, так чтобы запросы с любым доменом на белый IP вашего сервиса nginx-ingress-controller (тот что в нэймспэйсе ingress-nginx с типом LoadBalancer) шли на приложение
- Проверьте что при обращении из браузера на белый IP вы видите открывшееся приложение Prometheus
- В этом же неймспэйсе создайте DaemonSet node-exporter как в примере к лекции: practice/7.advanced-abstractions/daemonset.yaml
- Откройте в браузере интерфейс Prometheus.
  Попробуйте открыть Status -> Targets
  Тут вы должны увидеть все ноды своего кластера, которые Prometheus смог определить и собирает с ним метрики.

Так же можете попробовать на вкладке Graph выполнить запрос node_load1 - это минутный Load Average для каждой из нод в кластере.

Листинг манифестов

cat configmap_prometheus.yaml

```
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-config
data:
  prometheus.yml: |
    global:
      scrape_interval: 30s

    scrape_configs:
      - job_name: 'prometheus'
        static_configs:
        - targets: ['localhost:9090']
      - job_name: 'kubernetes-nodes'
        kubernetes_sd_configs:
        - role: node
        relabel_configs:
        - source_labels: [__address__]
          regex: (.+):(.+)
          target_label: __address__
          replacement: ${1}:9101
```

cat daemonset.yaml

```
---
apiVersion: apps/v1
kind: DaemonSet
metadata:
  labels:
    app: node-exporter
  name: node-exporter
```

```
spec:
  updateStrategy:
    rollingUpdate:
      maxUnavailable: 1
    type: RollingUpdate
  selector:
    matchLabels:
      app: node-exporter
  template:
    metadata:
      labels:
        app: node-exporter
    spec:
      containers:
      - args:
        - --web.listen-address=0.0.0.0:9101
        - --path.procfs=/host/proc
        - --path.sysfs=/host/sys
        - --collector.filesystem.ignored-mount-points=^/(dev|proc|sys|var/lib/
docker/.+)($|/)
        - --collector.filesystem.ignored-fs-types=^(autofs|binfmt_misc|cgroup|
configfs|debugfs|devpts|devtmpfs|fusectl|hugetlbfs|mqueue|overlay|proc|procfs|
pstore|rpc_pipefs|securityfs|sysfs|tracefs)$
        image: quay.io/prometheus/node-exporter:v0.16.0
        imagePullPolicy: IfNotPresent
        name: node-exporter
        volumeMounts:
        - mountPath: /host/proc
          name: proc
        - mountPath: /host/sys
          name: sys
        - mountPath: /host/root
          name: root
          readOnly: true
      hostNetwork: true
```

```
      hostPID: true
      tolerations:
        - effect: NoSchedule
          operator: Exists
      nodeSelector:
        beta.kubernetes.io/os: linux
      volumes:
      - hostPath:
          path: /proc
          type: ""
        name: proc
      - hostPath:
          path: /sys
          type: ""
        name: sys
      - hostPath:
          path: /
          type: ""
        name: root
```

cat ingress_prometheus.yaml

```
---
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: prometheus
spec:
  rules:
     - http:
```

```
        paths:
        - path: /
          pathType: Prefix
          backend:
            serviceName: prometheus
            servicePort: 80
```

cat service_acc_prometheus.yaml

```
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: prometheus
  namespace: default
---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRole
metadata:
  name: prometheus
rules:
- apiGroups: [""]
  resources:
```

```
  - nodes
  verbs: ["get", "list", "watch"]
---
apiVersion: rbac.authorization.k8s.io/v1beta1
kind: ClusterRoleBinding
metadata:
  name: prometheus
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: prometheus
subjects:
- kind: ServiceAccount
  name: prometheus
  namespace: default
```

cat service_prometheus.yaml

```
---
apiVersion: v1
kind: Service
metadata:
  name: prometheus
```

```
spec:
  selector:
    app: prometheus
  ports:
  - protocol: TCP
    port: 80
    targetPort: 9090
```

cat sts_prometheus3.yaml

```
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: prometheus
spec:
  serviceName: "prometheus"
  replicas: 1
  selector:
    matchLabels:
             app: prometheus
  template:
    metadata:
      labels:
         app: prometheus
    spec:
```

```
serviceAccount: prometheus
initContainers:
- name: "init-busybox-permission777"
  image: busybox
  imagePullPolicy: Always
  command: ["chmod","-R","777", "/data"]
  volumeMounts:
  - name: prometheus-data
    mountPath: /data
containers:
  - name: prometheus-server-2-19-2
    image: prom/prometheus:v2.19.2
    imagePullPolicy: "IfNotPresent"
    ports:
      - containerPort: 9090
    readinessProbe:
      httpGet:
        path: /-/ready
        port: 9090
      initialDelaySeconds: 30
      timeoutSeconds: 30
    livenessProbe:
      httpGet:
        path: /-/healthy
        port: 9090
      initialDelaySeconds: 30
      timeoutSeconds: 30
```

```
        resources:
          requests:
            cpu: 200m
            memory: 1000Mi
        volumeMounts:
          - name: config-volume-prometheus
            mountPath: /etc/prometheus
          - name: prometheus-data
            mountPath: /data
    terminationGracePeriodSeconds: 300
    volumes:
      - name: config-volume-prometheus
        configMap:
          name: prometheus-config
volumeClaimTemplates:
- metadata:
    name: prometheus-data
  spec:
    storageClassName: csi-ceph-hdd-dp1
    accessModes:
      - ReadWriteOnce
    resources:
      requests:
        storage: "5Gi"
```

Скриншоты работы kubernetes-cluster

```
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW7$ k get pvc
NAME                          STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS       AGE
prometheus-data-prometheus-0  Bound    pvc-7b64bf0d-15e7-4872-afef-031668abb7c0   5Gi        RWO            csi-ceph-hdd-dp1   17s
ps-pgdata                     Bound    pvc-183ae07a-2319-4ea5-a132-91fa99a8e6ab   10Gi       RWX            csi-ceph-hdd-dp1   6d17h
webdav                        Bound    pvc-40024748-9194-4a9b-a5f9-fe7837df9870   2Gi        RWX            csi-ceph-hdd-dp1   9d
```

```
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW7$ k get svc -A
NAMESPACE       NAME                                 TYPE           CLUSTER-IP       EXTERNAL-IP     PORT(S)                      AGE
default         kubernetes                           ClusterIP      10.254.0.1       <none>          443/TCP                      10d
default         prometheus                           ClusterIP      10.254.141.68    <none>          80/TCP                       37s
ingress-nginx   nginx-ingress-controller             LoadBalancer   10.254.128.123   89.208.87.242   80:30080/TCP,443:30443/TCP   10d
ingress-nginx   nginx-ingress-controller-metrics     ClusterIP      10.254.11.207    <none>          9913/TCP                     10d
ingress-nginx   nginx-ingress-default-backend        ClusterIP      10.254.244.136   <none>          80/TCP                       10d
kube-system     calico-node                          ClusterIP      None             <none>          9091/TCP                     10d
kube-system     csi-cinder-controller-service        ClusterIP      10.254.209.85    <none>          12345/TCP                    10d
kube-system     dashboard-metrics-scraper            ClusterIP      10.254.174.238   <none>          8000/TCP                     10d
kube-system     kube-dns                             ClusterIP      10.254.0.10      <none>          53/UDP,53/TCP,9153/TCP       10d
kube-system     kubernetes-dashboard                 ClusterIP      10.254.179.232   <none>          443/TCP                      10d
kube-system     metrics-server                       ClusterIP      10.254.209.248   <none>          443/TCP                      10d
magnum-tiller   tiller-deploy                        ClusterIP      10.254.126.244   <none>          44134/TCP                    10d
```

```
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW7$ k get sts
NAME         READY   AGE
prometheus   1/1     67s
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW7$ k get ing
NAME         CLASS    HOSTS   ADDRESS   PORTS   AGE
prometheus   <none>   *                 80      72s
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW7$ k get ep
NAME         ENDPOINTS              AGE
kubernetes   10.0.0.9:6443          10d
prometheus   10.100.180.144:9090    87s
```

```
lamo@lamo-K501LB:~/nout/GEEKBRAINS/Microservice_architecture_and_containerization/HW7$ k get po -owide
NAME                    READY   STATUS    RESTARTS   AGE    IP              NODE                                          NOMINATED NODE   READINESS GATES
node-exporter-5295h     1/1     Running   0          113s   10.0.0.9        kubernetes-cluster-5268-master-0              <none>           <none>
node-exporter-cdklx     1/1     Running   0          113s   10.0.0.14       kubernetes-cluster-5268-default-group-0       <none>           <none>
node-exporter-dpm56     1/1     Running   0          113s   10.0.0.19       kubernetes-cluster-5268-default-group-2       <none>           <none>
prometheus-0            1/1     Running   0          113s   10.100.180.144  kubernetes-cluster-5268-default-group-0       <none>           <none>
```

Подключаемся через браузер к нашему приложению.