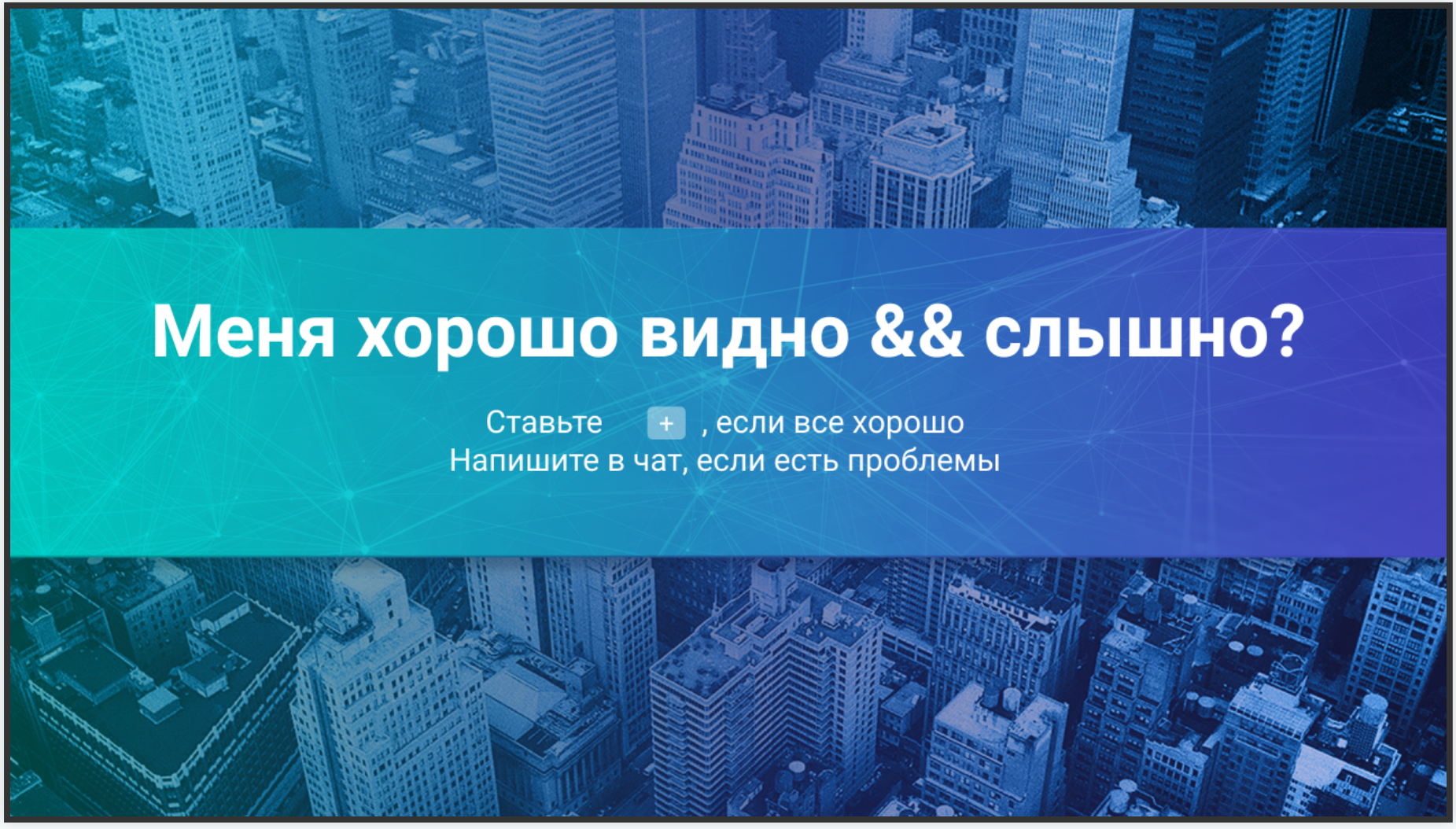

OTUS
ОНЛАЙН-ОБРАЗОВАНИЕ
Онлайн-образование


Меня хорошо видно && слышно?
Ставьте + , если все хорошо
Напишите в чат, если есть проблемы

План

- Сетевое взаимодействие
- Хранение данных
- Мониторинг

Pod

- Смертны
- Динамически создаются и удаляются
- IP-адреса появляются и исчезают вместе с Pod'ами
- Несколько Pod'ов могут выполнять одну и ту же функцию

Service

- Абстракция, описывающая набор Pod'ов и конфигурацию доступа к ним
- Позволяет отвязаться от использования конкретных Pod'ов

Service with selector

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```

Service without selector

```yaml
kind: Service
apiVersion: v1
metadata:
  name: my-service
spec:
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```

```yaml
kind: Endpoints
apiVersion: v1
metadata:
 name: my-service
subsets:
  - addresses:
      - ip: 1.2.3.4
    ports:
      - port: 9376
```

ExternalName Service

- Редирект на уровне DNS (прокси не используется)
- Возвращает CNAME запись

```
my-service.prod.svc.cluster.local => my.database.example.com
my-service => my.database.example.com
```

```
kind: Service
apiVersion: v1
metadata:
 name: my-service
 namespace: prod
spec:
  type: ExternalName
  externalName: my.database.example.com
```

DNS records

В рамках неймспейса - svc-name

Из другого неймспейса - svc-name.namespace

Через весь кластер - svc-name.namespace.svc.cluster.local

Поды - pod-ip-address.namespace.pod.cluster.local

Как опубликовать приложение

ТИПЫ СЕРВИСОВ

- ClusterIP - доступ только изнутри кластера
- NodePort - доступ снаружи кластера через порты физических хостов
- LoadBalancer - доступ снаружи кластера с использованием внешнего сервиса балансировки
- ExternalIP - доступ снаружи по внешнему IP адресу, указывающему на одну из нод

Ingress

Объект управляющий внешним доступом к сервисам внутри кластера.

Обеспечивает:

- Организацию единой точки входа в приложения снаружи
- Балансировку трафика
- Терминацию SSL
- Виртуальный хостинг на основе имен и т.д.
- Работает на L7 уровне.

Ingress Controller

Ingress - набор правил внутри кластера Kubernetes.
Для применения данных правил нужен Ingress Controller - плагин который состоит из 2-х функциональных частей:

- Приложение, которое отслеживает через Kubernetes API новые объекты Ingress и обновляет конфигурацию балансировщика
- Балансировщик (nginx, HAProxy, traefik, ...), который отвечает за управление сетевым трафиком

Ingress

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
spec:
  rules:
  - http:
      paths:
      - path: /testpath
        backend:
          serviceName: test
          servicePort: 80
```

Single Service Ingress

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
spec:
  backend:
    serviceName: test
    servicePort: 80
```

Simple Fanout

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - path: /foo
        backend:
          serviceName: s1
          servicePort: 80
      - path: /bar
        backend:
          serviceName: s2
          servicePort: 80
```

Name Based Virtual Hosting

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
spec:
  rules:
  - host: foo.bar.com
    http:
      paths:
      - backend:
        serviceName: s1
        servicePort: 80
  - host: bar.foo.com
    http:
      paths:
      - backend:
        serviceName: s2
        servicePort: 80
```

TLS termination

```
apiVersion: v1
kind: Secret
data:
  tls.crt: base64 encoded cert
  tls.key: base64 encoded key
metadata:
  name: testsecret-tls
  namespace: default
```

```
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: tls-example-ingress
spec:
  tls:
  - hosts:
    - sslexample.foo.com
    secretName: testsecret-tls
  rules:
  ...
```

Хранение данных в Kubernetes

Volumes

- Меньше ограничений по сравнению с Docker'ом
- Full lifecycle с привязкой к Pod'у
- Pod может использовать несколько типов Volume'ов одновременно

Local Volumes

- emptyDir - пустая директория, создается на Node и удаляется вместе с удалением Pod. Используется для обмена данными между контейнерами внутри Pod (в том числе init containers)
- hostPath - директория на Node, не удаляется при удалении Pod. Используется для доступа к информации на Node, например к /var/lib/docker/

Cloud Volumes

- AWS EBS
- AzureDisk и AzureFile
- gcePersistentDisk

Custom Volumes

- nfs
- CephFS и GlusterFS
- FC и iSCSI

Persistent Volumes

- PersistentVolume
- PersistentVolumeClaim
- StorageClass
- Local Volumes

Persistent Volumes

- Создаются на уровне кластера
- PV похожи на обычные Volume, но имеют отдельный от сервисов жизненный цикл

Persistent Volumes

- ReadWriteOnce
- ReadOnlyMany
- ReadWriteMany

Persistent Volumes: Phase

- Available — доступен, не смонтирован
- Bound — смонтирован
- Released — размонтирован
- Failed — что-то сломалось при запросе

Persistent Volumes: Reclaim Policy

- Retain — не удалять
- Recycle — переиспользовать (deprecated)
- Delete (default) — удалять

PersistentVolumeClaim

- Запрос на хранилище от пользователя
- PVC могут требовать определенный объем хранилища и прав доступа
- Создаются на уровне namespace

Мониторинг

Мониторинг

PIPELINES

- Core metrics pipeline
  - Kubelet
  - Resource estimator
  - Metrics-server
  - API server
- Monitoring pipeline
  - Expose metrics to end-users

МОНИТОРИНГ НОД КЛАСТЕРА

- Core metrics
  - CPU
  - RAM
  - Disk
  - Network

Мониторинг

KUBE-API SERVER

- Request Rates and Latencies
- Performance of controller work queues
- Etcd helper cache work queues and cache performance
- General process status (File Descriptors/Memory/CPU Seconds)
- Golang status (GC/Memory/Threads)

Мониторинг

ETCD

- Leader existence and leader change rate
- Proposals committed/applied/pending/failed
- Disk write performance
- Inbound gRPC stats
- Intra-cluster gRPC stats

Prometheus

КОМПОНЕНТЫ:

- Pushgateway
- Prometheus server
- Alertmanagers
- Web-UI

Prometheus хранение метрик

ОТДЕЛЬНЫЕ ХРАНИЛИЩА ДЛЯ ИСТОРИЧЕСКИХ ДАННЫХ:

- Victoria Metrics
- Cotrex
- Thanos
- Clickhouse ^_^

Prometheus operator

- Pod monitor
- Service monitor

ВАРИАНТЫ УСТАНОВКИ

- Helm chart
- Official repo
- Вкатить его руками ^_^

Prometheus operator

```
prometheus:
  prometheusSpec:
    retention: 3d
    ### Check this labels: kubectl get prometheus -o yaml -n moni
    serviceMonitorNamespaceSelector: {} ### Namespace for Service
    serviceMonitorSelectorNilUsesHelmValues: false
    serviceMonitorSelector: {} ### matchLabels for ServiceMonitor
# serviceMonitorSelector:
# matchLabels:
# prometheus: kube-prometheus
# release: prometheus-cluster-monitoring
grafana:
  adminPassword: XXXXXXXXXXXXXXXXXXXX
```

Service monitor

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: example-app
  labels:
    team: frontend
spec:
  selector:
    matchLabels:
      app: example-app
  endpoints:
  - port: web
```