K8S(Kubernetes)容器运维工程师

K8S集群(生产环境)部署及规划：

1. 基于二进制方式部署K8S多Master高可用集群，利用Ansible Playbook自动化部署集群、备份恢复集群以及管理Master,Worker,Etcd等节点。

2. 根据集群规模(节点数量、Pod数量、部署频率、访问量等多个角度衡量)来规划Master节点的机器配置，根据集群上应用程序所需总的资源 (CPU,内存)、副本数、高可用性等来规划Worker节点的机器配置以及数量。

3. 使用Helm部署K8S常用插件: Metrics-Server, Metallb, Nginx-Ingress/Traefik, Gitlab, Jenkins, Harbor, Elasticsearch, Fluentd/Log-Pilot, Kibana, Prometheus, Alertmanager, Grafana, Dashboard/Rancher等。

迁移公司传统虚拟机上运行的Java应用项目(Tomcat,Spring Boot,Spring Cloud/Dubbo微服务架构等)至K8S容器平台:

1. 熟悉Spring Cloud微服务架构常用组件：Eureka, Ribbon, Hystrix, Feign, Zuul, Config, Gateway, Oauth2, Admin等的基础上，与研发沟通，将微服务架构中各个组件及应用模块容器化，设计并规划好Docker镜像的构建方案。
3. 应用容器化后，将容器放入Pod中，同时需要考虑Pod的资源请求和限制，健康检查，数据持久化，调度约束，滚动更新，环境变量注入(比如通过k8s环境变量配置dockerfile容器启动命令参数以及springboot配置文件application.yml)，Network Policy，HPA等。
4. 单一Pod如果出现故障，就会影响业务连续性，所以需要多副本，对于无状态应用一般使用Deployment Controller(封装了Pod的副本管理、部署更新、回滚、扩容、缩容等)来管理Pod。而对于有状态的应用使用StatefulSet(比如Spring Cloud架构中服务注册组件Eureka在K8S中高可用部署)。
5. 使用Deployment通过多副本的方式保证了Pod的高可用和横向扩展，那么就需要考虑负载均衡，集群内部使用Service管理Pod访问，外部访问集群可以通过Service（Nodeport, Loadblancer）4层负载均衡以及Ingress(推荐使用)7层负载均衡来实现。
6. 容器中的存储都是临时的，因此Pod重启的时候，内部的数据会发生丢失，使用PV/PVC管理持久化数据。
7. 代码和配置分离，这样更容易实现DevOps流水线的编排，K8S中使用ConfigMap管理应用配置文件，推荐使用Apollo集中化管理应用不同环境、不同集群的配置。

**构建基于 Kubernetes, Jenkins, Gitlab, Harbor, Helm 的 CI/CD(持续集成/部署)完整DevOps流水线：**

1. **传统虚拟机上Jenkins一主多从方式存在主Master单点故障，每个Slave的编译配置环境不一样，资源分配不均衡且浪费的问题。而K8S上运行的Jenkins通过动态创建Jenkins Slave Pod来处理Job的方式解决传统的 Jenkins Slave 浪费大量资源、队列阻塞等问题。**
2. **配置动态Jenkins Slave，Jenkins安装kubernetes-plugin,并在Jenkinsfile中配置Jenkins Slave Pod模板，Jenkins Slave Pod中加入CI/CD过程中所需要用到的二进制命令工具，比如kubectl,docker,maven,gradle,npm,helm等。**
3. **构建CI/CD过程中关键是以下配置文件的编写，建议这些配置文件全部放入项目代码仓库中进行统一的管理：**
4. **Jenkinsfile文件: Jenkins Pipeline部署 Kubernetes 应用的CI/CD流程大体如下：1.git拉取代码、2.代码静态分析/单元测试、3.maven/gradle/npm编译打包、4.docker构建镜像(编写Dockerfile)并推送至harbor镜像仓库、5.**更改k8s yaml文件中docker镜像tag(编写k8s yaml文件)、6.利用kubectl/helm(编写helm chart相关文件)部署/更新应用到k8s平台，建议采用Declarative Pipeline编写Jenkinsfile文件。
5. **Dockerfile文件：在遵循官方Dockerfile最佳实践规范的基础上，尽量构建最小Docker镜像，推荐使用 Alpine 基础镜像。**
6. **Kubernetes Yaml文件：主要是常用K8S资源的配置：deployment, statefulset, service, ingress, persistentvolumeclaim, configmap等。其中deployment一定要配置pod的资源请求(resource request)以及健康检查(liveness/readiness probe)。**
7. **Helm Chart相关文件：编写values.yaml以及各类K8S资源的配置模板yaml文件。**

**K8S其它相关**

1. **日志：利用 Elasticsearch+** **Log-Pilot+Kibana 搭建Kubernetes日志解决方案，Log-Pilot能同时获取容器标准输出日志和容器内部日志文件。如果对ELK中日志查询有更高的安全性要求，比如设置不同用户对日志索引的访问权限等，推荐安装X-Pack插件，日志告警除了通过X-Pack实现外也可通过ElastAlert实现。**
2. **存储：使用Ceph-Deploy工具部署Ceph集群，在Kubernetes上创建分别基于Ceph RBD(块存储,** **ReadWriteOnce), CephFS(文件存储,** **ReadWriteMany)的StorageClass，用来动态创建Persistentvolume和Persistentvolumeclaim。注意动态创建的PV其默认回收策略为Delete，若删除对应的PVC时，PV以及PV所指向的后端存储数据也将删除。尤其用helm install部署应用时一般会使用Storageclass自动生成PVC,而使用helm delete卸载应用时，默认也会删除部署时生成的PVC。所以生产环境，建议更改回收策略为Retain。**
3. **网络：常用CNI插件有Flannel，Calico等，推荐生产环境使用Calico网络插件，支持Network Policy来定义容器间的访问策略。**
4. **监控告警：**

**灰度发布**

**可读可写，但只支持被单个Pod挂载。**

**Linux运维工程师**

**1. 负责Linux系统上日志平台(ELK: filebeat, kafka, logstash, elasticsearch, kibana)，CI/CD平台(Jenkins)，监控告警(Zabbix)，消息队列(RabbitMQ)，配置中心(Apollo)，任务调度(XXL-JOB)，链路追踪(SkyWalking),缓存(Redis)等运维相关公共平台的设计，部署及维护工作。**