红帽OpenShift 4 资源和工具（8小时）

部署

在本模块中，你将了解到有关部署和DeploymentConfigs以及不同的部署策略和它们的配置。你还将了解蓝绿、金丝雀和A/B测试部署策略。

部署

应用程序的部署审查

- Pods 不是持久性的。
- 标签用于分组，选择Pods
- 用于启动/停止/重启Pod的ReplicaSets和ReplicationControllers
- 用于为一组Pod提供固定的IP和DNS名称的服务
- Deployments和DeploymentConfigs用于控制更新和重新部署应用程序。

部署组件

- 容器模板：
  - 容器名称
  - 容器镜像
  - 容器资源(请求和限制)
  - 存储卷
  - 健康检查（准备度和有效性）。
- 部署策略
  - 如何重新启动/重新创建部署
- 部署触发器
  - 何时重新启动/重新创建部署

创建部署

- 使用YAML清单
- 搜索实例
- 定义元数据：名称和标签
  - 至少要有`app: _appname_`
- 定义部署的`spec`
  - 复制的数量
  - 选择器（`matchLabels`）：哪些Pod属于Deployment
- 定义容器模板
  - 镜像、名称、标签、端口、资源

检查状态

- `oc describe deployment NAME`
- `oc status`

- `oc get all`

扩展部署

- `oc scale deployment NAME --replicas=NUMBER`
- `oc rollout status deployment NAME`

应用更新

- 要在镜像更新时更新部署：
  - `oc edit deployment NAME` → 改变容器镜像标签
  - 或者 `oc set image deployment NAME CONTAINER=NEW_IMAGE` → 指向新的镜像标签
- 这个动作：
  - 创建新的ReplicaSet
  - 用新的镜像来扩大它的规模
  - 当新的容器准备好时，缩小旧的ReplicaSet的规模

DeploymentConfigs

- `oc new-app` 命令执行时 *DeploymentConfig* 会被创建
  - *ReplicationController* 也会被创建
- DeploymentConfig 会创建 `-deploy` Pod 去执行部署流程
- DeploymentConfig 的修订版本是可以跟踪的
- 如果新版本出现错误 DeploymentConfig 会自动回滚到最近的可用版本

参考

- 比较 Deployments 和 DeploymentConfigs

部署策略

滚动 是默认的

- 创建新的ReplicationController（RC）。
- 执行金丝雀策略
  - 用新的镜像将其扩展到1
  - 将当前的RC缩减1个副本
  - 持续到当前RC有0个副本为止
  - 唯一活着的副本是新的镜像

  当应用程序不支持滚动更新时使用*Recreate*策略
- 终止所有正在运行的Pod
- 创建新的RC，用新的镜像启动Pod
- 可以在终止旧Pod之前和之后执行额外的步骤
  - 也可以在启动新的Pod之后

蓝绿部署策略

- 有时你需要更多的测试来确保新版本的正常运行
- 你希望新的部署与当前的部署并行运行
- 你切换到新版本，但不自动删除旧版本

- 为此使用*Route*：
    - 在后端服务之间进行切换
    - 只有在确定新的部署是好的时候才删除旧的部署

金丝雀部署策略

- 使用*Route*在新旧版本之间分割应用程序流量
- 只有一小部分用户被发送到新版本
- 应用程序需要支持同时运行两个不同的版本

A-B测试策略

- 例子：测试两种不同风格的首页，并测量用户的反应
- 你同时运行两个版本，以50/50的比例发送流量
- 在技术上可以运行两个以上的版本
    - 使用两个以上的支持服务
- 应用程序需要支持同时运行两个不同的版本

总结

- 部署
- DeploymentConfigs
- 部署策略