第 2 章 基于 Jenkins 和 k8s 构建企业级 DevOps 容器云平台

本节所讲内容：

2.1 什么是 DevOps?

2.2 k8s 在 DevOps 中可实现的功能?

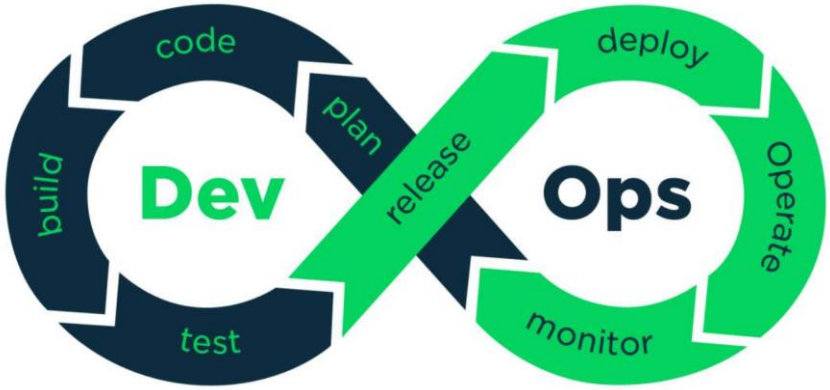
2.3 Jenkins+K8S+harbor+git+sonarqube+nexus 构建 DevOps 容器云平台流程

实战 1：在 k8s 集群安装 jenkins

实战 2：jenkins 安装 kubernetes 插件

2.1 什么是 DevOps?

DevOps 中的 Dev 是 Devlopment（开发），Ops 是 Operation（运维），用一句话来说 DevOps 就是打通开发运维的壁垒，实现开发运维一体化。DevOps 整个流程包括敏捷开发->持续集成->持续交付->持续部署。



2.1.1 敏捷开发

提高开发效率，及时跟进用户需求，缩短开发周期。

敏捷开发包括编写代码和构建代码两个阶段，可以使用 git 或者 svn 来管理代码，用 maven 对代码进行构建

2.1.2 持续集成（CI）

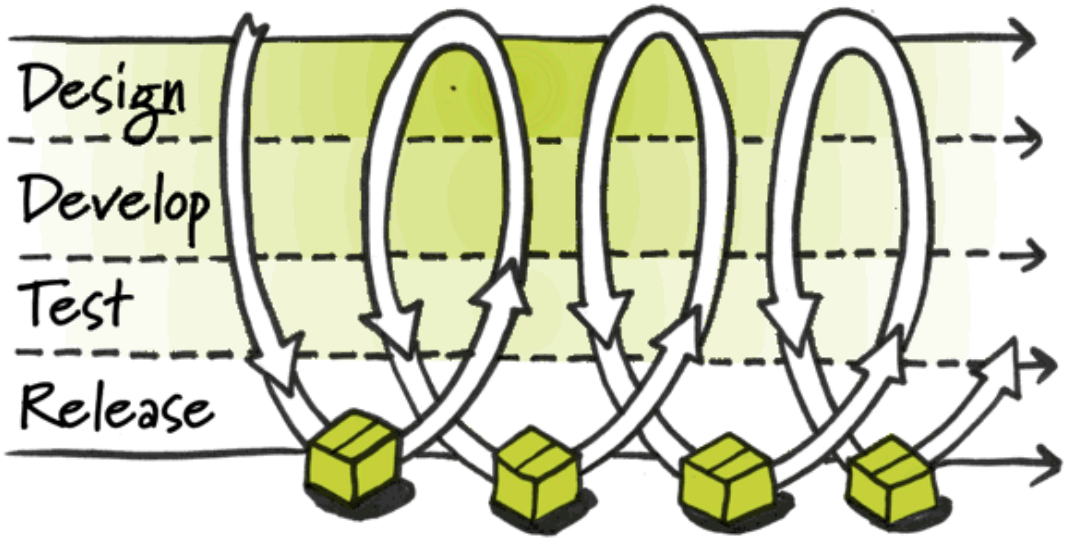
持续集成强调开发人员提交了新代码之后，立刻自动的进行构建、（单元）测试。根据测试结果，我们可以确定新代码和原有代码能否正确地集成在一起。持续集成过程中很重视自动化测试验证结果，对可能出现的一些问题进行预警，以保障最终合并的代码没有问题。

常见的持续集成工具：

1. Jenkins

Jenkins 是用 Java 语言编写的，是目前使用最多和最受欢迎的持续集成工具，使用 Jenkins，可以自动监测到 git 或者 svn 存储库代码的更新，基于最新的代码进行构建，把构建好的源码或者镜像发布到生产环境。Jenkins 还有个非常好的功能：它可以在多台机器上进行分布式地构建和负载测试

2. TeamCity
3. Travis CI
4. Go CD
5. Bamboo
6. GitLab CI
7. Codeship



它的好处主要有以下几点：

1）较早的发现错误：每次集成都通过自动化的构建（包括编译，发布，自动化测试）来验证，哪个环节出现问题都可以较早的发现。

2）快速的发现错误：每完成一部分代码的更新，就会把代码集成到主干中，这样就可以快速的发现错误，比较容易的定位错误。

3）提升团队绩效：持续集成中代码更新速度快，能及时发现小问题并进行修改，使团队能创造出更好的产品。

4）防止分支过多的偏离主干：经常持续集成，会使分支代码经常向主干更新，当单元测试失败或者出现 bug，如果开发者需要在没有调试的情况下恢复仓库的代码到没有 bug 的状态，只有很小部分的代码会丢失。

持续集成的目的是提高代码质量，让产品快速的更新迭代。它的核心措施是，代码集成到主干之前，必须通过自动化测试。只要有一个测试用例失败，就不能集成。

Martin Fowler 说过，"持续集成并不能消除 Bug，而是让它们非常容易发现和改正。"

互动：Martin Fowler 是谁?

马丁•福勒

马丁•福勒是一个软件开发方面的著作者和国际知名演说家，专注于面向对象分析与设计，统一建模语言，领域建模，以及敏捷软件开发方法，包括极限编程。

与持续集成相关的还有持续交付和持续部署。

2.1.3 持续交付

持续交付在持续集成的基础上，将集成后的代码部署到更贴近真实运行环境的「类生产环境」（production-like environments）中。交付给质量团队或者用户，以供评审。如果评审通过，代码就进入生产阶段。如果所有的代码完成之后一起交付，会导致很多问题爆发出来，解决起来很麻烦，所以持续集成，也就是每更新一次代码，都向下交付一次，这样可以及时发现问题，及时解决，防止问题大量堆积。

2.1.4 持续部署

持续部署是指当交付的代码通过评审之后，自动部署到生产环境中。持续部署是持续交付的最高阶段。Puppet，SaltStack 和 Ansible 是这个阶段使用的流行工具。容器化工具在部署阶段也发挥着重要作用。 Docker 和 k8s 是流行的工具，有助于在开发，测试和生产环境中实现一致性。 除此之外，k8s 还可以实现自动扩容缩容等功能。

2.2 k8s 在 DevOps 中可实现的功能

2.2.1 自动化

敏捷开发->持续集成->持续交付->持续部署。

2.2.2 多集群管理

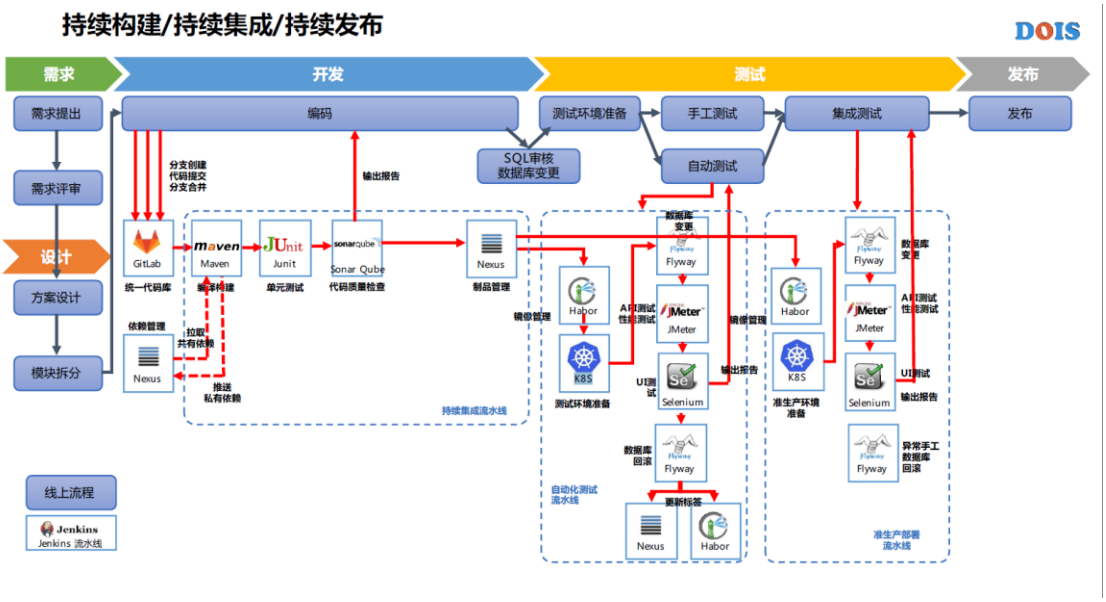
可以根据客户需求对开发，测试，生产环境部署多套 kubernetes 集群，每个环境使用独立的物理资源，相互之间避免影响。

2.2.3 多环境一致性

Kubernetes 是基于 docker 的容器编排工具，因为容器的镜像是不可变的，所以镜像把 OS、业务代码、运行环境、程序库、目录结构都包含在内，镜像保存在我们的私有仓库，只要用户从我们提供的私有仓库拉取镜像，就能保证环境的一致性。

2.2.4 实时反馈和智能化报表

每次集成或交付，都会第一时间将结果通过多途径的方式反馈给你，也可以定制适合企业专用的报表平台。

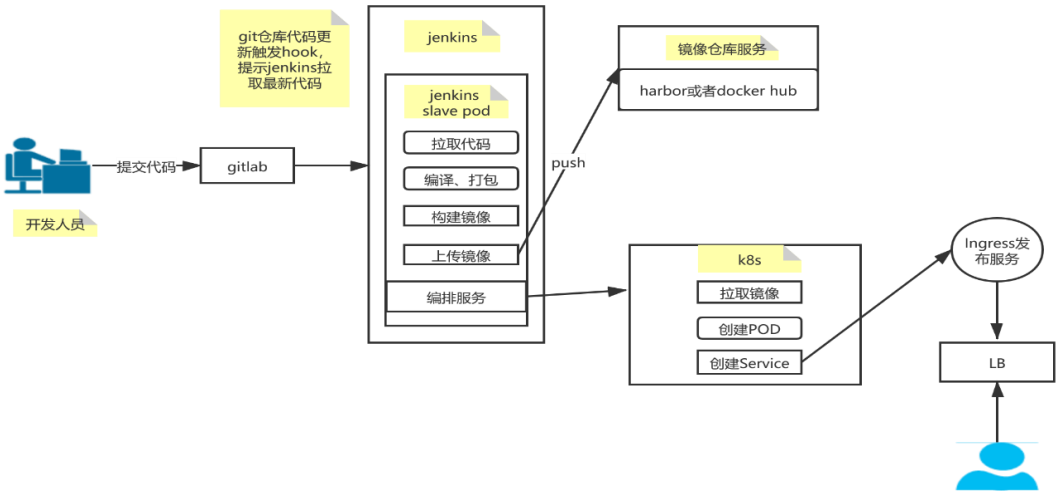


2.3 Jenkins+K8S+harbor+git+sonarqube+nexus 构建 DevOps 容器云平台流程

Jenkins 是开源的 CI&CD 工具， 提供超过 1000 个插件来支持构建、部署、自动化， 满足任何项目的需要。完整的 DevOps 流程：

开发提交代码到代码仓库 gitlab-→jenkins 检测到代码更新-→调用 k8s api 在 k8s 中创建 jenkins slave pod：

Jenkins slave pod 拉取代码---→通过 maven 把拉取的代码进行构建成 war 包或者 jar 包--->上传代码到 Sonarqube，进行静态代码扫描- -->基于 war 包构建 docker image-->把镜像上传到 harbor 镜像仓库-->基于镜像部署应用到开发环境-->部署应用到测试环境--->部署应用到生产环境。



实战 1：基于 Jenkins+k8s+Git 等技术链构建企业级 DevOps 自动化容器云平台

1、安装 Jenkins

可用如下两种方法

1）通过 docker 直接下载 jenkins 镜像，基于镜像启动服务
2）在 k8s 中部署 Jenkins 服务
2、安装 nfs 服务
#选择自己的任意一台机器，我选择 k8s 的控制节点 xuegod63
1）、在 xuegod63 上安装 nfs，作为服务端
[root@xuegod63 ~]# yum install nfs-utils -y
[root@xuegod63 ~]# systemctl start nfs
[root@xuegod63 ~]# systemctl enable nfs
2）、在 xuegod64 上安装 nfs，作为客户端
yum install nfs-utils -y
systemctl start nfs
systemctl enable nfs
3）、在 xuegod63 上创建一个 nfs 共享目录
[root@xuegod63 ~]# mkdir /data/v1 -p
[root@xuegod63 ~]# cat /etc/exports
/data/v1 192.168.1.0/24(rw,no_root_squash)
#使配置文件生效
[root@xuegod63 ~]# exportfs -arv
[root@xuegod63 ~]# systemctl restart nfs
3、在 kubernetes 中部署 jenkins
（1）创建名称空间
[root@xuegod63 ~]# kubectl create namespace jenkins-k8s
（2）创建 pv
[root@xuegod63 ~]# cat pv.yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: jenkins-k8s-pv
spec:
  capacity:
    storage: 10Gi
  accessModes:
  - ReadWriteMany
  nfs:
    server: 192.168.1.63
    path: /data/v1
#跟新资源清单文件
[root@xuegod63 ~]# kubectl apply -f pv.yaml
（3）创建 pvc
[root@xuegod63 ~]# cat pvc.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: jenkins-k8s-pvc

```
  namespace: jenkins-k8s
spec:
  resources:
    requests:
      storage: 10Gi
  accessModes:
  - ReadWriteMany
#更新资源清单文件
[root@xuegod63 ~]# kubectl apply -f pvc.yaml
#查看 pvc 和 pv 绑定是否成功
[root@xuegod63 ~]# kubectl get pvc -n jenkins-k8s
NAME          STATUS    VOLUME          CAPACITY    ACCESS MODES    STORAGECLASS
jenkins-k8s-pvc    Bound     jenkins-k8s-pv    10Gi          RWX
```

（4）创建一个 sa 账号

```
[root@xuegod63 ~]# kubectl create sa jenkins-k8s-sa -n jenkins-k8s
```

（5）把上面的 sa 账号做 rbac 授权

```
[root@xuegod63 ~]# kubectl create clusterrolebinding jenkins-k8s-sa-cluster -n
jenkins-k8s   --clusterrole=cluster-admin --serviceaccount=jenkins-k8s:jenkins-k8s-sa

#参考：
https://kubernetes.io/zh/docs/reference/access-authn-authz/rbac/
```

（6）通过 deployment 部署 jenkins

注：下面实验用到的镜像是 jenkins.tar.gz 和 jenkins-jnlp.tar.gz，把这两个压缩包上传到 k8s 的 xuegod64 节点，用如下方法手动解压：

```
docker load -i jenkins.tar.gz
docker load -i jenkins-jnlp.tar.gz
[root@xuegod63 ~]# cat jenkins-deployment.yaml
kind: Deployment
apiVersion: apps/v1
metadata:
  name: jenkins
  namespace: jenkins-k8s
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jenkins
  template:
    metadata:
      labels:
        app: jenkins
    spec:
      serviceAccount: jenkins-k8s-sa
```

```
      containers:
      - name: jenkins
        image:    xuegod/jenkins:v1
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 8080
          name: web
          protocol: TCP
        - containerPort: 50000
          name: agent
          protocol: TCP
        resources:
          limits:
            cpu: 1000m
            memory: 1Gi
          requests:
            cpu: 500m
            memory: 512Mi
        livenessProbe:
          httpGet:
            path: /login
            port: 8080
          initialDelaySeconds: 60
          timeoutSeconds: 5
          failureThreshold: 12
        readinessProbe:
          httpGet:
            path: /login
            port: 8080
          initialDelaySeconds: 60
          timeoutSeconds: 5
          failureThreshold: 12
        volumeMounts:
        - name: jenkins-volume
          subPath: jenkins-home
          mountPath: /var/jenkins_home
      volumes:
      - name: jenkins-volume
        persistentVolumeClaim:
          claimName: jenkins-k8s-pvc
#更新资源清单文件
kubectl apply -f jenkins-deployment.yaml
[root@xuegod63 ~]# kubectl get pods -n jenkins-k8s
NAME                          READY   STATUS              RESTARTS   AGE
```

```
jenkins-74b4c59549-qvmrt    0/1     CrashLoopBackOff    1          15s
#上面可以看到 CrashLoopBackOff，解决方法如下：
#查看 jenkins-74b4c59549-qvmrt 日志
[root@xuegod63 ~]# kubectl logs jenkins-74b4c59549-qvmrt -n jenkins-k8s
touch: cannot touch '/var/jenkins_home/copy_reference_file.log': Permission denied
#上面问题是因为/data/v1 目录权限问题，按如下方法解决：
[root@xuegod63 ~]# chown -R 1000.1000 /data/v1/
[root@xuegod63 ~]# kubectl delete -f jenkins-deployment.yaml
[root@xuegod63 ~]# kubectl apply -f jenkins-deployment.yaml
[root@xuegod63 ~]# kubectl get pods -n jenkins-k8s
NAME                            READY   STATUS    RESTARTS   AGE
jenkins-74b4c59549-gp95l    1/1     Running   0          16s
```

（7）把 jenkins 前端加上 service，提供外部网络访问

```
[root@xuegod63 ~]# cat jenkins-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: jenkins-service
  namespace: jenkins-k8s
  labels:
    app: jenkins
spec:
  selector:
    app: jenkins
  type: NodePort
  ports:
  - name: web
    port: 8080
    targetPort: web
    nodePort: 30002
  - name: agent
    port: 50000
    targetPort: agent

#更新资源清单文件
[root@xuegod63 ~]# kubectl apply -f jenkins-service.yaml
[root@xuegod63 ~]# kubectl get svc -n jenkins-k8s
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)
jenkins-service   NodePort   10.110.34.67   <none>
8080:30002/TCP,50000:32331/TCP
```

4、配置 Jenkins

在浏览器访问 jenkins 的 web 界面：

http://192.168.1.63:30002/login?from=%2F

入门

1）获取管理员密码

在 nfs 服务端，也就是我们的 xuegod63 节点获取密码：

[root@xuegod63 ~]# cat /data/v1/jenkins-home/secrets/initialAdminPassword

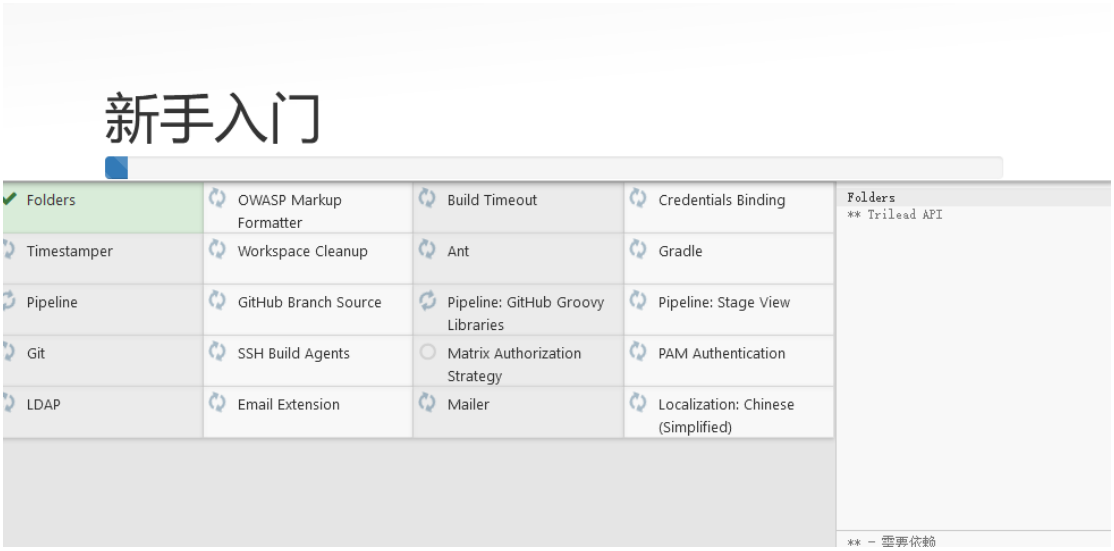
f9b0b4400c4a4d6eaec6762616db6d63

把获取到的密码拷贝到上面管理员密码下的方框里

点击继续，出现如下界面

2）安装插件

安装推荐的插件



离线安装 jenkins 插件：

https://plugins.jenkins.io/

插件安装好之后显示如下

3）创建第一个管理员用户

用户名和密码都设置成 admin，线上环境需要设置成复杂的密码

修改好之后点击保存并完成，出现如下界面

点击保存并完成，出现如下界面

点击保存并完成，出现如下界面

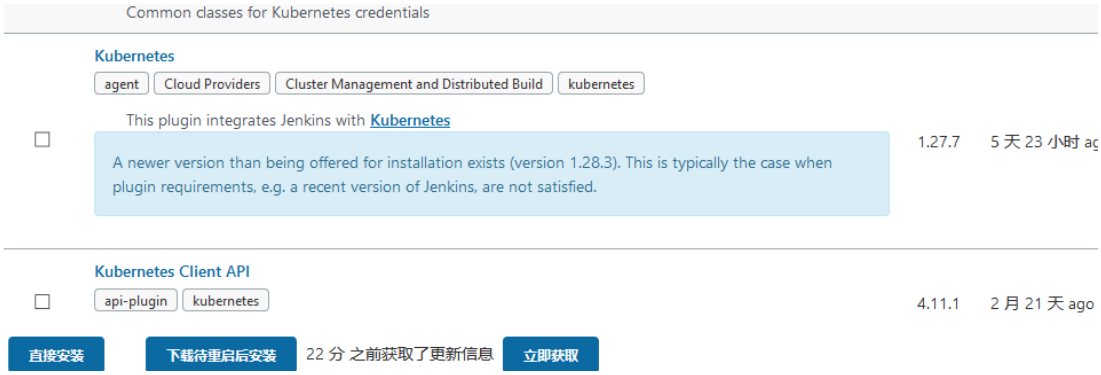
点击开始使用 Jenkins

5、测试 jenkins 的 CI/CD

#在 Jenkins 中安装 kubernetes 插件

（1）在 jenkins 中安装 k8s 插件

Manage Jnekins------>插件管理------>可选插件------>搜索 kubernetes------>出现如下



选中 kubernetes 之后------>点击下面的直接安装------>安装之后选择重新启动 jenkins--->http://192.168.1.63:30002/restart-->重启之后登陆 jenkins 即可

总结：

2.1 什么是 DevOps?

2.2 k8s 在 DevOps 中可实现的功能?

2.3 Jenkins+K8S+harbor+git+sonarqube+nexus 构建 DevOps 容器云平台流程

实战 1：在 k8s 集群安装 jenkins

实战 2：jenkins 安装 kubernetes 插件