

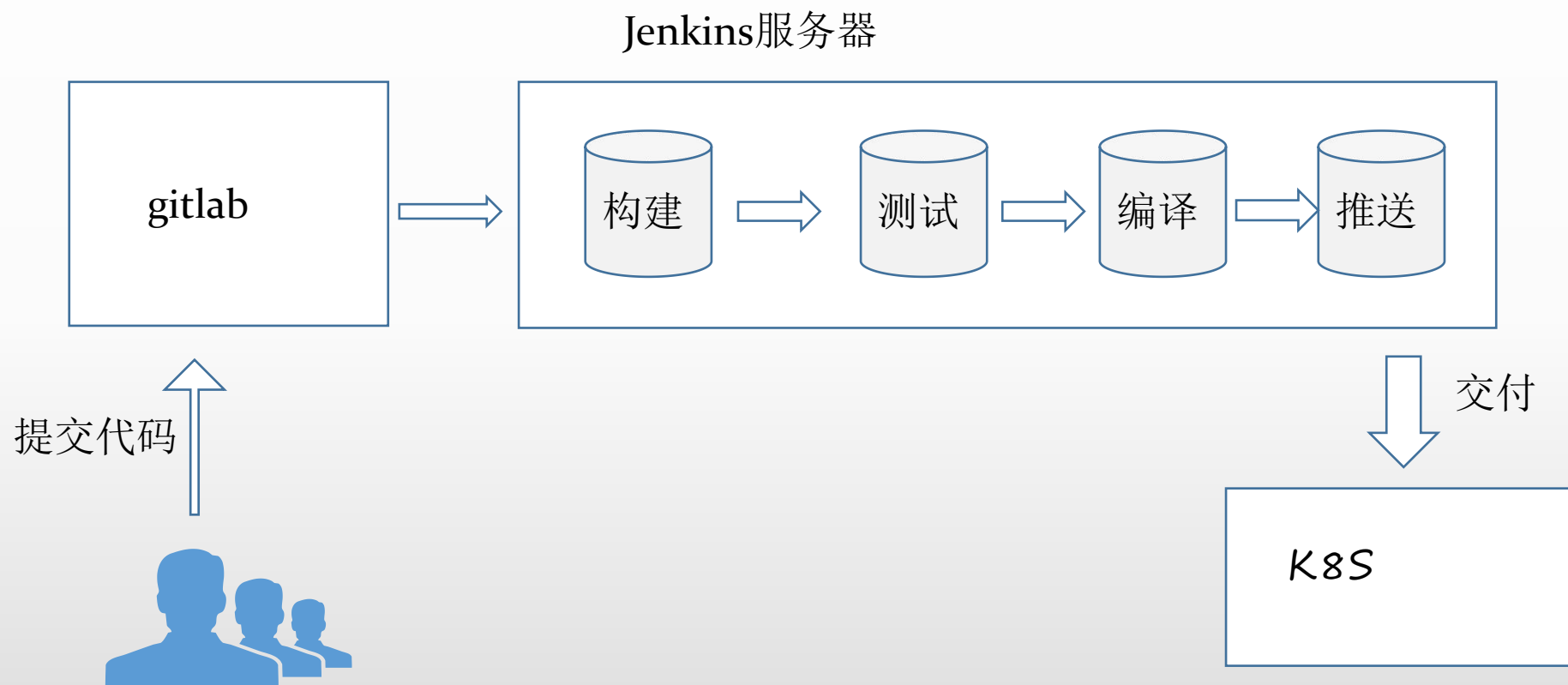
CKA-CI/CD/devops

讲师：老段 RHCE/RHCA/COA/CKA

devops的意义

- 介绍devops
- 什么是可持续集成CI
- 什么是可持续交付CD
- 什么是可持续部署CD

使用gitlab+Jenkins+k8s建立CI/CD解决方案

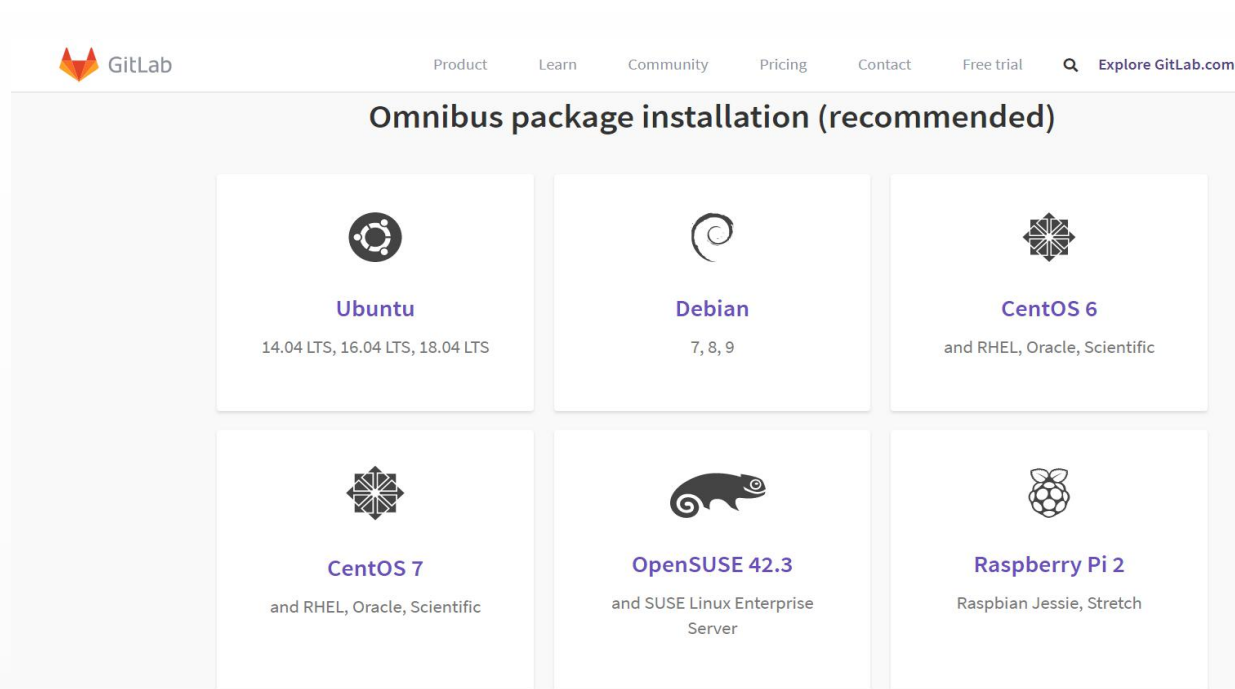


gitlab安装

<https://about.gitlab.com/installation/>

下载地址

<https://packages.gitlab.com/gitlab/gitlab-ce/>



配置yum源

```
curl https://packages.gitlab.com/install/repositories/gitlab/gitlab-ce/script.rpm.sh | sudo bash
```

安装

```
yum install gitlab-ce-11.6.2-ce.o.el7.x86_64
```

练习用离线包

```
vim /etc/gitlab/gitlab.rb  
13 external_url 'http://192.168.26.20'
```

gitlab-ctl reconfigure

(gitlab-ctl restart或reboot)



Thank you for installing GitLab!
GitLab was unable to detect a valid hostname for your instance.
Please configure a URL for your GitLab instance by setting `external_url`
configuration in /etc/gitlab/gitlab.rb file.
Then, you can start your GitLab instance by running the following command:

```
sudo gitlab-ctl reconfigure
```

For a comprehensive list of configuration options please see the Omnibus GitLab readme
<https://gitlab.com/gitlab-org/omnibus-gitlab/blob/master/README.md>

```
[root@vms20 ~]#
```

Recipe: gitlab::postgres-exporter

- * service[postgres-exporter] action restart
 - restart service service[postgres-exporter]
- * ruby_block[restart postgres-exporter svlogd configuration] action create
 - execute the ruby block restart postgres-exporter svlogd configuration
- * ruby_block[reload postgres-exporter svlogd configuration] action create
 - execute the ruby block reload postgres-exporter svlogd configuration

Running handlers:

Running handlers complete

Chef Client finished, 454/655 resources updated in 01 minutes 44 seconds

gitlab Reconfigured!

```
[root@vms20 ~]#
```



Please create a password for your new account.

GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

Change your password

New password

•••••

Confirm new password

•••••

Invalid Login or password.

GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

Sign in

Register

Username or email

root

Password

••••••••

☐ Remember me

[Forgot your password?](#)

Sign in

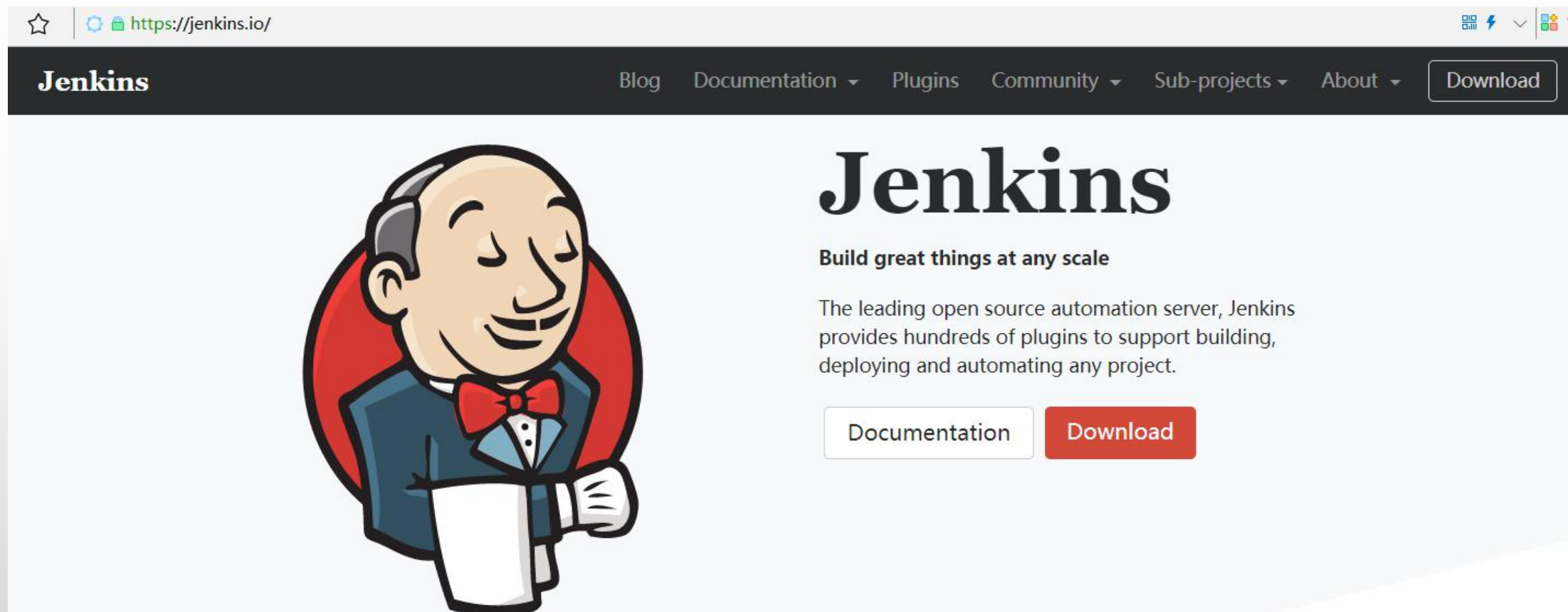
gitlab的操作

- web界面的操作

git命令的使用

- `git clone http://osp60.rhce.cc/root/dockerimage.git`
- `git clone http://osp60.rhce.cc/root/dockerimage.git /xxx`
- `git add .`
- `git commit 'xx'`
- `git push`

安装Jenkins



安装步骤

- `wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenkins.repo`
- `rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key`
- `yum install jenkins`
- 安装jdk
- `http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html`
- `/etc/init.d/jenkins start`
- `netstat -ntulp | grep 8080`

入门

解锁 Jenkins

为了确保管理员安全地安装 Jenkins，密码已写入到日志中 ([不知道在哪里?](#)) 该文件在服务器上：

```
/var/lib/jenkins/secrets/initialAdminPassword
```

请从本地复制密码并粘贴到下面。

管理员密码

 此连接不安全。输入的登录信息可能被窃取。 [详细了解](#)

继续

新手入门

自定义Jenkins

插件通过附加特性来扩展Jenkins以满足不同的需求。

安装推荐的插件

安装Jenkins社区推荐的插件。



选择插件来安装

选择并安装最适合的插件。



192.168.26.62:8080



搜索



新手入门

新手入门



Folders

OWASP Markup Formatter

Build Timeout

Credentials Binding

** JDK Tool
** Script Security

Timestamper

Workspace Cleanup

Ant

Gradle

Pipeline

GitHub Branch Source

Pipeline: GitHub Groovy Libraries

Pipeline: Stage View

Git

Subversion

SSH Slaves

Matrix Authorization Strategy

PAM Authentication

LDAP

Email Extension

Mailer

** - 需要依赖

Jenkins 2.140

新手入门

创建第一个管理员用户

用户名: 密码: 确认密码: 全名: 电子邮件地址:



Jenkins

查找



Jenkins

新建任务

用户

构建历史

系统管理

我的视图

凭据

新建视图

欢迎来到 Jenkins!

开始 [创建一个新任务](#).

构建队列

队列中没有构建任务


构建执行状态

1 空闲

2 空闲

192.168.26.62:8080/newJob

搜索


Jenkins


查找


输入一个任务名称


pl_test1


» 必填项


**构建一个自由风格的软件项目**
这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来构建你的项目,甚至可以构建软件以外的系统.

**流水线**
精心地组织一个可以长期运行在多个节点上的任务。适用于构建流水线（更加正式地应当称为工作流），增加或者组织难以采用自由风格的任务类型。

**构建一个多配置项目**
适用于多配置项目,例如多环境测试,平台指定构建,等等.

**GitHub Organization**
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

**多分支流水线**
根据一个SCM仓库中检测到的分支创建一系列流水线。

**文件夹**
创建一个可以嵌套存储的容器。利用它可以进行分组。视图仅仅是一个过滤器，而文件夹则是一个独立的命名空间，因此你可以有多个相同名称的内容，只要它们在不同的文件夹里即可。

确定

General构建触发器高级项目选项流水线

高级...

流水线

定义Pipeline script

脚本

1 node {

2 echo 'Hello World'

3 }

Hello World

☒ 使用 Groovy 沙盒

[流水线语法](#)

保存

应用

192.168.26.62:8080/job/pl_test1/

Jenkins

pl_test1

- 返回到工作台
- 状态
- 变更历史
- 立即构建**
- 删除 Pipeline
- 配置
- Full Stage View
- 重命名
- Pipeline Syntax

Pipeline pl_test1

hello world

最近变更

Stage View

This Pipeline has run successfully, but does not define any stages. Please use the `stage` step to define some stages in this Pipeline.

相关链接

Build History 构建历史

find

#1 2018-9-15 14:57

RSS 全部 RSS 失败

- 返回工作台
- 状态
- 变更历史
- 立即构建
- 删除 Pipeline
- 配置
- Full Stage View
- 重命名
- Pipeline Syntax

Build History 构建历史

find

#1 2019-1-5 上午1:44

变更历史

Console Output

编辑构建信息

删除本次生成

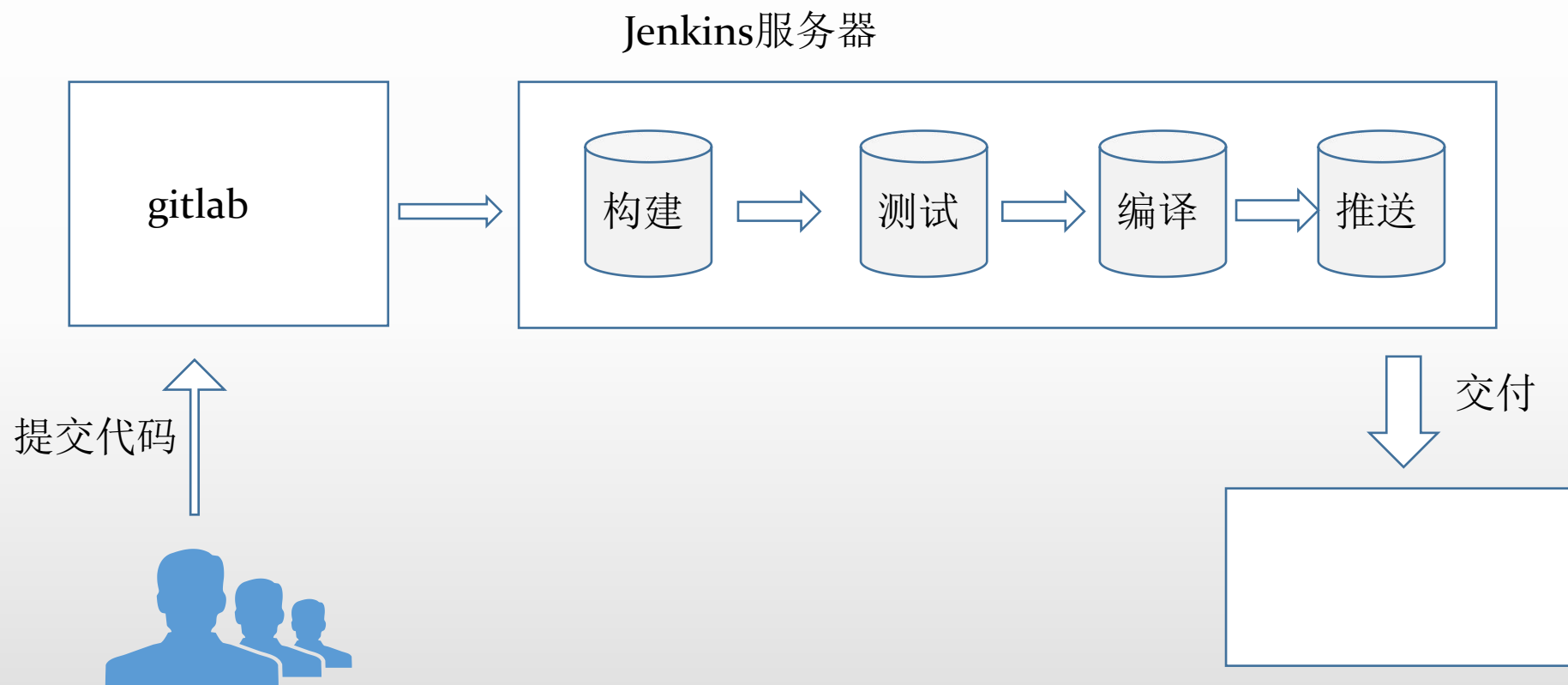
回放

Pipeline Steps

Workspaces

S 全部 RSS 失败

使用gitlab+Jenkins+k8s建立CI/CD解决方案



在gitlab上创建一个project, 类型为public

在客户端clone下来

创建一个Dockerfile和index.html文件

cat Dockerfile

FROM docker.io/nginx

MAINTAINER lduan

ADD index.html /usr/share/nginx/html/

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]

git add .

git commit -m 'xx'

git push

配置docker本地仓库

- `docker pull registry`
- `docker run -d --name registry -p 5000:5000 --restart=always -v /myreg:/var/lib/registry registry`
- 修改默认地址
`ADD_REGISTRY='--add-registry 192.168.26.70:5000'`

docker使用本地仓库

```
cat /etc/docker/daemon.json
```

```
{  
  "registry-mirrors": ["https://frz7i079.mirror.aliyuncs.com"],  
  "insecure-registries": ["192.168.26.70:5000"]  
}
```

```
curl http://192.168.26.70:5000/v2/_catalog
```

```
docker push 192.168.26.70:5000/rhce/busybox
```

```
docker pull 192.168.26.70:5000/rhce/busybox 或者
```

```
curl -XGET http://192.168.26.70:5000/v2/_catalog
```

```
curl -XGET http://192.168.26.70:5000/v2/rhce/nginx/tags/list
```

安装docker插件

- Jenkins---> 系统管理--->插件管理
- 安装所有和docker相关插件

<input checked="" type="checkbox"/>	Docker plugin This plugin integrates Jenkins with Docker	1.1.5	卸载
<input checked="" type="checkbox"/>	docker-build-step This plugin allows to add various docker commands to your job as build steps.	2.0	卸载
<input type="checkbox"/>	Durable Task Plugin Library offering an extension point for processes which can run outside of Jenkins yet be monitored.	1.25	卸载
<input checked="" type="checkbox"/>	Email Extension Plugin This plugin is a replacement for Jenkins's email publisher. It allows to configure every aspect of email notifications: when an email is sent, who should receive it and what the email says	2.63	卸载

配置gitlab和Jenkins联动

Jenkins的设置

系统管理-->系统设置



docker设置

vim /etc/sysconfig/docker

OPTIONS后增加-H tcp://0.0.0.0:2376 -H

unix:///var/run/docker.sock

Docker Builder

Docker URL:

Docker server REST API URL:

[高级...](#) [Test Connection](#)

跳到最下，
增加云

云

Docker

Name:

Docker Host URI:

Server credentials: - 无 - [Add](#)

[高级...](#) [Test Connection](#)

请仔细填写并验证，否则会出现错误

新建工程

输入一个任务名称

k8s1

» 必填项



构建一个自由风格的软件项目

这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来构建你的项目, 甚至可以构建软件以外的系统.



流水线

精心地组织一个可以长期运行在多个节点上的任务。适用于构建流水线（更加正式地应当称为工作流），增加或者组织难以采用自由风格的任务类型。

构建

执行 shell

命令

```
cd /zz/  
rm -rf /zz/*  
git clone http://192.168.26.20/root/nginx.git  
version=$(date +"%Y.%m.%d.%H.%M.%S")  
name=192.168.26.21:5000/cka/nginx:$version  
docker build -t $name nginx  
docker push $name
```

查看 [可用的环境变量列表](#)

高级...

mkdir /zz

chown jenkins.jenkins /zz

在编译过程中，如果出现

Got permission denied while trying to connect to the Docker daemon socket

在把jenkins加入到root组，同时重启jenkins

/etc/init.d/jenkins restart

配置gitlab和Jenkins联动

Jenkins的设置

系统管理-->全局安全设置

 **全局安全配置**

☒ 启用安全

不要记住我 ☐

访问控制

安全域

☒ Jenkins专用户数据库

☐ 允许用户注册

☐ LDAP

☐ Servlet容器代理

☐ Unix用户/组数据库

授权策略

☐ 任何用户可以做任何事(没有任何限制)

☐ 安全矩阵

☒ 登录用户可以做任何事

☒ 匿名用户具有可读权限

☐ 遗留模式

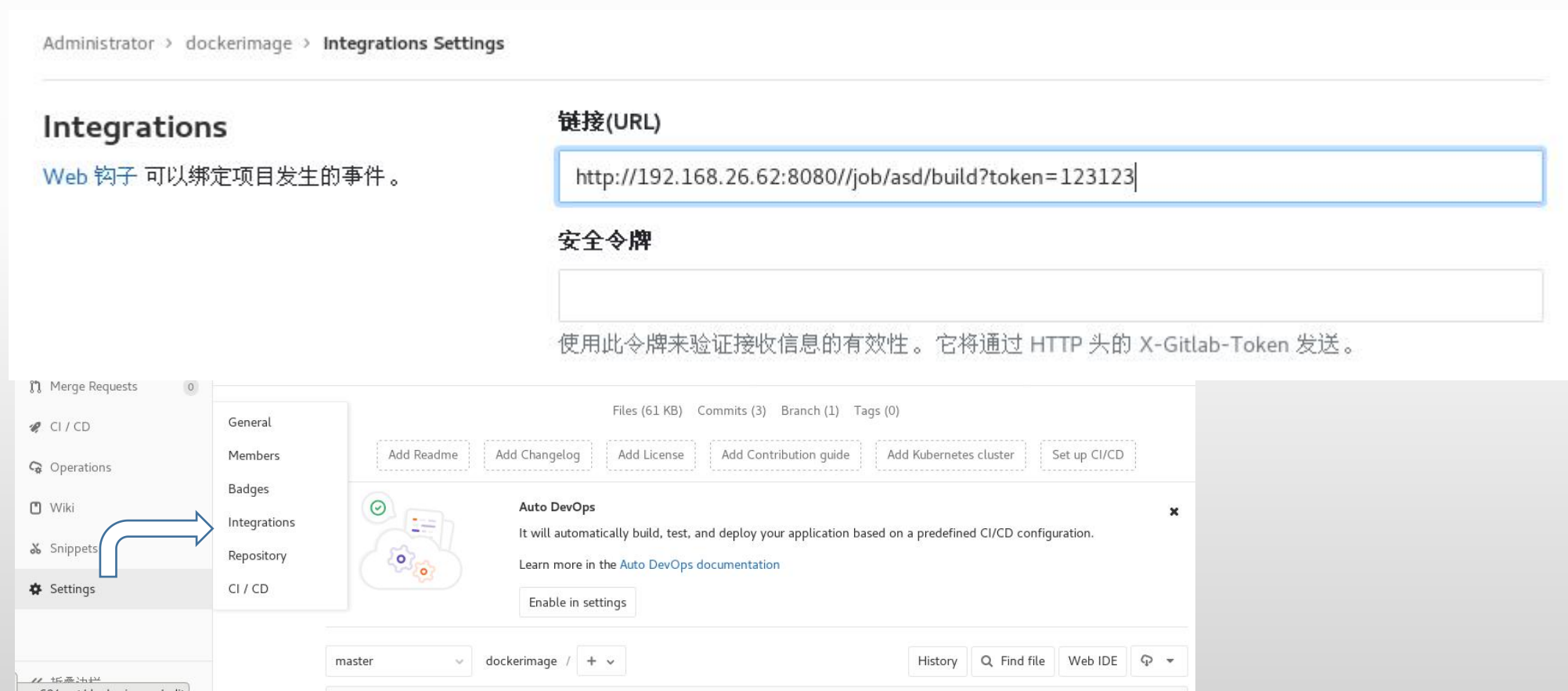
☐ 项目矩阵授权策略

跨站请求伪造保护

☐ 防止跨站点请求伪造

创建Jenkins工程和gitlab工程关联

- 在gitlab上设置



Integrations

Web 钩子 可以绑定项目发生的事件。

链接(URL)

<http://192.168.26.62:8080//job/asd/build?token=123123>

安全令牌

使用此令牌来验证接收信息的有效性。它将通过 HTTP 头的 X-Gitlab-Token 发送。

触发器



推送事件

此链接将在推送到版本仓库时触发



标签推送事件

此链接将在推送新标签到版本仓库时触发

SSL 证书验证



开启 SSL 证书验证

增加 Web 钩子



Web 钩子 (1)

<http://192.168.26.62:8080//job/asd/build?token=123123>

Push Events

SSL Verification: enabled

Edit

Test ▼



创建Jenkins工程和gitlab工程关联

在Jenkins工程里创建构建触发器

源码管理

☒ 无
☐ Git
☐ Subversion

构建触发器

☒ 触发远程构建 (例如,使用脚本)

身份验证令牌

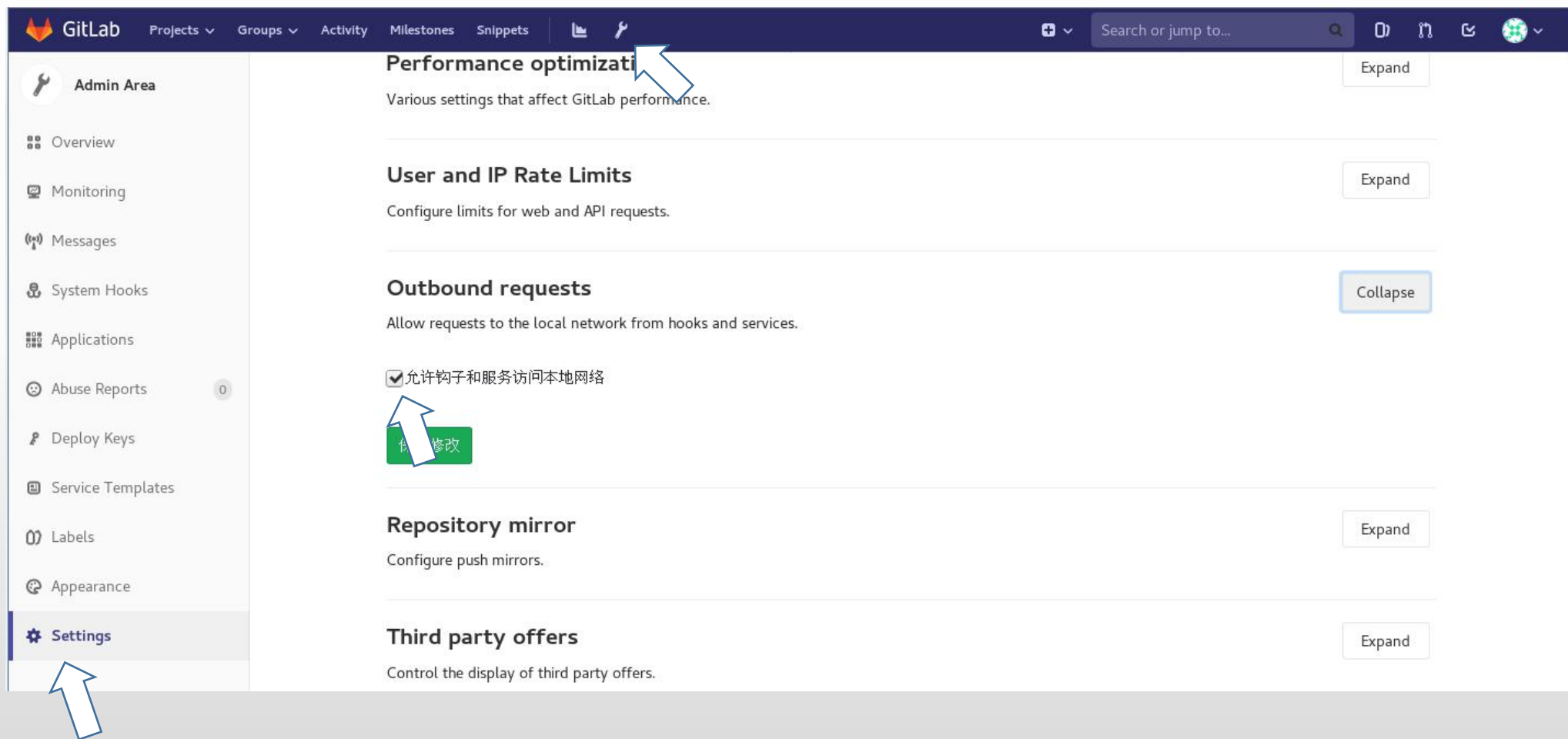
Use the following URL to trigger build remotely: JENKINS_URL/job/asd/build?token=TOKEN_NAME 或者
/buildWithParameters?token=TOKEN_NAME
Optionally append &cause=Cause+Text to provide text that will be included in the recorded build cause.

☐ GitHub hook trigger for GITScm polling
☐ 其他工程构建后触发
☐ 定时构建
☐ 轮询 SCM

此处身份令牌可以随便写

注意下面圈起来的部分，这部分是要填写到gitlab里的

如遇: url is blocked:requests to local



测试提交代码自动触发jenkins

- git clone <http://osp60/root/dockerimage.git>
- cd dockerimage/

```
[root@osp60 dockerimage]# ls  
aa.txt  Dockerfile  epel.repo  
[root@osp60 dockerimage]#
```

- git add .
- git commit -m 'xx'
- git push

部署CI/CD

在k8s上运行一个deployment nginx，并创建服务

为tom设置权限

```
kubectl create clusterrolebinding myclusterbind1 --clusterrole=cluster-admin --user=tom
```

在客户端上测试

```
kubectl -s="https://192.168.26.51:6443" --insecure-skip-tls-verify=true --username="tom" --password="redhat" get pods -n ns01
```

设置权限

```
cat /etc/sudoers.d/jenkins
```

```
jenkins  ALL=(root) NOPASSWD: /usr/bin/rm
```

构建的shell

```
cd /zz/
```

```
sudo rm -rf /zz/*
```

```
git clone http://192.168.26.20/root/nginx.git
```

```
version=$(date +"%Y.%m.%d.%H.%M.%S")
```

```
name=192.168.26.21:5000/cka/nginx:$version
```

```
docker build -t $name nginx
```

```
docker push $name
```

```
kubectl -s="https://192.168.26.51:6443" --insecure-skip-tls-verify=true --username="tom" --
```

```
password="redhat" set image deployment/nginx nginx="$name" -n ns01
```

