

CI CD 详解

1、 安装 jenkins

由于 jenkins.war,由 java 编写 , 需要配置 jdk 的环境 , jdk 的版本要求 1.8

1、 配置 JDK 环境

```
# rz //上传 jdk-8u211-linux-x64.tar.gz
# tar xf jdk-8u211-linux-x64.tar.gz -C /usr/local
# cd /usr/local/
# mv jdk1.8.0_211/ java1.8/
# vim /etc/profile.d/java.sh
(添加以下内容)
#!/bin/bash
JAVA_HOME=/usr/local/java1.8
PATH=$JAVA_HOME/bin:$PATH
export JAVA_HOME PATH

# source /etc/profile.d/java.sh
```

验证 :

```
[root@QF local]# java -version
java version "1.8.0_211"
Java(TM) SE Runtime Environment (build 1.8.0_211-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.211-b12, mixed mode)
[root@QF local]#
```

2、 配置 tomcat , 启动 jenkins 服务

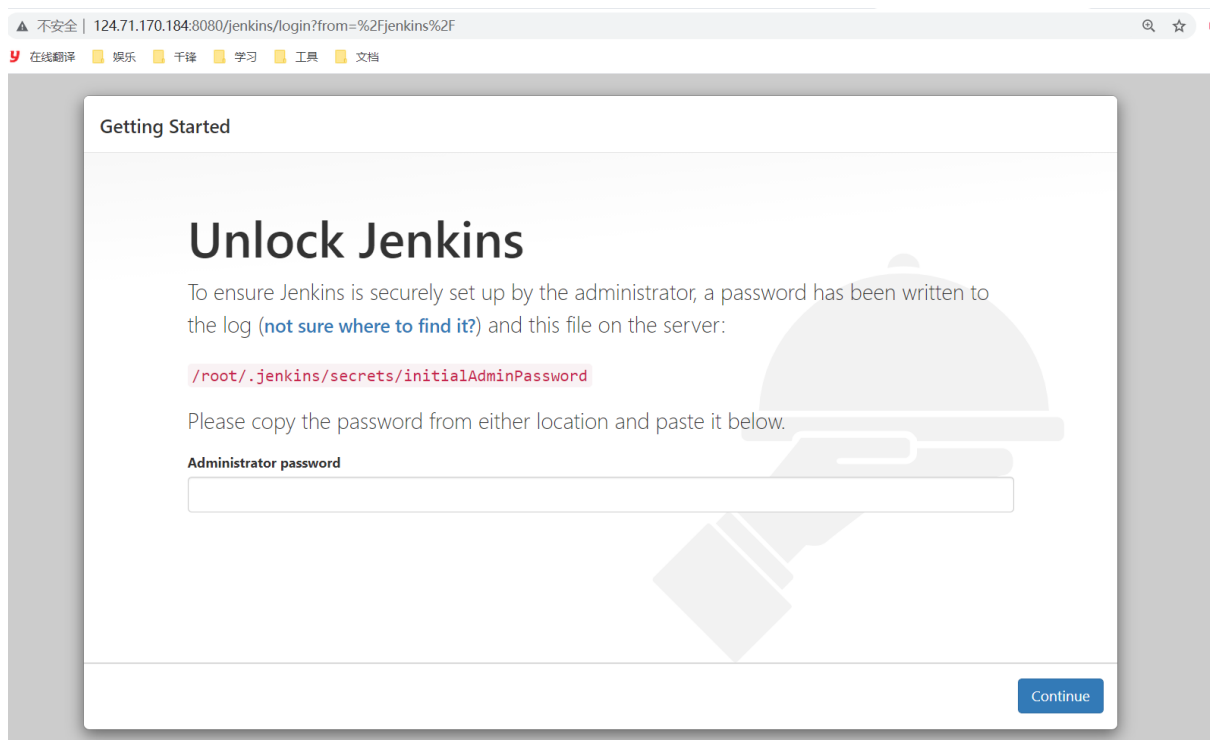
```
# rz //上传 apache-tomcat-7.0.94.tar.gz
# tar xf apache-tomcat-7.0.94.tar.gz -C /usr/local
# cd /usr/local
# mv apache-tomcat-7.0.94/ tomcat7

# rz //上传 jenkins.war
```

```
# cp jenkins.war /usr/local/tomcat7/webapps/  
# /usr/local/tomcat7/bin/shutdown.sh  
# /usr/local/tomcat7/bin/startup.sh  
//重启 tomcat 会自动帮 jenkins.war 进行解压  
# netstat -tnlp          //查看 8080 端口是否运行
```

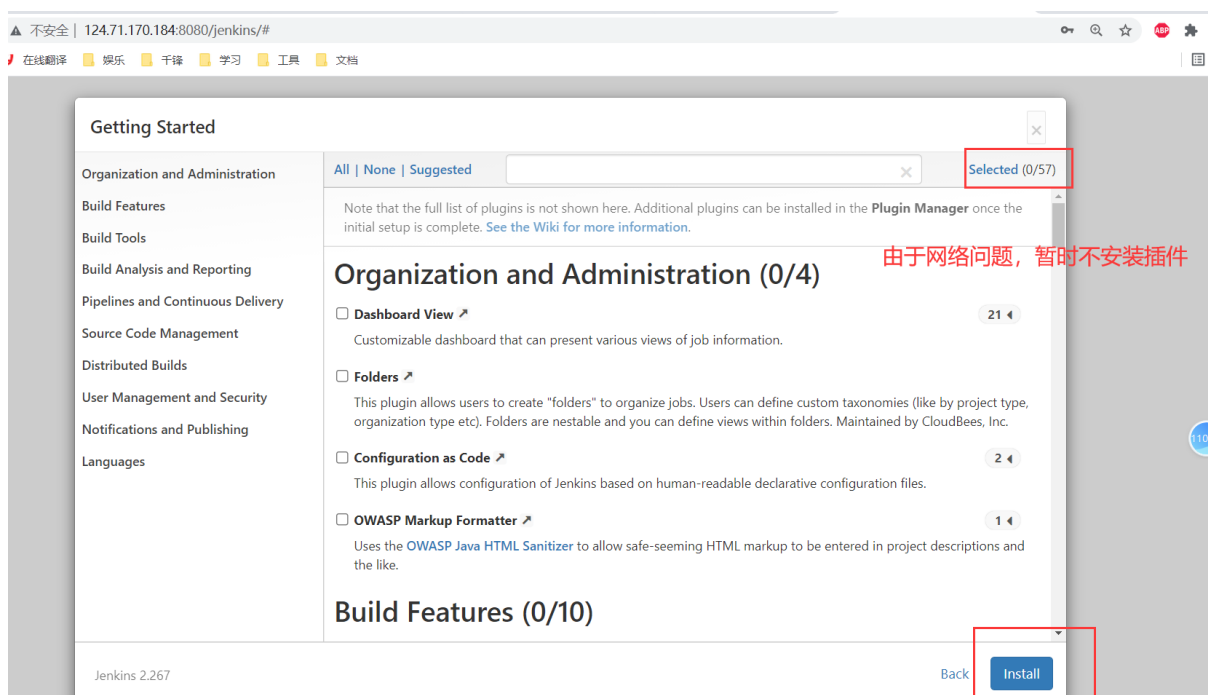
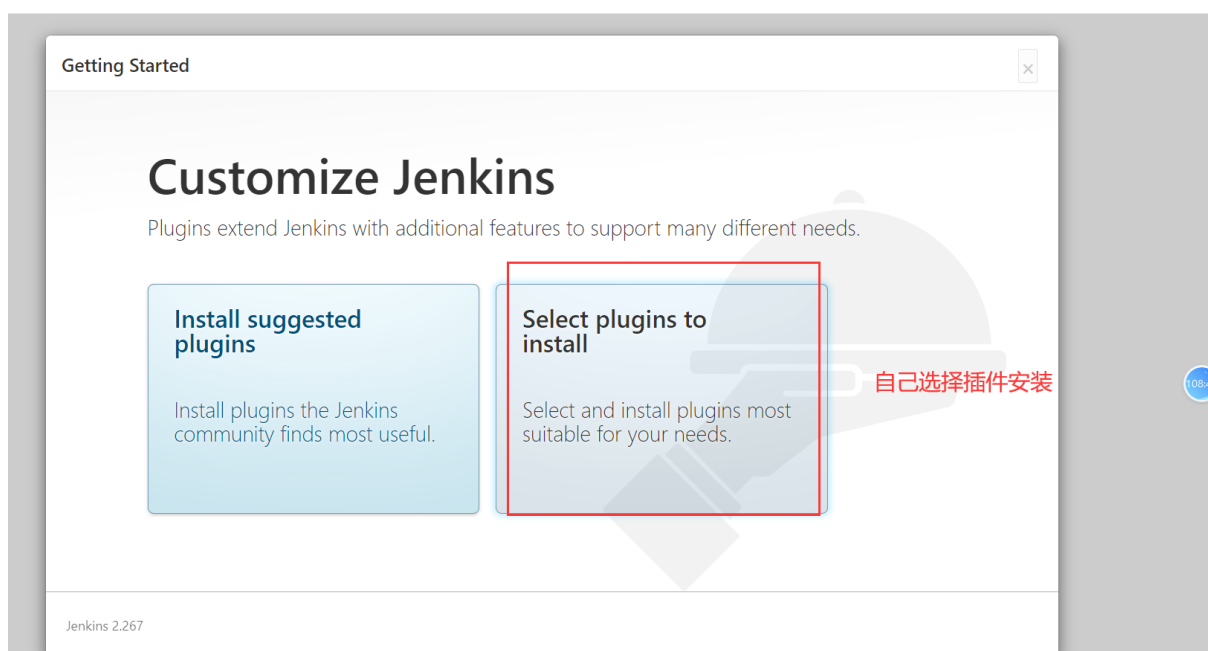
3、配置安装 jenkins

通过浏览器访问 **ip:8080/jenkins**



要求输入密码：

```
# cat /root/.jenkins/secrets/initialAdminPassword  
//把查看出来的密码输入到上面的框中，点击继续
```



▲ 不安全 | 124.71.170.184:8080/jenkins/#

在线翻译 娱乐 千锋 学习 工具 文档

Getting Started

Create First Admin User

Username:

Password:

Confirm password:

Full name:

Jenkins 2.267

[Skip and continue as admin](#) [Save and Continue](#)

然后就一直点下一步

安装插件，这里提供了插件的集成压缩包

```
# rz //上传 plugins2.267.tar.gz
# rm -rf /root/.jenkins/plugins
# tar xf plugins2.267.tar.gz -C /root/.jenkins/
```

▲ 不安全 | 124.71.170.184:8080/jenkins/restart

在线翻译 娱乐 千锋 学习 工具 文档

Jenkins

search

Are you sure you want to restart Jenkins?

[Yes](#)

进行jenkins重启，重新加载插件

设置jenkins 中文界面

应用 百度 在线翻译 娱乐 千锋 学习 工具 文档

Dashboard

- New Item
- People
- Build History
- Manage Jenkins**
- My Views
- Lockable Resources
- New View

Build Queue

No builds in the queue.

Build Executor Status

- Idle
- Idle

124.71.170.184:8080/jenkins/manage

Manage Jenkins

New version of Jenkins (2.311) is available for download (changelog).

⚠ Your container doesn't use UTF-8 to decode URLs. If you use non-ASCII characters as a job name etc, this will cause problems. See Containers and Tomcat i18n for more details.

Warnings have been published for the following currently installed components: [Go to plugin manager](#) [Configure which of these warnings are shown](#)




Jenkins 2.267 core and libraries:

- Denial of service vulnerability in bundled Jetty
- Multiple security vulnerabilities in Jenkins 2.274 and earlier, LTS 2.263.1 and earlier
- Multiple security vulnerabilities in Jenkins 2.286 and earlier, LTS 2.277.1 and earlier
- Privilege escalation vulnerability in bundled Spring Security library
- Multiple security vulnerabilities in Jenkins 2.299 and earlier, LTS 2.289.1 and earlier

Credentials Plugin 2.3.14:
Reflected XSS vulnerability

Matrix Authorization Strategy Plugin 2.6.4:
Incorrect permission checks may allow accessing some items

System Configuration

-  **Configure System**
Configure global settings and paths.
-  **Global Tool Configuration**
Configure tools, their locations and automatic installers.
-  **Manage Plugins**
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
🔴 There are updates available

☐ Environment variables

☐ Tool Locations

Locale

Default Language

zh_CN

☒ Ignore browser preference and force this language to all users

2、 安装 gitlab

新建

```
# vim /etc/yum.repos.d/gitlab-ce.repo
```

内容为

```
[gitlab-ce]
name=Gitlab CE Repository
baseurl=https://mirrors.tuna.tsinghua.edu.cn/gitlab-ce/yum/el$releasever/
gpgcheck=0
enabled=1
```

再执行

```
# yum install gitlab-ce -y
```

配置 gitlab

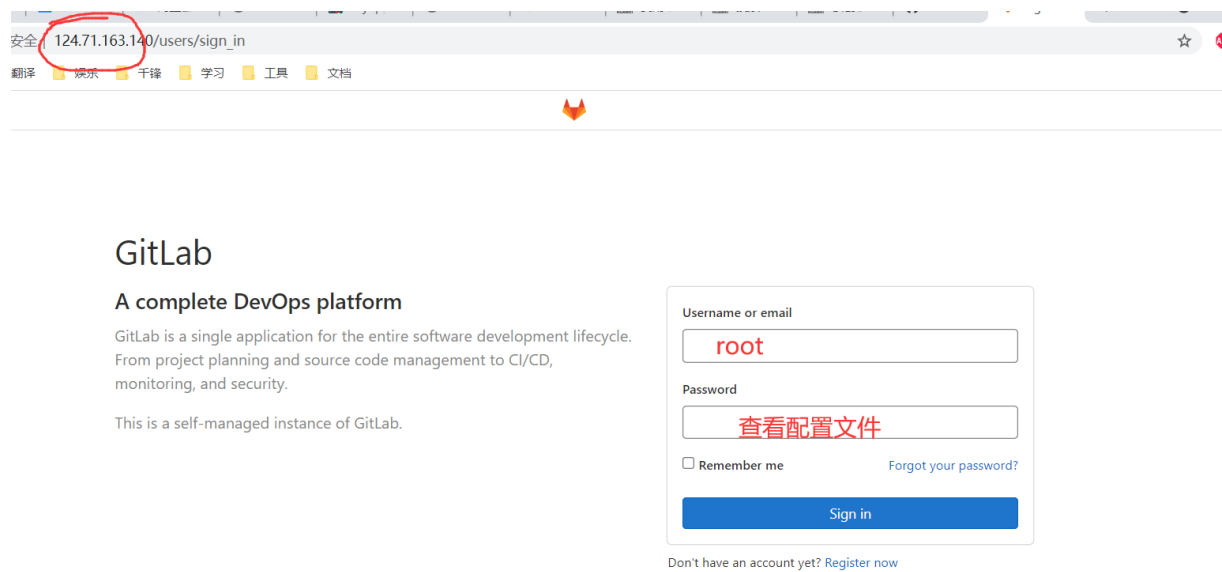
```
# vim /etc/gitlab/gitlab.rb    //域名改为 IP
```

```
###! On AWS EC2 instances, we also attempt to fetch the  
###! address from AWS. For more details, see:  
###! https://docs.aws.amazon.com/AWSEC2/latest/UserGui  
external_url 'http://124.71.163.140'
```

初始化启动 gitlab

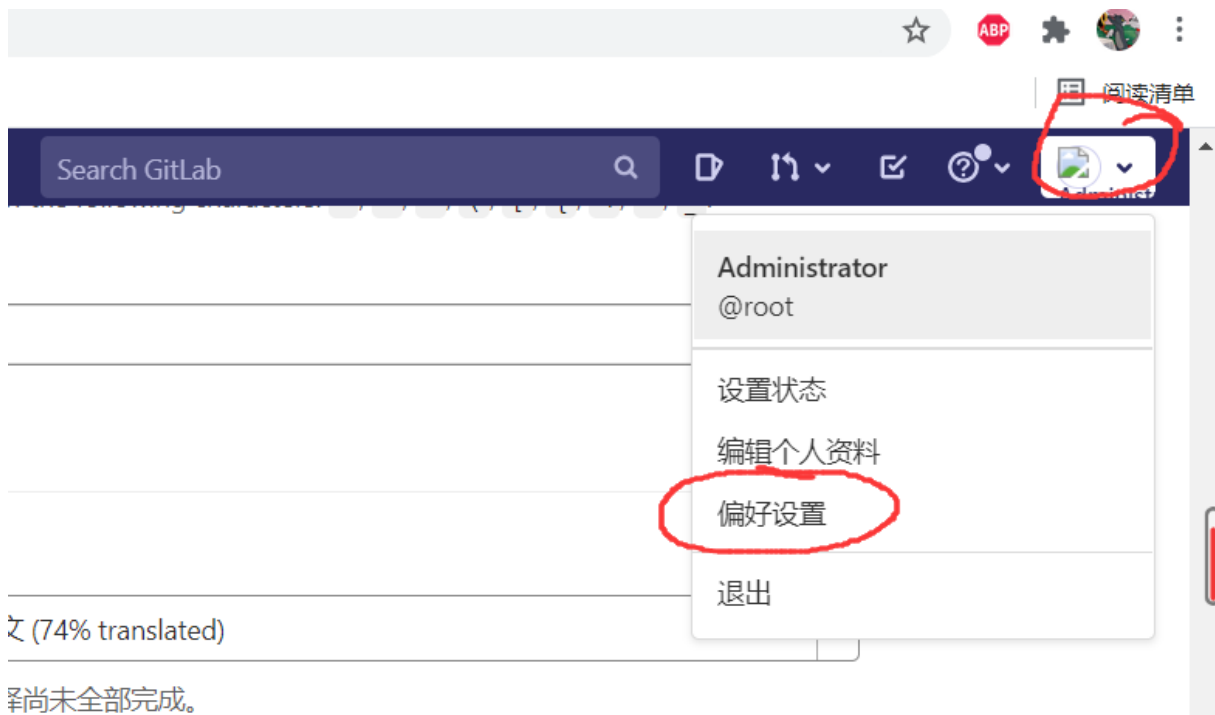
```
# gitlab-ctl reconfigure //这一步时间很长  
# gitlab-ctl stop //停止  
# gitlab-ctl start //启动
```

登入 gitlab



密码：# cat /etc/gitlab/initial_root_password

webUI 设置中文



3、 配置 gitlab 机器可以免密拉取 gitlab 的代码

1、 新建项目

项目

您的项目 2 星标项目 0 浏览项目

Filter by name... 名称

所有 个人

M GitLab Instance / Monitoring Owner
This project is automatically generated and helps monitor this GitLab instance. [Learn more.](#) ★ 0 ♾ 0 0 0 更新于 1小时前

M Administrator / myapp-test Maintainer
这是一个测试web项目 ★ 0 ♾ 0 0 0 更新于 2分钟前

创建空白项目

创建一个空白项目来存放您的文件，规划您的工作，并在代码等方面进行协作。

新建项目 创建空白项目

项目名称 Myapp Test

项目 URL http://124.71.163.140/ root

项目标识串 myapp-test

项目描述(可选)

描述格式 可写可不写

可见性级别

☒ 私有
项目访问必须明确授予每个用户。如果此项目是在一个群组中，群组成员将会获得访问权限。

☐ 内部
除外部用户外，任何登录用户均可访问该项目。

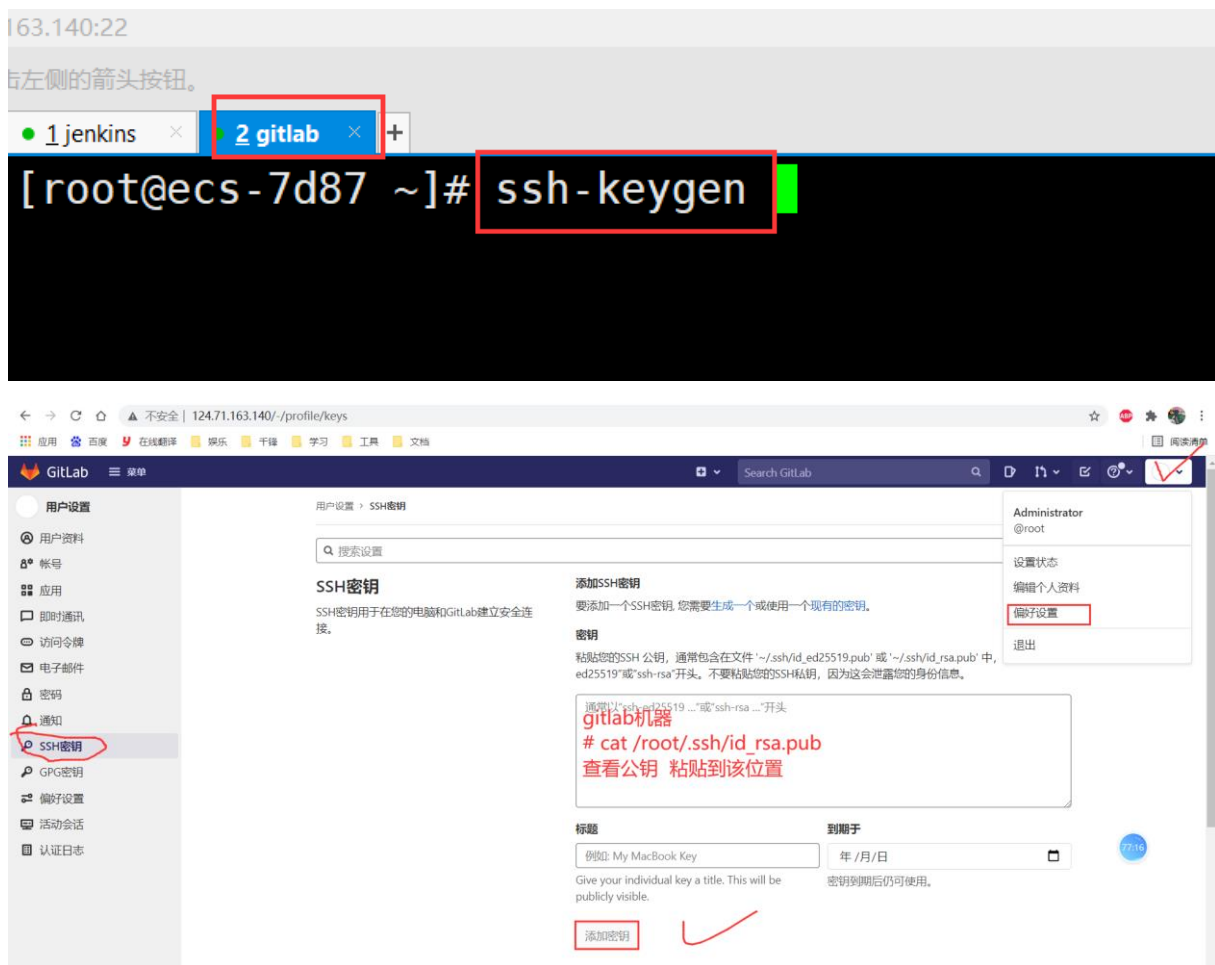
☒ 公开
该项目允许任何人访问。

☒ 使用自述文件初始化仓库
允许您立即克隆这个项目的仓库。如果您计划推送一个现有的仓库，请跳过这个步骤。

新建项目 取消

2、 给 gitlab 去上传 ssh 公钥

```
# ssh-keygen //一直回车
# # ls /root/.ssh/
authorized_keys id_rsa id_rsa.pub known_hosts
//id_rsa 私钥
//id_rsa.pub 公钥
```

3、 push 代码到 gitlab

在客户端，这边暂时 gitlab 机器作为客户端

```
# git clone git@192.168.10.13:root/my-webs.git #把仓库 clone 到客户端
```

```
# cd my-webs
```

```
# echo "hello" > index.html
```

```
# git add index.html
```

```
# git commit -m "V1" #打标签 V1
```

```
# git config --global user.email "root@host4"
```

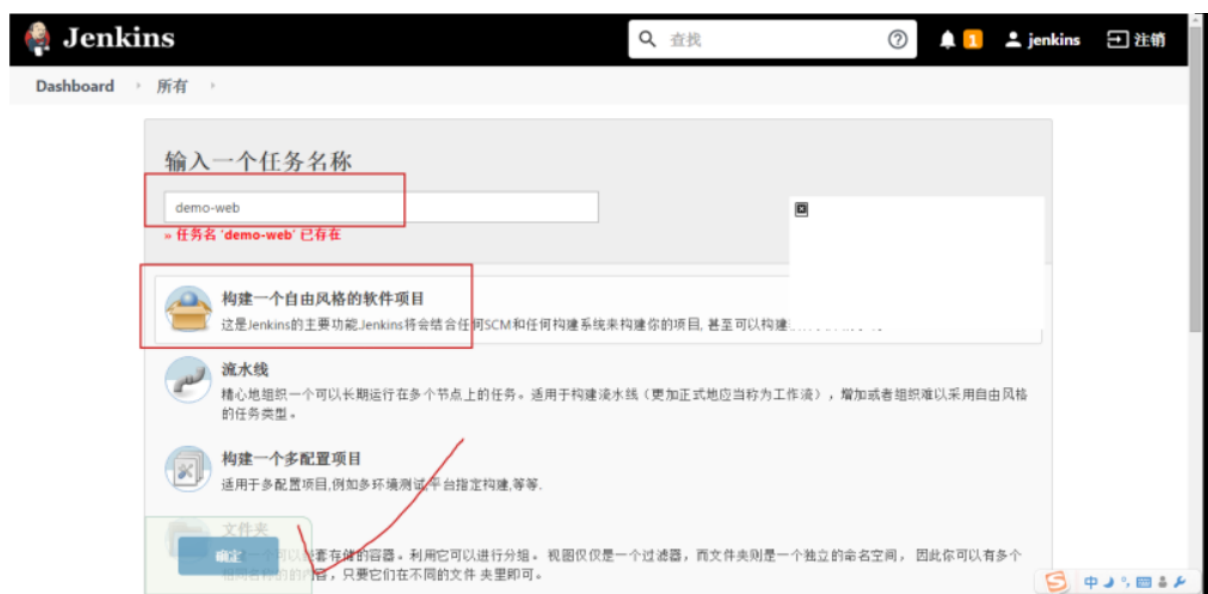
```
# git config --global user.name "root"
```

```
# 邮箱: 用户名@主机名 用户: 用户名
```

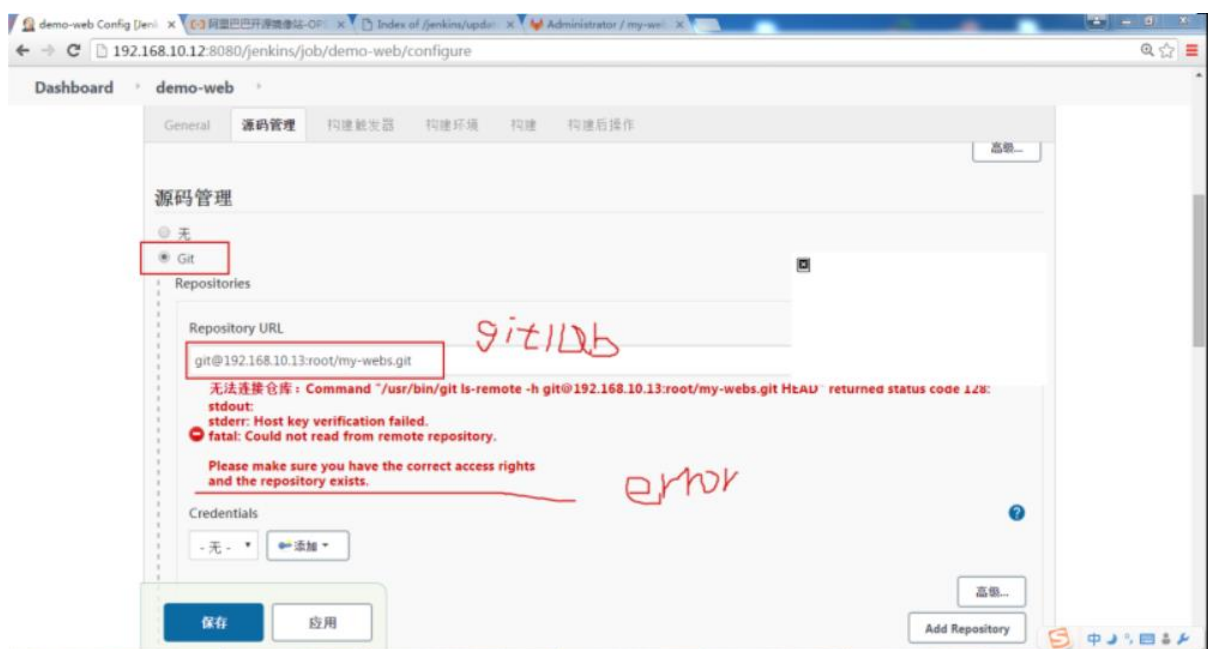
```
# git push -u origin main #进行提交
```

案例 1：向 gitlab 上 push 简单 html 页面，jenkins 进行上线

1、在 jenkins 创建一个任务，myapp-test



2、配置 jenkins 任务，告诉 jenkins 去哪里获取代码



报错的原因：

jenkins 跟 gitlab 没有免密通信

jenkins 机器

```
# ssh-keygen //生成一对密钥
```

```
# cat /root/.ssh/id_rsa.pub //把公钥传给 gitlab web UI 界面
```

```
# cat /root/.ssh/id_rsa //把私钥内容放到 jenkins
```

Git

Repositories

Repository URL

git@124.71.163.140:root/myapp-test.git

Credentials

root (jenkins id-rsa) 添加

高级...

Add Repository

Branches to build

指定分支 (为空白代表any)

*/main

注意 gitlab新版本 主分支叫main

Credentials

root (from gitlab) 添加

Jenkins

board demo-web

General 源码管理 构建触发器 构建环境 构建 构建后操作

Jenkins 凭据提供者: Jenkins

添加凭据

Domain

全局凭据 (unrestricted)

类型

Username with password

Username with password

GitHub App

SSH Username with private key

Secret file

Secret text

Certificate

密码



4、在jenkins 写一个自动发布的脚本

```
# vim /tmp/httpd.sh
```

（添加以下内容）

```
#!/bin/bash
```

```
which httpd || /bin/yum install httpd -y
```

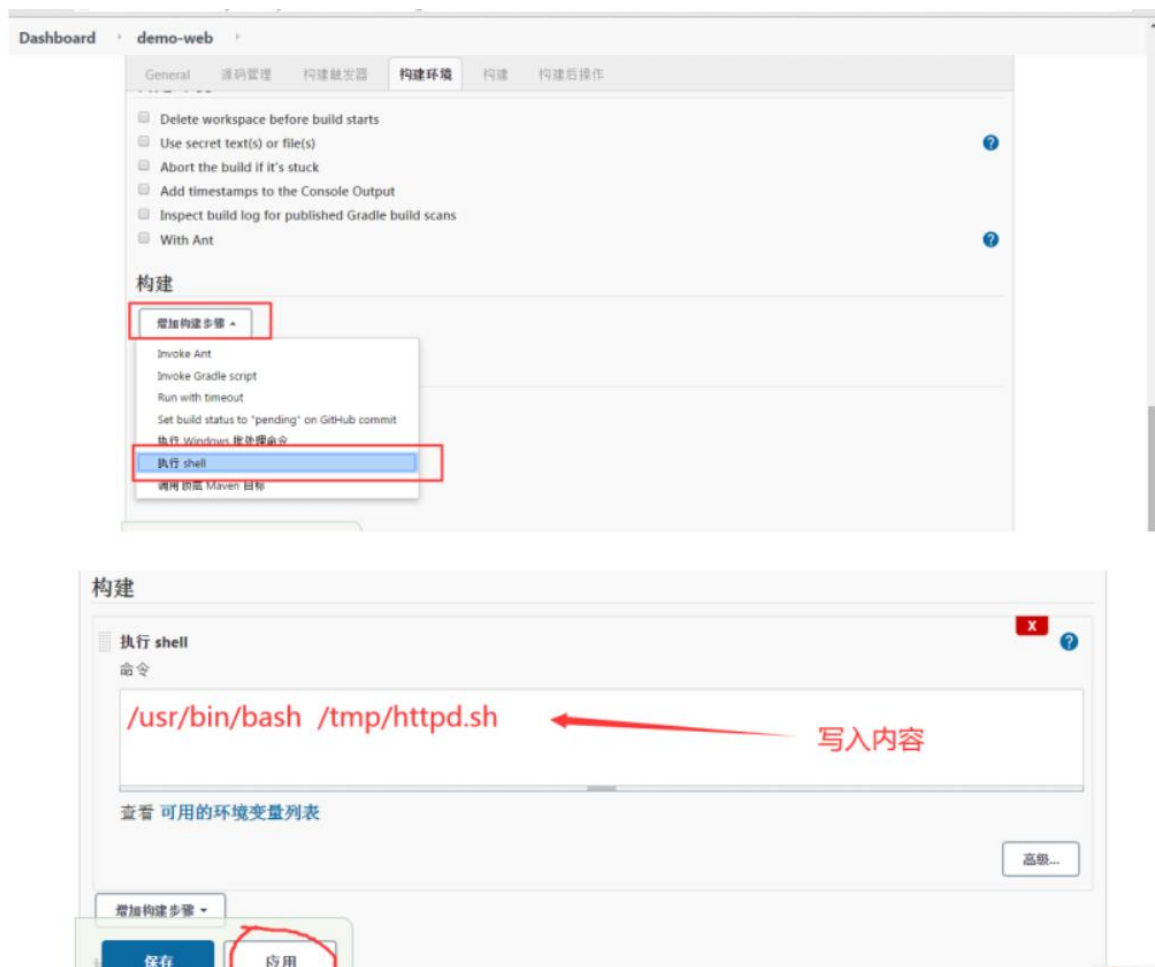
```
/usr/bin/rm -rf /var/www/html/*
```

```
/usr/bin/cp /root/.jenkins/workspace/myapp-test-1/* /var/www/html
```

```
/usr/bin/systemctl restart httpd
```

6、jenkins 设置构建任务

在任务配置里面找到对应位置执行该脚本



最后点击立即构建，之后构建成功去访问jenkins 机器的 IP

观察是否是 gitlab 里面的代码

案例 2：通过 gitlab 插件去自动构建发布

1、安装插件



是否安装插件：



2. 安装好后在项目配置中找到构建触发器,可以看到增加了一个触发器



3. 勾选进入配置

设置需要跟踪变化的分支，根据上面的选项配置，可以是允许全部分支的变化触发构建，也可以设置只是具体的某些分支触发，这里示例是允许 master 分支上的变化触发构建

Allowed branches

☐ Allow all branches to trigger this job 点击更多 ?

☒ Filter branches by name ?

Include

⚠ Following patterns don't match any branch in source repository: master

Exclude

☐ Filter branches by regex ?

☐ Filter merge request by label

4.生成token:点击generate,会生成一个token

Secret token ?

5、在 Gitlab 上找到要自动构建的项目，点击左侧边栏的 Settings -> Webhooks

maven-test > spring > Integrations Settings

Integrations

Webhooks can be used for binding events when something is happening within the project.

URL 此处填第2步中的url

Secret Token 此处填第4步生成的token

Use this token to validate received payloads. It will be sent with the request in the X-Gitlab-Token HTTP header.

Trigger ☒ Push events 勾选

This URL will be triggered by a push to the repository

点击最下面的测试，观察最上面返回是否是 200 状态码

ⓘ Hook executed successfully: HTTP 200

测试jenkisl 能自动拉取代码：

在 gitlab 机器 push 代码到 gitlab，观察jenkisl 是否能自动构建