第十五章　GitLab 与 Jenkins 持续集成平台使用方法

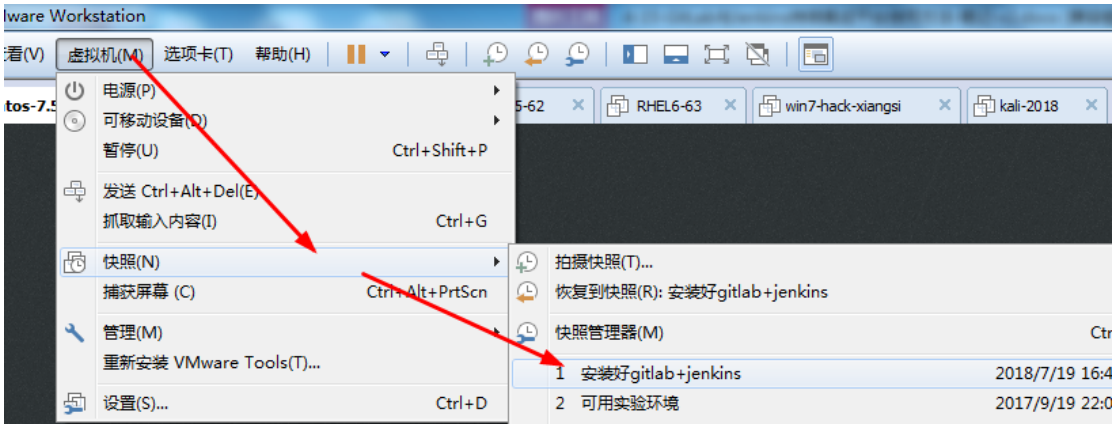
本节所讲内容：

15.1　配置 Jenkins 使用 gitlib 更新代码

15.2　实现 gitlib 触发 Jenkins 自动部署

15.3　增加 Build Pipeline 插件以流程图的形式展示各个 Job 的顺序

准备实验环境：恢复到以一下快照：



恢复好快照后，启动虚拟机

gitlab 和 jenkins 服务，我在之前已经做了开机启动，所以这里可以直接使用了。

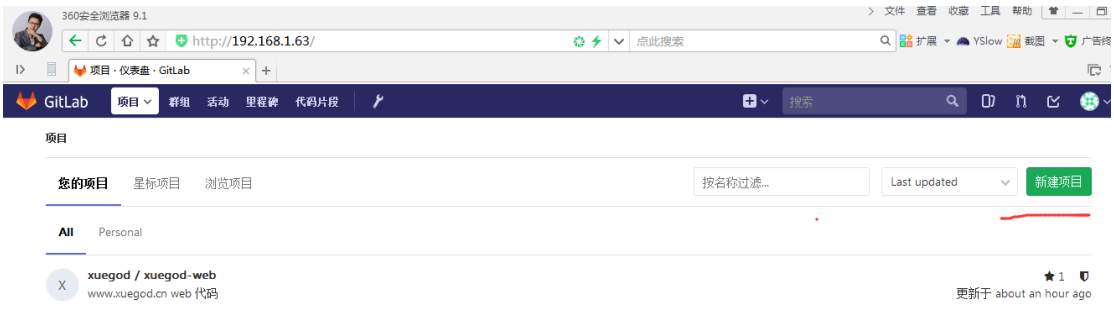
15.1　配置 Jenkins 使用 gitlib 更新代码

15.1.1　查看 GitLab 上创建的项目：xuegod-web

登录 gitlab　　http://192.168.1.63/　　用户名：root　密码：xuegod.cn

登录 jenkins　　http://192.168.1.63:198/　　用户名：admin　密码：123456

查看之前创建的 xuegod-web 项目：



导出 xuegod 项目:

[root@xuegod63 ~]# rm -rf xuegod-web/

[root@xuegod63 ~]# git clone　http://192.168.1.63/xuegod/xuegod-web.git

正克隆到 'xuegod-web'...

Username for 'http://192.168.1.63': jianmingbasic

Password for 'http://jianmingbasic@192.168.1.63': xuegod.cn

15.1.2　安装 Jenkins 插件

安装以下插件：

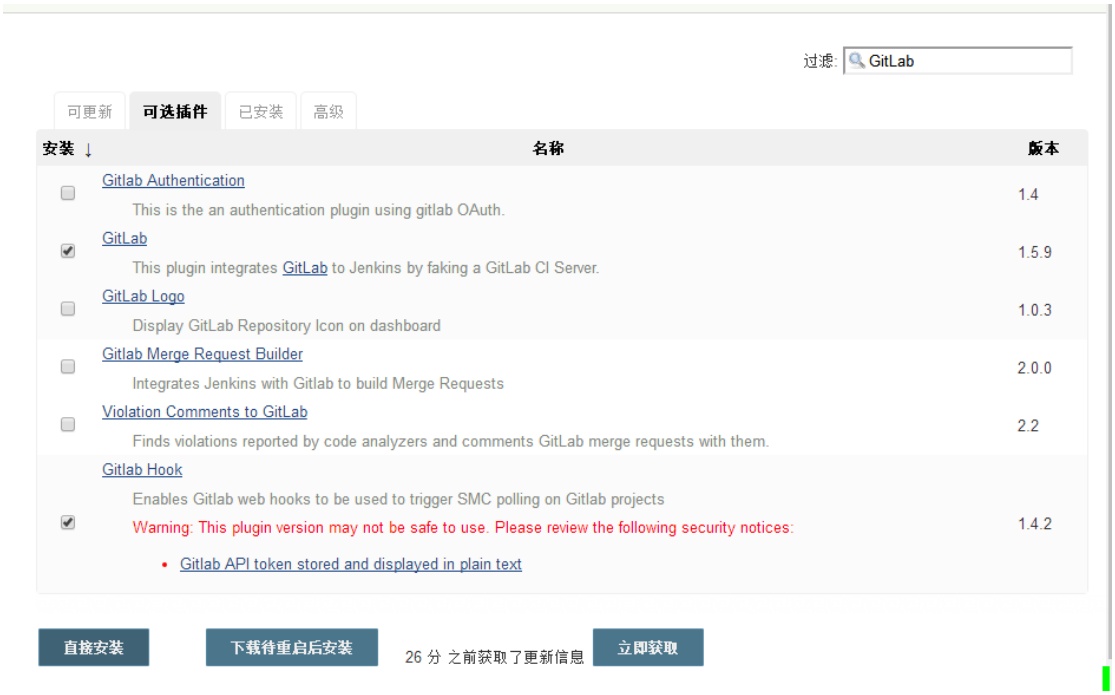
Credentials Plugin（默认已经安装） #签名证书管理插件
Gitlab Plugin #安装后从 gitlab 获取代码
Git Plugin 和 Git Client Plugin ##用于 jenkins 在 gitlab 中拉取源码
GitLab Hook #gitlab 触发 jenkins 构建项目 #hook [hʊk] 钩子 ； plugin ['plʌgɪn] 插件
gitlab Authentication # gitlab 和 jenkins 认证相关的插件
SSH Plugin #远程执行 shell 脚本
Publish Over SSH ##用于通过 ssh 部署应用

系统管理–>插件管理–>选择你需要的

安装插件：


过滤: git
可更新
可选插件
已安装
高级
安装 ↓
名称
版本
GitHub Authentication
A Jenkins authentication plugin that delegates to GitHub. We also implement an Authorization Strategy that uses the acquired OAuth token to interact with the GitHub API to determine a user's level of access to Jenkins.
0.29
Gitlab Authentication
A Jenkins authentication plugin that delegates to gitlab. We also implement an Authorization Strategy that users the acquired OAuth token to interact with the Gitlab API to determine a users level of access to Jenkins.
1.4
Gitcolony Build Notification
Updated live branches and pull requests build status into Gitcolony
1.1
GitHub SQS Build Trigger
2.1
GitLab
1.5.8

过滤: Email Extension
可更新
可选插件
已安装
高级
安装 ↓
名称
版本
Email Extension Template
1.0
直接安装
下载待重启后安装
34 分 之前获取了更新信息
立即获取


过滤: GitLab
可更新
可选插件
已安装
高级
安装 ↓
名称
版本
Gitlab Authentication
This is the an authentication plugin using gitlab OAuth.
1.4
GitLab
This plugin integrates GitLab to Jenkins by faking a GitLab CI Server.
1.5.9
GitLab Logo
Display GitLab Repository Icon on dashboard
1.0.3
Gitlab Merge Request Builder
Integrates Jenkins with Gitlab to build Merge Requests
2.0.0
Violation Comments to GitLab
Finds violations reported by code analyzers and comments GitLab merge requests with them.
2.2
Gitlab Hook
Enables Gitlab web hooks to be used to trigger SMC polling on Gitlab projects
Warning: This plugin version may not be safe to use. Please review the following security notices:
Gitlab API token stored and displayed in plain text
1.4.2
直接安装
下载待重启后安装
26 分 之前获取了更新信息
立即获取

安装插件方式选择：

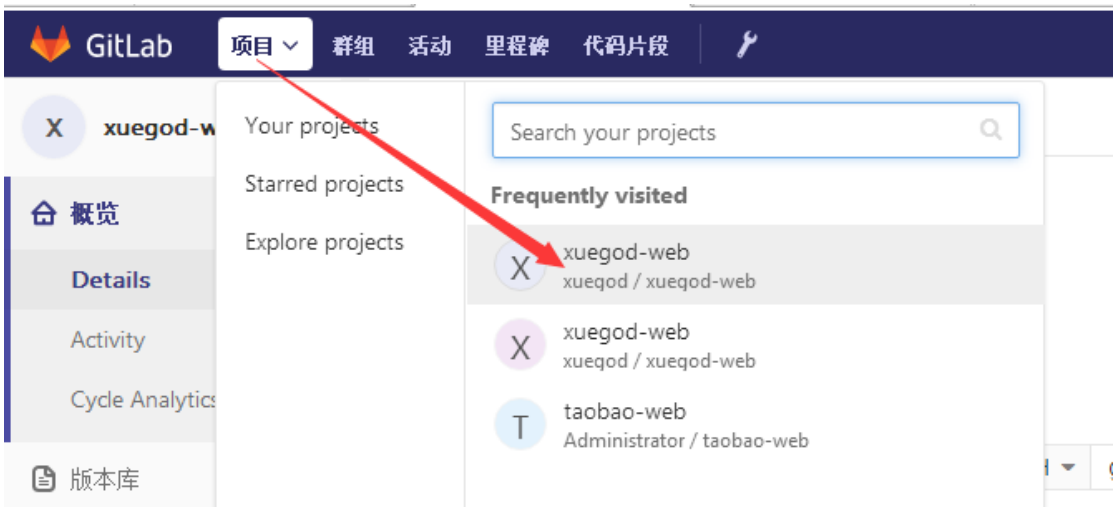
方法一.如果服务器可以上网,那边选择在线安装最好不过了,安装流程为: 系统管理(Configure System)----插件管理(Manage Jenkins)---选择需要的插件直接安装即可

方法二.如果服务器不能上网,那么就只能离线安装,首先去 http://updates.jenkins-ci.org/download/plugins/ 下载需要的 plugin,选择匹配的版本号,下载到本地,然后打开:系统管理()---插件管理—高级---找到”上传插件”(浏览,找到扩展名为.hpi 的插件，上传之后默认直接就安装了。重启 jenkins，安装的插件就可以使用了。安装插件依赖解决：

插件安装不上去，一定是某个地方出现问题。在 Jenkins 的终端日志中，可以查看到对应的错误。需要针对性的解决。 比如：在安装 subversion 的时候，的错误提示。 一般的情况安装对应的插件，会把需要依赖的插件安装上。如出现对应的依赖问题，就需要一一安装上对应的依赖包。

15.1.3 设置 gitlab 用户 root 可以通过 ssh 公钥直接下载代码

部署 key，让 root 用户可以不输入密码，获取 gitlab 上的源代码



[root@xuegod63 ~]# ssh-keygen　#生成的公钥。 一路回车，不要输入保护私钥的密码。

[root@xuegod63 ~]# cat .ssh/id_rsa.pub　#查看公钥

ssh-rsa

AAAAB3NzaC1yc2EAAAADAQABAAABAQCfAxKT6/F09mKFpyZuWJwSDzu2ZaB4Q975BRLIa
La1aN4/95KmQQZKF4F9L+NeLvIEOPoZKDnxa2iSZXO2d2JNzualf/hZeJ3uJXnHGgELVZgh5v
NKI2IP/iA5Q+Hx5LZ0fJKmje2n25EFEH7Nc5N7xc7Fkch6dDtWfTf0SsyFaTgaAsAOe+nctRg5
dyUhDss0+qV2a89Qd423qhHwnv+LmbDec2PR1SbQwweyjTV/DT3DvORG/eT28Lj876LyWm
IjV+/vHgygbmIqDywF2WKnStgbXYqn6iJG4mCzry536LvP696vx63u0QI7yPyMsU1+yKhJPW
LIARduXRrtwK25 root@xuegod63.cn

测试部署 key，发现可以不用用户名密码后直接获取代码 ：

测试：

[root@xuegod63 ~]# rm -rf xuegod-web/

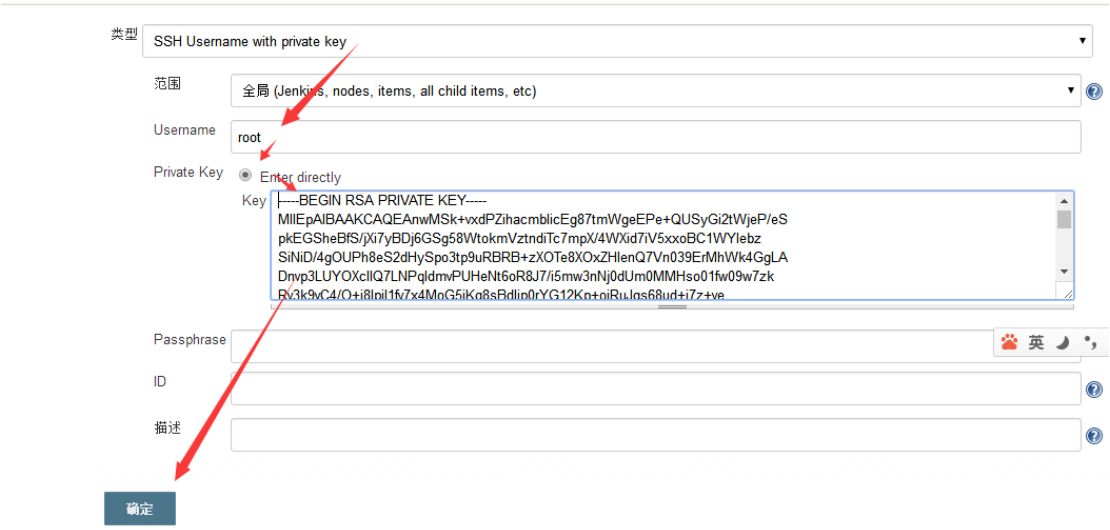
[root@xuegod63 ~]# git clone git@192.168.1.63:xuegod/xuegod-web.git #发现可以直接下载代码

15.1.4 Jenkins 添加 git 用户 root 的私钥

复制 xuegod63 中 id_dsa.pub 里面的公钥添加到 Jenkins（private key 选项）

[root@xuegod63 ~]# cat .ssh/id_rsa

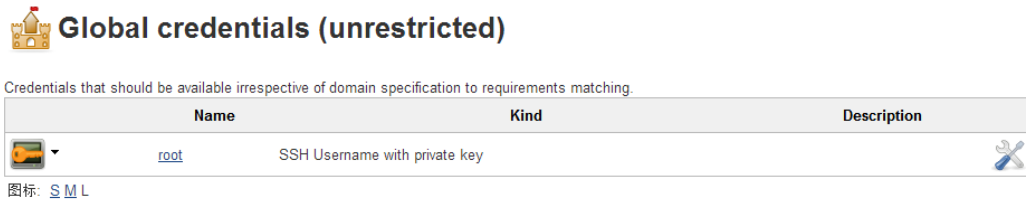
Credentials [krəˈdenʃlz] 资格证书



注：这里直接复制/root/.ssh/id_rsa 中的内容。

root 公钥在 gitlab，root 私钥在 jenkins，这样 jenkins 就可以直接拉取 gitlab 上的代码

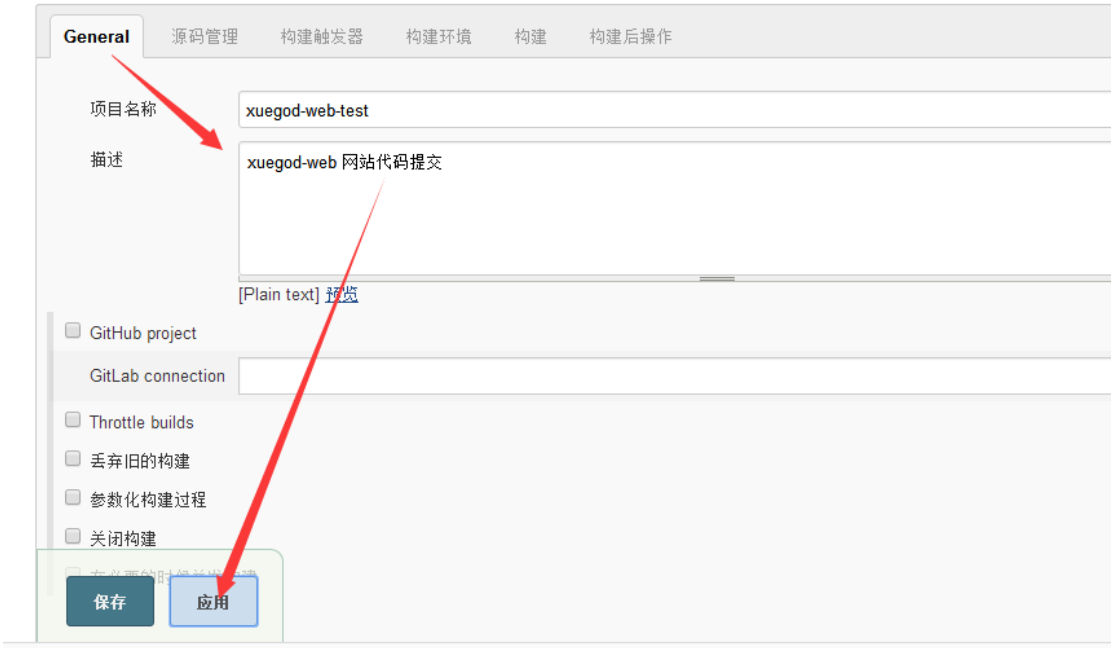
全局凭证（不受限制） unrestricted [ˌʌnrɪˈstrɪktɪd] 无限制的



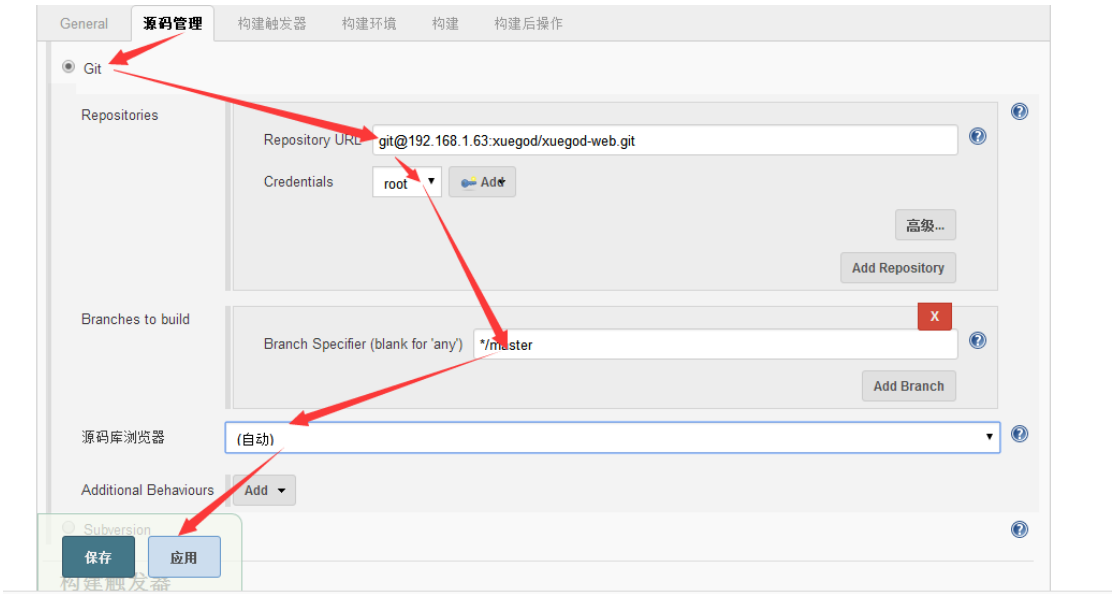
15.1.5 Jenkins 创建项目

新建任务：

xuegod-web 网站代码提交



配置适用 git 源： git@192.168.1.63:xuegod/xuegod-web.git



点保存。

查看状态：

构建成功。

15.1.6 将代码发布到 web 服务器

可以通过执行命令或脚本的方式进行代码发布，我使用 root 用户在各个 web 服务器上发布代码。大家也可以建立一些普通用户如果 www 或 apache 来发布。如果使用普通用户发布代码，要保持所有服务器上的用户 id 一致，这样没有权限的问题。

安装 apache 服务

[root@xuegod63 yum.repos.d]# yum install httpd -y

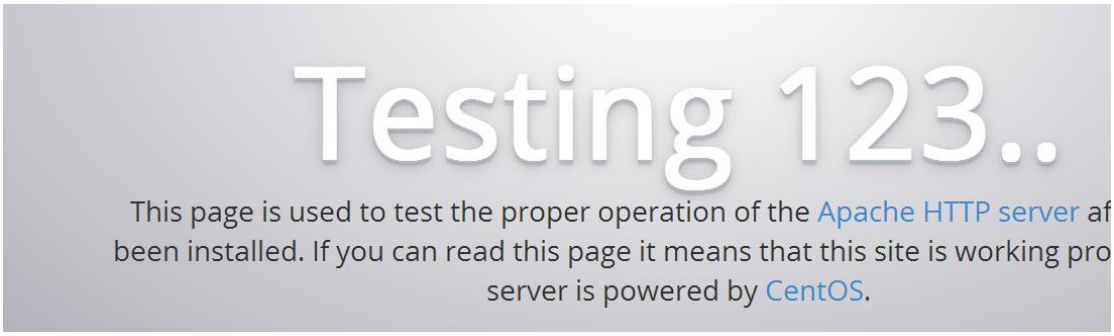
[root@xuegod63 yum.repos.d]# vim /etc/httpd/conf/httpd.conf

改：Listen 80

为：Listen 81

[root@xuegod63 ~]# systemctl restart httpd

访问正常：http://192.168.1.63:81/



15.1.7 准备自动部署 web 代码事项

1、实现 root 用户可以无密码直接登录系统：

[root@xuegod63 ~]# ssh-copy-id root@192.168.1.63

[root@xuegod63 ~]# ssh 192.168.1.63

注：需要在哪台机器上自动发布代码，就需要让 jenkins 这样机器的 root 用户可以免费密码登录到那台服务器上。

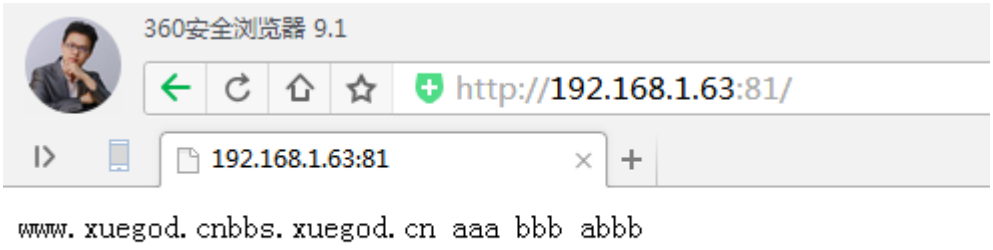
2、配置 Jenkins 用户 sudu 权限并写

```
[root@xuegod63 ~]# visudo     #最后添加
jenkins ALL=(ALL)          NOPASSWD: /usr/bin/ssh
#不需要使用密码即可执行 ssh
```

3、创建自动上传 web 代码的脚本

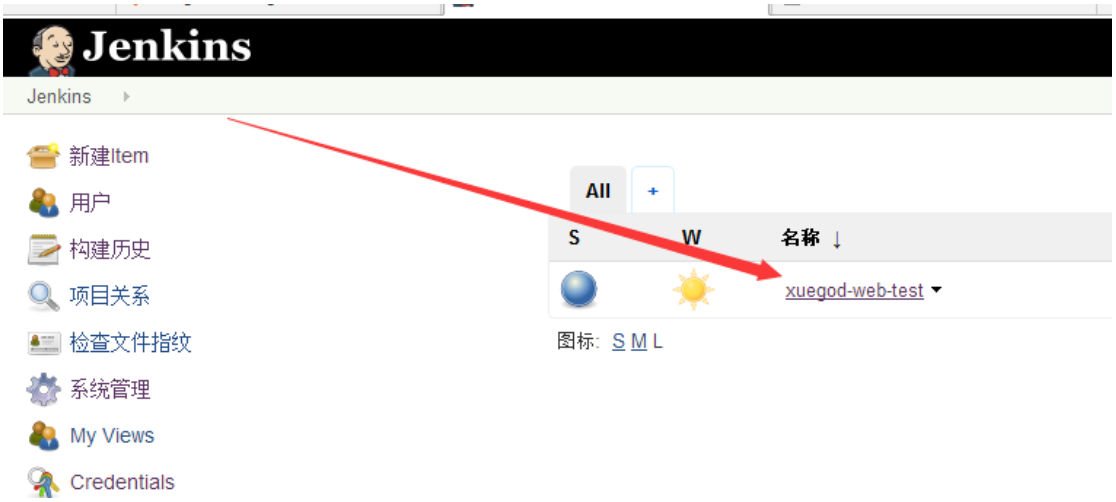
```
[root@xuegod63 ~]#   vim   deploy.sh   #插入以下内容
#!/bin/bash
echo $USER
cd /root/xuegod-web
git pull
scp -r ./* root@192.168.1.63:/var/www/html/
[root@xuegod63 ~]# chmod +x deploy.sh
[root@xuegod63 ~]# ./deploy.sh   #确保可以无交互执行成功
```

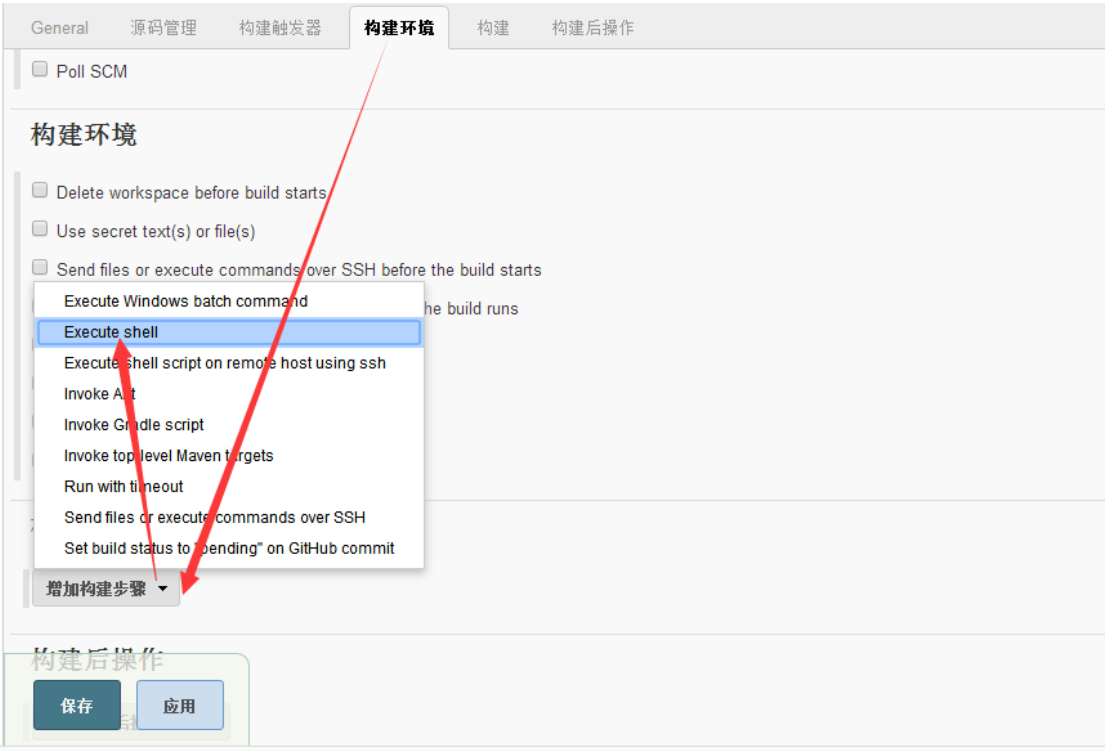
测试自动上传代码：



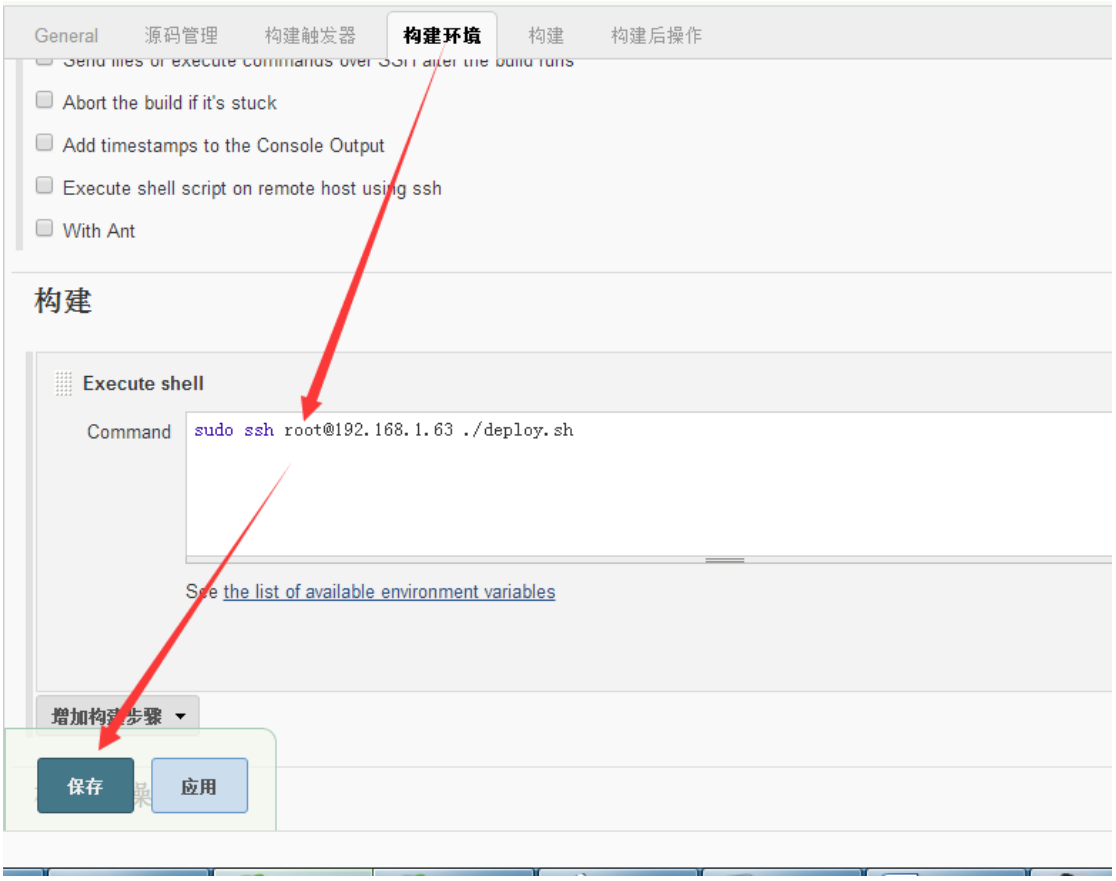
[root@xuegod63 ~]# rm -rf /var/www/html/index.html   #把网页删除一下，一会还要测试 jenkins 部署

15.1.8 在 jenkins 的”增加构建步骤“写入上传代码脚本



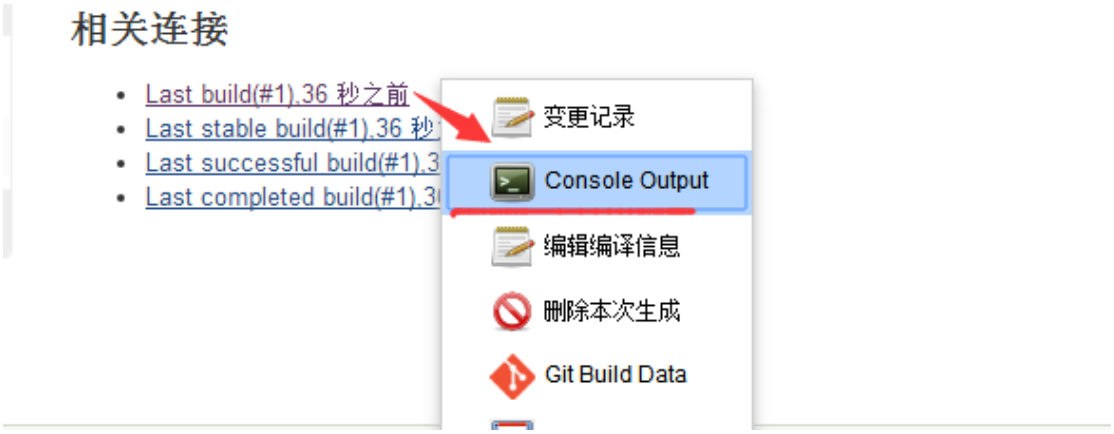


sudo ssh root@192.168.1.63 ./deploy.sh



手动点立即构建：

查看输出的内容：




```
Started by user mk
Building in workspace /var/lib/jenkins/workspace/xuegod-web-test
Cloning the remote Git repository
Cloning repository git@192.168.1.63:xuegod/xuegod-web.git
 > git init /var/lib/jenkins/workspace/xuegod-web-test # timeout=10
Fetching upstream changes from git@192.168.1.63:xuegod/xuegod-web.git
 > git --version # timeout=10
using GIT_SSH to set credentials
 > git fetch --tags --progress git@192.168.1.63:xuegod/xuegod-web.git +refs/heads/*:refs/remotes/origi
 > git config remote.origin.url git@192.168.1.63:xuegod/xuegod-web.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url git@192.168.1.63:xuegod/xuegod-web.git # timeout=10
Fetching upstream changes from git@192.168.1.63:xuegod/xuegod-web.git
using GIT_SSH to set credentials
 > git fetch --tags --progress git@192.168.1.63:xuegod/xuegod-web.git +refs/heads/*:refs/remotes/origi
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision ec7684d9773f07367caf0ab0347a039315e58ba2 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f ec7684d9773f07367caf0ab0347a039315e58ba2
Commit message: "111"
First time build. Skipping changelog.
[xuegod-web-test] $ /bin/sh -xe /tmp/jenkins7737484825274992855.sh
+ sudo ssh root@192.168.1.63 ./deploy.sh
root
Already up-to-date.
Finished: SUCCESS
```

到此单台测试已经成功。

15.1.9 开始自动部署多台 web 服务器

启动 xuegod64.cn 虚拟机

[root@xuegod64 ~]# yum install httpd -y

[root@xuegod64 yum.repos.d]# vim /etc/httpd/conf/httpd.conf

改： 42 Listen 81

为： 42 Listen 80

[root@xuegod64 yum.repos.d]# systemctl restart httpd

[root@xuegod64 ~]# iptables -F

登录 xuegod63，上传到 root 用户公钥

[root@xuegod63 ~]# ssh-copy-id root@192.168.1.64

[root@xuegod63 ~]# ssh root@192.168.1.64 #测试可以直接登录成功

```
[root@xuegod63 ~]# vim deploy.sh   #追加一行
#!/bin/bash
echo $USER
cd /root/xuegod-web
git pull
scp -r ./* root@192.168.1.63:/var/www/html/
scp -r ./* root@192.168.1.64:/var/www/html/
```

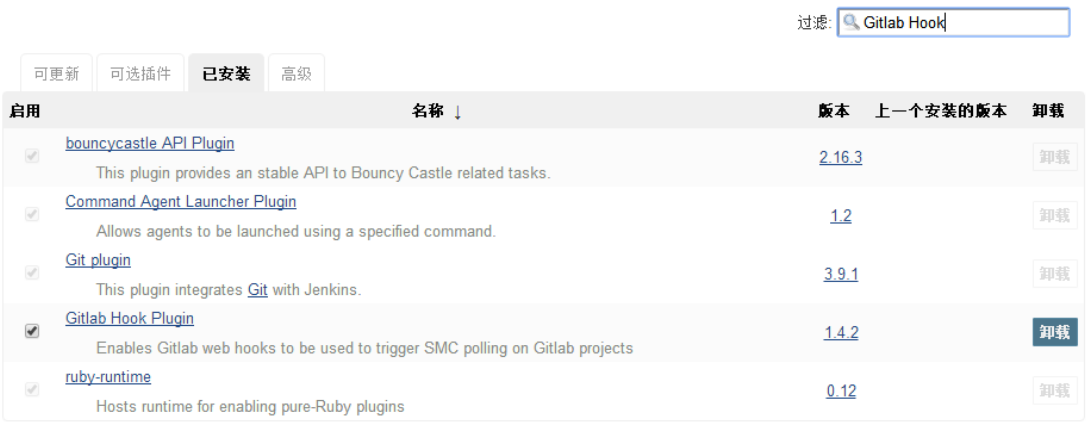
15.2 实现 gitlib 触发 Jenkins 自动部署

实战场景：在公司的测试环境当中，当开发人员向 gitlab 仓库提交代码后，gitlab 自动通知 jenkins 进行构建项目、代码质量测试然后部署至测试环境，这里先暂时部署到测试环境。对于生产环境，后期使用手动部署代码。

15.2.1 安装 Gitlab Hook Plugin 插件

1、安装 Gitlab Hook Plugin 插件：

#系统管理-管理插件-可选插件，安装这两个插件：Gitlab Hook Plugin 和 Build Authorization Token Root 和 Build Token Trigger



15.2.2 配置身份验证令牌

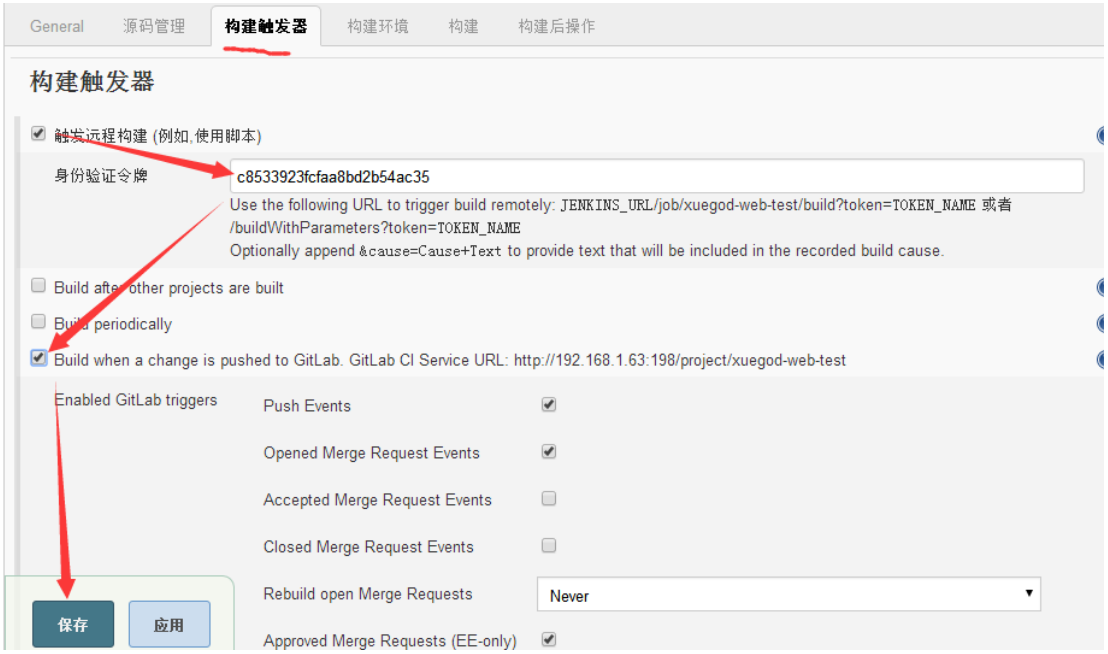
生成 Token ，openssl rand 生成随机密码

```
[root@xuegod63 ~]# openssl  rand -hex 12
c8533923fcfaa8bd2b54ac35
```

注：-hex：输出结果为 16 进制数据; 数字 12 是指定生成的随机字符为 12 字节，即 24 位 16 位进制数

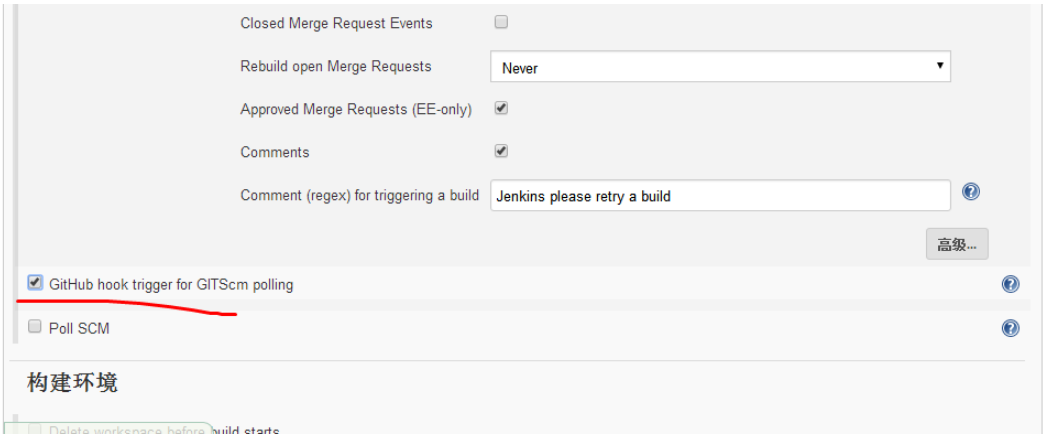
配置触发器 http://192.168.1.63:198/job/xuegod-web-test/configure

身份验证令牌： c8533923fcfaa8bd2b54ac35



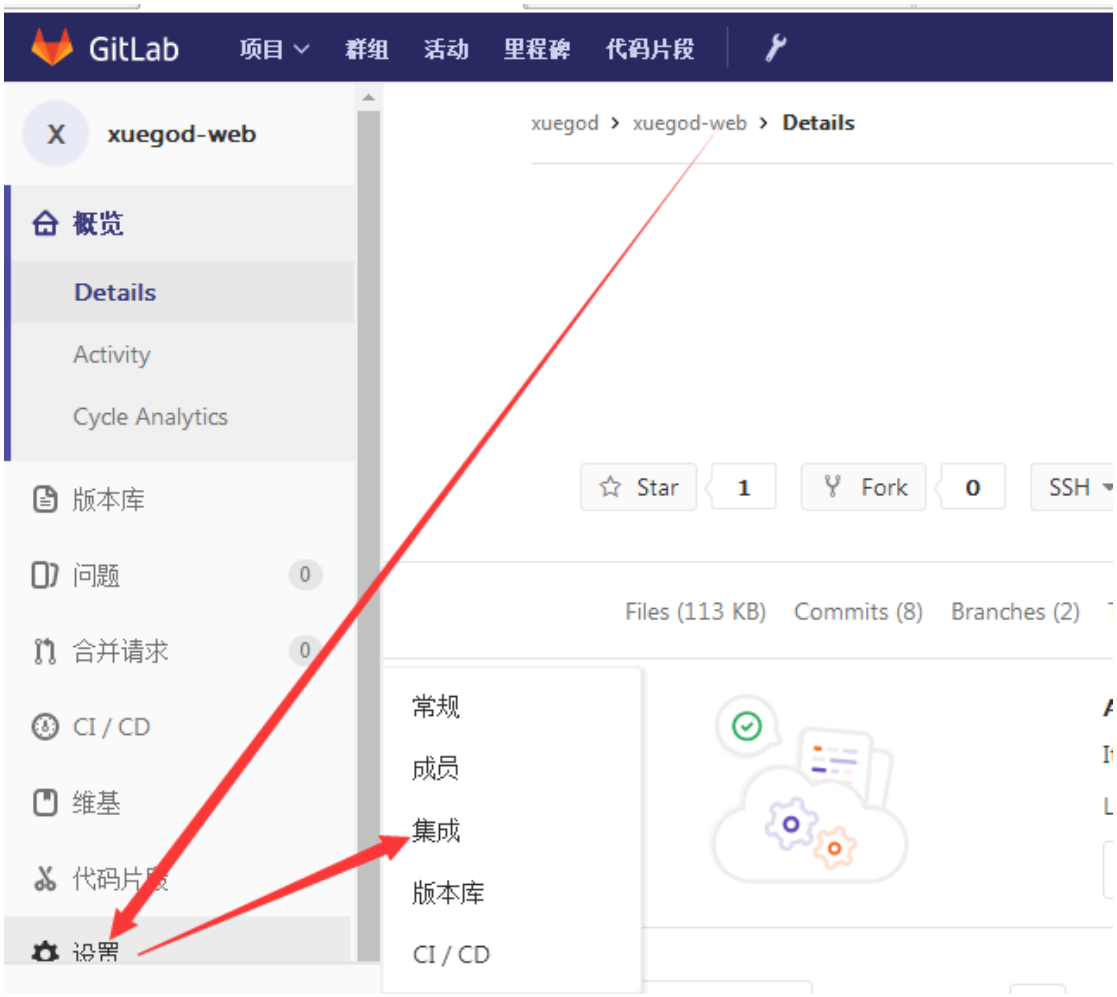
http://192.168.1.63:198/project/xuegod-web-test

构建触发器，勾上“GitHub hook trigger for GITScm polling”



15.2.2 在 git 项目配置界面设置链接和 token

登录 gitlab,在这个项目下找到 web 钩子配置的地方

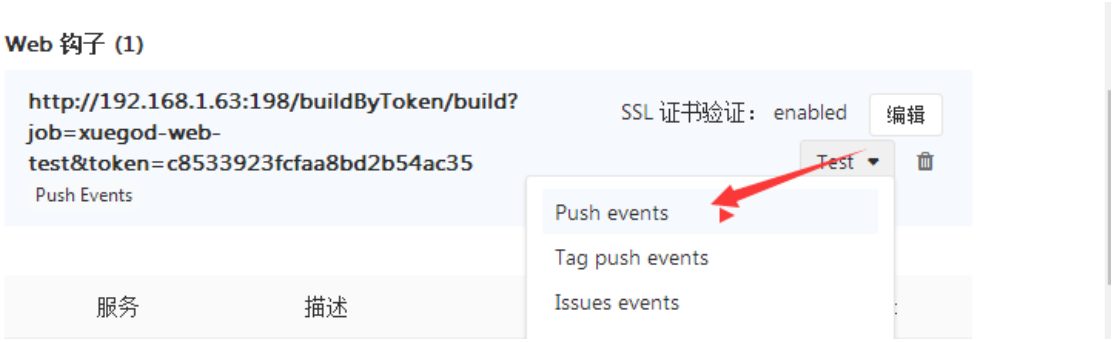


使用这个：

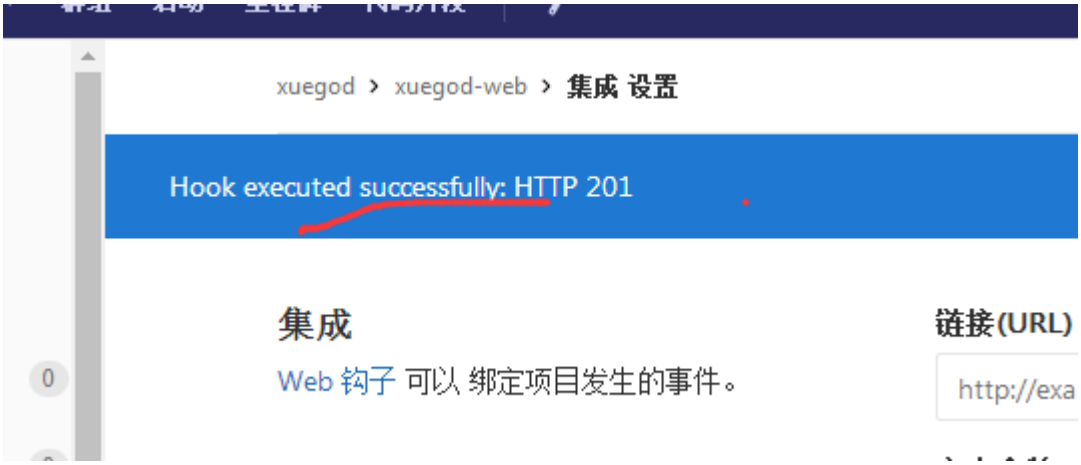
http://192.168.1.63:198/buildByToken/build?job=xuegod-web-test&token=c8533923fcfaa8bd2b54ac35

格式：http://jenkins 地址/buildByToken/build?job=jenkins 项目名&token=token 值

测试：



返回这个，说明成功了：



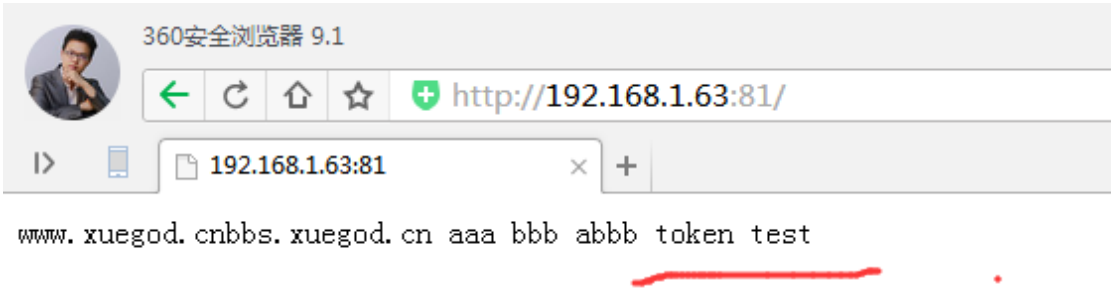
15.2.2 测试 gitlab 触发 jenkins 自动部署

向 git 服务器提交代码，验证是否可以自动部署：

```
[root@xuegod63 xuegod-web]# cd /root/xuegod-web/
[root@xuegod63 xuegod-web]# echo "token test" >> index.html
[root@xuegod63 xuegod-web]# git add index.html
[root@xuegod63 xuegod-web]# git commit -m "modify index.html"
[root@xuegod63 xuegod-web]# git push -u origin master
```

访问 web 界面验证代码是否最新的：

http://192.168.1.63:81/ 已经是最新的了



总结步骤： gitlab 创建代码项目-》gitlab 部署公钥-》jenkins 安装插件-》创建新项目-》绑定私钥-》添加触发器-》测试部署

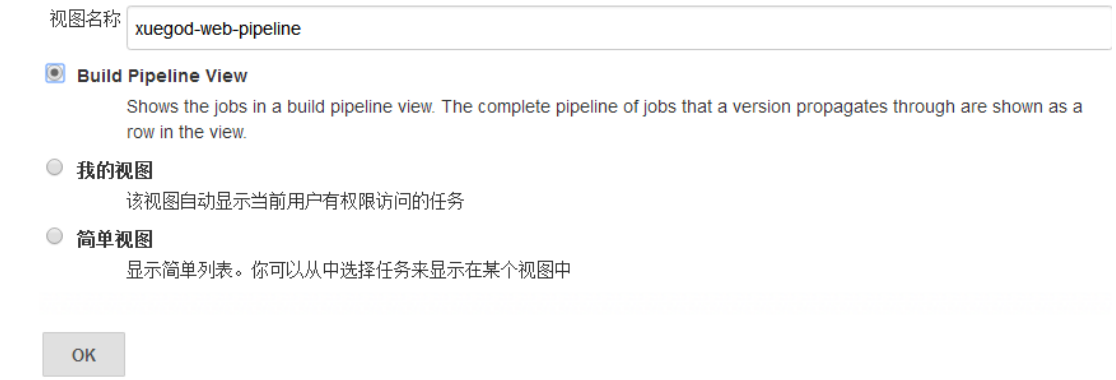
15.3 增加 Build Pipeline 插件以流程图的形式展示各个 Job 的顺序

安装 Build Pipeline 插件 ,Build Pipeline 插件作用： 这个插件能够以流程图的形式展示各个 Job 的顺序，依赖关系等等。

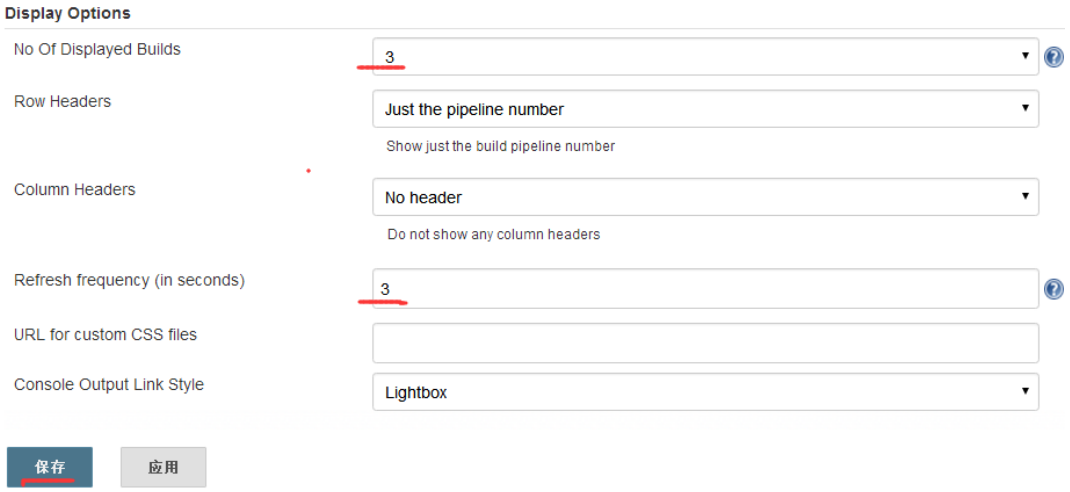
Build Pipeline

添加一个视图，配置项目 xuegod-web-test 构建完成后自动构建 xuegod-web 项目：

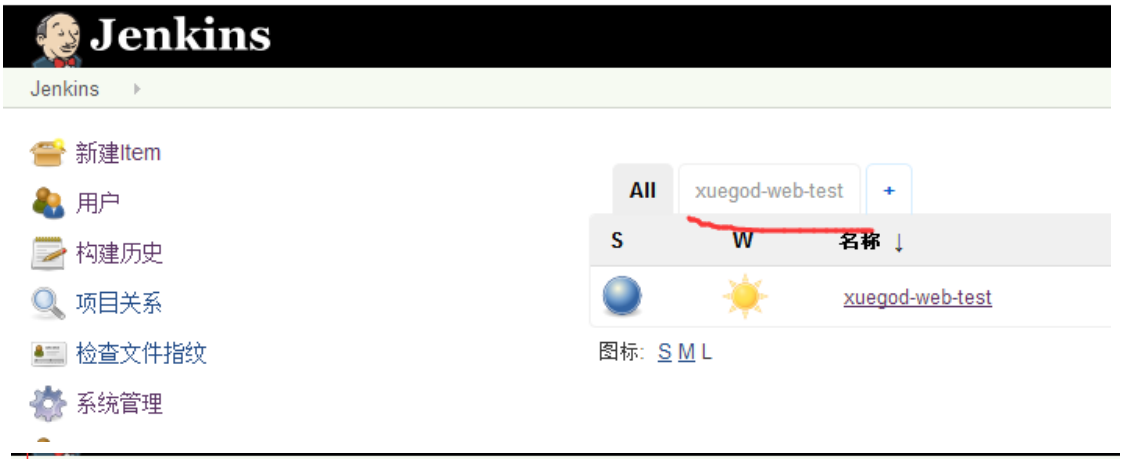
视图名称： xuegod-web-pipeline

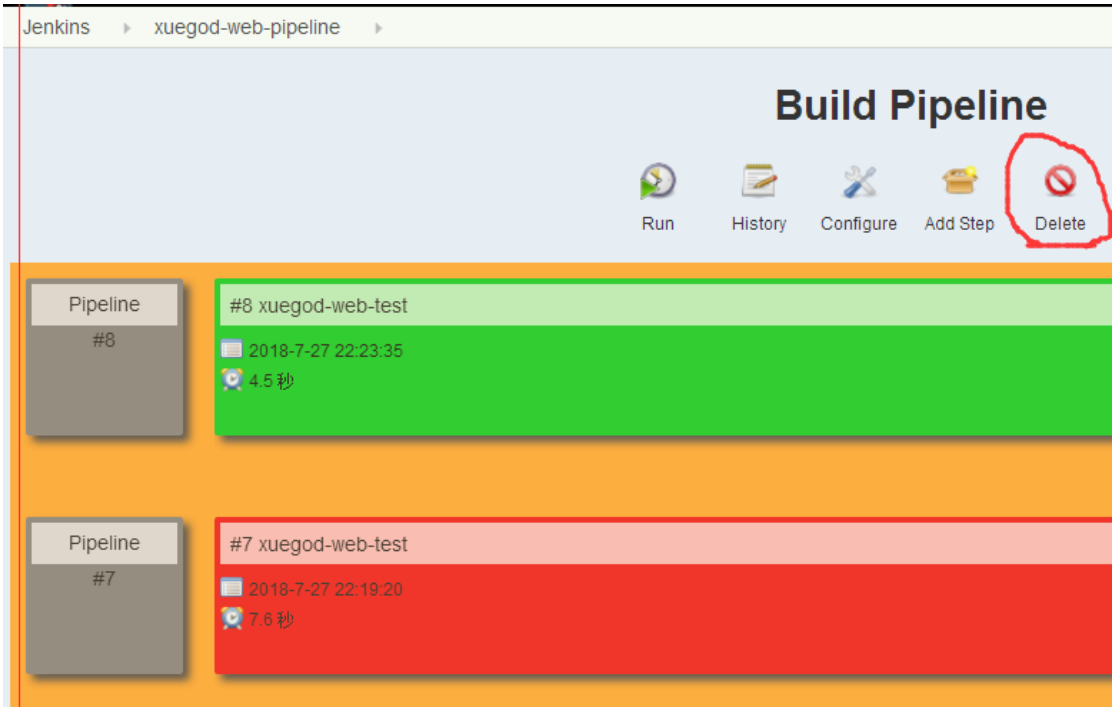


显示最近 3 次构建的结果：



查看效果：





总结：

15.1 配置 Jenkins 使用 gitlib 更新代码

15.2 实现 gitlib 触发 Jenkins 自动部署

15.3 增加 Build Pipeline 插件以流程图的形式展示各个 Job 的顺序