Docker Jenkins + SVN| GitLab CI/CD

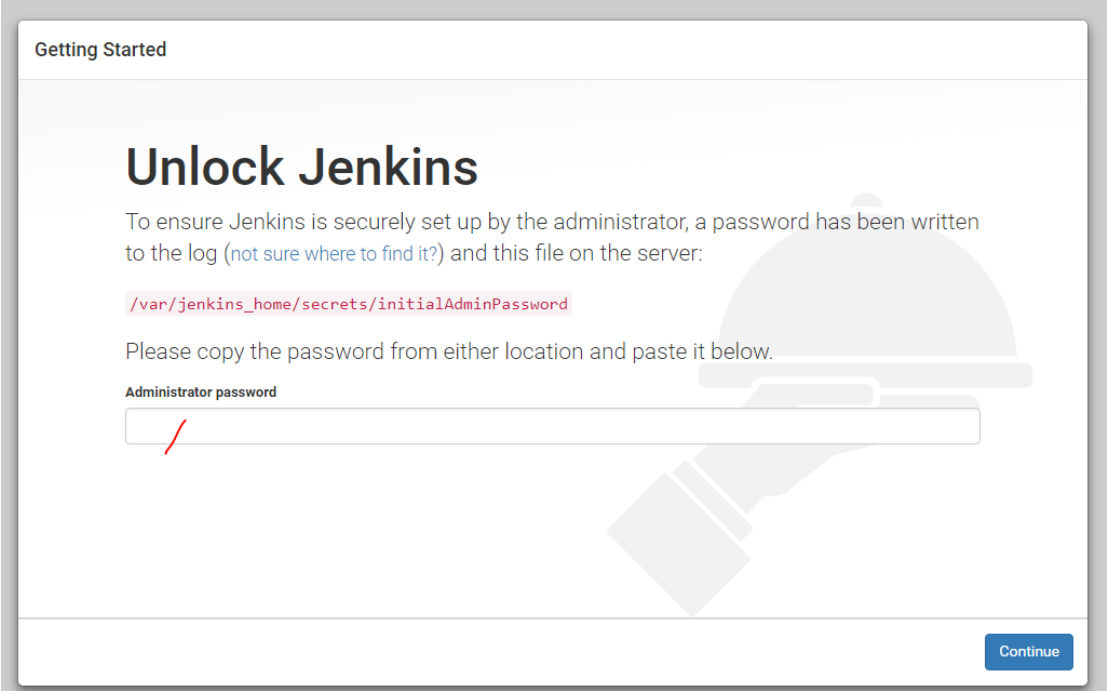
1 Docker 容器中启动jenkins 2
2 选择插件安装 3
3 设置超级管理员帐号 4
4 依赖环境 4
5 SVN+Jenkins自动构建 4
5.1 创建一个自由风格的软件项目 4
5.2 源码管理选择Subversion 5
5.3 配置构建触发器 5
6 Git+Jenkins自动构建 8
附录： 16

作者	日期	版本
姜鹏	2019/03/25	V0.1

1 Docker 容器中启动jenkins

版本：Jenkins 2.60.3

$docker run -u root --rm -d --name jenkins -p 8099:8080 -p 50001:50000 -v /data/jenkins/:/var/jenkins_home/ -v /usr/home/softwares/apache-maven-3.6.0:/var/lib/maven -v /var/run/docker.sock:/var/run/docker.sock jenkinsci/blueocean



Password: 0c15823f28a240378e3480bb3ba402d1$docker

进入容器：

$exec -it Jenkins sh

查看密码：

$ cat /var/jenkins_home/secrets/initialAdminPassword

若容器启动jenkins失败，可以尝试：

清理此容器的网络占用

$docker network disconnect --force bridge jenkins

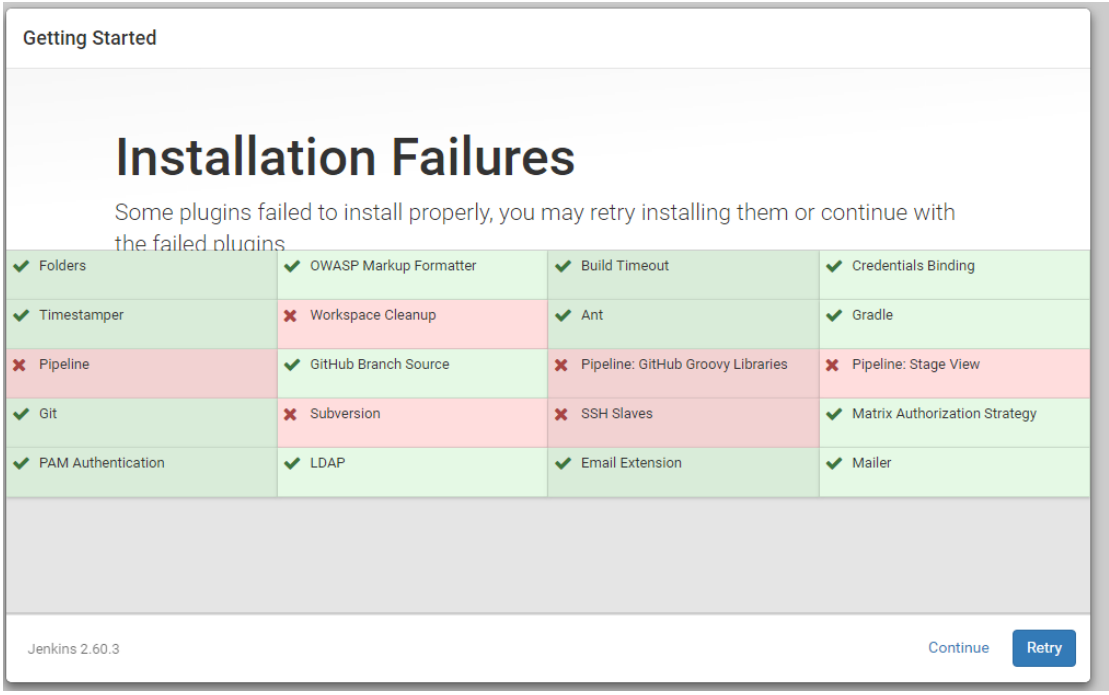
查看是否还有同名容器占用

$docker network inspect jenkins

systemctl start docker

systemctl stop docker

2 选择插件安装



3 设置超级管理员帐号

jenkins/Jenkins

4 依赖环境

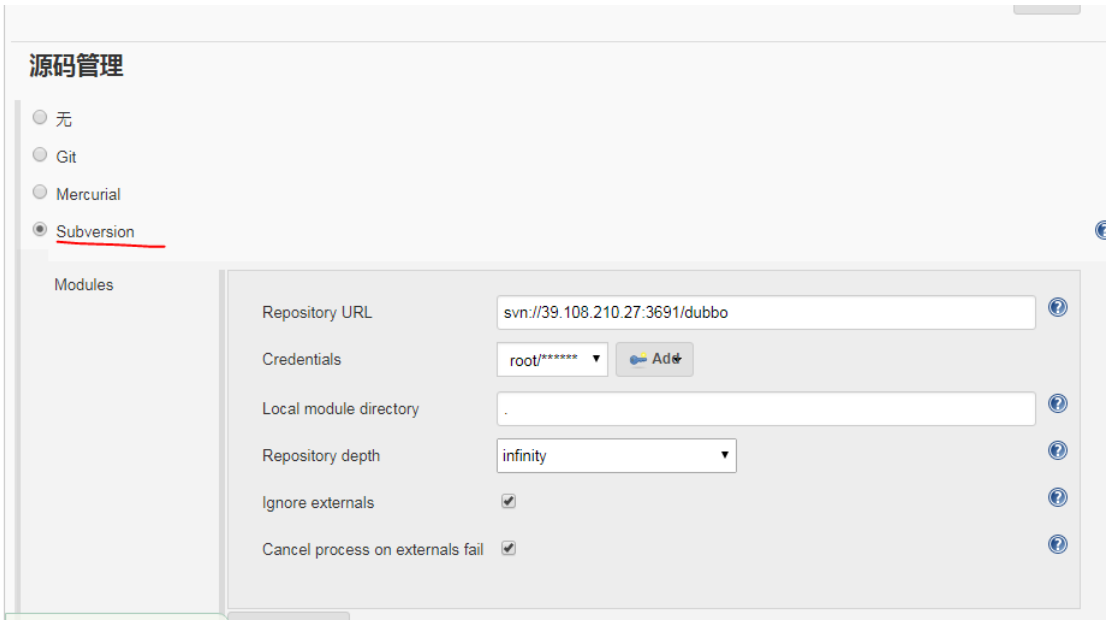
➢ JDK: /docker-java-home

```
$ java -version
openjdk version "1.8.0_171"
OpenJDK Runtime Environment (build 1.8.0_171-8u171-b11-1~deb9u1-b11)
OpenJDK 64-Bit Server VM (build 25.171-b11, mixed mode)
$ pwd
/docker-java-home/bin
$
```

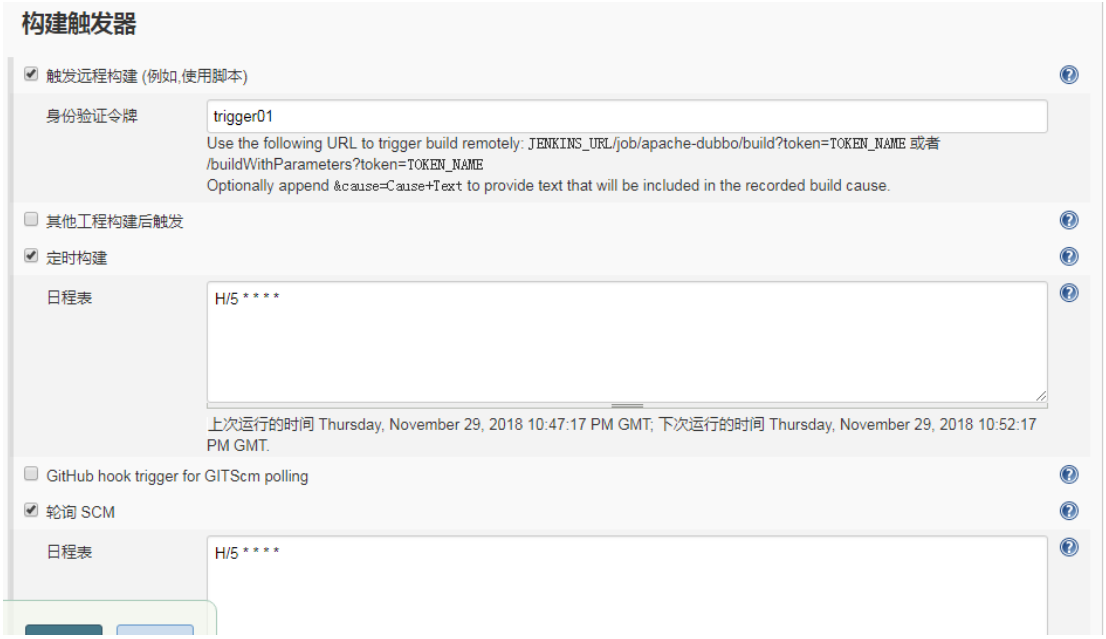
5 SVN+Jenkins自动构建

5.1 创建一个自由风格的软件项目

5.2 源码管理选择Subversion



5.3 配置构建触发器



输入：

http://39.108.210.27:8099/job/apache-dubbo/build?token=trigger01

可以触发一次构建

(1) 定时构建：不管SVN或Git中数据有无变化，均执行定时化的构建任务 ；

(2) 轮询SCM：只要SVN或Git中数据有更新，则执行构建任务；

构建语法说明：

(1)格式为：* * * * *（五个星）；

(2) 第一个*表示分钟，取值0~59

(3) 第二个*表示小时，取值0~23

(4) 第三个*表示一个月的第几天，取值1~31

(5) 第四个*表示第几月，取值1~12

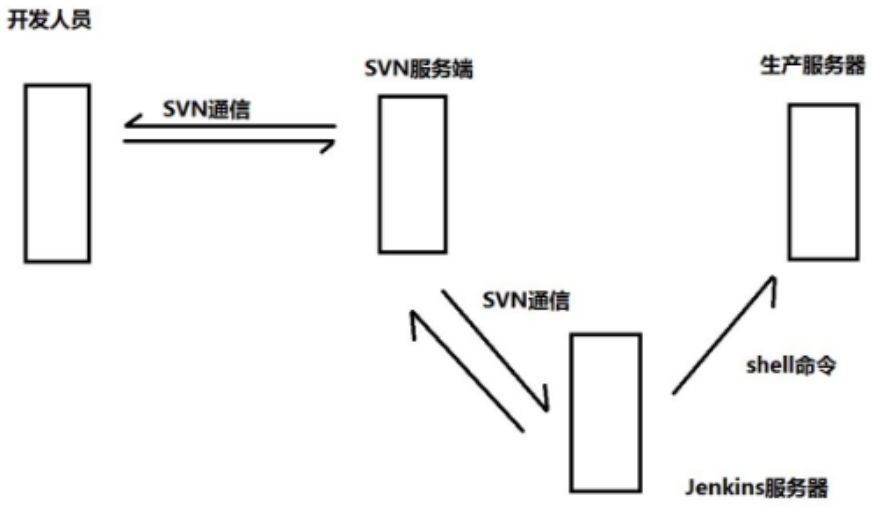
(6) 第五个*表示一周中的第几天，取值0~7，其中0和7代表的都是周日

使用举例：

每隔10分钟构建一次：　　H/10 * * * *

每隔1小时构建一次：　　H H/1 * * *

每月30号构建一次：　　H H 30 * *



6 Git+Jenkins自动构建

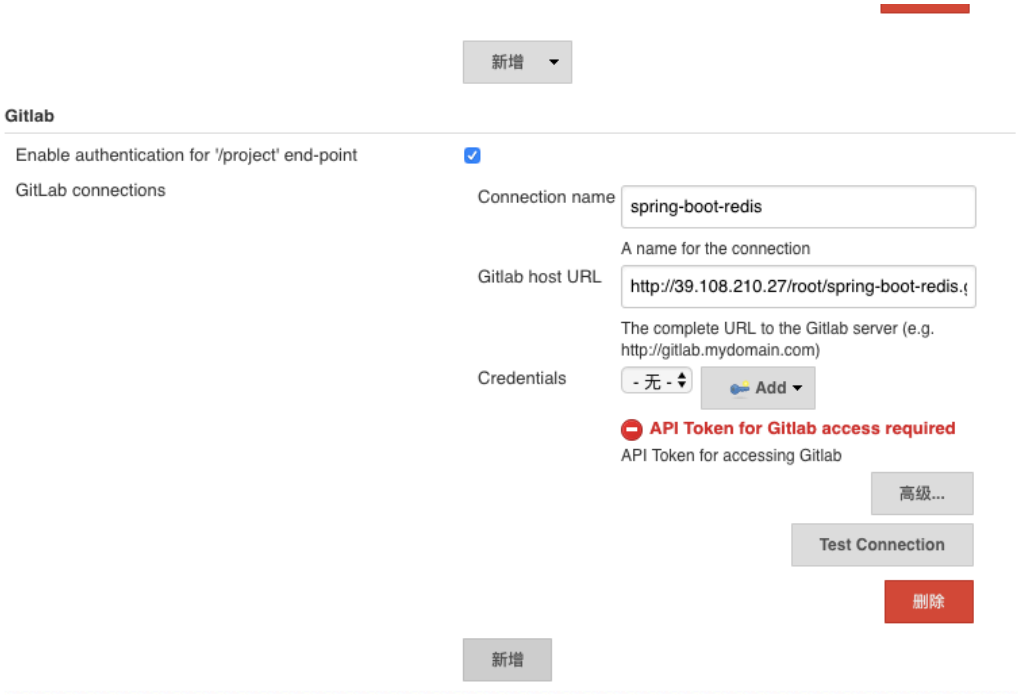
Git项目：https://github.com/alanjiang/hook.git

```
JDK1.8 : /usr/bin/java

Git1.8.3.1: /usr/bin/git

Maven3.6.0: /usr/home/softwares/apache-maven-3.6.0/
```

- **第一步：**

进入jenkins，点击“系统管理”， 点击 “系统设置”。 进入gitlab 选项：

要求填写 URL和凭据。



- **第二 步：配置gitlab 公匙+私匙**

例如：我的gitlab帐号为：root，对应的邮箱为：admin@example.com

（1）生成公、私匙放在本机

/Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa

$mkdir -p /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/

$touch /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa

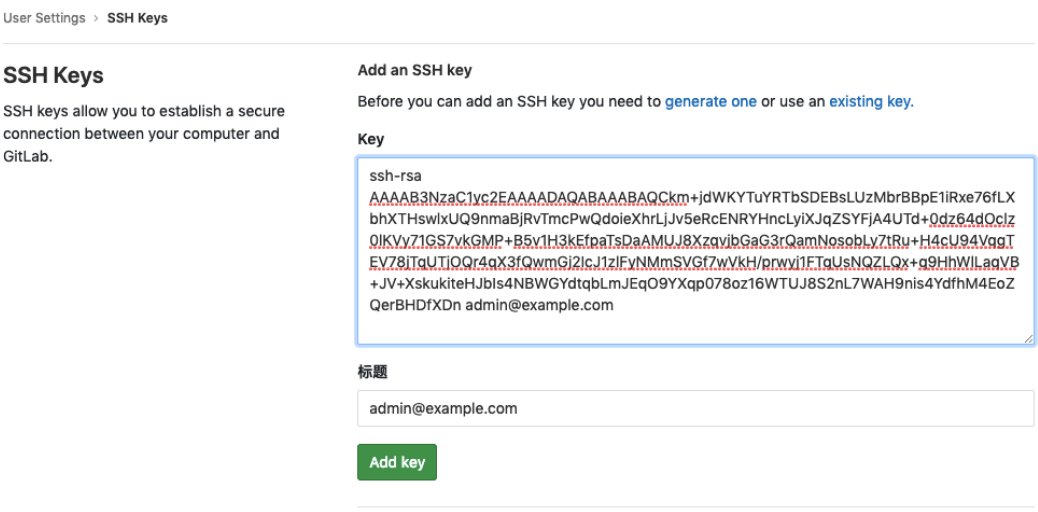
执行：

$ssh-keygen -t rsa -C "admin@example.com"

最后生成了公匙和私匙，如下：

```
zhangxiaodeMacBook-Air:.ssh zhangxiao$ pwd
/Users/zhangxiao/Desktop/keys-gitlab/root/.ssh
zhangxiaodeMacBook-Air:.ssh zhangxiao$ ls
id_rsa          id_rsa.pub
```

（2）将公匙id_rsa.pub的内容配置在gitlab上。



点击“Add Key” 完成配置。

(3) 配置全局用户名变量

git config --global user.name "root"

git config --global user.name "admin@example.com"

(4) 添加sshkey至ssh-agent

$ssh-add /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa

```
zhangxiaodeMacBook-Air:gitlab-workspace zhangxiao$ ssh-add /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa
Enter passphrase for /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa:
Identity added: /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa (admin@example.com)
```

测试ssh-agent是否生效, 执行:

$ssh-agent

测试一下使用ssh协议git clone 一个gitlab上的项目:

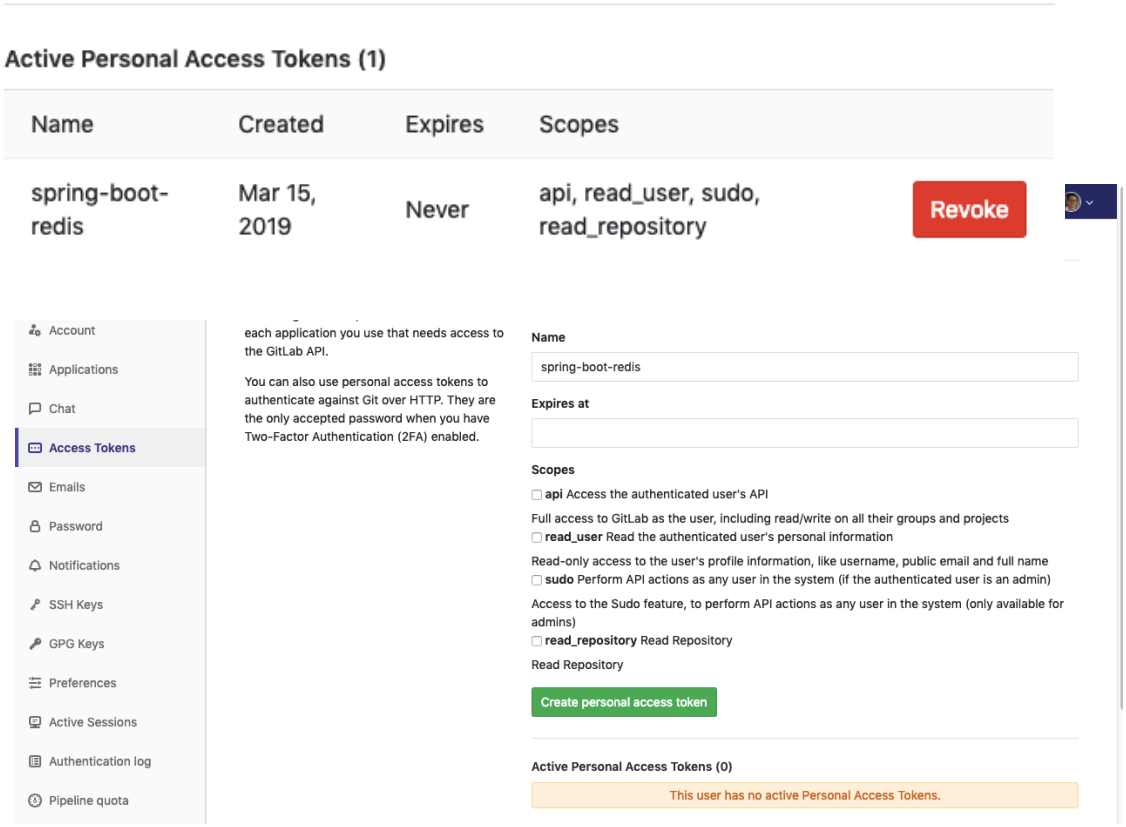
```
zhangxiaodeMacBook-Air:gitlab-workspace zhangxiao$ git clone git@39.108.210.27:root/spring-boot-redis.git
Cloning into 'spring-boot-redis'...
remote: Counting objects: 568, done.
remote: Compressing objects: 100% (384/384), done.
remote: Total 568 (delta 145), reused 535 (delta 117)
Receiving objects: 100% (568/568), 246.26 KiB | 153.00 KiB/s, done.
Resolving deltas: 100% (145/145), done.
```

(5)gitlab 上生成一个 “个人访问的Token”。

请记住这个gitlab的API Token:

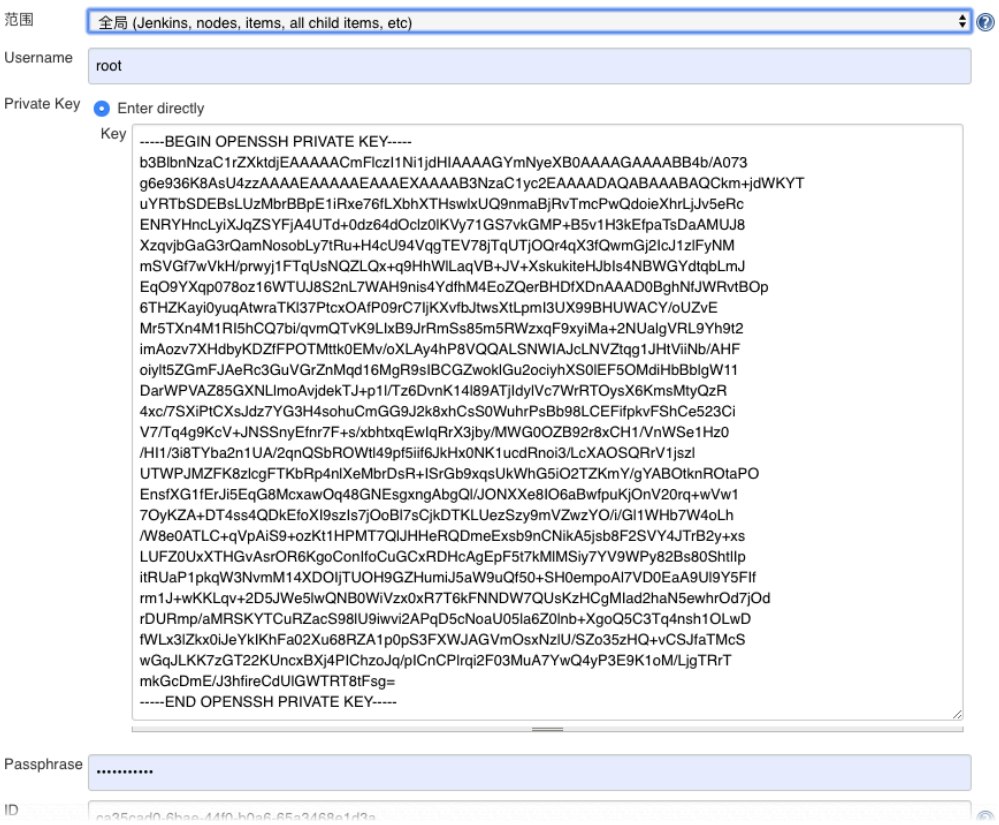
g97g3_wRyVcDRc9ELa_6

Rnner Token: wLKrAnSAYzFzPbDQcedm



(6) 在jenkins上配置凭据 ： 选择 Gitlab API token ，填写上一步的API Token。

也可以创建一个基于SSH公、私匙的凭据： 如下图：

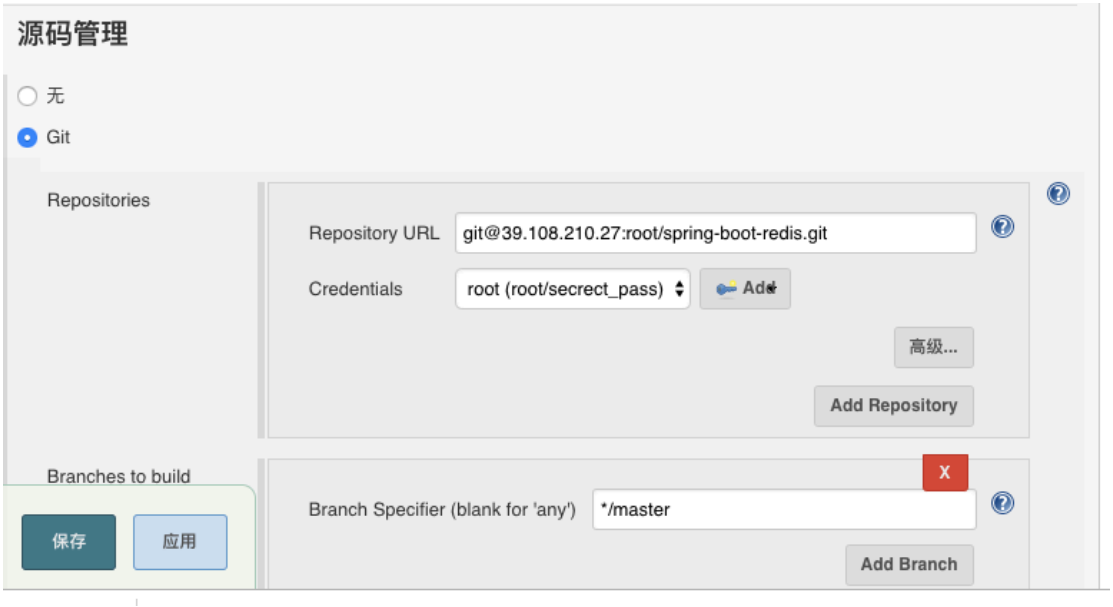


注意：这里的私匙内容与gitlab中设置的公匙是成对的。

（7）再次回到“系统管理“ –》 ”系统设置“：

（8）点“配置”

“源码管理”选项卡中填写 git项目的URL， credentials 中选取上步骤中（6)的设置的SSH凭据，也可以是API TOKEN，这儿演示SSH凭据。



（9）构建选项卡中，选择 “执行SHELL”

可以将MAVEN的构建、打包命令写在执行的SHEEL脚本中。

可以包括：

运行sonar:sonar 分析；

将WAR包部署在远程服务器上，并启动WEB应用；

可提交一个PULL REQUEST；

GIT分支合并。。。

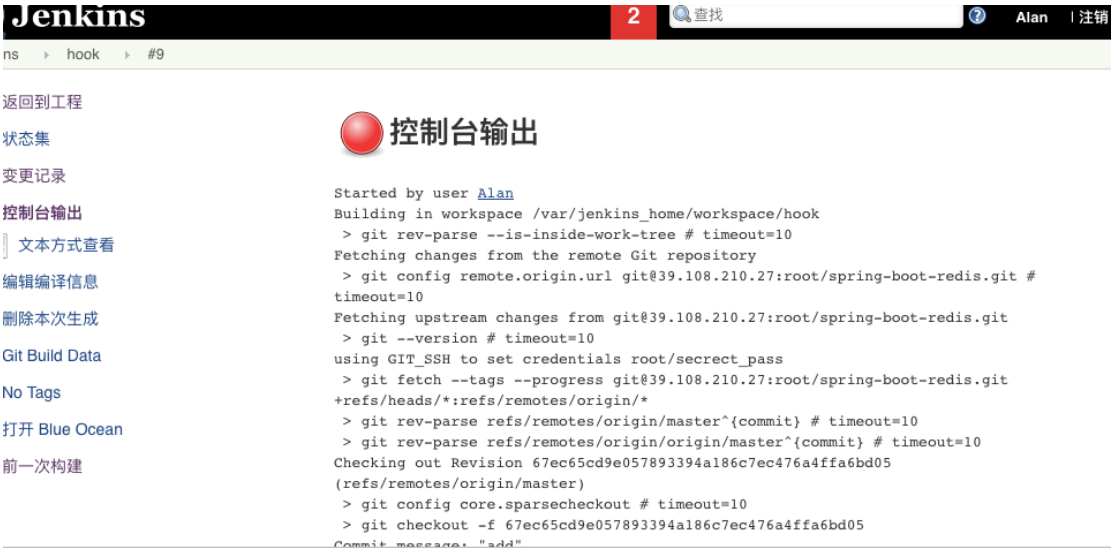
轻松完成CI/CD。

（10）构建后操作，可以配置邮箱。

点击“构建历史”，可以看见所有构建历史项目。

点击项目的名称 链接，进入构建的项目列表，可以查看日志等信息。

查看“控制台输出“， 可以查看输出的报告。



附录：

云服务中应用到的命令：

启动 停止Docker

$ sudo systemctl start docker

$sudo systemctl stop docker

$service docker status

Gitlab 服务：

`gitlab-ctl stop|start|restart`

参考：

https://jenkins.io/doc/