day02

gitlab

- 它是一个开源的git仓库服务器。用于实现代码集中托管。
- 分为企业版和CE社区版。
- 部署方式：软件包部署、容器部署。

通过容器部署gitlab服务器

- 将虚拟机192.168.4.20作为gitlab服务器。它需要4GB以上内存。
- 将/linux-soft/2/gitlab_zh.tar拷贝到192.168.4.20

```
[root@zzgrhel8 ~]# scp /linux-soft/2/gitlab_zh.tar
192.168.4.20:/root
```

- 部署gitlab容器

```
# 安装容器管理软件podman
[root@git ~]# yum install -y podman

# 修改192.168.4.20的ssh端口号。因为gitlab容器也要用到22端口，有冲突。
# +17是打开文件时，光标直接定位到第17行。
[root@git ~]# vim +17 /etc/ssh/sshd_config
 17  Port 2022
[root@git ~]# systemctl restart sshd
# ssh连接退出，再登陆时，需要指定端口号
[root@zzgrhel8 ~]# ssh -p2022 192.168.4.20

# 导入镜像。一个镜像可以创建很多容器。镜像是只读的，容器是可以改变的。
# 容器相当于是精简的虚拟机，可以像虚拟机一样，对外提供服务。
[root@git ~]# podman load < gitlab_zh.tar

# 查看导入的镜像
```

```
[root@git ~]# podman images
REPOSITORY            TAG      IMAGE ID       CREATED
    SIZE
localhost/gitlab_zh   latest   1f71f185271a   3 years
ago   1.73 GB

# 容器如果出现故障，首先的排错方法是重启它；如果无效，
删掉重建。
# 为了删容器，不丢失数据，需要把容器需要的数据保存在宿
主机上。在哪台主机上启动容器，哪台主机就是宿主机。
# 在192.168.4.20上创建用于保存容器数据的目录
[root@git ~]# mkdir -p /srv/gitlab/{config,logs,data}
[root@git ~]# ls /srv/gitlab/
config  data  logs

# gitlab容器需要/etc/resolv.conf文件。不存在则创建它
[root@git ~]# touch /etc/resolv.conf

# 创建容器
# -d后台运行。-h gitlab设置容器的主机名。--name gitlab
是podman ps查看到的容器名；-p指定发布的端口号，当访问宿
主机443/80/22端口时，这样的请求就发给容器的相关端口；--
restart always是开机自启；-v是映射路径，将容器中指定的
路径，映射到宿主机，以便保存容器产生的数据；最后的
gitlab_zh是镜像名。
[root@git ~]# podman run -d -h gitlab --name gitlab -p
443:443 -p 80:80 -p 22:22 --restart always -v
/srv/gitlab/config:/etc/gitlab -v
/srv/gitlab/logs:/var/log/gitlab -v
/srv/gitlab/data:/var/opt/gitlab gitlab_zh
# 查看容器
```

```
[root@git ~]# podman ps
# 如果一切正常，几分钟后，可以访问http://192.168.4.20/
```

配置gitlab

- 第一次登陆时，要求改密码。密码需要是复杂密码，如1234.com。修改之后，登陆的用户名是root。
- 改变显示配置。

项目


192.168.4.20/admin
GitLab
项目
群组
更多
应用
违规举报
0
部署密钥
服务模板
标记
外观
设置
折叠边栏
192.168.4.20/admin/appearance

登录/注册页面：

点击下面的保存后，LOGO图标将会改变。退出后，登陆界面也会有变化。

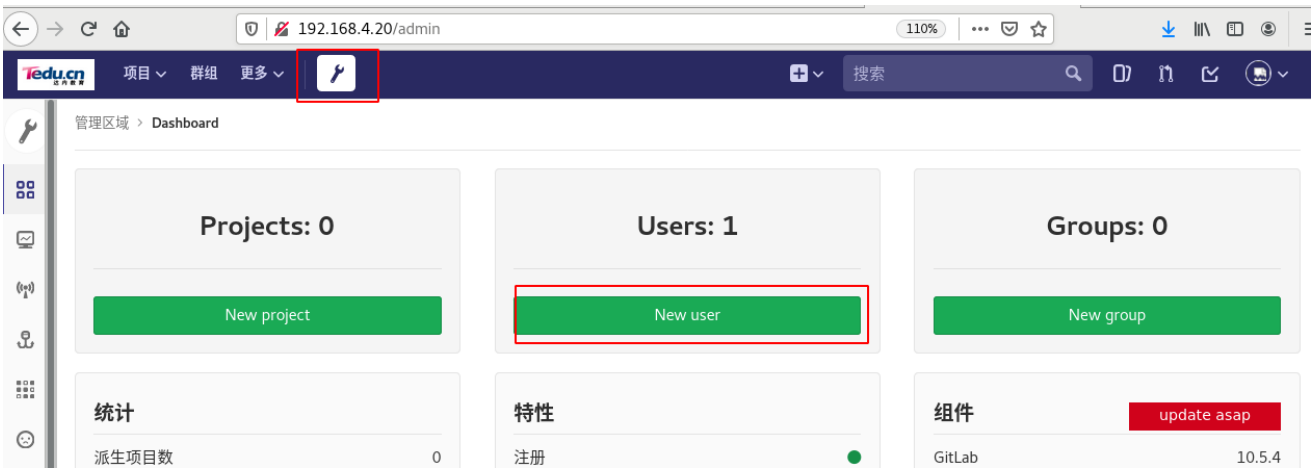
TEDU-云计算学院

达内云计算学院内部git服务器

```
echo 'Hello World!'
```

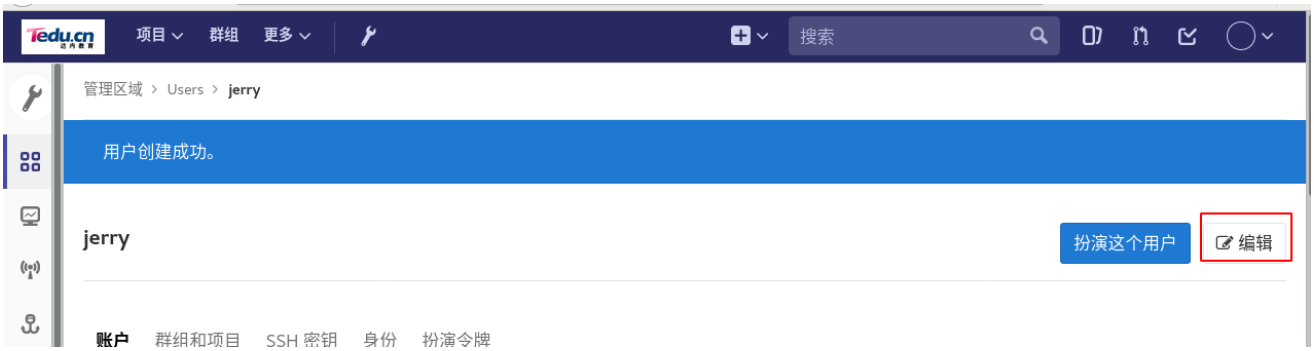
gitlab中主要的概念

- 用户：为使用gitlab的用户创建的账号。
- 组：用户的集合。一般可以为部门创建组。将来可以在项目上为组授权，组 中所有的用户都会得到相应的权限。
- 项目：用于保存代码文件的空间。
- 创建用户



填写截图上的几项后，其他使用默认配置，点保存。

创建好用户后，点击编辑，可以为他/她设置密码：

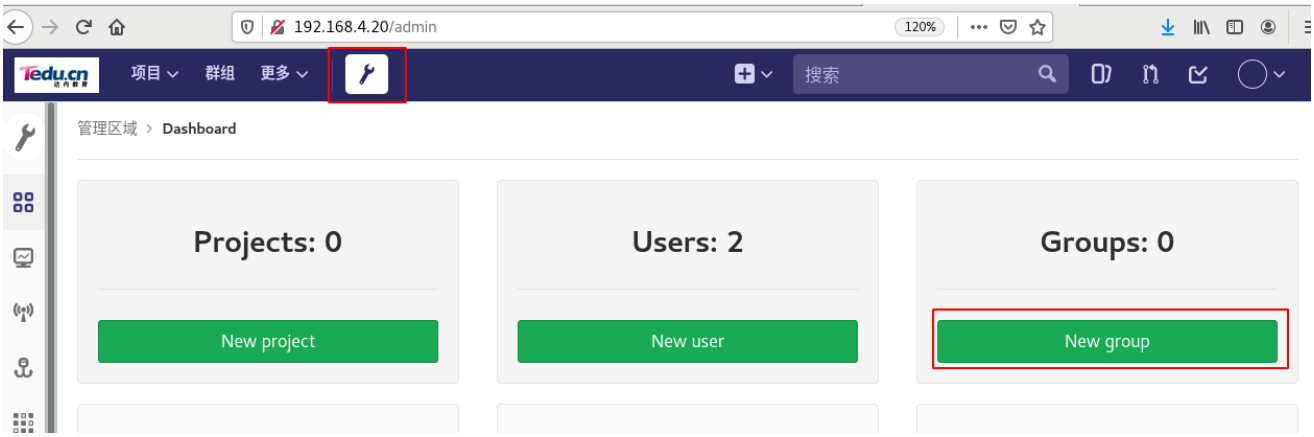


保存修改后，退出当前账号，使用新账号登陆测试。第一次登陆时，也是要求修改密码，新密码可以设置与旧密码一样。新建的jerry用户因为权限较小，所以看到的界面，没有root的功能多。

> 容器的删除

```
# 查看容器的名字和ID号，删除时，使用名字或ID号均可
[root@git ~]# podman ps
# 强制删除容器
[root@git ~]# podman rm -f gitlab
# 新建容器
[root@git ~]# podman run -d -h gitlab --name gitlab
-p 443:443 -p 80:80 -p 22:22 --restart always -v
/srv/gitlab/config:/etc/gitlab -v
/srv/gitlab/logs:/var/log/gitlab -v
/srv/gitlab/data:/var/opt/gitlab gitlab_zh
```

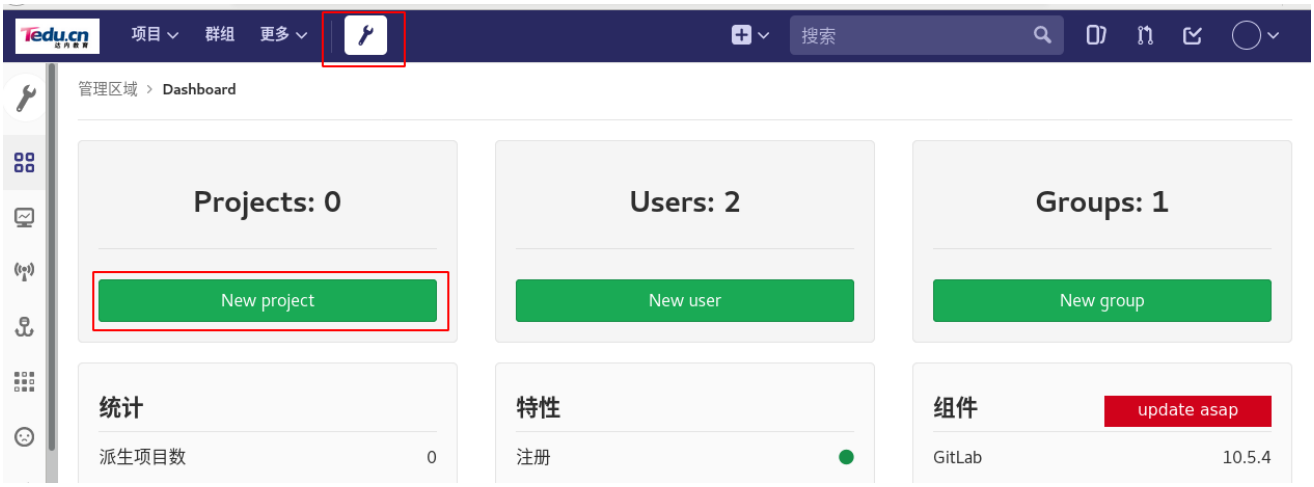
- 创建组。注意，需要使用root账号



点击“创建群组”

- 将jerry加到devops组中，角色是“主程序员”

- 创建项目



客户端上传代码到gitlab服务器

查看项目路径，采用http方式上传

- 查看项目说明

- 切换jerry用户，设置jerry的密码。

登录 注册

用户名或邮箱

jerry

密码

••••••••

记住我 忘记密码?

登录

尚未收到确认邮件? 重新发送确认邮件。

设置新密码

请立即设置一个新密码。
密码被成功修改后将会重定向到登录页面。

当前密码 ••••••••

密码 ••••••••

确认密码 ••••••••

设置新密码

Tedu.cn 达内教育

TEDU-云计算学院

登录 | 注册

用户名或邮箱

jerry

密码

••••••••

记住我 忘记密码?

登录

尚未收到确认邮件? 重新发送确认邮件。

- 在客户端192.168.4.10上下载项目，编写代码并上传

```
[root@develop ~]# git clone
http://192.168.4.20/devops/myproject.git
正克隆到 'myproject'...
warning: 您似乎克隆了一个空仓库。
[root@develop ~]# ls    # 本地出现一个myproject目录
anaconda-ks.cfg  myproject

# 创建说明文件并上传。一般来说，git服务器在首页默认可以显示readme文件的内容
[root@develop ~]# cd myproject/
[root@develop myproject]# vim README.md
- 这是我的第1个测试项目
```
echo 'Hello World!'
```
[root@develop myproject]# git add .    # 保存到暂存区
```

```
[root@develop myproject]# git commit -m "init data"  # 确认到版本库
# 将master分支推送到origin仓库。origin是默认仓库名。
[root@develop myproject]# git push -u origin master
Username for 'http://192.168.4.20': jerry   # 用户名
Password for 'http://jerry@192.168.4.20': 1234.com   # 密码
# 在服务器上刷新web页面


# 将来就可以重得操作：写代码、确认到版本库、上传到服务器
[root@develop myproject]# cp /etc/hosts .
[root@develop myproject]# git add .
[root@develop myproject]# git commit -m "add hosts"
[root@develop myproject]# git push   # 不必再使用-u选项
Username for 'http://192.168.4.20': jerry
Password for 'http://jerry@192.168.4.20': 1234.com


# 模拟另一个客户端同步数据
[root@zzgrhel8 ~]# ssh 192.168.4.10
[root@develop ~]# cd /var/tmp/
[root@develop tmp]# git clone http://192.168.4.20/devops/myproject.git
[root@develop tmp]# ls
myproject
[root@develop tmp]# cd myproject/
[root@develop myproject]# ls
hosts  readme.md
```

```
# 在家目录的myproject中上传新文件
[root@develop myproject]# cp /etc/issue .
[root@develop myproject]# ls
hosts  issue  readme.md
[root@develop myproject]# git add .
[root@develop myproject]# git commit -m "add issue"
[root@develop myproject]# git push
Username for 'http://192.168.4.20': jerry
Password for 'http://jerry@192.168.4.20': 1234.com

# 在/tmp/myproject中同步数据
[root@develop myproject]# git pull
[root@develop myproject]# ls
hosts  issue  readme.md
```

使用ssh免密推送代码

- 本质上与ssh免密登陆服务器一样。

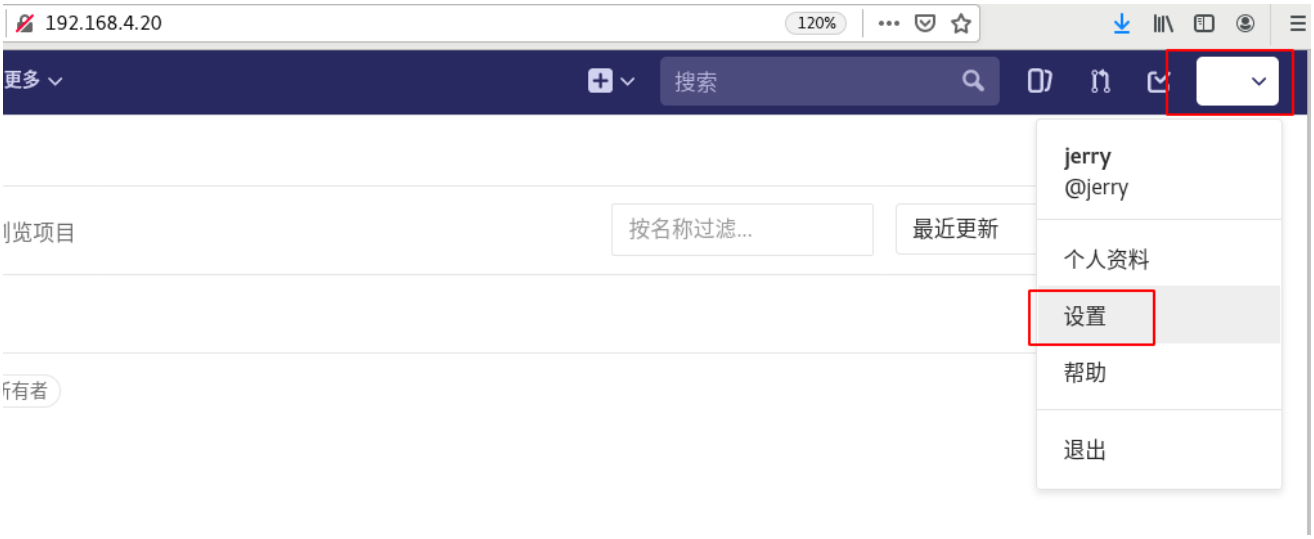
1. 在客户端192.168.4.10上生成密钥对

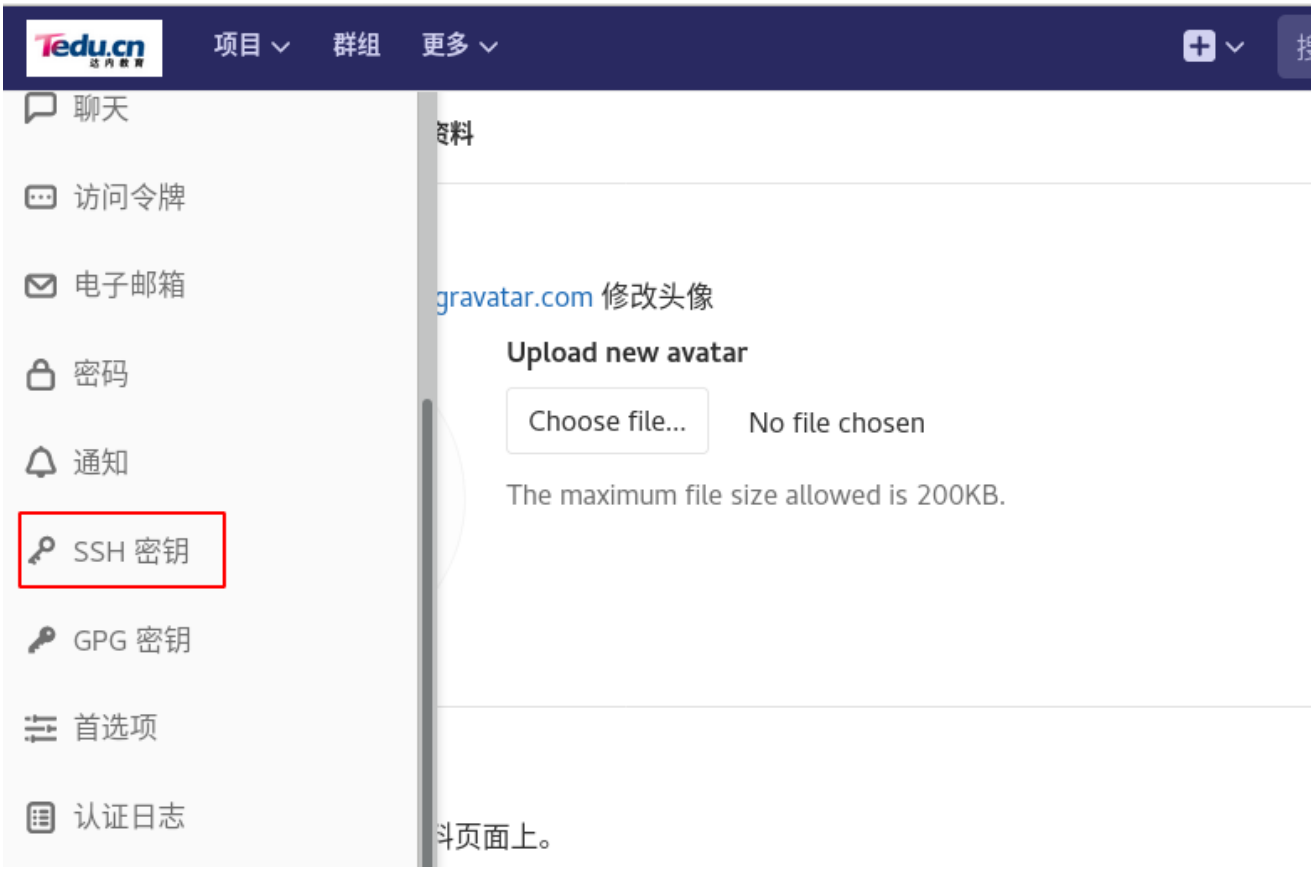
```
[root@develop myproject]# ssh-keygen   # 三个问题，都直接回车
```

2. 将公钥保存到gitlab服务器

```
# 查看并复制公钥内容
[root@develop myproject]# cat ~/.ssh/id_rsa.pub
```

在gitlab上切换Jerry用户登陆





把jerry的公钥粘贴到密钥框中

3. 将推送代码的方式改为ssh

查看ssh路径

在192.168.4.10上将推送代码的路径改为ssh的方式

```
# 查看仓库信息，当前是http方式
[root@develop myproject]# git remote -v
```

```
origin  http://192.168.4.20/devops/myproject.git
(fetch)
origin  http://192.168.4.20/devops/myproject.git (push)

# 删除http的路径
[root@develop myproject]# git remote remove origin

# 添加ssh路径
[root@develop myproject]# git remote add origin
git@192.168.4.20:devops/myproject.git

# 查看修改后的路径
[root@develop myproject]# git remote -v
origin  git@192.168.4.20:devops/myproject.git (fetch)
origin  git@192.168.4.20:devops/myproject.git (push)

# 推送代码测试
[root@develop myproject]# cp /etc/passwd .
[root@develop myproject]# git add .
[root@develop myproject]# git commit -m "add passwd"
[root@develop myproject]# git push -u origin master  #
不再需要密码
[root@develop myproject]# git push
```

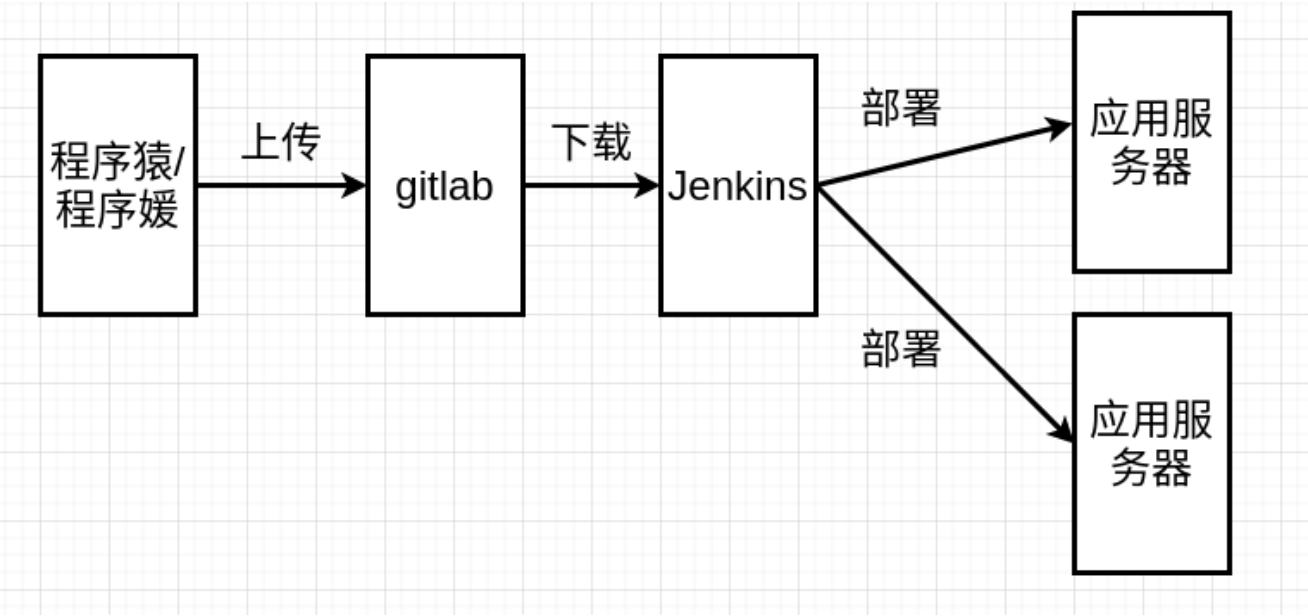
巩固练习

1. 在gitlab上新建名为myweb的项目，为devops组创建，可见等级为公开。

2. 将192.168.4.10上的myweb项目关联到gitlab的myweb，以ssh方式关联。
3. 在192.168.4.10上，把myweb目录中文件上传。

```
[root@develop ~]# cd myweb/
[root@develop myweb]# git remote add origin
git@192.168.4.20:devops/myweb.git
[root@develop myweb]# git push -u origin master
# 在gitlab上查看myweb项目
```

CI（持续集成）/CD（持续交付）



软件程序上线流程

1. 程序员将代码上传到gitlab服务器
2. 云计算工程师，通过jenkins服务器自动下载gitlab上的代码
3. 云计算工程师编写自动部署到服务器上的脚本

安装Jenkins服务器

- jenkins的IP地址是：192.168.4.30。它必须能与其他主机通信
- 关闭selinux/防火墙
- 安装jenkins

```
# 安装依赖包
# jenkins需要通过git下载代码，所以装git。
# jenkins是java程序，所以装java
# postfix和mailx是邮件程序，jenkins可以通过它们给管理员发邮件
[root@jenkins ~]# yum install -y git postfix mailx java-11-openjdk

# 把jenkins软件包拷贝到192.168.4.30
[root@zzgrhel8 ~]# ls /linux-soft/2/jenkins*
/linux-soft/2/jenkins-2.263.1-1.1.noarch.rpm
/linux-soft/2/jenkins_plugins.tar.gz
```

```
[root@zzgrhel8 ~]# scp /linux-soft/2/jenkins*
192.168.4.30:/root/

# 在192.168.4.30上安装jenkins
[root@jenkins ~]# yum install -y jenkins-2.263.1-
1.1.noarch.rpm

# 启动服务，并设置为开机自启
[root@jenkins ~]# systemctl enable jenkins
jenkins.service is not a native service, redirecting to
systemd-sysv-install.   # 注意：这里不是错误，忽略即可
Executing: /usr/lib/systemd/systemd-sysv-install enable
jenkins
[root@jenkins ~]# systemctl start jenkins
```

- 访问http://192.168.4.30:8080，进行初始化

```
# 查看初始化密码
[root@jenkins ~]# cat
/var/lib/jenkins/secrets/initialAdminPassword
2c58512973be4a44aec3ef5c1463d00a
```

把查看到的密码粘贴到文本框中，如下：

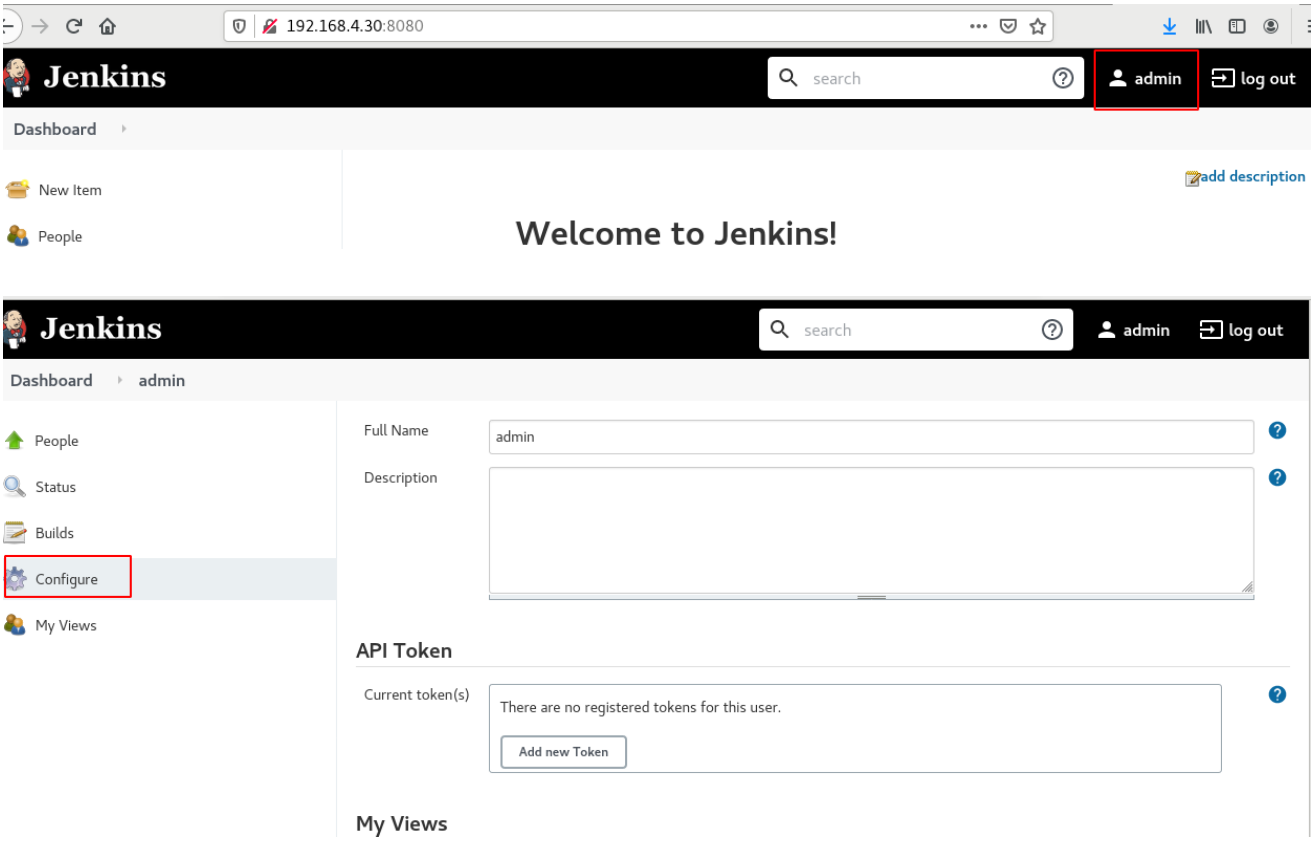
不用创建管理员，使用自带的admin


新手入门
创建第一个管理员用户
Username:
Password:
Confirm password:
Full name:
Jenkins 2.263.1
使用admin账户继续
保存并完成


新手入门
实例配置
Jenkins URL:
http://192.168.4.30:8080/
Jenkins URL 用于给各种Jenkins资源提供绝对路径链接的根地址。 这意味着对于很多Jenkins特色是需要正确设置的，例如：邮件通知、PR状态更新以及提供给构建步骤的BUILD_URL环境变量。
推荐的默认值显示在尚未保存，如果可能的话这是根据当前请求生成的。 最佳实践是要设置这个值，用户可能会需要用到。这将会避免在分享或者查看链接时的困惑。
Jenkins 2.263.1
现在不要
保存并完成


新手入门
Jenkins已就绪！
你已跳过创建admin用户的步骤。要登录请使用用户名：'admin' 及用于访问安装向导的管理员密码。
Jenkins安装已完成。
开始使用Jenkins
Jenkins 2.263.1

- 修改admin密码


192.168.4.30:8080
Jenkins
search
admin
log out
Dashboard
New Item
People
add description
Welcome to Jenkins!

Jenkins
search
admin
log out
Dashboard
admin
People
Status
Builds
Configure
My Views
Full Name
admin
Description
API Token
Current token(s)
There are no registered tokens for this user.
Add new Token
My Views

Password

Password: ••••••

Confirm Password: ••••••

SSH Public Keys

SSH Public Keys

Session Termination

Terminate All Sessions

Setting for search

Save Apply

Welcome to Jenkins!

admin

••••••

Sign in

Keep me signed in