# gitlab的搭建与使用

本文搭建的gitlab是基于yum的方式进行安装的，具体的安装步骤如下：

cat /opt/gitlab/embedded/service/gitlab-rails/VERSION

## 安装

（1）、配置yum源

vim /etc/yum.repos.d/gitlab-ce.repo #复制如下内容并保存，注意配置是centos7的地址

[gitlab-ce]

name=gitlab-ce

baseurl=https://mirrors.tuna.tsinghua.edu.cn/gitlab-ce/yum/el7/

Repo\_gpgcheck=0

Enabled=1

Gpgkey=https://packages.gitlab.com/gpg.ke

如果是centos6的，baseurl修改为

[gitlab-ce]

name=gitlab-ce

baseurl=https://mirrors.tuna.tsinghua.edu.cn/gitlab-ce/yum/el6/

Repo\_gpgcheck=0

Enabled=1

Gpgkey=https://packages.gitlab.com/gpg.ke

1. 、更新本地yum缓存

Yum clean all

yum makecache

1. 安装指定版本，此处为9.5.3

yum install gitlab-ce-9.5.3 #(yum intall gitlab-ce自动安装最新版)

注意安装的过程中可能会报如下错误

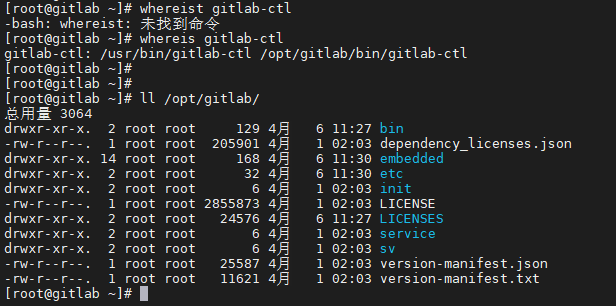


解决办法：

yum -y install gitlab-ce-9.5.3 --nogpgcheck # ####跳过校验安装



查看git安装位置：



（5）、更改配置

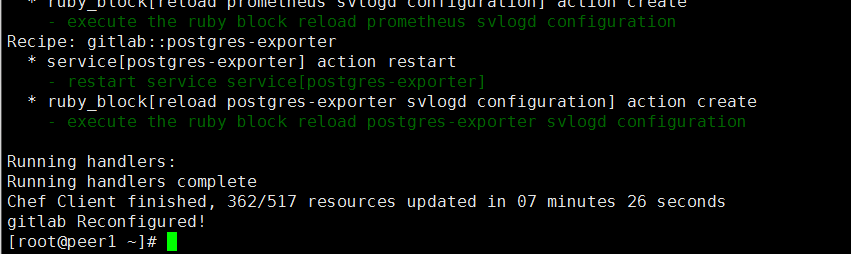
vim /etc/gitlab/gitlab.rb

找到 external\_url 'http://localhost:8081'

修改成你的地址，此处为http://192.168.10.235

（6）、对GitLab进行编译

gitlab-ctl reconfigure #修改配置后，执行 需要较长时间



（7）、清除缓存

gitlab-rake cache:clear RAILS\_ENV=production #需要等待较长时间，无反馈

（8）、启动gitlab服务，以及关闭防火墙等操作

gitlab-ctl start # 启动所有 gitlab 组件；并加入开机启动项里面:/etc/rc.local

gitlab-ctl stop # 停止所有 gitlab 组件；

gitlab-ctl restart # 重启所有 gitlab 组件；

gitlab-ctl status # 查看服务状态；

gitlab-ctl tail # 查看日志；

关闭防火墙：

service iptables stop ###centos6关闭防火墙

systemctl stop firewalld.service ###centos7关闭防火墙

systenctl disable firewalld.service ###cent0s7 关闭防火墙

设置selinux：

[root@peer1 ~]# getenforce

Enforcing #查看selinux模式，Enforcing为强制模式

[root@peer1 ~]# setenforce 0 #设置selinux模式：1为强制模式、2为宽松模式

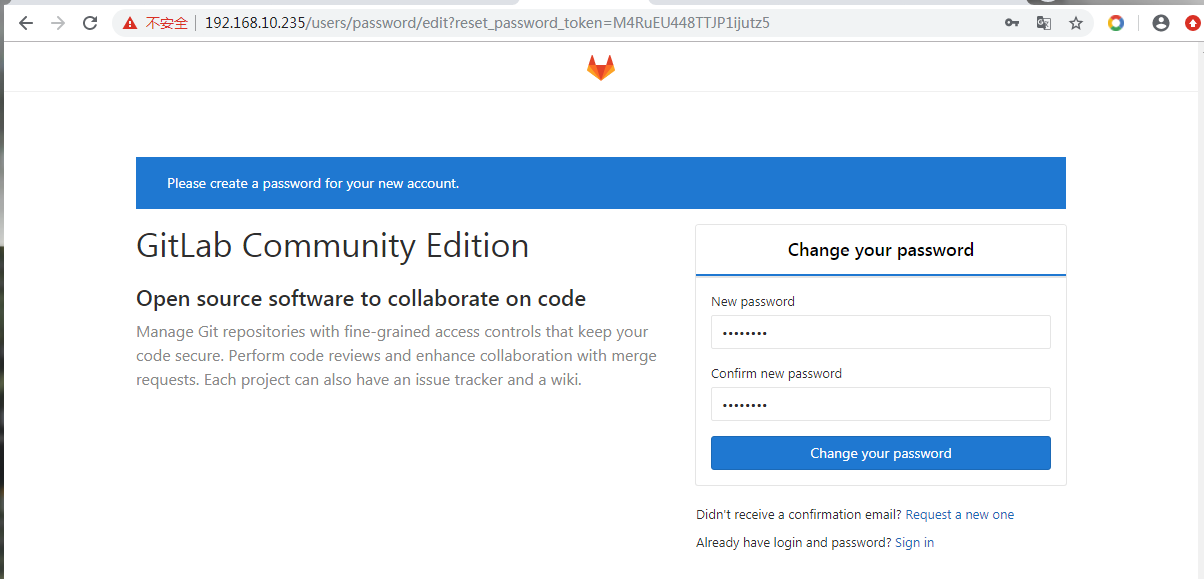
[root@peer1 ~]# getenforce

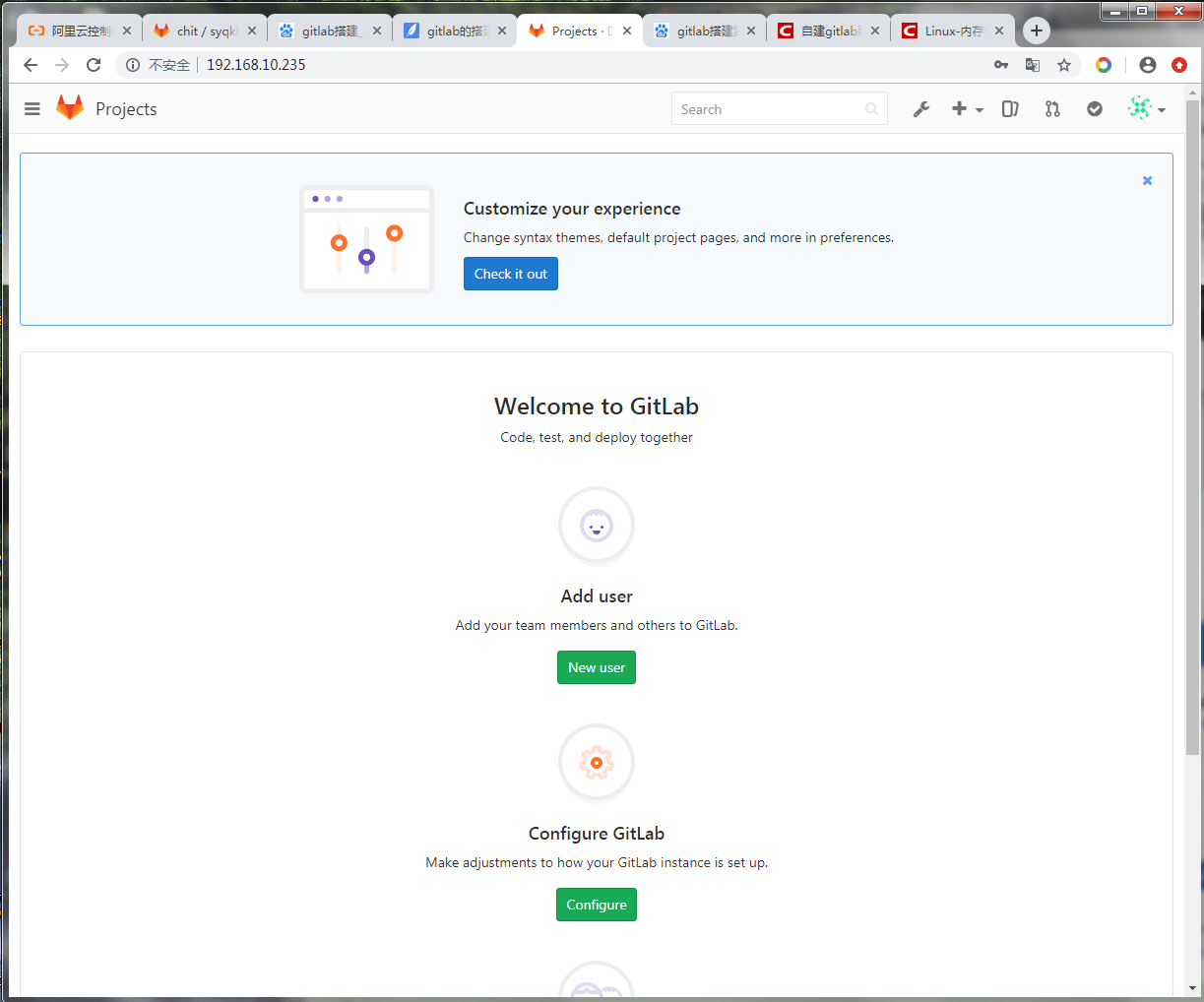
Permissive #Permissive为宽松模式

1. 、访问gitlab：192.168.1.157:8081，gitlab默认的root账号密码为qazwsxed，首次进入需要修改root密码

如果访问报错：502或者无法连接，可能是服务器的内存不够，正常至少要4G的内存

新密码为：123qqq...AAA





## Gitlab数据恢复：

[root@peer1 backups]# gitlab-ctl restart #启动gitlab

ok: run: gitaly: (pid 8608) 0s

ok: run: gitlab-monitor: (pid 8610) 0s

ok: run: gitlab-workhorse: (pid 8616) 1s

ok: run: logrotate: (pid 8633) 0s

ok: run: nginx: (pid 8640) 0s

ok: run: node-exporter: (pid 8645) 1s

ok: run: postgres-exporter: (pid 8649) 0s

ok: run: postgresql: (pid 8653) 1s

ok: run: prometheus: (pid 8661) 0s

ok: run: redis: (pid 8665) 1s

ok: run: redis-exporter: (pid 8672) 0s

ok: run: sidekiq: (pid 8677) 0s

ok: run: unicorn: (pid 8684) 1s

[root@peer1 backups]# gitlab-ctl stop unicorn #停止unicorn服务

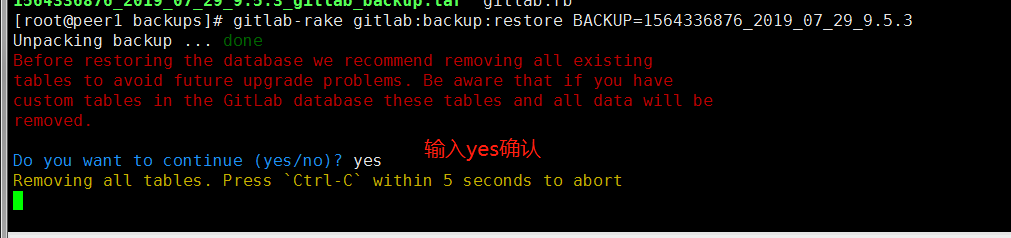
ok: down: unicorn: 1s, normally up

[root@peer1 backups]# gitlab-ctl stop sidekiq #停止sidekiq服务

ok: down: sidekiq: 0s, normally up

[root@peer1 backups]# gitlab-rake gitlab:backup:restore BACKUP=1564336876\_2019\_07\_29\_9.5.3

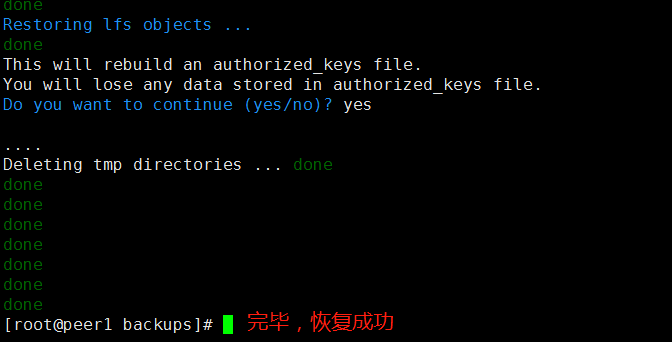
恢复期间第一个需要确认的：警告输入yes



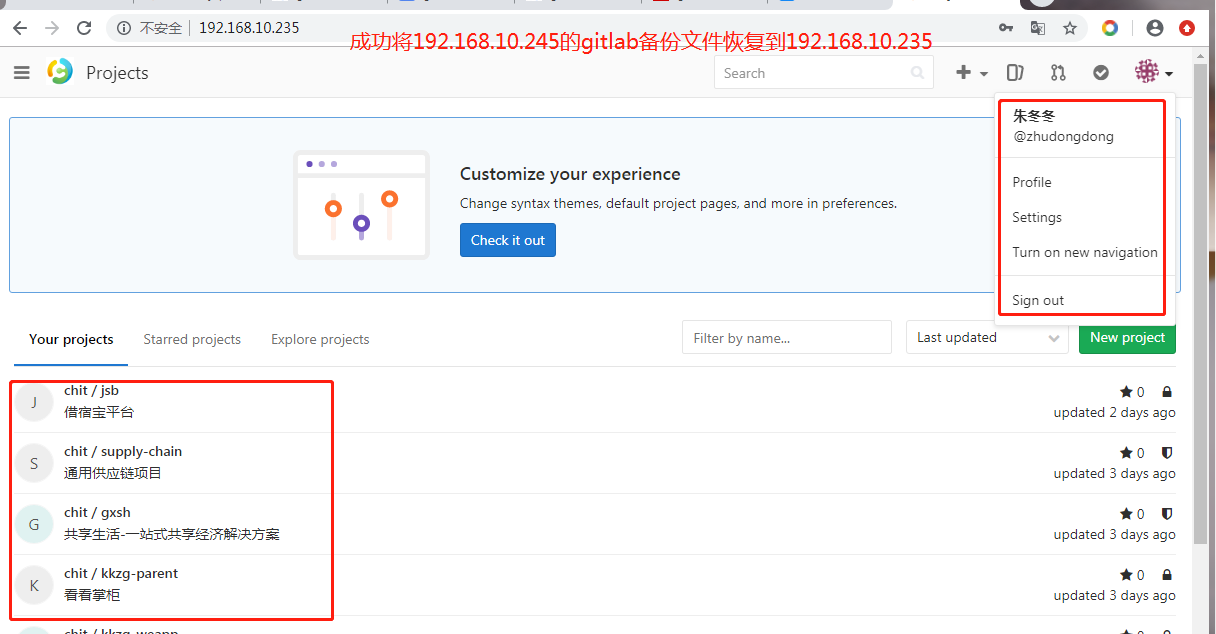
恢复期间第二个需要确认的：

英文翻译： 这将重建授权的\u密钥文件。

您将丢失任何存储在授权密钥文件中的数据。



登陆验证：



# Gitlab升级

## Gitlab同版本迁移

新的服务器安装相同版本gitlab, 将备份包放在gitlab默认备份目录（/var/opt/gitlab/backup）

cat /opt/gitlab/embedded/service/gitlab-rails/VERSION #先查看版本

9.5.3

[root@localhost ~]# yum install gitlab-ce-9.5.3 –nogpgcheck #安装相同版本

[root@gitlab ~]# vim /etc/gitlab/gitlab.rb #配置访问地址

external\_url 'http://192.168.10.246'

[root@gitlab ~]# vim /etc/selinux/config #修改selinux改为disabled

[root@gitlab ~]# setenforce 0

[root@gitlab ~]# getenforce

[root@gitlab ~]# yum -y install net-tools

[root@gitlab ~]# netstat -unltp |grep 80

[root@gitlab ~]# gitlab-ctl status

run: gitaly: (pid 21060) 365s; run: log: (pid 20355) 445s

………..

run: redis: (pid 19895) 575s; run: log: (pid 19894) 575s

run: redis-exporter: (pid 20801) 409s; run: log: (pid 20800) 409s

run: sidekiq: (pid 20294) 451s; run: log: (pid 20293) 451s

run: unicorn: (pid 20226) 457s; run: log: (pid 20225) 457s

[root@gitlab ~]#

[root@gitlab ~]# gitlab-ctl stop unicorn

ok: down: unicorn: 0s, normally up

[root@gitlab ~]#

[root@gitlab ~]# gitlab-ctl stop sidekiq

ok: down: sidekiq: 0s, normally up

[root@gitlab ~]# chmod 777 /var/opt/gitlab/backups/1585850463\_2020\_04\_03\_9.5.3\_gitlab\_backup.tar #赋予权限

[root@gitlab ~]# gitlab-rake gitlab:backup:restore BACKUP=1585850463\_2020\_04\_03\_9.5.3

**遇到的两个问题：**

[root@gitlab ~]# gitlab-rake gitlab:backup:restore BACKUP=1585850463\_2020\_04\_03\_9.5.3

No backups found in /var/opt/gitlab/backups #要把备份下来的包放到默认备份目录/var/opt/gitlab/backups

Please make sure that file name ends with \_gitlab\_backup.tar

[root@gitlab ~]# mv 1585850463\_2020\_04\_03\_9.5.3\_gitlab\_backup.tar /var/opt/gitlab/backups/

[root@gitlab ~]# gitlab-rake gitlab:backup:restore BACKUP=1585850463\_2020\_04\_03\_9.5.3

Unpacking backup ... tar: 1585850463\_2020\_04\_03\_9.5.3\_gitlab\_backup.tar：无法 open: 权限不够

tar: Error is not recoverable: exiting now #拷贝来的备份文件没又权限

unpacking backup failed

[root@gitlab ~]# gitlab-ctl start sidekiq #启动sidekiq服务

ok: run: sidekiq: (pid 22774) 0s

[root@gitlab ~]# gitlab-ctl start unicorn #启动unicorn 服务

ok: run: unicorn: (pid 22788) 0s

## gitlab自动备份：

备份命令： gitlab-rake gitlab:backup:create

[root@gitlab ~]# cd /var/opt/gitlab/ backups/

[root@gitlab backups]# vim auto\_backup\_to\_remote.sh

+++++++++自动备份脚本内容++++++++++++

#! /bin/bash

gitlab-rake gitlab:backup:create #创建备份文件

# gitlab 服务器备份路径

LocalBackDir=/var/opt/gitlab/backups

# 远程备份服务器 gitlab备份文件存放路径

RemoteBackDir=/io/gitlab\_backup

# 远程备份服务器 登录账户

RemoteUser=root

# 远程备份服务器 IP地址

RemoteIP=192.168.10.243

#当前系统日期

DATE=`date +"%Y-%m-%d"`

#Log存放路径

LogFile=$LocalBackDir/log/$DATE.log

# 查找 gitlab本地备份目录下 时间为300分钟之内的，并且后缀为.tar的gitlab备份文件

BACKUPFILE\_SEND\_TO\_REMOTE=$(find $LocalBackDir -type f -mmin -300 -name '\*.tar\*')

#新建日志文件

touch $LogFile

#追加日志到日志文件

echo "Gitlab auto backup to remote server, start at $(date +"%Y-%m-%d %H:%M:%S")" >> $LogFile

echo "---------------------------------------------------------------------------" >> $LogFile

# 输出日志，打印出每次scp的文件名

echo "---------------------The file to scp to remote server is: $BACKUPFILE\_SEND\_TO\_REMOTE-------------------------------" >> $LogFile

#备份到远程服务器

scp $BACKUPFILE\_SEND\_TO\_REMOTE $RemoteUser@$RemoteIP:$RemoteBackDir

#追加日志到日志文件

echo "---------------------------------------------------------------------------" >> $LogFile

scp /etc/gitlab/gitlab.rb $RemoteIP:$RemoteBackDir

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

[root@gitlab backups]# chmod +x auto\_backup\_to\_remote.sh

**设置免密登录远程主机：**

[root@gitlab backups]# ssh-keygen

[root@gitlab backups]# ssh-copy-id 192.168.10.243

[root@gitlab backups]# ssh 192.168.10.243 #测试免密登录

[root@gitlab backups]# vim /etc/gitlab/gitlab.rb #配置gitlab配置文件，

gitlab\_rails['manage\_backup\_path'] = true

gitlab\_rails['backup\_path'] = "/var/opt/gitlab/backups"

gitlab\_rails['backup\_keep\_time'] = 604800 #保存一周以内的备份，单位秒，一天为86400

[root@gitlab backups]# gitlab-ctl reconfigure #更新配置

**配置任务计划：**

[root@gitlab backups]# crontab -e #或者直接编辑/var/spool/cron/root

crontab: installing new crontab #添加内容无误之后提示创建新的任务

30 3 \* \* \* sh /var/opt/gitlab/backups/auto\_backup\_to\_remote.sh #添加的内容

[root@gitlab backups]# crontab -l

30 3 \* \* \* sh /var/opt/gitlab/backups/auto\_backup\_to\_remote.sh

**Contab 配置说明：**

* （系统级的）做系统级配置我们会直接配置编辑 vi /etc/crontab，配置完需要重启crond服务

Vi /etc/crontab这种方法只有root用户能用，这种方法更加方便与直接直接给其他用户设置计划任务，而且还可以指定执行shell等等，

/etc/rc.d/init.d/crond restart

* （用户级的）一般还是建议大家使用 crontab -e ，这样系统也会帮着检查我们配置的脚本语法。

这样执行以后，属于用户自定义的，会被写到/var/spool/cron 目录下，生成一个和当前登录用户名一致的文件，文件内容就是我们编辑的定时脚本。

1. crontab -e是某个用户的周期计划任务； /etc/crontab是系统的周期任务
2. vi /etc/crontab 编辑定时脚本，不会检查脚本语法 内容如下：
3. crontab -e 然后直接编辑定时脚本，会检查脚本语法：
4. /etc/crontab 修改语法格式不一样，后者多一个 user 指定
   * 01 22 \* \* \* /sbin/shutdown -h now crontab -e配置案例
   * 01 22 \* \* \* root   /sbin/shutdown -h now /etc/crontab配置案例

## gitlab升级

官方网站：<https://docs.gitlab.com/ee/policy/maintenance.html#upgrade-recommendations>

Gitlab不支持跨版本升级，需要按照官方网站指南一步一步更换版本升级，例如：当前版本为9.2.6升级到12.8.5官方推荐需要从9.2.6升级到9.5.10再升级到10.8.7…. 具体为：9.2.6-> 9.5.10-> 10.8.7-> 11.11.8-> 12.0.12->12.8.5