# 安装环境

|  |  |
| --- | --- |
| 操作系统 | CentOS Linux release 7.2.1511 (Core) 64位 |
| 容器 | Docker version 1.12.2, build bb80604 |
| Gitlab | latest 最新稳定版（实际版本GitLab Community Edition 8.12.7 7429b21 Check） |
| 数据库 | Omnibus gitlab内置postgresql和redis |
| Web服务器 | Omnibus gitlab内置Nginx |

**命令说明：由于gitlab安装在docker容器中，凡是已gitllab\*开头的命令都需要进入容器执行。进入容器的命令如下:**

**# sudo docker exec –it gitlab /bin/bash**

# 安装Docker Engine

## 安装Docker Engine

安装Docker Engine需要root或者sudo用户的管理员权限。

* 添加docker的yum仓库

$ sudo tee /etc/yum.repos.d/docker.repo <<-'EOF'

[dockerrepo]

name=Docker Repository

baseurl=https://yum.dockerproject.org/repo/main/centos/7/

enabled=1

gpgcheck=1

gpgkey=https://yum.dockerproject.org/gpg

EOF

* 安装docker引擎

$ sudo yum -y install docker-engine

* 启用docker服务

$ sudo systemctl enable docker.service

* 启动docke进程

$ sudo systemctl **start** docker

* 验证docker是否正常运行

$ sudo docker run --rm hello-world

出现类似如下消息，即代表成功



## 创建docker用户和用户组

当你使用docker命令时，为了避免必须使用sudo，可以创建一个docker用户组，并且添加用户到这个组里面，当docker进程启动时，它会使得docker用户组获得操作系统的读写所有权。

* 创建docker用户组

$ groupadd docker

* 创建git用户，并添加到docker用户组中

$ useradd git

将git添加到docker用户组

$ usermod -aG docker git

* 使用git用户登录，将不需要使用sudo，即可操作以下命令

$ docker **run** hello-world

## 设置docker自启动

$ sudo systemctl enable docker

## 卸载docker

* 列出已经安装的docker包

$ sudo yum **list** installed | grep docker

* 删除安装包

$ sudo yum -y remove docker-engine.x86\_64

* 删除所有镜像、容器等

$ sudo rm -rf /var/lib/docker

# 安装gitlab

## 安装gitlab

执行以下命令将会去下载安装gitlab-ce版本容器。并且gitlab容器文件都存储在/data/gitlab目录中

$ sudo docker run --detach \

--hostname 10.163.37.9 \

--publish 443:443 --publish 80:80 --publish 88:22 \

--name gitlab \

--restart always \

--volume /data/gitlab/config:/etc/gitlab \

--volume /data/gitlab/logs:/var/log/gitlab \

--volume /data/gitlab/data:/var/opt/gitlab \

gitlab/gitlab-ce:latest

## 数据文件描述

数据文件描述如下

|  |  |  |
| --- | --- | --- |
| 本地位置 | 对应在docker容器位置 | 描述 |
| /data/gitlab/config | /etc/gitlab | 存储配置文件信息 |
| /data/gitlab/logs | /var/log/gitlab | 存储日志 |
| /data/gitlab/data | /var/opt/gitlab | 存储应用程序数据 |

## gitlab配置文件

由于是这个容器是使用官方的Omnibus GitLab安装包，因此所有的配置都是在/etc/gitlab/gitlab.rb文件里面做。

* 你可以先进入docker容器，然后编辑配置文件。

$ docker exec -it gitlab /bin/bash

* 也可以直接编辑配置文件。

$ docker exec -it gitlab vi /etc/gitlab/gitlab.rb

* 配置文件修改后需要输入如下命令使其生效

$ gitlab-ctl reconfigure

$ gitlab-ctl restart

## 配置smtp

修改配置文件/etc/gitlab/gitlab.rb为如下

gitlab\_rails['gitlab\_email\_from'] = ' git@lifeisgreat.com.cn'

gitlab\_rails['smtp\_enable'] = true

gitlab\_rails['smtp\_address'] = "smtp. lifeisgreat.com.cn "

gitlab\_rails['smtp\_port'] = 25

gitlab\_rails['smtp\_user\_name'] = "git@lifeisgreat.com.cn"

gitlab\_rails['smtp\_password'] = "Git,1234"

gitlab\_rails['smtp\_domain'] = "smtp. lifeisgreat.com.cn "

gitlab\_rails['smtp\_authentication'] = "login"

gitlab\_rails['smtp\_enable\_starttls\_auto'] = true

## 配置ldap

修改配置文件为/etc/gitlab/gitlab.rb如下

gitlab\_rails['ldap\_enabled'] = true

gitlab\_rails['ldap\_servers'] = YAML.load <<-'EOS' # remember to close this block with 'EOS' below

main: # 'main' is the GitLab 'provider ID' of this LDAP server

label: 'LDAP'

host: '192.168.16.226'

port: 389

uid: 'sAMAccountName'

method: 'plain' # "tls" or "ssl" or "plain"

bind\_dn: ' liliangang@headquarter.cn.lifeisgreat '

password: 'Abcd1234'

active\_directory: true

allow\_username\_or\_email\_login: true

block\_auto\_created\_users: false

base: 'ou=Accounts,dc=headquarter,dc=cn,dc=lifeisgreat'

user\_filter: ''

EOS

# 备份与还原

## 备份gitlab配置信息

进入定时任务编辑文件。

$ sudo crontab -e -u root

编辑备份脚本（每天凌晨3点备份配置文件）

$ 0 3 \* \* \*  tar cfz /data/gitlabbak/$(date”+git-\%s.tar”) /data/gitlab/config

## 还原gitlab配置信息

进入备份文件目录下。

$ cd /data/gitlabbak

找到对应备份文件\*.tar，解压并还原。

$ tar -xfz gitlab-xxxx.tar

还原配置文件后需要进行如下操作是配置生效。（需要进入容器）

$ gitlab-ctl reconfigure

$ gitlab-ctl restart

## 修改备份gitlab系统路径

进入容器

$ docker exec -it gitlab /bin/bash

编辑配置文件

$ vim /etc/gitlab/gitlab.rb

编辑内容如下（默认路径为/var/opt/gitlab/backups）

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

修改配置文件后需要使配置文件生效

$ gitlab-ctl reconfigure

$ gitlab-ctl restart

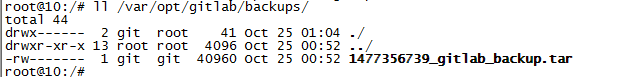
## 备份gitlab系统

执行备份语句

$ gitlab-rake gitlab:backup:create

进入备份目录查看备份情况

$ ll /var/opt/gitlab/backups



## 还原gitlab系统

停止进程连接数据库

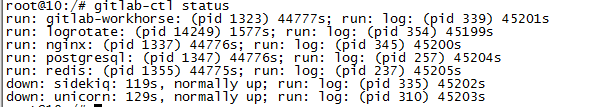
$ gitlab-ctl stop unicorn

$ gitlab-ctl stop sidekiq



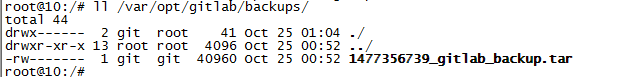
查看gitlab状态，看是否已停止连接数据库

$ gitlab-ctl status



查看备份路径下的备份文件

$ ll /var/opt/gitlab/backups



执行还原备份（以下1477356739是备份文件中的序列号）

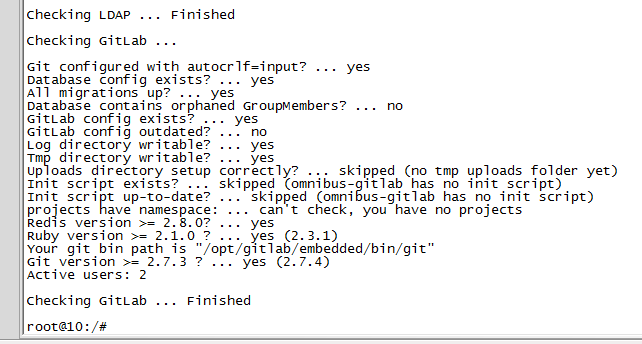
$ gitlab-rake gitlab:backup:restore BACKUP=1477356739

启动gitlab

$ gitlab-ctl start

校验是否恢复成功

$ gitlab-rake gitlab:check SANITIZE=true



# 常用命令

* 进入docker容器命令

$ sudo docker exec –it gitlab /bin/bash

* 使配置文件生效（在进入docker容器后执行）

$ gitlab-ctl reconfigure

$ gitlab-ctl restart

* gitlab服务启动关闭和重启（在进入docker容器后执行）

$ gitlab-ctl start|stop|restart

* 启动关闭和重启docker的gitlab镜像

$ sudo docker start|stop|restart gitlab

* 移除doker的gitlab镜像

$ sudo docker rm gitlab