How to Install GitLab Community Edition (CE) 11.x on CentOS 7

Last Updated: Fri, Oct 12, 2018

CentOSLinux GuidesProgrammingServer Apps

Using a Different System?

Since GitHub was acquired by Microsoft, quite a few developers have planned to migrate their own code repositories from github.com to an alternative self-hosted solution. GitLab Community Edition (CE) is the most common choice. As a sophisticated and flexible solution, GitLab CE can be deployed using various methods, but only the officially recommended method, the Omnibus package installation, will be covered herein.

Prerequisites

* A fresh Vultr CentOS 7 server instance with at least 4GB of memory. 8GB or more is recommended for serving up to 100 users. Say its IPv4 address is 203.0.113.1.
* A [sudo user](https://www.vultr.com/docs/how-to-use-sudo-on-debian-centos-and-freebsd).
* A domain gitlab.example.com being pointed towards the instance mentioned above.

Note: *When deploying on your own server instance, be sure to replace all example values with actual ones.*

Step 1: Perform basic tasks for hosting GitLab CE

Fire up an SSH terminal, and log in to your CentOS 7 server instance as a sudo user.

Add a swap partition and tweak the swappiness setting

When deploying GitLab CE 11.x on a machine with 4GB of memory, it's required to setup a 4GB swap partition for a smooth running.

sudo dd if=/dev/zero of=/swapfile count=4096 bs=1M

sudo chmod 600 /swapfile

sudo mkswap /swapfile

sudo swapon /swapfile

echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab

free -m

Note: *If you are using a different server size, the size of the swap partition may vary.*

For system performance purposes, it is recommended to configure the kernel's swappiness setting to a low value like 10:

echo 'vm.swappiness=10' | sudo tee -a /etc/sysctl.conf

sudo sysctl -p

cat /proc/sys/vm/swappiness

The output of the cat command will be 10.

Setup the machine's hostname and fully qualified domain name (FQDN)

Use the following commands to setup a hostname, gitlab, and an FQDN, gitlab.example.com, for the machine:

sudo hostnamectl set-hostname gitlab

cat <<EOF | sudo tee /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

203.0.113.1 gitlab.example.com gitlab

EOF

You can confirm the results:

hostname

hostname -f

Modify firewall rules

Allow inbound HTTP and HTTPS traffic:

sudo firewall-cmd --permanent --add-service=http

sudo firewall-cmd --permanent --add-service=https

sudo systemctl reload firewalld.service

Install the EPEL YUM repo and then update the system

sudo yum install -y epel-release

sudo yum -y update && sudo shutdown -r now

When the system is up and running again, log back in as the same sudo user to move on.

Step 2: Install required dependencies

Before installing GitLab CE, you need to install required dependencies:

sudo yum install -y curl policycoreutils-python openssh-server openssh-clients

Also, if you want to use Postfix to send notification messages, you need to install Postfix and modify firewall rules as follows:

sudo yum install -y postfix

sudo systemctl enable postfix.service

sudo systemctl start postfix.service

sudo firewall-cmd --permanent --add-service=smtp

sudo firewall-cmd --permanent --add-service=pop3

sudo firewall-cmd --permanent --add-service=imap

sudo firewall-cmd --permanent --add-service=smtps

sudo firewall-cmd --permanent --add-service=pop3s

sudo firewall-cmd --permanent --add-service=imaps

sudo firewall-cmd --reload

Having Postfix installed, you need to configure Postfix by editing its main config file /etc/postfix/main.cf in accordance with your actual server settings.

Note: *In addition to the above instructions, you need to submit a support ticket to cancel Vultr's default block on SMTP port 25.*

Alternatively, if you want to use another messaging solution, just skip installing Postfix and choose to use an external SMTP server after GitLab CE has been installed.

Step 3: Setup the GitLab RPM repo and then install GitLab CE

Setup the GitLab CE RPM repository on your system:

cd

curl -sS https://packages.gitlab.com/install/repositories/gitlab/gitlab-ce/script.rpm.sh | sudo bash

Next, install GitLab CE 11.x:

sudo EXTERNAL\_URL="http://gitlab.example.com" yum install -y gitlab-ce

The installation may take a while.

Finally, point your favorite web browser to http://gitlab.example.com, and then submit a new password as prompted to finish the installation.

From now on, use the below credentials to log in as the administrator:

* Username: root
* Password: <your-new-password>

Step 4: Enable HTTPS access by integrating a Let's Encrypt SSL certificate

For now, you have successfully installed GitLab CE 11.x on your server instance, and users can already visit the site using the HTTP protocol. For security purposes, its recommended to enable HTTPS access to your GitLab server by integrating a Let's Encrypt SSL certificate.

Use the vi editor to open the GitLab CE config file:

sudo vi /etc/gitlab/gitlab.rb

Find the following two lines:

external\_url 'http://gitlab.example.com'

# letsencrypt['contact\_emails'] = [] # This should be an array of email addresses to add as contacts

Replace them accordingly:

external\_url 'https://gitlab.example.com'

letsencrypt['contact\_emails'] = ['admin@example.com']

Save and quit:

:wq!

Run the following command to reconfigure GitLab CE using the updated settings:

sudo gitlab-ctl reconfigure

The reconfiguration will take a while.

After the reconfiguration is done, all users will be forced to use the HTTPS protocol when accessing the GitLab site.

Note: *After switching from HTTP to HTTPS, legacy cookies may cause a GitLab 422 error. Clearing cookies fixes this issue.*