FreeBSD Administration - Support #461

Installing GitLab 7.2 on FreeBSD

09/21/2014 06:52 PM - Daniel Curtis

Status:	Closed	**Start date:**	09/21/2014
Priority:	Normal	**Due date:**	
Assignee:	Daniel Curtis	**% Done:**	100%
Category:	Source Code Management	**Estimated time:**	2.00 hours
Target version:	FreeBSD 9	**Spent time:**	4.00 hours

Description

Setting up the Environment

- Start by making sure everything is up to date:

```
pkg update
pkg upgrade
```

- Install a few dependencies

```
pkg install -y git redis icu libxml2 libxslt python2 bash sudo gcc47 gmake autoconf automake l
ibtool bison readline libyaml sqlite3 gdbm cmake postgresql-libpqxx
```

- Add the GitLab user

```
pw add user -n git -m -s /usr/local/bin/bash -c "GitLab"
```

Install Redis

- Start Redis and enable it to start at boot:

```
echo 'redis_enable="YES"' >> /etc/rc.conf
service redis start
```

Install MariaDB 5.5

- This environment will be setup with MariaDB 5.5 for its MySQL server:

```
pkg install mariadb55-server mariadb55-client
```

- Ensure you have MySQL version 5.5.14 or later

```
mysql --version
```

- Start and enable MySQL at boot:

```
echo 'mysql_enable="YES"' >> /etc/rc.conf
service mysql-server start
```

- Secure your installation:

```
mysql_secure_installation
```

- Login to MySQL

```
mysql -u root -p
```

1. Create a user for GitLab

   ```
   CREATE USER 'git'@'localhost' IDENTIFIED BY 'SuperSecretPassword';
   ```

   NOTE: Change *SuperSecretPassword* to what ever password desired
2. Ensure you can use the InnoDB engine which is necessary to support long indexes.

   ```
   SET storage_engine=INNODB;
   ```

   NOTE: If this fails, check your MySQL config files (e.g. `/etc/mysql/*.cnf`, `/etc/mysql/conf.d/*`) for the setting "innodb = off"
3. Create the GitLab production database

   ```
   CREATE DATABASE IF NOT EXISTS `gitlabhq_production` DEFAULT CHARACTER SET `utf8` COLLATE `utf8_unicode_ci`;
   ```

4. Grant the GitLab user necessary permissions on the table.

   ```
   GRANT SELECT, LOCK TABLES, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER ON `gitlabhq_production`.* TO 'git'@'localhost';
   ```

5. Quit the database session

   ```
   quit
   ```

- Try connecting to the new database with the new user

```
sudo -u git -H mysql -u git -p -D gitlabhq_production
```

  - Type the MySQL git password set earlier

    ```
    SuperSecretPassword
    ```

- You should now see a mysql> prompt, quit the database session:

```
quit
```

WARNING: GitLab suggests using PostgreSQL for the database used; this is due to the inability to support case-sensitive table names. Since my use case is small, I will not need case-sensitivity, and will use a MySQL derivative instead.

Ruby

- Switch to the GitLab user and install RVM:

```
su - git
curl -sSL https://get.rvm.io | bash -s stable
source /home/git/.rvm/scripts/rvm
```

- Install Ruby 2.1.2 using RVM:

```
rvm install 2.1.2
```

- Check the Ruby install path:

```
which ruby
```

  - Example output:

```
/home/git/.rvm/rubies/ruby-2.1.2/bin/ruby
```

- Check the installed Ruby version:

```
ruby -v
```

  - Example output:

```
ruby 2.1.2p95 (2014-05-08 revision 45877) [x86_64-freebsd10.0]
```

Install GitLab 7.2

- Change to the git user home directory and download GitLab 7.2

```
cd
git clone https://gitlab.com/gitlab-org/gitlab-ce.git -b 7-2-stable gitlab
```

- Create a config file

```
cd /home/git/gitlab
cp config/gitlab.yml.example config/gitlab.yml
```

- Edit the gitlat config file:

```
vi config/gitlab.yml
```

  - Change the gitlab: host: to git.example.com
  - If using https change the gitlab: port: from 80 to **443**
  - Change the git: bin_path: to **/usr/local/bin/git**
  - Change the satellites: path: to **/usr/home/git/gitlab-satellites/**
  - Change the gitlab_shell: path: to **/usr/home/git/gitlab-shell/**
  - Change the gitlab_shell: repos_path: to **/usr/home/git/repositories/**
  - Change the gitlab_shell: hooks_path: to **/usr/home/git/gitlab-shell/hooks/**

- Create a unicorn.rb file:

```
cp config/unicorn.rb.example config/unicorn.rb
```

- Create a rack_attack.rb file:

```
cp config/initializers/rack_attack.rb.example config/initializers/rack_attack.rb
```

- Create a database configuration file:

```
cp config/database.yml.mysql config/database.yml
```

- Configure the GitLab system user:

```
git config --global user.name "GitLab"
git config --global user.email "example@example.com"
```

- Setup the directories and permissions:

```
chown -R git log/
chown -R git tmp/
chmod -R u+rwX log/
chmod -R u+rwX tmp/
chmod -R u+rwX tmp/pids/
chmod -R u+rwX tmp/sockets/
chmod -R u+rwX  public/uploads
```

- Create the gitlab-satellites:

```
mkdir /home/git/gitlab-satellites
chmod u+rwx,g=rx,o-rwx /home/git/gitlab-satellites
```

- Install bundler:

```
gem install bundler
export CC=gcc47
export CXX=c++47
export CPP=cpp47
```

- Install GitLab gems

```
bundle install --deployment --without development test postgresql aws
```

Install GitLab Shell

- Run the installation task for gitlab-shell (replace `REDIS_URL` if needed):

```
bundle exec rake gitlab:shell:install[v1.9.7] REDIS_URL=redis://localhost:6379 RAILS_ENV=produ
ction
```

By default, the gitlab-shell config is generated from your main gitlab config.

- **NOTE**: When using GitLab with HTTPS please change the following:

1. Provide paths to the certificates under `ca_file` and `ca_path` options.
2. The `gitlab_url` option must point to the https endpoint of GitLab.
3. In case you are using self signed certificate set `self_signed_cert` to `true`.

*You can review (and modify) the gitlab-shell config as follows:

```
vi /home/git/gitlab-shell/config.yml
```

- Initialize Database and Activate Advanced Features

  ```
  bundle exec rake gitlab:setup RAILS_ENV=production
  ```

- Verify that the install worked:

  ```
  bundle exec rake gitlab:env:info RAILS_ENV=production
  ```

- Precompile the assets used by GitLab:

  ```
  bundle exec rake assets:precompile RAILS_ENV=production
  ```

- Exit out of the GitLab user, back into the root account:

  ```
  exit
  ```

- Create an init script

  ```
  cp /usr/home/git/gitlab/lib/support/init.d/gitlab /usr/local/etc/rc.d/gitlab
  ```

- Start GitLab

  ```
  service gitlab start
  ```

- Enable GitLab to start at boot

  ```
  echo 'gitlab_enable="YES"' >> /etc/rc.conf
  ```

Using Nginx to Host GitLab

- Install nginx:

  ```
  pkg install -y nginx
  ```

- Copy the GitLab nginx config into the nginx directory

  ```
  cp /usr/home/git/gitlab/lib/support/nginx/gitlab /usr/local/etc/nginx/gitlab.conf
  ```

  - Change the hostname: **git.example.com**
  - Change the proxy_pass to the correct address and port: http://127.0.0.1:8080

- Add the following piece of code in /usr/local/etc/nginx/nginx.conf, before the last "}":

```
include /usr/local/etc/nginx/gitlab.conf;
```

- Prepare the directories needed by nginx:

```
mkdir -p /var/tmp/nginx /var/log/nginx
chown -R www: /var/log/nginx /var/tmp/nginx
```

- Start and enable nginx at boot:

```
echo 'nginx_enable="YES"' >> /etc/rc.conf
service nginx start
```

- Default username and password:
  - user: **root**
  - pass: **5iveL!fe**

(Optional) Using Apache with Passenger to Host GitLab

Running GitLab from an apache server with passenger installed is a simple.

- Edit the virtual host definition and add/modify it according to your needs:

```
vi /usr/local/etc/apache24/Vhosts/git.example.com.conf
```

  - And add/edit the following virtual host definition:

```
<VirtualHost *:80>
  DocumentRoot "/home/git/gitlab/public"
  ServerName git.example.com
  ErrorLog "/var/log/git.example.com-error_log"
  CustomLog "/var/log/git.example.com-access_log" common
  PassengerRuby /home/git/.rvm/wrappers/ruby-2.1.2/ruby

  <Directory /home/git/gitlab/public>
      Options -MultiViews
      AllowOverride All
      Order allow,deny
      Allow from all
      Require all granted
  </Directory>
</VirtualHost>
```

- Restart apache:

```
service apache24 restart
```

- Start and enable apache24 at boot:

```
echo 'apache24_enable="YES"' >> /etc/rc.conf
service apache24 start
```

Resources

http://luizgustavo.pro.br/blog/2014/08/21/instalacao-gitlab-no-freebsd/
https://gitlab.com/gitlab-org/gitlab-ce/blob/7-2-stable/doc/install/installation.md
https://gitlab.com/gitlab-org/gitlab-ce/blob/7-2-stable/doc/install/database_mysql.md

Related issues:

Copied to FreeBSD Administration - Support #532: Installing GitLab 7.6 on Fre...	**Closed**	**09/21/2014**

History

#1 - 09/21/2014 07:55 PM - Daniel Curtis

- Description updated

- % Done changed from 70 to 80

#2 - 09/22/2014 03:06 PM - Daniel Curtis

- Description updated

- Status changed from New to Resolved

- % Done changed from 80 to 90

#3 - 09/22/2014 06:42 PM - Daniel Curtis

- Project changed from 98 to FreeBSD Administration

- Description updated

- % Done changed from 90 to 100

#4 - 09/28/2014 02:29 PM - Daniel Curtis

- Description updated

- Status changed from Resolved to Closed

#5 - 01/10/2015 05:44 PM - Daniel Curtis

- Copied to Support #532: Installing GitLab 7.6 on FreeBSD added

#6 - 02/14/2015 10:37 AM - Daniel Curtis

- Subject changed from Installing GitLab 7.2 on FreeBSD 9.2-RELEASE to Installing GitLab 7.2 on FreeBSD

- Target version set to FreeBSD 9

#7 - 02/14/2015 11:49 AM - Daniel Curtis

- Category set to Source Code Management