# Buglliza-5.x

# 安装部署文档

|  |  |
| --- | --- |
| 作者: | HHL |
| 日期: | 2018-03-02 |

# **How to install Bugzilla 5.0 on CentOS 7**

来源：<https://www.howtoforge.com/tutorial/install-bugzilla-50-on-centos-7/>

This HowTo will walk you through the installation of Bugzilla 5.0 on CentOS 7. Bugzilla is an advanced bug tracking system, developed by the Mozilla Foundation (the organization that develops the famous Firefox browser). Bugzilla allows you to track defects and code changes in your applications, allows you to communicate in your dev team easily and submit and review patches.

## **Prerequisites for CentOS7.2**

### **Internet Connection**

You should have a minimal CentOS 7 installation with the latest updates and IP address and hostname set. Your machine should be able to reach the internet. If you can not reach the internet directly and you have to use a proxy server, you should configure yum to use a proxy by editing*/etc/yum.conf*. Add the following lines (change them to your needs):

*# The proxy server - proxy server:port  
proxy=http://192.168.178.1:8080  
# The account details for yum connections  
# proxy\_username=yum-user  
# proxy\_password=qwerty*

When we use command line tools, that require internet access and you are behind a proxy server, make sure you always execute the following two commands before using commands that need an internet connection:

*export http\_proxy=http://192.168.178.1:8080/  
export https\_proxy=http://192.168.178.1:8080/*

With this method, you should be able to use this How-to even if you are behind a proxy.

### **SELinux**

Security-Enhanced Linux (SELinux) is a mandatory access control (MAC) security mechanism implemented in the kernel. It is an excellent security mechanism but it will break Bugzilla until an official Bugzilla 5.0 rpm-package is released that will also configure SELinux. So at this point we will set SELinux to permissive mode. In this mode, SELinux is enabled but it will not enforce the security policy.

#### **Disable SELinux**

Execute the following command to change SELinux mode from enforcing to permissive.

*sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/g' /etc/selinux/config*

The file should now look like this:

*# This file controls the state of SELinux on the system.  
# SELINUX= can take one of these three values:  
#     enforcing - SELinux security policy is enforced.  
#     permissive - SELinux prints warnings instead of enforcing.  
#     disabled - No SELinux policy is loaded.  
SELINUX=permissive  
# SELINUXTYPE= can take one of three two values:  
#     targeted - Targeted processes are protected,  
#     minimum - Modification of targeted policy. Only selected processes are protected.  
#     mls - Multi Level Security protection.  
SELINUXTYPE=targeted*

After that change, reboot your machine.

### **Account used for installation**

If not explicitly told otherwise, all commands in this How-to are executed as the user '**installer**' that is an administrative user on the CentOS 7 machine and is allowed to execute commands with '**sudo**'.

### **Additional Repositories**

We will need a lot of packages that are not in the official CentOS repositories, therefor we enable an additional repository on our CentOS box.

#### **Enable Epel repository**

Execute the following commands to enable the epel repository.

*sudo yum install deltarpm epel-release  
sudo yum update*

Info: You have to accept the epel GPG key when installing the first package from the epel repository.

## **Install Apache with mod\_ssl and mod\_perl**

*sudo yum install httpd httpd-devel mod\_ssl mod\_ssl mod\_perl mod\_perl-devel*

This will install Apache, mod\_ssl, mod\_perl and wget with all of the needed dependencies.

### **Enable and start httpd.service**

CentOS 7 uses systemd, so we will enable and start Apache the 'systemd way'.

#### **Start httpd.service**

*sudo systemctl start httpd.service*

#### **Check status of httpd.service**

*sudo systemctl status httpd.service*

This command should tell you, that httpd.service is running (some lines stripped out)

*httpd.service - The Apache HTTP Server  
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)  
   Active: active (running) since Tue 2015-08-04 11:18:03 CEST; 6s ago  
 Main PID: 11930 (/usr/sbin/httpd)  
...*

#### **Permanently enable httpd.service**

After verifying the above output we enable httpd.service for autostarting with:

*sudo systemctl enable httpd.service*

The system will automatically create a symbolic link to enable http.service for automatic starting.

#### **Open port 80 in the local firewall to allow HTTP requests**

CentOS 7 is using FirewallD, so we have to use firewall-cmd to change the firewall setting to allow incoming connections on port 80 (HTTP).

*sudo firewall-cmd --zone=public --add-port=80/tcp --permanent  
# success  
sudo firewall-cmd --reload  
# success*

#### **Test the Apache web server**

Use a browser and open http://ip-of-your-server/ (replace ip-of-your-server with the IP address of your server). You should see the Apache Test Page. - Apache is working!

## **Install MariaDB**

CentOS 7 comes with MariaDB instead of MySQL. MariaDB is an open source equivalent to MySQL. To install MariaDB execute the following commands:

*sudo yum install mariadb-server mariadb mariadb-devel php-mysql*

#### **Make MariaDB autostarting at boot**

To autostart MariaDB at boot, we set it up as a service the same way we did before with httpd.service. Execute the following three commands to start, check the status and enable MariaDB as a permanent service:

*sudo systemctl start mariadb.service  
sudo systemctl status mariadb.service  
sudo systemctl enable mariadb.service*

#### **Set MariaDB root password**

Open an SQL prompt into your MariaDB server by executing the following command:

*mysql -u root*

You should land on the MariaDB prompt. In the following commands, you have to replace myrootpassword with the password you want to use for your MariaDB root user!

*MariaDB [ (none) ]> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('myrootpassword');*

*Query OK, 0 rows affected (0.00 sec)*

*MariaDB [ (none) ]> \q*

*Bye*

Now try if you can login with the root user

*mysql -u root -p  
Enter password: myrootpassword*

*Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MariaDB connection id is 5  
Server version: 5.5.41-MariaDB MariaDB Server  
Copyright (c) 2000, 2014, Oracle, MariaDB Corporation Ab and others.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.*

*MariaDB [ (none) ]> \q*

*Bye*

#### **Set max\_allowed\_packets for MariaDB**

Bugzilla needs a minimum size of 'max\_allowed\_packet' configured within MariaDB. So let us change the generic MariaDB configuration to set the 'max\_allowed\_packet' size to 4 MBytes.  
Open '/etc/my.cnf' with your editor and add the following lines below the '[mysqld]' section:

*# Bugzilla  
# maximum allowed size of an attachment upload  
#change this if you need more!  
max\_allowed\_packet=4M*

After that change the entire 'my.cnf' looks like this:

*[mysqld]  
datadir=/var/lib/mysql  
socket=/var/lib/mysql/mysql.sock  
# Disabling symbolic-links is recommended to prevent assorted security risks  
symbolic-links=0  
# Settings user and group are ignored when systemd is used.  
# If you need to run mysqld under a different user or group,  
# customize your systemd unit file for mariadb according to the  
# instructions in http://fedoraproject.org/wiki/Systemd  
  
# Bugzilla  
# maximum allowed size of an attachment upload  
#change this if you need more!  
max\_allowed\_packet=4M  
  
[mysqld\_safe]  
log-error=/var/log/mariadb/mariadb.log  
pid-file=/var/run/mariadb/mariadb.pid  
  
#  
# include all files from the config directory  
#  
!includedir /etc/my.cnf.d*

The changes will take affect after restaring MariaDB.

*sudo systemctl restart mariadb.service*

### **Additional Packages**

Bugzilla needs a lot of additional packages, mainly perl-related, some preriquisites to make the perl modules work and a setup to be able to download and install perl modules. Therefor we install all necessary dependencies. We also make sure, that we

**DO NOT INSTALL the package perl-homedir, because this would break the Bugzilla installation.**

('perl-homedir' would install perl modules in user folders which won't be accessible by Bugzilla so absolutely do not install it!)  
  
The following command will make sure perl-homedir is not installed and will install the other required packages:

*sudo yum install gcc gcc-c++ graphviz graphviz-devel patchutils gd gd-devel wget perl\* -x perl-homedir*

About 1300 packages will be installed!

## **Install Bugzilla**

Now that we have all the preparing done we are ready to download the latest Bugzilla, create the database for it and do some final setup.

### **Create a database for Bugzilla**

To create a database for Bugzilla on our MariaDB server we have to open the MariaDB root prompt again:

*mysql -u root -p  
Enter password: myrootpassword*

*Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MariaDB connection id is 5  
Server version: 5.5.41-MariaDB MariaDB Server  
Copyright (c) 2000, 2014, Oracle, MariaDB Corporation Ab and others.  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.*

At the MariaDB root prompt enter the following commands to create a the database 'bugs' and allow the user 'bugs' to fully access it. (Please change 'bugsuserpassword' to your password!)

*MariaDB [ (none) ]> create database bugs;*

*Query OK, 1 row affected (0.00 sec)*

*MariaDB [ (none) ]> grant all on bugs.\* to bugs@localhost identified by 'bugsuserpassword';*

*Query OK, 0 row affected (0.00 sec)*

*MariaDB [ (none) ]> \q*

*Bye*

### **Download and extract Bugzilla**

While there are different options to get BugZilla, we use wget to download the tarball We want to get the Stable Release (5.0).

*cd  
wget 'https://ftp.mozilla.org/pub/mozilla.org/webtools/bugzilla-5.0.tar.gz'*

After the download is complete, we extract the archive:

*tar -xzvf bugzilla-5.0.tar.gz*

### **Copy Bugzilla folder to webserver folder**

Now let's copy the extracted folder (bugzilla-5.0) to it's target loation (/var/www/html/bugzilla) with the following command

*cd  
sudo cp -R ./bugzilla-5.0/ /var/www/html/bugzilla/*

### **Final setup using Bugzilla's installation script**

The following commands must be executed as the root user, therefor we open a root shell with the following command:

*sudo su  
[root@localhost installer]#*

Our prompt has now a '#' at the end, meaning we are root. Please be careful with all the commands now! The howto will now always show the prompt to tell you that you should be root to execute the commands!  
  
Now (as root) we we change to the webserver folder where bugzilla resides

*[root@localhost installer]# cd /var/www/html/bugzilla  
[root@localhost bugzilla]#*

Now execute the following command to let the Bugzilla installation script check the status of our setup

*[root@localhost bugzilla]# ./checksetup.pl*

Usually the output of the above command will tell us, that some perl modules are missing from our installation (optional and required ones) and that we have to install at least the required ones to be able to continue. You could of course install all the required (and if you like) the optional modules by hand executing the commands the installation script will show you, but with this single command you will install all necessary perl modules at once (required and optional). So let's execute it:

*[root@localhost bugzilla]# /usr/bin/perl install-module.pl --all*

The setup script will now setup a temporary CPAN configuration and tries to install all of the perl modules Bugzilla needs (required and optional). Only some minor warnings should appear during the process.  
  
At the end, let's again verify the status of the installation by executing checksetup.pl

*[root@localhost bugzilla]# ./checksetup.pl*

All dependencies should be okay now, except one module ( DBD-Oracle (v1.19)  which is not found. This is okay because this module will only install if it finds a working Oracle installation which we do not have!  
  
The setup script will now show the following text:

*Reading ./localconfig...  
  
This version of Bugzilla contains some variables that you may want to  
change and adapt to your local settings. The following variables are  
new to ./localconfig since you last ran checksetup.pl:  
  
create\_htaccess, webservergroup, use\_suexec, db\_driver, db\_host,  
db\_name, db\_user, db\_pass, db\_port, db\_sock, db\_check,  
db\_mysql\_ssl\_ca\_file, db\_mysql\_ssl\_ca\_path, db\_mysql\_ssl\_client\_cert,  
db\_mysql\_ssl\_client\_key, index\_html, interdiffbin, diffpath,  
site\_wide\_secret  
  
Please edit the file ./localconfig and then re-run checksetup.pl  
to complete your installation.*

So as stated above we open ./localconfig with our editor and change it to our needs. Here is the one I used (comments removed), make sure you change the database names and passwords to your needs!

*$create\_htaccess = 1;  
$webservergroup = 'apache';  
$use\_suexec = 0;  
$db\_driver = 'mysql';  
$db\_host = 'localhost';  
$db\_name = 'bugs';  
$db\_user = 'bugs';  
$db\_pass = 'bugsuserpassword';  
$db\_port = 0;  
$db\_sock = '';  
$db\_check = 1;  
$db\_mysql\_ssl\_ca\_file = '';  
$db\_mysql\_ssl\_ca\_path = '';  
$db\_mysql\_ssl\_client\_cert = '';  
$db\_mysql\_ssl\_client\_key = '';  
$index\_html = 0;  
$interdiffbin = '/bin/interdiff';  
$diffpath = '/bin';  
$site\_wide\_secret = 'ifKuihguW8nlxLcxeNU4whHzFbxDIGWSvtR6S7Ul38cFQn004YDcVzuBJfnF8M9X';*

Now let's run the setup script again. It should now detect the correct database configuration and starting to access the MariaDB server for a final setup.

*[root@localhost bugzilla]# ./checksetup.pl*

Output (some lines stripped)

*...  
...  
Adding new table bz\_schema...  
Initializing bz\_schema...  
Creating tables...  
Converting attach\_data maximum size to 100G...  
Setting up choices for standard drop-down fields:  
   priority bug\_status rep\_platform resolution bug\_severity op\_sys  
Creating ./data directory...  
...  
...  
Precompiling templates...done.  
Fixing file permissions...  
Initializing "Dependency Tree Changes" email\_setting ...  
Initializing "Product/Component Changes" email\_setting ...  
Marking closed bug statuses as such...  
Creating default classification 'Unclassified'...  
Setting up foreign keys...  
Setting up the default status workflow...  
Creating default groups...  
Setting up user preferences...  
  
Looks like we don't have an administrator set up yet. Either this is  
your first time using Bugzilla, or your administrator's privileges  
might have accidentally been deleted.  
  
Enter the e-mail address of the administrator:*

Now follow the prompts and finalize your setup (use your values for the answers here!):

*Enter the e-mail address of the administrator: youremail@yourdomain.tld  
Enter the real name of the administrator: Der PCFreak  
Enter a password for the administrator account: adminpassword  
Please retype the password to verify: adminpassword  
youremail@yourdomain.tld is now set up as an administrator.*

When everything went fine you should see:

*Creating initial dummy product 'TestProduct'...  
  
Now that you have installed Bugzilla, you should visit the 'Parameters'  
page (linked in the footer of the Administrator account) to ensure it  
is set up as you wish - this includes setting the 'urlbase' option to  
the correct URL.  
checksetup.pl complete.*

At this point we are nearly done. Execute the following line to comment out a line in the .htaccess file that the Bugzilla installation script created:

*[root@localhost bugzilla]# sed -i 's/^Options -Indexes$/#Options -Indexes/g' ./.htaccess*

### **Configure Apache to host our Bugzilla installation**

Apache still does not know anything about Bugzilla. So let's create the file /etc/httpd/conf.d/bugzilla.conf with the following content to introduce our Bugzilla website to the httpd.service.

*#/etc/httpd/conf.d/bugzilla.conf  
<VirtualHost \*:80>  
DocumentRoot /var/www/html/bugzilla/  
</VirtualHost>  
<Directory /var/www/html/bugzilla>  
AddHandler cgi-script .cgi  
Options +Indexes +ExecCGI  
DirectoryIndex index.cgi  
AllowOverride Limit FileInfo Indexes  
</Directory>*

After the file has been created we restart Apache for the changes to take effect:

*[root@localhost bugzilla]# systemctl restart httpd.service*

#### **Switch back to normal user**

Working as root is no longer required, you can use sudo now if you need root privileges. So exeute the following command to exit the root shell:

*[root@localhost bugzilla]# exit  
[installer@localhost ~]$*

#### **Test the Bugzilla installation with your browser**

Use a browser and open http://ip-of-your-server/ (replace ip-of-your-server with the ip address of your server). You should now see the Bugzilla page instead of the default Apache Test Page.  
You can now login with the credentials you provided to the Bugzilla installation script. In this How-To this was:

*Email Address: youremail@yourdomain.tld  
Password     : adminpassword*

Voila! You have now a working Bugzilla 5.0 installation on CentOS 7. You can now continue to setup the details of Bugzilla within the Bugzilla web interface.

**附: 汉化方法：**

将汉化包里的zh-CN拷贝到bugzilla下的template文件夹下，刷新网页后页面右上角有切换语言的链接！