How to install Amusewiki

Marco Pessotto, Alexander Krotov
Contents

General installation ................................................. 3
Install prerequisites .................................................. 3
Install Amusewiki application ...................................... 3
Configure database ..................................................... 4
Configure initial site .................................................. 4
Start the application .................................................. 4
Starting the application manually ................................... 4
Starting the application with systemd ............................. 5
Web server configuration .............................................. 5
Additional configuration .............................................. 6
Configuration file ....................................................... 6
Mail with SMTP ........................................................ 6
Multiple installations .................................................. 7
OS-specific instructions ............................................... 7
FreeBSD 12.0 ............................................................ 7
CentOS 7 ................................................................. 9
The recommended way to install Amusewiki is via Debian repository. If you want to install manually, read on. If you need to do some post-installation tweaking for ports and webserver logging, jump to the configuration section. Otherwise feel free to skip this document entirely.

**General installation**

The installation of Amusewiki takes time (depending on speed of the machine and of the network) and requires about 5 Gb of disk space because of the full installation of TeX Live. If you’re short on disk space, don’t even start to install. The app will create files at full speed anyway, so consider 10 Gb for a reasonable start.

The procedure is fully automated, so start it, check if it bails out at the beginning, forget about it for some time (run it under screen), then come back later and finish it up to complete it for the operations which require root privileges (notably the webserver configuration).

You can speed up the process by installing TeX Live from the OS repositories, which is the suggested approach because this way the security fixes are delegated to the standard tools.

Amusewiki installation is similar for all *nix operating systems, but you may want to additionally consult OS-specific instructions section which contain instructions for less common operating systems. Windows is not supported in any way.

**Install prerequisites**

In addition, the following software should be installed:

- a database (MySQL, PostgreSQL, SQLite are supported)
- Perl and Carton: carton package on Debian
- fontconfig (install it before installing TeX Live)
- GraphicsMagick (for thumbnails) and ImageMagick (for preview generation)
- a mime-info database: shared-mime-info on Debian
- a dedicated system user (with a clean home) which is going to run the site
- SSL binaries and development libraries (openssl and libssl-dev)
- Xapian libraries and development files (xapian-tools libxapian-dev)
- commonly used utilities: unzip, wget, git, rsync
- TeX Live full > 2012, either from system repo (recommended) or from https://www.tug.org/texlive/

Log in as the user you want to run the site.

If you installed TeX Live from the installer, tweak the shell rc file to include the binaries in the PATH.

Logout and login again.

**Install Amusewiki application**

Unpack the sources (or clone the repo) and change directory into them.

Run installation script to check that you have installed prerequisites and complete the installation:

```
./script/install.sh
```
Configure database

Create a database for the application. E.g., for MySQL:

```sql
mysql> create database amusewiki DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_general_ci;
mysql> grant all privileges on amusewiki.* to amusewiki@localhost identified by 'Pa55Word';
```

Or, for PostgreSQL:

```
su - postgres
psql
create user amusewiki with password 'XXXX';
create database amusewiki owner amusewiki;
```

For SQLite no setup is required.

Copy `dbic.yaml.<dbtype>.example` to `dbic.yaml` and adjust the credentials, and `chmod` it to `0600`. (For SQLite is good as it is).

Configure initial site

If you have multiple Amusewiki instances (please note, multiple sites are just fine on a single instance) on the same machine, see below before proceeding (you probably want to tweak the configuration).

Configure initial site with:

```
./script/configure.sh [hostname]
```

Please note that the installation procedure will create a mirror of amusewiki.org under the subdomain `amusewiki.<your domain>`, where `<your domain>` is the output of `hostname -d`. Nothing you can't change later from the admin console, but you need to access it. You can pass the desired hostname as the first argument to the configure script.

Start the application

Starting the application manually

To set the number of FCGI workers, set the environment variable `AMW_WORKERS` (defaults to 3).

```
export AMW_WORKERS=5
```

To start/stop/restart the application:

```
./init-all.sh start
./init-all.sh stop
./init-all.sh restart
```

This is not needed if you use systemd.
Starting the application with systemd

If your OS uses systemd for initialization, you can generate systemd units with

```
script/generate-systemd-unit-files
```

It will generate unit files for three services:

- amusewiki-web
- amusewiki-jobber
- amusewiki-cgit

Follow the instructions printed on the screen to enable amusewiki-web and amusewiki-jobber. Do **not** enable amusewiki-cgit. cgit is managed by the web server, which we'll configure in the next section.

Web server configuration

The supported and recommended setup is nginx + FCGI. The FCGI setup should work with Apache HTTP Server as well, but it's not actively supported, even if perldoc Catalyst::Manual::Deployment::Apache::FastCGI should help.

Nginx configuration

To regenerate the nginx configuration after adding a site:

```
carton exec script/amusewiki-generate-nginx-conf
```

Follow the printed instructions to install generated configuration.

Caddy configuration

Another easy to setup webserver is Caddy, but you will have to configure it manually and your configuration may become incomplete in the future.

Here, Amusewiki is assumed to be installed in `/home/amusewiki/amusewiki/` and hostname is amusewiki.local.

When downloading Caddy, enable `http.cgi` plugin. To run Amusewiki, create a Caddyfile in the Amusewiki installation directory with the following contents:

```
amusewiki.local:8080
  tls off
  log ./access.log
  errors ./error.log
  root /usr/home/amusewiki/amusewiki/root

  cgi {
    except /git/cgit.css
    except /git/cgit.png
    match /git
    exec /home/amusewiki/amusewiki/root/git/cgit.cgi
  }

  fastcgi / unix:/usr/home/amusewiki/amusewiki/var/amw.sock {
```
except /static/
except /git/
}

Start caddy.

### Additional configuration

#### Configuration file

Normally, you don't need to change anything. However, you may need to do some tweaking to the webserver configuration. This is done via the configuration file.

If amusewiki was installed with a debian package, the location is /etc/amusewiki.conf otherwise you should create a file called amusewikifarm_local.conf in the application directory, which will override the existing settings in amusewikifarm.conf.

Example with the defaults:

```xml
<Model::Webserver>
  ## cgit port
cgit_port 9015
  ## nginx log format
log_format combined
  ## nginx root
nginx_root /etc/nginx
  ## string to identify this installation
instance_name amusewikidebian
  webserver_root /usr/share/perl5/AmuseWikiFarm/root
  fcgi_socket /var/lib/amusewiki/amusewiki.socket
</Model::Webserver>
```

#### Mail with SMTP

You need to set the desired parameter as environment variable (in the systemd unit file or in the user starting the application). See [https://metacpan.org/pod/Email::Sender::Manual::QuickStart](https://metacpan.org/pod/Email::Sender::Manual::QuickStart) and [https://metacpan.org/pod/Email::Sender::Transport::SMTP](https://metacpan.org/pod/Email::Sender::Transport::SMTP) for details.

Previously we used the application config file, but that's sloppy because it prevents the jobber to send mails properly.

Example:

```bash
$ export EMAIL_SENDER_TRANSPORT=SMTP
$ export EMAIL_SENDER_TRANSPORT_host=smtp.example.com
$ export EMAIL_SENDER_TRANSPORT_port=2525
./init-all.sh restart
```

If you use systemd unit files to start/stop/restart the application, you need to override them and set the environment variables instead.

```bash
cp /lib/systemd/system/amusewiki-web.service \\   /lib/systemd/system/amusewiki-jobber.service \\   /etc/systemd/system
```

Add in the [Service] stanza the needed variables, like this
Multiple installations

If you run a Debian machine and you have only one instance running and if you have the port 9015 free, you don’t need any of this.

Please note: “multiple instances” doesn’t mean “multiple sites”. On a single instance you can have as many sites as you want.

The interaction between nginx and the application, including cgit, is controlled by the Webserver model. You can configure it creating a file in the application root named amusewikifarm_local.conf with this content (here listed with the defaults).

```html
<Model::Webserver>
  # cgit port
  cgit_port 9015
  log_format combined
  nginx_root /etc/nginx
  instance_name amusewiki
  fcgiwrap_socket /var/run/fcgiwrap.socket
</Model::Webserver>
```

The instance_name is just a string used to create the nginx configuration files to avoid conflicts with other installations. So you may have one instance named “testing” and the other “live”.

OS-specific instructions

The following sections contain installation instructions for operating systems other than Debian. Note that these sections may be outdated, so read general installation instructions first.

FreeBSD 12.0

Install prerequisites:

```
pkg install perl5 p5-App-cpanminus \p5-carton git texlive-full \cgit GraphicsMagick shared-mime-info \xapian-core xapian-bindings \nginx fcgiwrap unzip rsync wget bash
```

Create amusewiki user:

```
[root@freebsd ~]# adduser
Username: amusewiki
Full name: Amusewiki
Uid (Leave empty for default):
Login group [amusewiki]:
Login group is amusewiki. Invite amusewiki into other groups? []: Login class [default]:
Shell (sh csh tcsh zsh rzsh bash rbash git-shell nologin) [sh]: bash
Home directory [/home/amusewiki]:
```
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]: no
Lock out the account after creation? [no]:
Username : amusewiki
Password : <disabled>
Full Name : Amusewiki
Uid : 1001
Class :
Groups : amusewiki
Home : /home/amusewiki
Home Mode :
Shell : /usr/local/bin/bash
Locked : no
OK? (yes/no): yes
adduser: INFO: Successfully added (amusewiki) to the user database.
Add another user? (yes/no): no
Goodbye!

Add to /etc/rc.conf

nginx_enable=YES
fcgiwrap_enable=YES
fcgiwrap_user=www
fcgiwrap_socket="unix:/var/run/fcgiwrap/socket"

# create an include sites-enabled directory for nginx and log dir
mkdir -p /usr/local/etc/nginx/sites-enabled
mkdir -p /var/log/nginx/
# and add an include directive in nginx.conf inside the http { } stanza, at the end.
# include /usr/local/etc/nginx/sites-enabled/*;
vi /usr/local/etc/nginx/nginx.conf

And start the services

service nginx start
service fcgiwrap start

su - amusewiki
git clone https://github.com/melmothx/amusewiki.git
cd amusewiki
CXX=c++ ./script/install.sh
# create the config
cat << EOF > amusewikifarm_local.conf
<Model::Webserver>
  nginx_root /usr/local/etc/nginx
  fcgiwrap_socket /var/run/fcgiwrap/socket
</Model::Webserver>
EOF

Then decide to which hostname you want to serve this and run

./script/configure.sh amw.localdomain
Read the output (you’ll need to configure a database), do what it says and run it again, taking note of the credentials, and follow the instructions.

If you want to use MySQL, run as root:

```bash
pkg install mysql57-server
echo 'mysql_enable=YES' >> /etc/rc.conf
service mysql-server start
cat ~/.mysql_secret # see password
mysql -h localhost -p
```

Create the database as described above in the generic section and then finalize the configuration with the amusewiki user:

```bash
# su - amusewiki
$ cd amusewiki
$ cpanm -L local DBD::mysql
$ cp dbic.yaml.mysql.example dbic.yaml
$ vi dbic.yaml # edit db credentials
$ ./script/configure.sh amw.localdomain # reconfigure
$ ./init-all start
```

For PostgreSQL, run as root:

```bash
pkg install postgresql96-server
echo 'postgresql_enable=YES' >> /etc/rc.conf
service postgresql initdb
service postgresql start
su - postgres
```

Create the database as described above in the generic section and then finalize the configuration with the amusewiki user:

```bash
# su - amusewiki
$ cd amusewiki
$ cpanm -L local DBD::Pg
$ cp dbic.yaml.pg.example dbic.yaml
$ vi dbic.yaml # edit db credentials
$ ./script/configure.sh amw.localdomain # reconfigure
$ ./init-all start
```

**CentOS 7**

Most of these instructions will apply to other GNU/Linux systems with systemd and SELinux. TeX Live is obsolete, so we will install it from CTAN. The same goes with Perl.

From a fresh install:

```bash
# yum install epel-release
# yum install git nginx perl-local-lib sqlite cgit perl-App-cpanminus fontconfig GraphicsMagick ImageMagick shared-mime-info openssl openssl-devel xapian-core xapian-core-devel unzip wget libxml2 libxml2-devel expat-devel policycoreutils setroubleshoot
# yum groupinstall 'Development Tools'
```

Tweak nginx configuration to conform to Debian standard
# mkdir /etc/nginx/sites-enabled

Modify /etc/nginx/nginx.conf adding this line:

```
include /etc/nginx/sites-enabled/*;
```

Right after include /etc/nginx/conf.d/*.conf;
(Probably you may also want to add client_max_body_size 8m; as the default is way too low).
Start nginx:

```
# systemctl enable nginx
# systemctl start nginx
```

Open the firewall

```
# firewall-cmd --get-active-zones
# firewall-cmd --zone=public --add-service=http --permanent
# firewall-cmd --zone=public --add-service=https --permanent
# firewall-cmd --reload
```

Prepare the installation directory for your user (say amusewiki, but any other will do).
Please note that we install it under /var/www/ to avoid problems with SELinux.

```
# mkdir /var/www/amusewiki
# chown amusewiki:amusewiki /var/www/amusewiki
```

As the user which is going to run Amusewiki, install a fresh Perl. This way we simplify and make our install independent from the base system, which is lacking way too many modules.

```
$ eval `perl -Mlocal::lib`
$ cpanm Perl::Build
$ perl-build -j 3 --test 5.24.1 $HOME/amw-perl
$ cd /vaw/www/amusewiki
$ git clone https://github.com/melmothx/amusewiki.git
$ cd amusewiki # you're now in /vaw/www/amusewiki/amusewiki, our app home
$ ./script/install-texlive.sh # install texlive
$ echo 'export PATH=$HOME/amw-perl/bin:$HOME/texlive/2018/bin/x86_64-linux:$PATH' >> $HOME/.bashrc
$ chmod 755 $HOME # open the home so plackup is accessible to nginx

Logout and login again and check the program paths to point to the newly installed ones

```
$ which perl # should be: ~/amw-perl/bin/perl
$ which xelatex # should be ~/texlive/2018/bin/x86_64-linux/xelatex or equivalent
```

Install cpanm and the dependencies

```
$ perl -MCOPAN -e 'install Carton'
$ which carton # should be ~/amw-perl/bin/carton
$ cd /var/www/amusewiki/amusewiki/
$ ./script/install.sh
```

Decide the initial server name to serve Amusewiki (to get access to the admin), e.g. amw.localdomain.
See above if you want a MySQL or PostgreSQL database. The following command will create a sample site with the current Amusewiki documentation.


$ ./script/configure.sh amw.localdomain

Remove the egit wrapper, we're going to install systemd unit files.

$ rm root/git/cgit.cgi
$ ./script/generate-systemd-unit-files
$ carton exec ./script/amusewiki-generate-nginx-conf

Read the output and install the fresh nginx configuration.

Finally, open the permission for SELinux. As root:

```
cd /var/www/amusewiki/amusewiki/doc/centos/
checkmodule -M -m -o amusewiki.mod amusewiki.te
semodule_package -o amusewiki.pp -m amusewiki.mod
semodule -i amusewiki.pp
```

Reboot to be sure everything is ok, open with a browser the location you configured (say, amw.localdomain, you may want to add the entry /etc/hosts to access it) and login.

You probably want to head to the admin panel under /admin/sites to create a new site.