AMuseWiki

How to install AmuseWiki

Marco Pessotto

Marco Pessotto
How to install AmuseWiki

amusewiki.org

Contents

Caddy webserver

Although recommended and supported setup in nginx, Caddy server may be easier to setup, at least for development purposes.

Here, Amusewiki is assumed to be installed in `/home/amusewiki/amusewi` 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 Amusewiki services with `./init-all.sh start` and then start `caddy`.

The recommended way to install Amusewiki is via `http://packages.amusewiki` where a Debian repository with the packages is provided. 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 final section. Otherwise feel free to skip this document entirely.

Windows is not supported in any way.

General installation

The installation of Amusewiki takes time (depending on speed of the machine and of the network) and requires about 5Gb 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 10Gb for a reasonable start).

On a slow server, it takes 2 hours (mostly spent on installing and testing Perl modules, and 20 minutes to download and install the full TeX Live 2015), but the procedure is fully automated, so start it, check if it bails out at the beginning, forget about it for an hour or two (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 installing the available modules and TeX Live from the OS repositories, which is the suggested approach because this way the security fixes are delegated to the standard tools.

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

Prerequisites:

- a database (MySQL, PostgreSQL, SQLite are supported)
- a working Perl (i.e., you can install modules with cpanm without warnings). Usually this means you have to install `liblocal-lib-perl` and `cpanminus` in Debian (or you install Perl in your home).
- fontconfig (install it before installing TeX Live)
- GraphicsMagick (for thumbnails) and ImageMagick (for preview generation)
- a mime-info database: `shared-mime-info` on Debian

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` and `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:

```
$ 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.

```
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

```
Environment="EMAIL_SENDER_TRANSPORT=SMTP"
Environment="EMAIL_SENDER_TRANSPORT_host=smtp.example.com"
Environment="EMAIL_SENDER_TRANSPORT_port=2525"
```

Configuration file

Normally, you don't need to change anything. However, 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:

```
<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>
```


- 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 have a system wide perl, to install modules in your home you should install local::lib (liblocal-lib-perl on Debian) and add to .profile (or equivalent):

```
eval `perl -Mlocal::lib`
```

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

Logout and login again.

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

Install the prerequisites and complete the installation with:

Complete the installation with:

```
./script/install.sh
```

Configure the initial site

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

```
mysql> create database amuse DEFAULT CHARACTER SET utf8 DEF
mysql> grant all privileges on amuse.* to amuse@localhost ic
```

Or, for PostgreSQL:
Login as root.

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

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).
If you have multiple Amusewiki instance (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 the 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 first argument to the configure script.

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

```
$ rm root/git/cgit.cgi
$ ./script/generate-systemd-unit-files
$ ./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.

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, c
$ ./script/install-texlive.sh # install texlive
$ echo 'export PATH=$HOME/amw-perl/bin:$HOME/texlive/2017/bin/
$ chmod 755 $HOME  # open the home so plackup is accessible to
```

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/2017/bin/x86_64-linux/xe
```

Install cpanm and the dependencies

```
$ perl -MCPAN -e 'install App::cpanminus'
$ which cpanm # should be ~/amw-perl/bin/cpanm
$ 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
```


Start the application

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.

To regenerate the nginx configuration after adding a site:

```
./script/amusewiki-generate-nginx-conf
```

(and read the output).

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).

```
<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".

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:

```
# yum install epel-release
# yum install git nginx perl-local-lib sqlite cgit \
    perl-App-cpanminus fontconfig GraphicsMagick ImageMagick shared
    xapian-core xapian-core-devel unzip wget libxml2 libxml2-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
```

```
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
eval `perl -Mlocal::lib`
./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
```

Take note of the credentials, and follow the instructions.

FreeBSD

Install as much as possible from the repository:

```
pkg install perl5 p5-App-cpanminus git texlive-full-20150521 \
    cgit ImageMagick \
    GraphicsMagick shared-mime-info xapian-core xapian-bindings \
    nginx fcgiwrap unzip rsync wget bash \
    p5-Module-Install \
    p5-local-lib \
    p5-Catalyst-Devel \
    p5-DBIx-Class p5-DBIx-Class-Schema-Loader p5-DBIx-Class-Passphr
    p5-DBIx-Class-DeploymentHandler p5-DBIx-Class-Helpers p5-DBIx-C
    p5-DateTime-Format-RFC3339 \
    p5-Locale-Maketext-Lexicon \
    p5-Locale-PO p5-File-MimeInfo \
    p5-Protocol-ACME \
    p5-CAM-PDF                    \
    p5-Test-Differences                       \
    p5-Catalyst-Runtime                       \
    p5-Catalyst-Plugin-ConfigLoader           \
    p5-Catalyst-Plugin-Authentication         \
    p5-Catalyst-Plugin-Session                \
    p5-Catalyst-Plugin-Session-Store-FastMmap \
    p5-Catalyst-Plugin-Session-State-Cookie   \
    p5-Catalyst-Plugin-Authorization-Roles    \
    p5-Catalyst-View-TT                       \
    p5-Catalyst-Action-RenderView             \
    p5-Moose                                  \
    p5-namespace-autoclean                    \
    p5-Test-WWW-Mechanize-Catalyst            \
    p5-Term-Size-Any                          \
    p5-MIME-Types                             \
    p5-FCGI                                   \
    p5-FCGI-ProcManager                       \
```

```
p5-Unicode-Collate                          \
p5-DBIx-Class                               \
p5-DBD-SQLite                               \
p5-Daemon-Control                           \
p5-MooseX-NonMoose                          \
p5-JSON                                     \
p5-JSON-XS                                  \
p5-DBIx-Class-Schema-Loader                 \
p5-SQL-Translator                           \
p5-DBIx-Class-Schema-Config                 \
p5-DBIx-Class-PassphraseColumn              \
p5-DBIx-Class-InflateColumn-Authen-Passphrase \
p5-DateTime                                 \
p5-DateTime-Format-SQLite                   \
p5-DateTime-Format-MySQL                    \
p5-DateTime-Format-Pg                       \
p5-DateTime-Format-Strptime                 \
p5-XML-FeedPP                               \
p5-XML-Atom                                 \
p5-Git-Wrapper                              \
p5-Text-Wrapper                             \
p5-Email-Valid                              \
p5-File-Copy-Recursive                      \
p5-Search-Xapian                            \
p5-Catalyst-Model-Adaptor                   \
p5-Text-Unidecode                           \
p5-Log-Contextual                           \
p5-Log-Log4perl                             \
p5-Log-Dispatch                             \
p5-Log-Dispatch-File-Stamped                \
p5-Email-Sender                             \
p5-HTTP-Tiny            \
p5-MooseX-MarkAsMethods \
p5-PDF-API2 \
p5-Bytes-Random-Secure \
p5-Crypt-OpenSSL-X509 \
p5-HTTP-BrowserDetect \
p5-Type-Tiny \
p5-Archive-Zip \
p5-Template-Tiny \
```



```
    p5-Catalyst-Model-DBIC-Schema
```

```
[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]: bas
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 /usr/local/etc/nginx/sites-enabled
mkdir /var/log/nginx/
# and add an include directive in nginx.conf inside the http { } st
# include /usr/local/etc/nginx/sites-enabled/*;
```