trapper-project Documentation

Release 1.1.0

Open Science Conservation Fund (OSCF)

Jan 17, 2022

Contents:

CHAPTER 1

Overview

Trapper is an open source, django based web application to manage camera trapping projects. Motion-triggered camera trapping is increasingly becoming an important tool in ecological research. Because of the nature of collected data (multimedia files) even relatively small camera-trapping projects can generate large and complex datasets. The organization of these large collections of multimedia files and efficient querying for a particular subsets of data, especially in a spatio-temporal context, is often a challenging task. Without an appropriate software solution this can become a serious data management problem, leading to delays and inaccessibility of data in the long run. We propose a new approach which, in contrary to available software solutions, is a fully open-source web application using spatially enabled data that can handle arbitrary media types (both pictures and videos), supports collaborative work on a project and data sharing between system users. We used state of the art and well-recognized open-source software components and modern, general purpose programming language Python to design a flexible software framework for data management in camera trapping studies.

1.1 TRAPPER in a nuthsell

- it is open-source
- provides a spatially-enabled database backend
- can handle both pictures & videos
- provides a flexible model of classifications
- promotes data re-use
- supports collaborative work on a project
- provides API

1.2 Demo

https://demo.trapper-project.org

1.3 Read more

Bubnicki, J. W., Churski, M. and Kuijper, D. P. (2016), TRAPPER: an open source web-based application to manage camera trapping projects. Methods Ecol Evol, 7: 1209-1216. doi:10.1111/2041-210X.12571

https://besjournals.onlinelibrary.wiley.com/doi/10.1111/2041-210X.12571

For more news about TRAPPER please visit the Open Science Conservation Fund (OSCF) website:

https://os-conservation.org

CHAPTER 2

Installation

TRAPPER is provided as Docker container and can be deployed on both Linux- and Windows-based servers. However, we recommend a Linux distribution based on Debian e.g. Ubuntu Server. TRAPPER uses Docker composer, a tool for defining and running multi-container Docker applications. With this technology all basic services needed to run TRAPPER (including database and FTP server) can be automatically deployed and configured on a dedicated server with a possibility left open to configure particular services manually (e.g. a scenario with an external database server).

2.1 Installing Docker Community Edition (CE) & Docker Compose

1) Follow the steps from the official Docker documentation:
   - Ubuntu: https://docs.docker.com/install/linux/docker-ce/ubuntu/
   - Windows: https://hub.docker.com/editions/community/docker-ce-desktop-windows
2) Install Docker Compose: https://docs.docker.com/compose/

Note: To learn more about Docker technology including the management of Docker images and containers please read the official Docker documentation.

2.2 Getting the source code of TRAPPER

1) First, clone the source code of TRAPPER to your local repository:

```
$ git clone https://gitlab.com/oscf/trapper-project.git
```

2) To get the most up-to-date version of TRAPPER switch to a branch `development`:

```
$ git checkout development
```

Note: It is much easier to use Git Bash for cloning git repositories on Windows instead of using a standard Windows command line: https://git-scm.com/download/win

2.3 Running TRAPPER

Note: If you need a custom configuration of TRAPPER (e.g. external database, ftp and email services) please check the *Configuration* section for details.

1) Copy the `trapper.env` file to the `.env` file, you can do it using the command:

```
$ cp trapper.env .env
```

2) Adjust the variables in the `.env` file if you use a non-standard installation, e.g. the external postgresql database.
3) Run TRAPPER with the following command:
   - Production

   ```
   $ ./start.sh -pb prod
   ```

   - Development

   ```
   $ ./start.sh -pb dev
   ```

Note: Flags `-p` and `-b` tell the script to `pull` all required docker images and to `build` the `trapper` image locally. To learn more about all possible running options simply type `./start.sh`. For example, to run TRAPPER with an external postgresql database you can use flag `-d` (but first check the *Using an external database* section).

Warning: The process of pulling, building & starting all docker containers can take a while!

4) Now your TRAPPER instance should be up!

2.4 Creating the admin (superuser) account

Enter the `trapper` docker container and create the `superuser`:

```
$ docker exec -it trapper bash
$ python /app/trapper/trapper-project/manage.py createsuperuser
```

After providing a username, email and password you should be able to login to TRAPPER.

2.5 SSL certificates

If you already have the SSL certificates for your TRAPPER website you can use them by providing proper paths in the *.env* configuration file. See the *SSL certificates* section for details. These certificates will be used to configure the

HTTPS `nginx` proxy-server within a dedicated container. When there is no SSL certificates provided they will be generated automatically using the openssl toolkit.

Note: For example, you can get free SSL certificates from Let's Encrypt.

CHAPTER 3

Configuration

You can override many default settings in TRAPPER by adjusting the environmental variables listed in the file *.env*. Further in this section we provide some examples of how to customize selected aspects of TRAPPER.

3.1 Basic settings

Provide a name of your TRAPPER instance domain:

```
DOMAIN_NAME=demo.trapper-project.org
```

The `SECRET_KEY` is used to provide cryptographic signing, and should be set to a unique, unpredictable value (see django docs):

```
SECRET_KEY='cudv86xkzva)o1lkt-ow72)y__*xwsb1@43wtp1cc@zak=i@@e'
```

Define users that are admins of your TRAPPER instance. If the *Email service* is properly configured they will be notifying about all server errors (500), new users registrations and requests for approving new research projects.

```
ADMIN1=admin1,admin1@oscf.org # {username},{email}
ADMIN2=admin2,admin2@oscf.org
```

HTTP & HTTPS ports mapping definition is used to configure the trapper-nginx docker container (see nginx docker docs):

```
HTTP_PORTS=80:80
HTTPS_PORTS=443:443
```

3.2 SSL certificates

If you already have the SSL certificates for your TRAPPER website you can use them by setting the following variables:

```
# default configuration
SSL_CERTIFICATE=${PWD}/ssl/cert.pem
SSL_CERTIFICATE_KEY=${PWD}/ssl/key.pem
```

The example with the Let's Encrypt certificates:

```
SSL_CERTIFICATE=/etc/letsencrypt/live/trapper/fullchain.pem
SSL_CERTIFICATE_KEY=/etc/letsencrypt/live/trapper/privkey.pem
```

These certificates will be used to configure the HTTPS nginx proxy-server within a dedicated container (see trapper-nginx) and to secure the FTP connections (TLS) to the pure-ftpd docker container.

Note: When there is no SSL certificates provided they will be generated automatically using the openssl toolkit (self-signed certificates).

3.3 Using an external database

TRAPPER comes with a pre-configured postgresql docker image (see trapper-posgresql) and has been proved to work well with the recent postgresql-12 and postgis-3.0. The basic credentials needed to access this dockerized postgresql database volume can be configured with these parameters:

```
POSTGRES_USER=trapper
POSTGRES_PASSWORD=trapper
POSTGRES_DB=trapper
```

However, for a production-grade TRAPPER instance we recommend to configure an external postgresql database either running on the same virtual machine or on different remote server. In the *.env* file there are the following variables that can be used to configure a connection with an external database:

```
DB_NAME=trapper
DB_USER=trapper
DB_PASS=trapper
DB_HOST=172.17.0.1
DB_PORT=5432
```

Note: The host address `172.17.0.1` is for the case when the postgresql database is running on the same machine as TRAPPER docker container. This host ip should work for most of the standard Linux distributions, including the recent Ubuntu Server LTS 20.04.

Now you can run TRAPPER using the script *start.sh* and the extra flag *-d*:

```
$ ./start.sh -pbd prod
```

Note: You will still need to properly configure your external postgresql server which is out of the scope of this documentation. Here we only give you a couple of quick cues to help you make the connection between TRAPPER and postgresql working as expected (but be aware that likely you need more restrictive configuration!):

1) In the file `/etc/postgresql/12/main/postgresql.conf` update the following line:

```
listen_addresses = '*'
```

2) In the file `/etc/postgresql/12/main/pg_hba.conf` add the following entry:

```
# TYPE   DATABASE    USER   ADDRESS       METHOD
host     all         all    172.0.0.0/8   md5
```

172.0.0.0/8 matches all ip addresses starting with 172.X.X.X

3.4 Storage

The multimedia files uploaded to TRAPPER (i.e. camera trapping images & videos) can be stored either locally, i.e. using a storage infrastructure available at a virtual machine running TRAPPER docker container, or remotely in a cloud storage (see the subsection *Cloud storage* for details about supported cloud storage providers).

To use a local storage (default option) leave the following parameter empty:

```
DEFAULT_FILE_STORAGE=
```

Optionally, you can provide paths to local directories which will be used to mount the corresponding docker volumes:

```
VOL_MEDIA=/storage/trapper_data/media
VOL_EXTERNAL_MEDIA=/storage/trapper_data/external_media
```

Note: In a directory mounted as the volume `VOL_MEDIA` all images & videos uploaded to TRAPPER are stored in the following format:

```
# original files
protected/storage/resources/{user_id}/{date_uploaded}/

# preview files (for images)
protected/storage/resources/{user_id}/{date_uploaded}/previews/

# thumbnails (for both images & videos)
protected/storage/resources/{user_id}/{date_uploaded}/thumbnails/
```

The volume `VOL_EXTERNAL_MEDIA` provides a storage for users' data including FTP data (e.g. users' uploaded `.zip` files with multimedia collections) and all data packages exported from TRAPPER.

3.4.1 Cloud storage

The TRAPPER support for a cloud storage is based on a wonderful django package django-storages. Here is the list of cloud storage providers currently supported by TRAPPER and their corresponding configuration parameters:

1) Azure

```
DEFAULT_FILE_STORAGE=azure

AZURE_ACCOUNT_NAME=trapper
AZURE_ACCOUNT_KEY=key
AZURE_CONTAINER=trapper_data
```

(continues on next page)

(continued from previous page)

```
AZURE_CUSTOM_DOMAIN=trapper.blob.core.windows.net
MEDIA_LOCATION=media
```

2) Amazon S3

AWS_ACCESS_KEY_ID and *AWS_SECRET_ACCESS_KEY* - credentials for authorizing access to S3 storage.

AWS_STORAGE_BUCKET_NAME - S3 bucket name for storing all media files

AWS_S3_REGION_NAME - Name of AWS S3 region to use. Optional.

AWS_S3_ENDPOINT_URL - S3 Endpoint, required in order to use alternative S3-compatible storages such as Minio or Backblaze B2

Setting default file storage as *amazon_s3* is required in order to use S3 integration in TRAPPER

```
DEFAULT_FILE_STORAGE=amazon_s3
```

3.5 Email service

The working email service is required to make the notification framework working correctly. This feature is turned off by default. To configure the email notification backend you should set the following parameters (example with gmail):

```
EMAIL_NOTIFICATIONS=True
EMAIL_NOTIFICATIONS_RESEARCH_PROJECT=True
EMAIL_HOST=smtp.gmail.com
EMAIL_PORT=587
EMAIL_HOST_USER=project@gmail.com
EMAIL_HOST_PASSWORD=password
EMAIL_USE_TLS=True
```

Note: If local SMTP server is used it has to be configured to be able to send email messages. Configuration of the SMTP server is out of the scope of this documentation.

3.6 Users registration

Here you can define if users registration (signing-up) is open or not and optionally secure a user registration form with Google's recaptcha authentication.

```
USER_REGISTRATION_OPEN=True

# Google recaptcha settings
USE_RECAPTCHA=True
RECAPTCHA_PUBLIC_KEY=public_key
RECAPTCHA_PRIVATE_KEY=private_key
```

3.7 Django Debug Toolbar

If you are a developer of TRAPPER you can turn on a very cool tool for debugging called Django Debug Toolbar:

```
USE_DEBUG_TOOLBAR=True
DEBUG_TOOLBAR_USERS=admin1,admin2
```

3.8 Advanced system-level variables

These variables are set automatically when starting TRAPPER with the script `start.sh`. You can override them, but please keep defaults when you are not really sure what you can change here.

1) Set UID & GID for a user that will be used to run TRAPPER Gunicorn server within a docker container (by default these values are taken from a user starting docker containers using the `start.sh` script). The example custom configuration:

   ```
   TRAPPER_USER_UID=1000
   TRAPPER_USER_GID=1000
   ```

2) Set GID for a docker group which will be created within a TRAPPER docker container. The example custom configuration:

   ```
   DOCKER_GID=131
   ```

CHAPTER 4

Administration

Coming soon!

CHAPTER 5

Tutorial

Coming soon!

CHAPTER 6

Technology used - software components

6.1 Backend

The following is the list of the most important software components that have been used to design and implement Trapper:

- Ubuntu A base operating system under which Trapper was developed and tested.
- PostgreSQL The open source and industry standard relational database management system (RDBMS) with PostGIS as its spatial extension.
- nginx nginx [engine x] is an HTTP and reverse proxy server, a mail proxy server, and a generic TCP proxy server.
- Gunicorn This pure-python application is used to serve Trapper project.
- Supervisor This application is used to control all project related external applications like celery.
- Celery This python application is used to run various tasks in asnynchronous mode allowing Trapper to work faster and more efficient. The celery is used to generate thumbnails from large files or videos, or to process uploaded collections by users.
- Django The core component of Trapper, a high-level Python web framework maintained by the Django Software Foundation. Django framework simplifies and significantly speeds up the creation of complex, database-driven websites and emphasizes reusability and pluggability of their components. Additionally, a variety of 3rd party, open source django applications have been used to develop Trapper. For a complete list of these applications see **‘this file <>‘_**
- uMAP uMap lets you create maps with OpenStreetMap layers in a minute and embed them in your site. It uses django-leaflet-storage and Leaflet.Storage, built on top of Django and Leaflet.

6.2 Frontend

Trapper’s front-end is developed based on three independent solutions:

1) HTML (version 5) templates and their CSS styles (powered by SASS)
2) set of scripts written in pure JavaScript (ECMAscript 5).
3) all the external libraries and frameworks included in the project:

- Twitter Bootstrap Bootstrap Sass Official which is official SASS version of Twitter Bootstrap. This library provides a set of HTML components and CSS styles used for Trapper scaffold creation.
- Font Awesome Font Awesome - webfont of vector icons used in the project.
- Angular JS Angular JS all the grids/tables including their filters has been build on top of this Google's framework
- Angular Cookies Official angular module for cookies management.
- Angular Sanitize Official angular module which improves angular templates data binding.
- Moment Extremaly powerful library for date parsing & manipulation.
- Select2 Complete solution that extends default HTML select controls.
- Select2 Bootstrap CSS CSS styles for Select2 so it fits Twitter Bootstrap feel & look.
- Bootstrap WYSIHTML5 Javascript Plugin which brings WYSIWYG text editor to the table.
- Bootstrap Datepicker http://eternicode.github.io/bootstrap-datepicker/ Javascript Widget - simple datepicker.
- Jquery Timepicker Jquery Widget which is just a timepicker.
- Bootstrap Datetimepicker Javascript Widget that combines both time and date picker.
- Video JS This library extends standard HTML5 video players.
- Video JS Rangeslider Video JS plugin that allows to set and get video sequences.

CHAPTER 7

License

CHAPTER 8

Indices and tables

- genindex
- modindex
- search