

---

# **GreedyBear**

***Release 0.2.0***

**Matteo Lodi**

**Jul 15, 2022**



# CONTENTS

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Public feeds . . . . .	1
<b>2</b>	<b>Installation</b>	<b>3</b>
<b>3</b>	<b>Usage</b>	<b>5</b>
<b>4</b>	<b>Contribute</b>	<b>7</b>
4.1	Rules . . . . .	7
4.2	Code Style . . . . .	7
4.3	How to start (Setup project and development instance) . . . . .	8
4.4	Create a pull request . . . . .	8
<b>5</b>	<b>Tests</b>	<b>9</b>
<b>6</b>	<b>GreedyBear API docs</b>	<b>11</b>
<b>7</b>	<b>GreedyBear API Redoc</b>	<b>13</b>
	<b>HTTP Routing Table</b>	<b>15</b>



## INTRODUCTION

The project goal is to extract data of the attacks detected by a [TPOT](#) or a cluster of them and to generate some feeds that can be used to prevent and detect attacks.

[Official announcement here.](#)

### 1.1 Public feeds

There are public feeds provided by The Honeynet Project in this site: [greedybear.honeynet.org](http://greedybear.honeynet.org). [Example](#)

To check all the available feeds, Please refer to our [usage guide](#)

Please do not perform too many requests to extract feeds or you will be banned.

If you want to be updated regularly, please download the feeds only once every 10 minutes (this is the time between each internal update).



## INSTALLATION

Start by cloning the project

```
# clone the Greedybear project repository
git clone https://github.com/honeynet/GreedyBear
cd GreedyBear/

# construct environment files from templates
cp .env_template .env
cd docker/
cp env_file_template env_file
cp env_file_postgres_template env_file_postgres
cd ..
```

Now you can start by building the image using docker-compose and run the project.

```
# build the image locally
docker-compose -p greedybear build

# start the app
docker-compose -p greedybear up

# now the app is running on http://localhost:80

# shut down the application
docker-compose -p greedybear down
```

Note that GreedyBear *needs* a running instance of ElasticSearch or a TPoT to function. If you don't have one, you can make the following changes to make GreedyBear spin up its own ElasticSearch and Kibana instances. (...Care! This option would require enough RAM to run the additional containers. Suggested is >=16GB):

1. In `docker/env_file`, set the variable `ELASTIC_ENDPOINT` to `http://elasticsearch:9200`.
2. Add `:docker/elasticsearch.yml` to the last defined `COMPOSE_FILE` variable or uncomment the `# local development with elasticsearch` container block in `.env` file.





## USAGE

GreedyBear is created with the aim to collect the information from the TPOTs and generate some actionable feeds, so that they can be easily accessible and act as valuable information to prevent and detect attacks.

Check either the [Redoc specification](#) to get all the details about how to use the available APIs.



## CONTRIBUTE

### 4.1 Rules

GreedyBear welcomes contributors from anywhere and from any kind of education or skill level. We strive to create a community of developers that is welcoming, friendly and right.

For this reason it is important to follow some easy rules based on a simple but important concept: **Respect**.

- Before starting to work on an issue, you need to get the approval of one of the maintainers. Therefore please ask to be assigned to an issue. If you do not that but you still raise a PR for that issue, your PR can be rejected. This is a form of respect for both the maintainers and the other contributors who could have already started to work on the same problem.
- When you ask to be assigned to an issue, it means that you are ready to work on it. When you get assigned, take the lock and then you disappear, you are not respecting the maintainers and the other contributors who could be able to work on that. So, after having been assigned, you have a week of time to deliver your first *draft* PR. After that time has passed without any notice, you will be unassigned.
- Before asking questions regarding how the project works, please read *through all the documentation* and `install` the project on your own local machine to try it and understand how it basically works. This is a form of respect to the maintainers.
- Once you started working on an issue and you have some work to share and discuss with us, please raise a draft PR early with incomplete changes. This way you can continue working on the same and we can track your progress and actively review and help. This is a form of respect to you and to the maintainers.
- When creating a PR, please read through the sections that you will find in the PR template and compile it appropriately. If you do not, your PR can be rejected. This is a form of respect to the maintainers.

### 4.2 Code Style

Keeping to a consistent code style throughout the project makes it easier to contribute and collaborate. We make use of `psf/black` and `isort` for code formatting and `flake8` for style guides.

## 4.3 How to start (Setup project and development instance)

To start with the development setup, make sure you go through all the steps in [Installation Guide](#) and properly installed it.

Please create a new branch based on the **develop** branch that contains the most recent changes. This is mandatory.

```
git checkout -b myfeature develop
```

Then we strongly suggest to configure [pre-commit](#) to force linters on every commits you perform:

```
# create virtualenv to host pre-commit installation
python3 -m venv venv
source venv/bin/activate
# from the project base directory
pip install pre-commit
pre-commit install
```

Remember that whenever you make changes, you need to rebuild the docker image to see the reflected changes.

### 4.3.1 NOTE about documentation:

If you made any changes to an existing model/serializer/view, please run the following command to generate a new version of the API schema and docs:

```
docker exec -it greedybear_uwsgi python manage.py spectacular --file docs/source/schema.
↪ yml && make html
```

## 4.4 Create a pull request

### 4.4.1 Remember!!!

Please create pull requests only for the branch **develop**. That code will be pushed to master only on a new release.

Also remember to pull the most recent changes available in the **develop** branch before submitting your PR. If your PR has merge conflicts caused by this behavior, it won't be accepted.

### 4.4.2 Install testing requirements

Run `pip install -r requirements/test-requirements.txt` to install the requirements to validate your code.

#### Pass linting and tests

Run `psf/black` to lint the files automatically, then `flake8` to check and `isort`.

(if you installed `pre-commit` this is performed automatically at every commit)

if you get any errors, fix them. Once you make sure that everything is working fine, please squash all of our commits into a single one and finally create a pull request.

---

CHAPTER  
**FIVE**

---

**TESTS**



## GREEDYBEAR API DOCS

### GET /api/enrichment

Request if a specific observable (domain or IP address) has been listed by GreedyBear

#### Status Codes

- 200 OK –

### GET /api/feeds/{feed\_type}/{attack\_type}/{age}.{format\_}

Request if a specific observable (domain or IP address) has been listed by GreedyBear

#### Parameters

- **age** (*string*) –
- **attack\_type** (*string*) –
- **feed\_type** (*string*) –
- **format** (*string*) –

#### Status Codes

- 200 OK – No response body





## **GREEDYBEAR API REDOC**



# HTTP ROUTING TABLE

/api  
GET /api/enrichment, 11  
GET /api/feeds/{feed\_type}/{attack\_type}/{age}.{format\_},  
11