Introducing SubCrawl

A Framework for the Analysis and Clustering of Hacking Tools Found Using Open Directories

Josh Stroschein

Malware Analyst, HP Inc.

josh.stroschein@hp.com
@jstrosch

Patrick Schläpfer

Malware Analyst, HP Inc.

patrick.schlaepfer@hp.com
@stoerchl

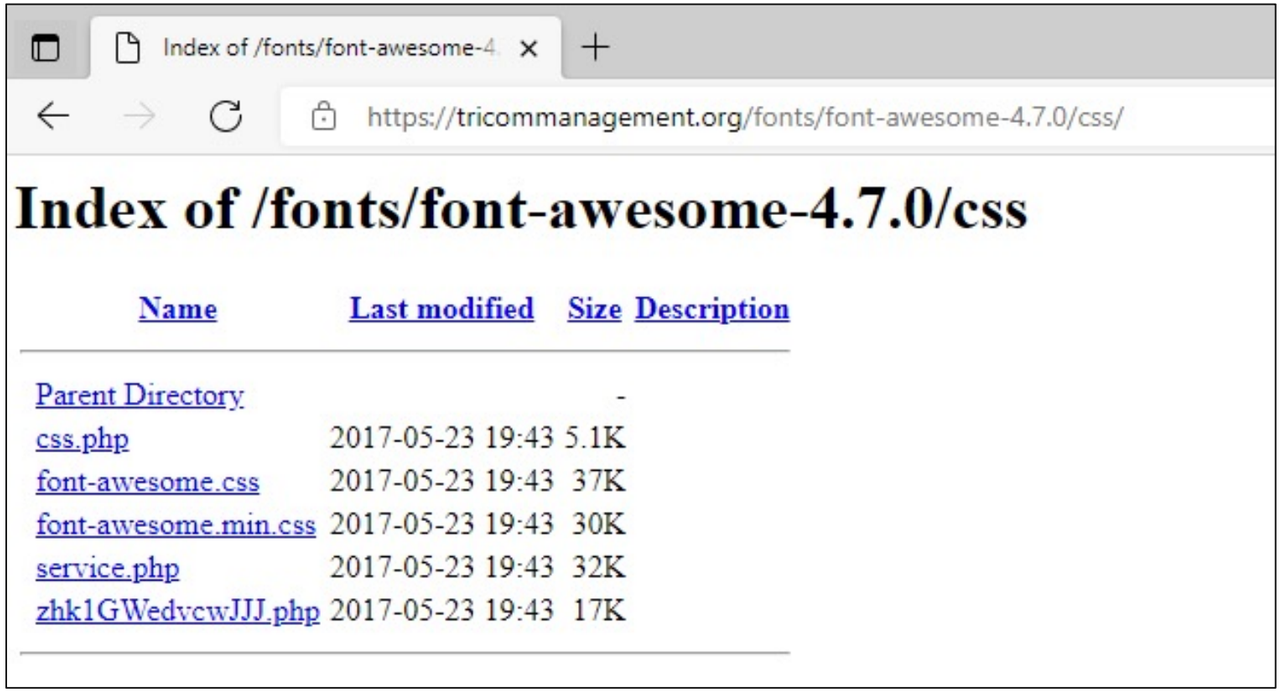
Alex Holland

Malware Analyst, HP Inc.

alex.holland@hp.com
@cryptogramfan

What Are Open Directories?

- An open directory shows the file listing on a webserver
- Sometimes show additional information like timestamps, filesizes and descriptions
- Allow users to download files which are normally interpreted by the webserver
- Usually start with “Index of…”
- Parent and/or child directories can be traversed



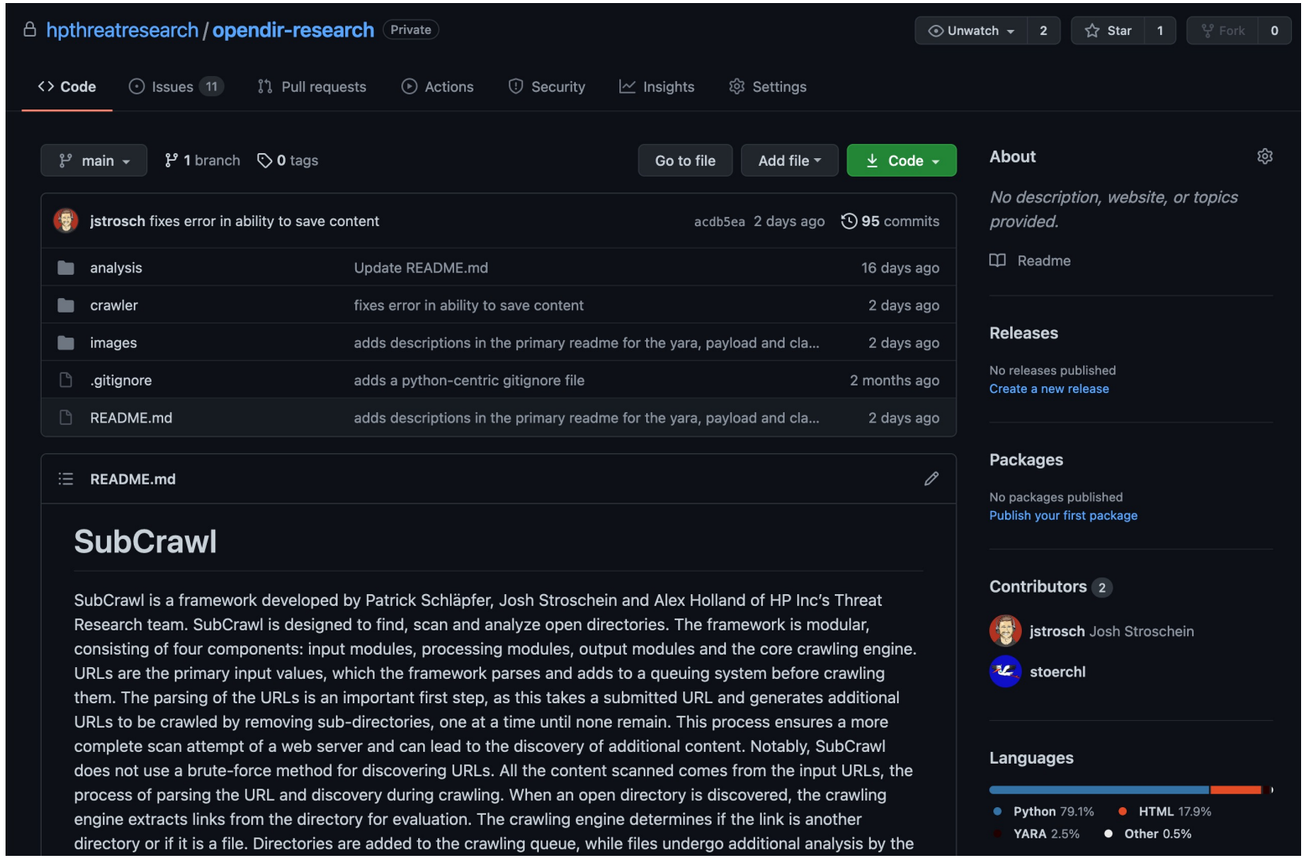
Why Open Directories?

From time to time, open directories are found on websites that are used for malware hosting and can thus provide malware analysts with a deeper insight into the used tools. Scanning and analyzing a large amount of open directories hosting malware can potentially answer two interesting questions:

- Do threat actors compromise the websites or buy access to it?
- How are the websites compromised?

Introducing SubCrawl

- To systematically detect and analyze such open directories, we developed SubCrawl.
- SubCrawl is a modular framework that scans URLs for open directory properties and analyzes their content using various processing modules.
- A few examples of such processing modules are:
  - *PayloadProcessing*
  - *YARAProcessing*
  - *ClamAVProcessing*
  - *TLSHProcessing*



https://github.com/hpthreatresearch/subcrawl

SubCrawl Modes of Operation

- SubCrawl offers two different operation modes.
  Run-Once Mode
  Service Mode

- The run-once mode is suitable if you want to quickly scan a website and evaluate the results directly on the console.

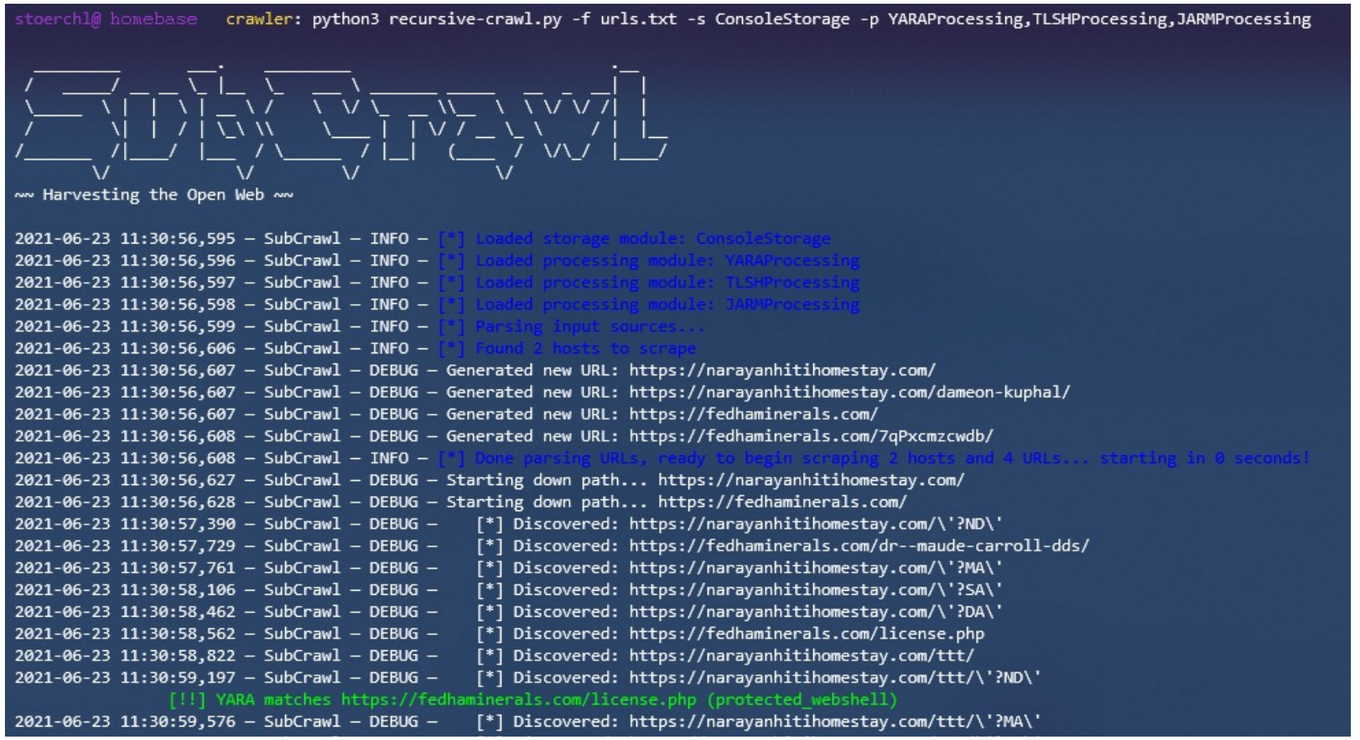
- The service mode is intended for scanning a larger number of websites over a longer period of time. The results are saved for later analysis using a configured storage module.

exe, tlp:green, opendir	283	misp@stoerchl.ch	2021-06-24	jx2.info
tlp:green	3860	patrick.schlapfer@hp.com	2021-05-03	Domain track event
32, exe, asyncrat, tlp:green	21	misp@stoerchl.ch	2021-06-24	upload.vina-host.com
32, exe, tlp:green	34	misp@stoerchl.ch	2021-06-24	www.56kdw.com
32, exe, tlp:green	16	misp@stoerchl.ch	2021-06-24	simhisancak.xyz
32, exe, tlp:green	21	misp@stoerchl.ch	2021-06-24	downloadas.xyz
32, exe, tlp:green	17	misp@stoerchl.ch	2021-06-24	down.esudaa.com
32, redlinestealer, exe, tlp:green	18	misp@stoerchl.ch	2021-06-24	immigrationentry.xyz

SubCrawl Storage Modules

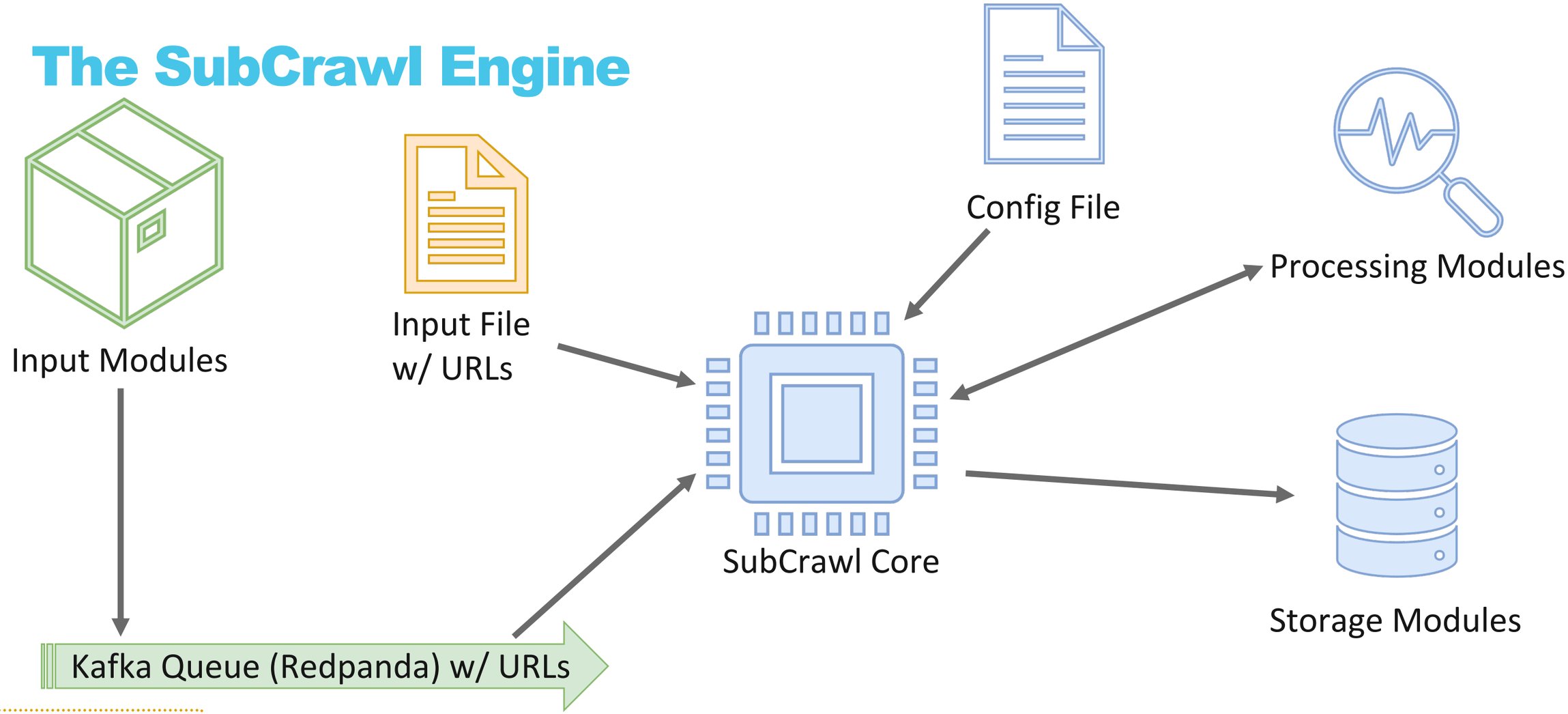
- The results from the processing modules can then be printed on the command line or saved in the desired form for later analysis based on the activated storage modules.

- Currently the following additional storage modules are implemented:
  SQLiteStorage
  MISPStorage



SubCrawl Run-once Mode Demo

The SubCrawl Engine



Service Mode Setup

- To make not only the run-once mode easy to use, but also the service mode, we wrapped it in a Docker container.
- As already shown in the framework architecture, the service mode relies on a queuing system. For this purpose, we use Redpanda, which is also started directly as a separate Docker container.

crawler > docker-compose.yml

```
version: "3.9"
services:
  web:
    build: .
```

```
FROM python:3.8-slim-buster
WORKDIR /subcrawl
RUN apt-get -y update
RUN apt-get -y upgrade
RUN apt-get -y install build-essential gcc \
    yara magic supervisor clamav-daemon \
    clamav-freshclam clamav-unofficial-sigs
COPY requirements.txt requirements.txt
COPY supervisor/supervisord.conf /etc/supervisor/
RUN mkdir /var/log/subcrawl
RUN pip3 install -r requirements.txt
RUN freshclam
EXPOSE 8000
COPY . .
CMD ["bash", "run.sh"]
```

```
    - PLAINTEXT://0.0.0.0:29092,OUTSIDE://0.0.0.0:9092
    - --advertise-kafka-addr
    - PLAINTEXT://redpanda:29092,OUTSIDE://localhost:9092
    image: docker.vectorized.io/vectorized/redpanda:latest
    container_name: redpanda-1
    ports:
    - 9092:9092
    - 29092:29092
```

Service Mode Setup

- The two developed modules from URLhaus and Phishtank are used as URL input. The URLs to be scanned are added to the queue, which is continuously processed by the crawler.

- The processing and storage modules that are to be used must be entered in the config file

```
processing_modules:
  - ClamAVProcessing
  - WebshellProcessing
  - JARMProcessing
  - TLSHProcessing
  - YARAProcessing
storage_modules:
  - SqliteStorage
```

Service Mode Setup

- To build and start the container execute the following command:

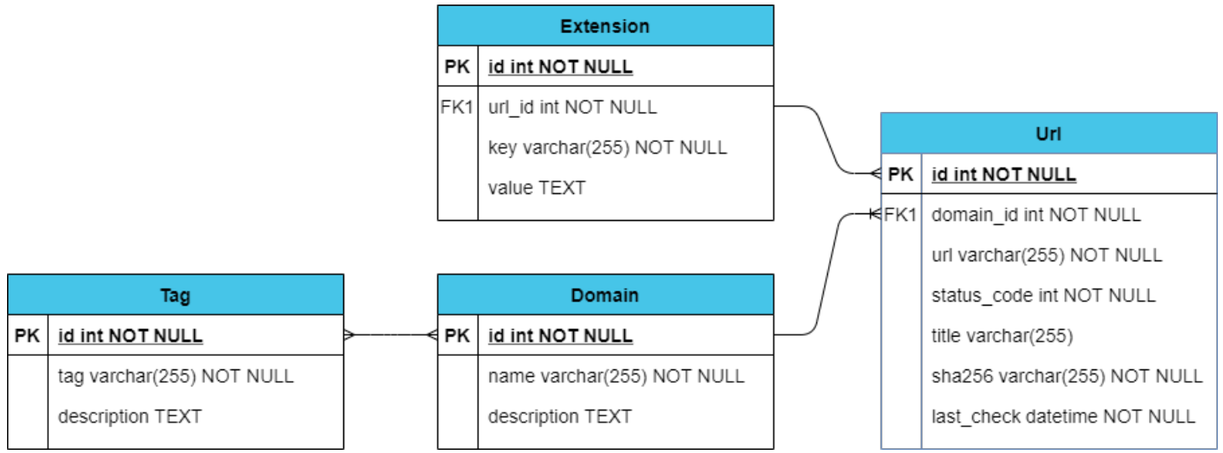
docker-compose up –build

- As soon as the containers are built and running, the web interface is reachable on port 8000

```
web_1       | Starting ClamAV daemon: clamd .
web_1       | Starting supervisor: supervisord.
web_1       | [2021-06-23 12:19:06 +0000] [368] [INFO] Starting gunicorn 20.0.4
web_1       | [2021-06-23 12:19:06 +0000] [368] [INFO] Listening at: http://0.0.0.0:8000 (368)
web_1       | [2021-06-23 12:19:06 +0000] [368] [INFO] Using worker: sync
web_1       | [2021-06-23 12:19:06 +0000] [373] [INFO] Booting worker with pid: 373
web_1       | [2021-06-23 12:19:06 +0000] [374] [INFO] Booting worker with pid: 374
web_1       | [2021-06-23 12:19:06 +0000] [375] [INFO] Booting worker with pid: 375
web_1       | [2021-06-23 12:19:06 +0000] [377] [INFO] Booting worker with pid: 377
```

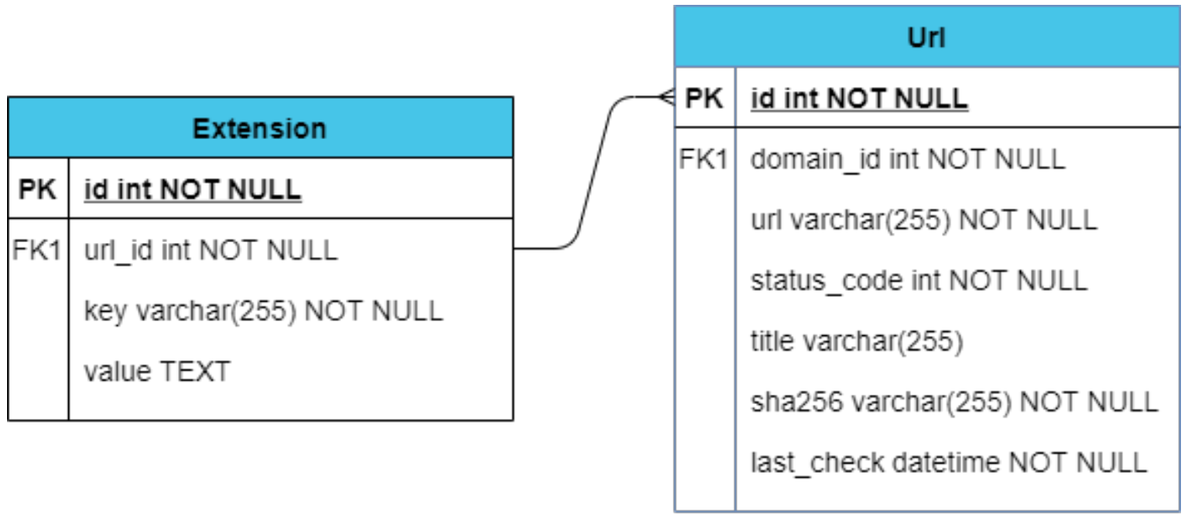
IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
crawler_web	"bash run.sh"	7 minutes ago	Up 7 minutes	0.0.0.0:8000->8000/tcp,	crawler_web_1
docker.vectorized.io/vectorized/redpanda:latest	"/usr/bin/rpk redpan…"	2 days ago	Up 7 minutes	0.0.0.0:9092->9092/tcp, 0.0.0.0:29092->29092/tcp, 9644/tcp	redpanda-1

SQLite Storage Module - Database



- The SQLite Storage module is intended for longer storage of scan results.
- SQLite is configured as default storage module in service mode if the supplied Docker container is used.
- The database schema is kept very simple with most tables being self-explanatory.

SQLite Storage Module - Database



- The “Extension” table is used to save HTTP headers of the scanned website as well as results of the processing modules
- Simple key – value table connected with an URL

SQLite Storage Module - Import

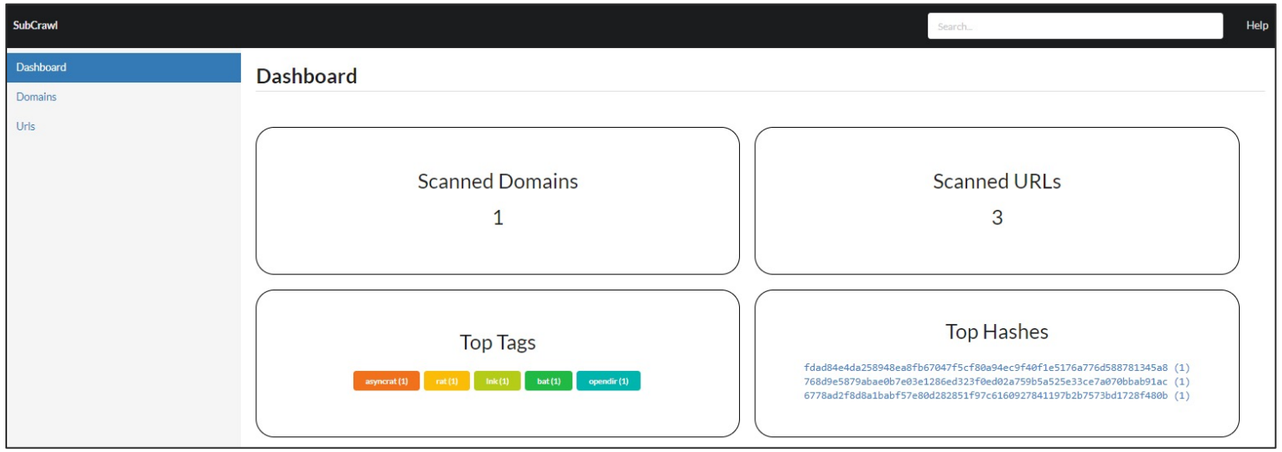
```
def store_result(self, result_data):
    # Load URLHaus tags
    url_info = dict()
    r = requests.get(self.cfg["misp"]["urlhaus_api"], allow_redirects=True)
    csv_data = io.StringIO(r.content.decode("utf-8"))
    counter = 0
    while counter < 8:
        next(csv_data)
        counter += 1

    csv_reader = csv.DictReader(csv_data)
    for row in csv_reader:
        domain = urlparse(row["url"]).netloc
        if domain not in url_info:
            url_info[domain] = set()
        url_info[domain].update(row["tags"].lower().split(","))
```

- The implementation of the SQLite storage module imports the database model, which is based on peewee from the utils folder.
- To enrich the data with additional contextual information, the storage module imports the tags from URLhaus and links them to the scanned domains.

SQLite Storage Module - Visual

- In order to analyze and manage the results of the scanned web pages in a user-friendly way, we have developed a simple web application.
- This allows the user to view, partially edit, delete or also search for data.
- On the home page, we implemented simple metrics that show how many domains and URLs have already been scanned and which tags and hashes occur the most.

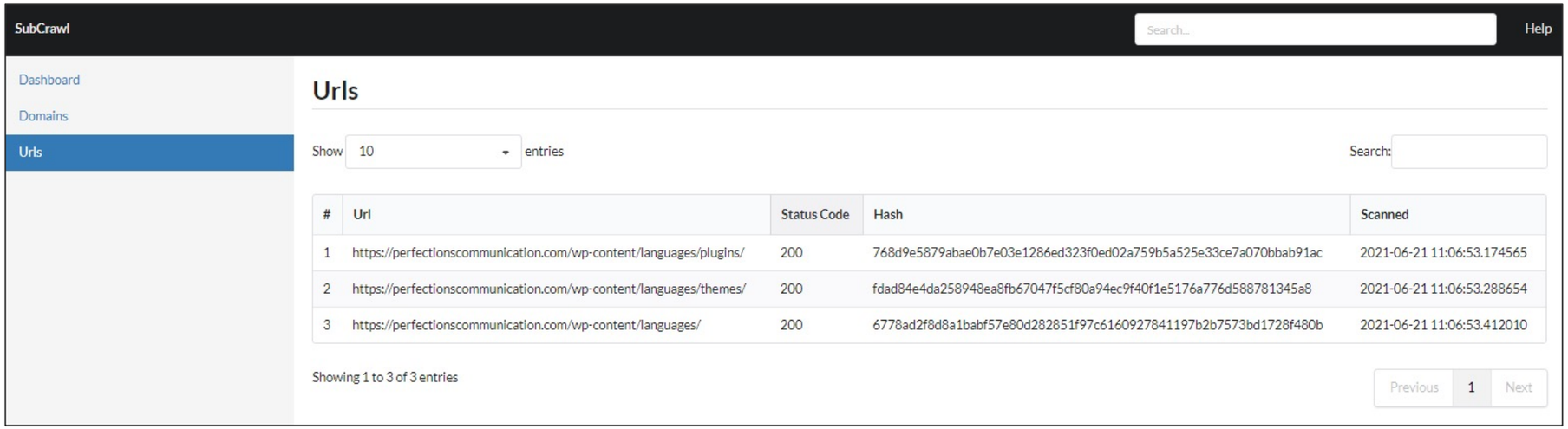


SQLite Storage Module - Visual

- The search can be used to query all the fields of the URLs, using the following syntax:

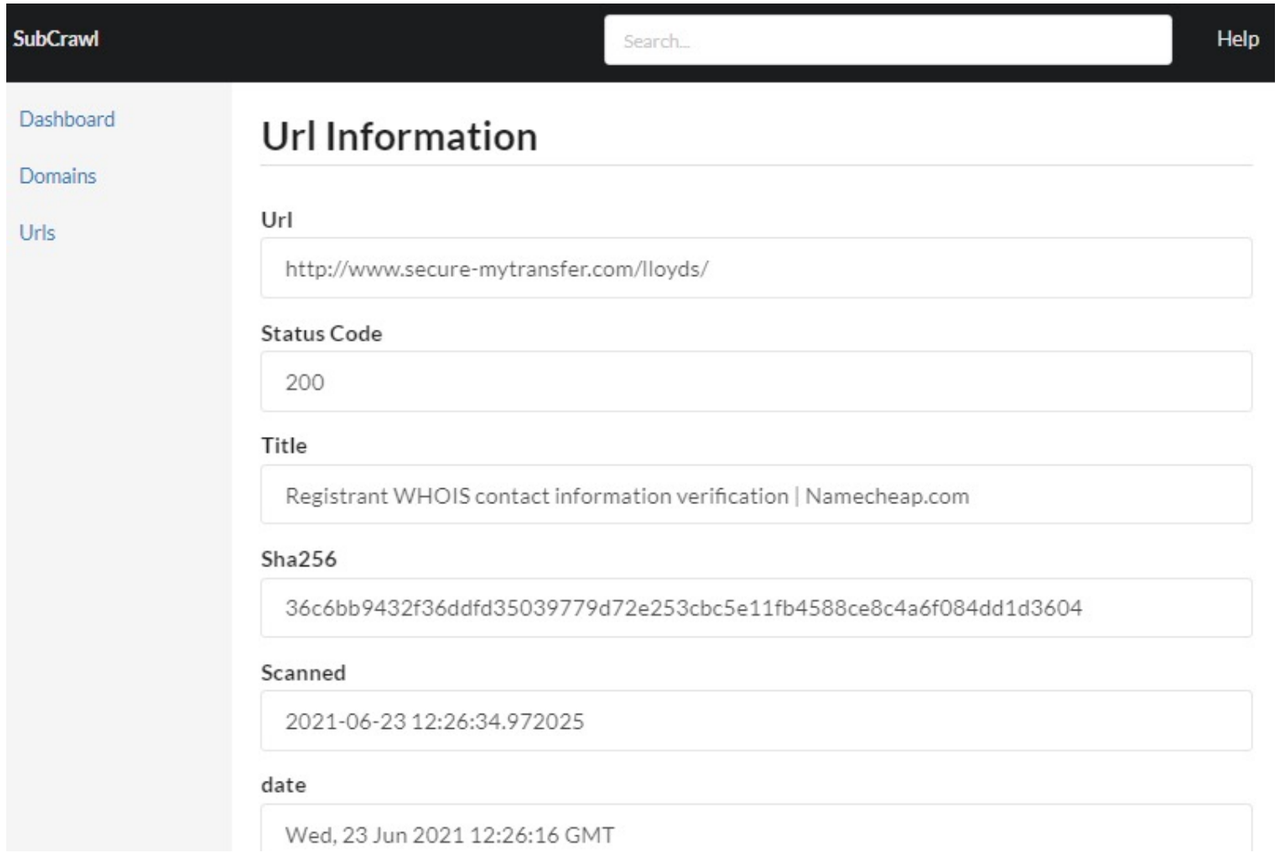
fieldname:value

- The result is then displayed in a list of found URLs.



SQLite Storage Module - Visual

- Additionally, there exists a detail view for each url and domain.
- Most of the fields can not be modified.
- It is possible to add a description to a domain



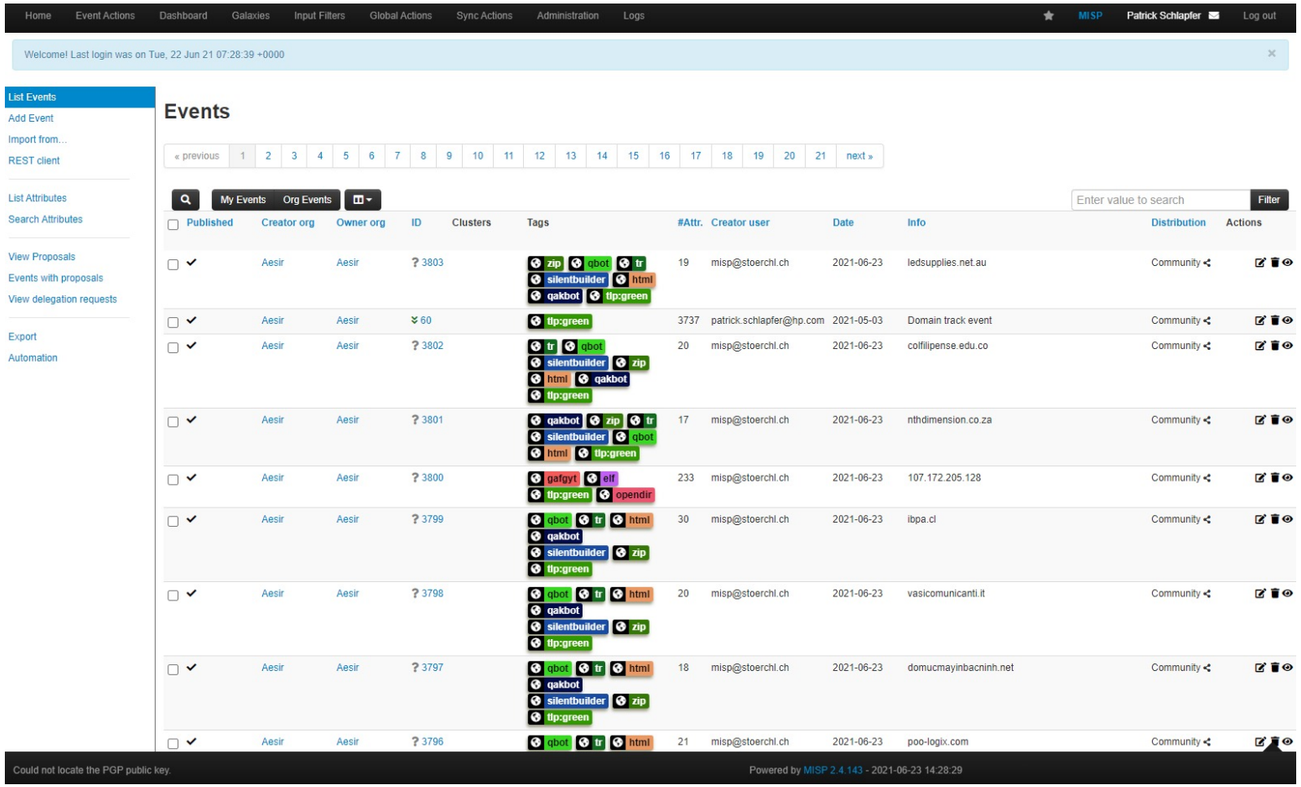
SubCrawl Service Mode Demo

MISP Storage Module

- Our SQLite user interface is rather basic and does not offer extended analysis capabilities
- MISP has a lot more functionalities and is widely used in the security community
- We did not choose MISP as default storage module because of the installation and configuration complexity
- Many MISP Docker images exist but they still need to be configured by the user

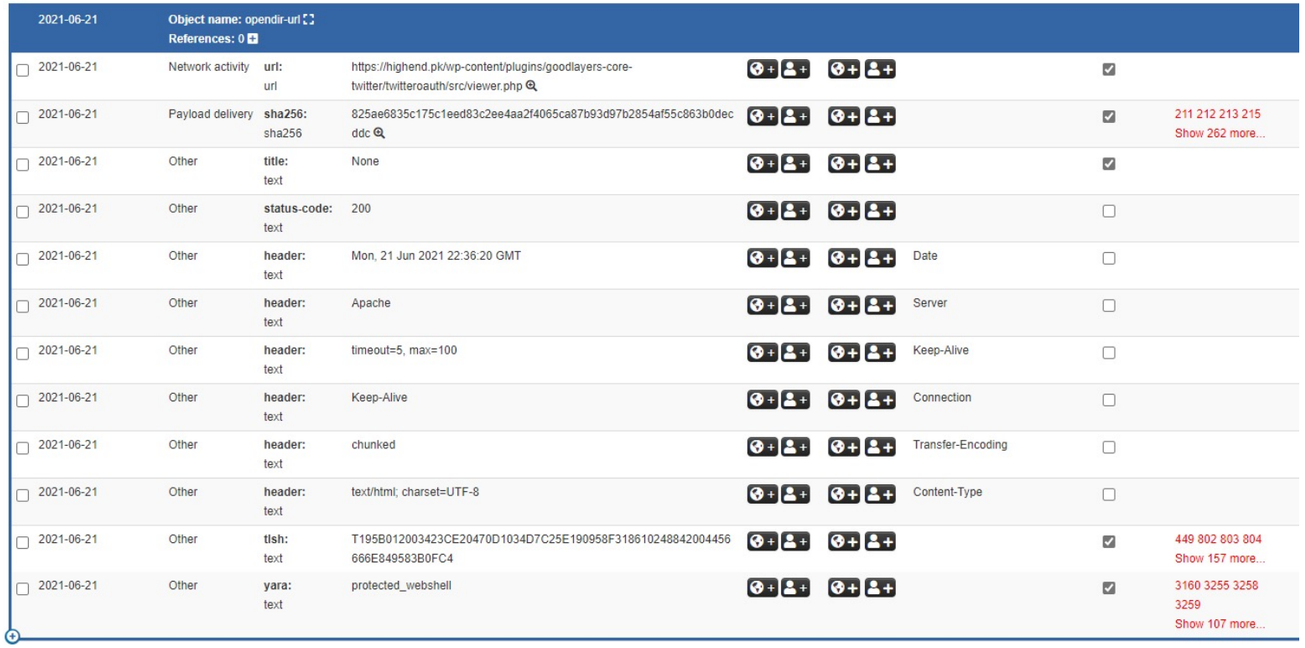
```
misp:
  misp_url: https://localhost
  misp_api_key: API_KEY_GOES_HERE
  urlhaus_api: https://urlhaus.abuse.ch/downloads/csv_recent/
  domain_event: 0
```

MISP Storage Module – Domain Events



- One event is created per domain
- Events are enriched with tags taken from URLhaus like the SQLite storage module
- One defined event is used to track already scanned domains

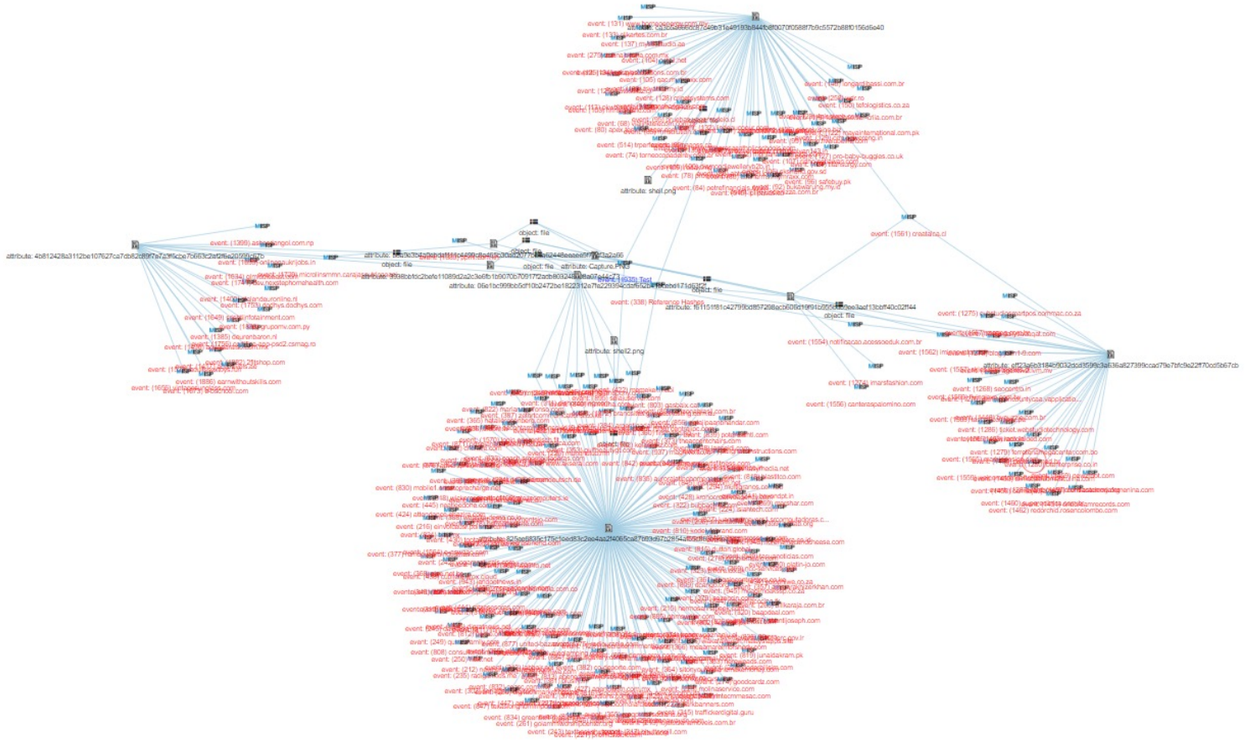
MISP Storage Module – opendir Object



2021-06-21	Object name: opendir-url References: 0						
2021-06-21	Network activity	**url:** url	https://highend.pk/wp-content/plugins/goodlayers-core-twitter/twitteroauth/src/viewer.php			☑	
2021-06-21	Payload delivery	**sha256:** sha256	825ae6835c175c1eed83c2ee4aa2f4065ca87b93d97b2854af55c863b0decddc			☑	211 212 213 215 Show 262 more...
2021-06-21	Other	**title:** text	None			☑	
2021-06-21	Other	**status-code:** text	200			☐	
2021-06-21	Other	**header:** text	Mon, 21 Jun 2021 22:36:20 GMT		Date	☐	
2021-06-21	Other	**header:** text	Apache		Server	☐	
2021-06-21	Other	**header:** text	timeout=5, max=100		Keep-Alive	☐	
2021-06-21	Other	**header:** text	Keep-Alive		Connection	☐	
2021-06-21	Other	**header:** text	chunked		Transfer-Encoding	☐	
2021-06-21	Other	**header:** text	text/html; charset=UTF-8		Content-Type	☐	
2021-06-21	Other	**tlsh:** text	T195B012003423CE20470D1034D7C25E190958F318610248842004456666E849583B0FC4			☑	449 802 803 804 Show 157 more...
2021-06-21	Other	**yara:** text	protected_webshell			☑	3160 3255 3258 3259 Show 107 more...

- Each scanned URL is combined into one MISP object with its attributes
- MISP displays all related events for each attribute
- A related event also contains such an attribute
- E.g., interesting for found webshell logins as displayed in image

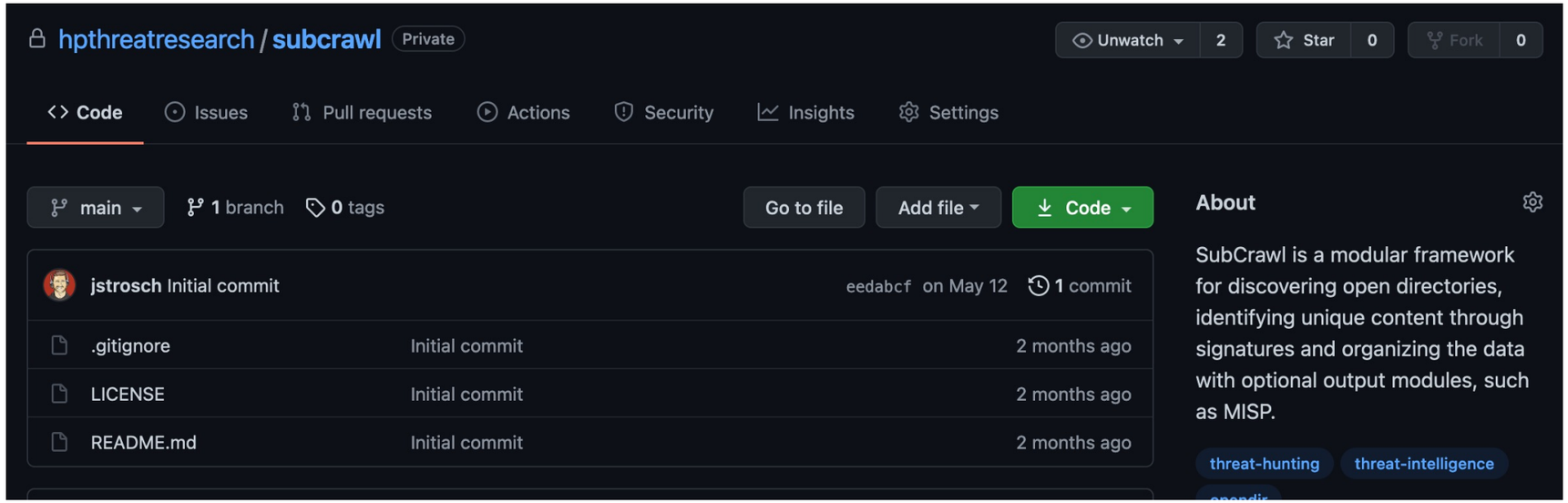
MISP Storage Module – Correlation Graph



- MISP offers the out of the box functionality to create correlation graphs
- With a large number of scanned domains, the clustering of used webservers, scripting technologies or found hashes can be interesting
- However, one limitation of MISP is that clustering based on similarity hashes is not possible
- Scripts for this can be found directly in our GitHub repository

SubCrawl is Now Available!

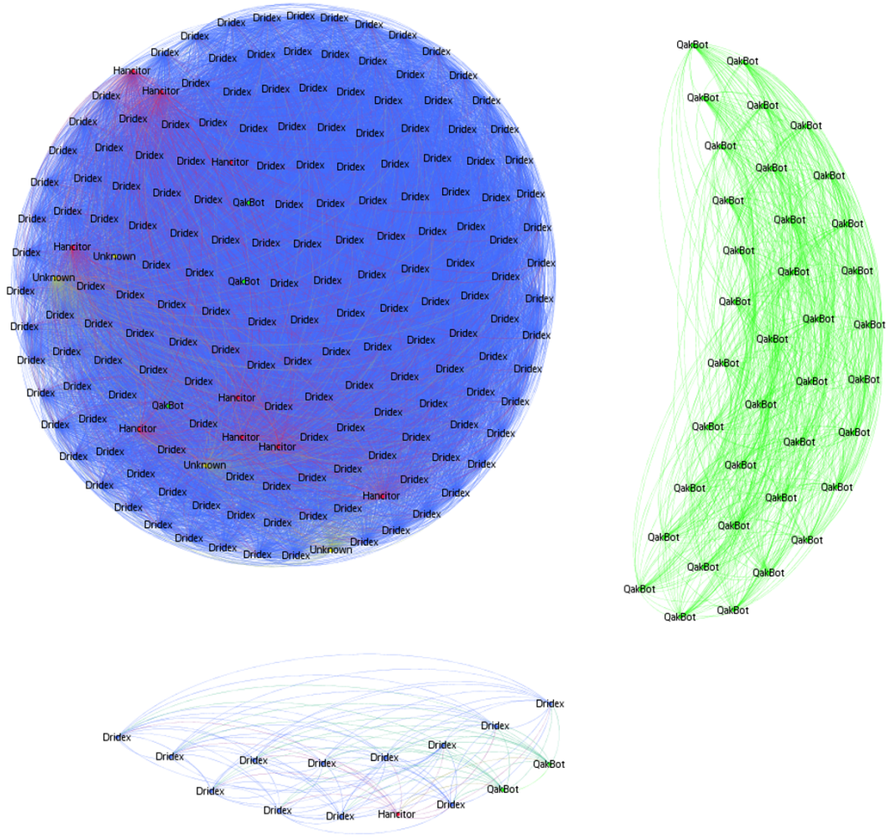
- Github - https://github.com/hpthreatresearch/subcrawl
- @jstrosch @stoerchl @cryptogramfan



Upcoming Events

- Virus Bulletin Localhost – Oct 2021
- https://vblocalhost.com/presentations/introducing-subcrawl-a-framework-for-the-analysis-and-clustering-of-hacking-tools-found-using-open-directories/

- Focus on data that we discovered, patterns in threat actor activity and how we were able to track tools such as web shells



Prevalent Dridex webshells on compromised hosts

Thank You!!