AIL Framework for Analysis of Information Leaks

workshop - A generic analysis information leak open source software

CIRCL
Computer Incident
Response Center
Luxembourg

Alexandre Dulaunoy
alexandre.dulaunoy@circl.lu
Sami Mokaddem
sami.mokaddem@circl.lu
Aurélien Thirion
aurelien.thirion@circl.lu

info@circl.lu

January 11, 2019

Objectives of the workshop

Our objectives of the workshop

- Demonstrate why data-analysis is critical in information security
- Explain challenges and the design of the AIL framework
- Learn how to install and start AIL
- Learn how to properly feed AIL with custom data
- Learn how to manage current modules
- Learn how to create new modules
- Practical part: Workshop

Sources of leaks

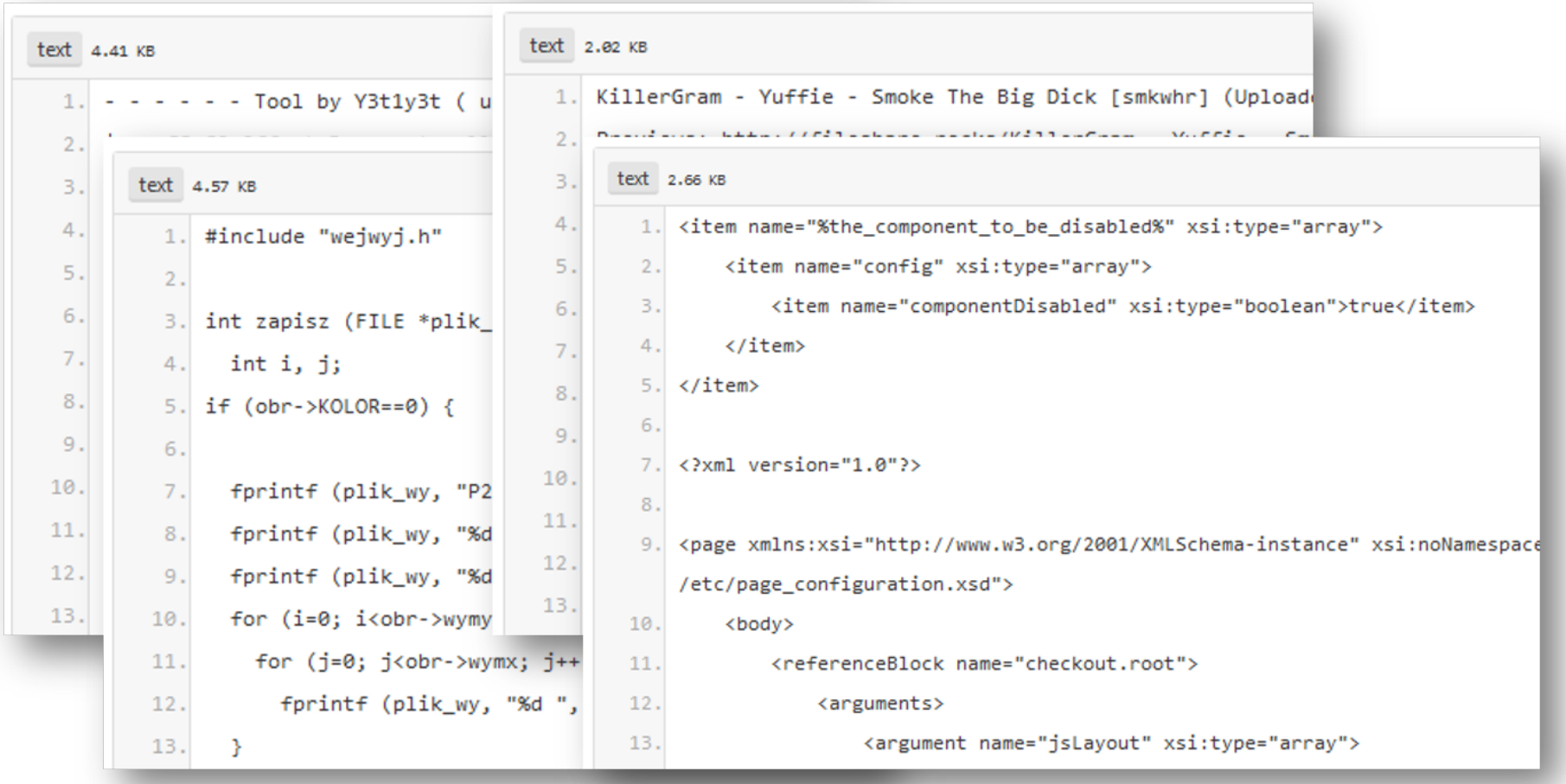
Sources of leaks: Paste monitoring

- Example: `http://pastebin.com/`
  - Easily storing and sharing text online
  - Used by programmers and legitimate users
    $\rightarrow$ Source code & information about configurations

Sources of leaks: Paste monitoring

- Example: `http://pastebin.com/`
  - Easily storing and sharing text online
  - Used by programmers and legitimate users
    $\rightarrow$ Source code & information about configurations
- Abused by attackers to store:
  - List of vulnerable/compromised sites
  - Software vulnerabilities (e.g. exploits)
  - Database dumps
    $\rightarrow$ User data
    $\rightarrow$ Credentials
    $\rightarrow$ Credit card details
  - More and more ...

Examples of pastes



text 4.41 KB

```
- - - - - - Tool by Y3t1y3t ( u
```

text 2.02 KB

```
KillerGram - Yuffie - Smoke The Big Dick [smkwhr] (Upload
```

text 4.57 KB

```
#include "wejwyj.h"

int zapisz (FILE *plik_
  int i, j;
if (obr->KOLOR==0) {

  fprintf (plik_wy, "P2
  fprintf (plik_wy, "%d
  fprintf (plik_wy, "%d
  for (i=0; i<obr->wymy
    for (j=0; j<obr->wymx; j++
      fprintf (plik_wy, "%d ",
  }
```

text 2.66 KB

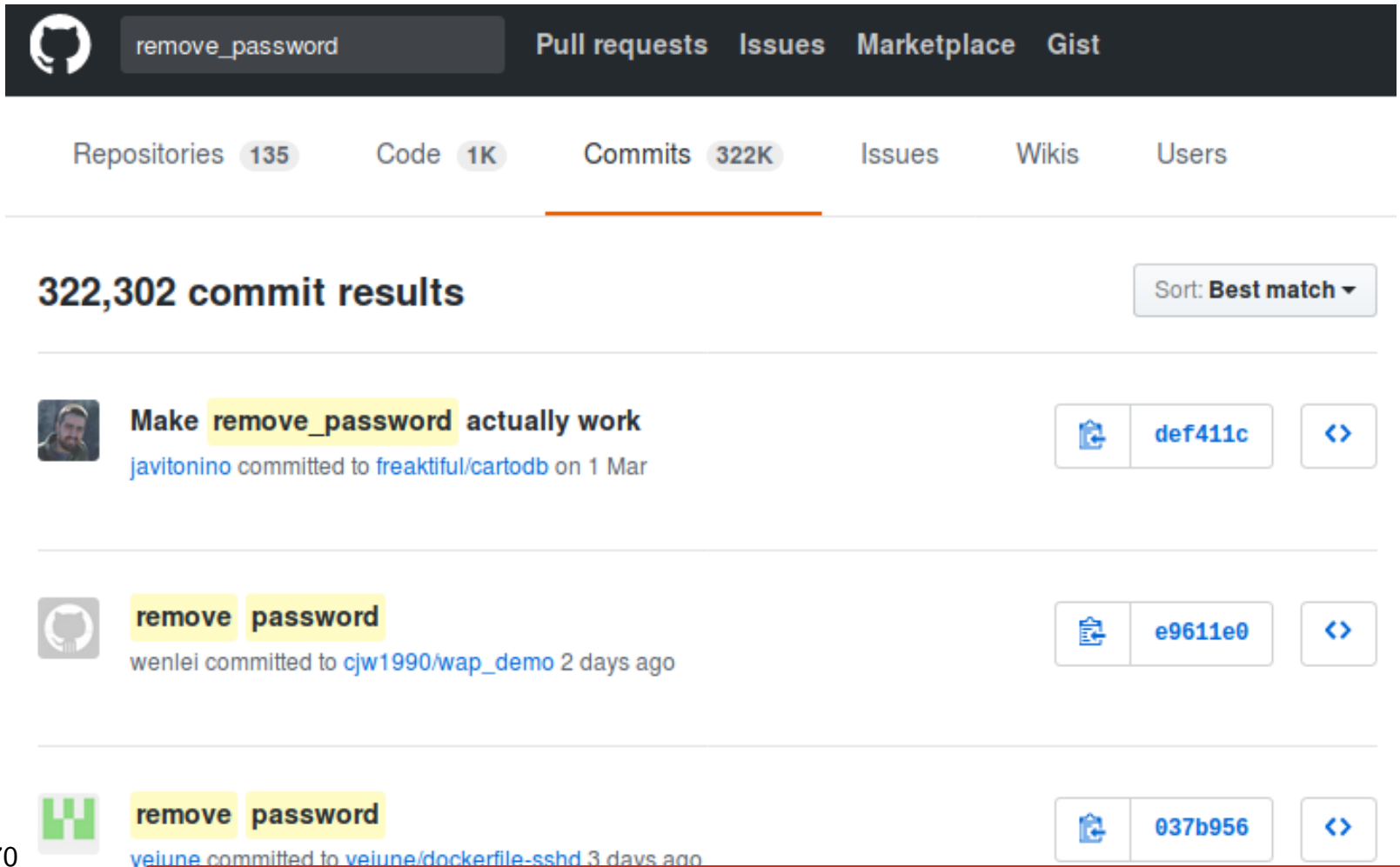
```
<item name="%the_component_to_be_disabled%" xsi:type="array">
    <item name="config" xsi:type="array">
        <item name="componentDisabled" xsi:type="boolean">true</item>
    </item>
</item>

<?xml version="1.0"?>

<page xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
/etc/page_configuration.xsd">
    <body>
        <referenceBlock name="checkout.root">
            <arguments>
                <argument name="jsLayout" xsi:type="array">
```

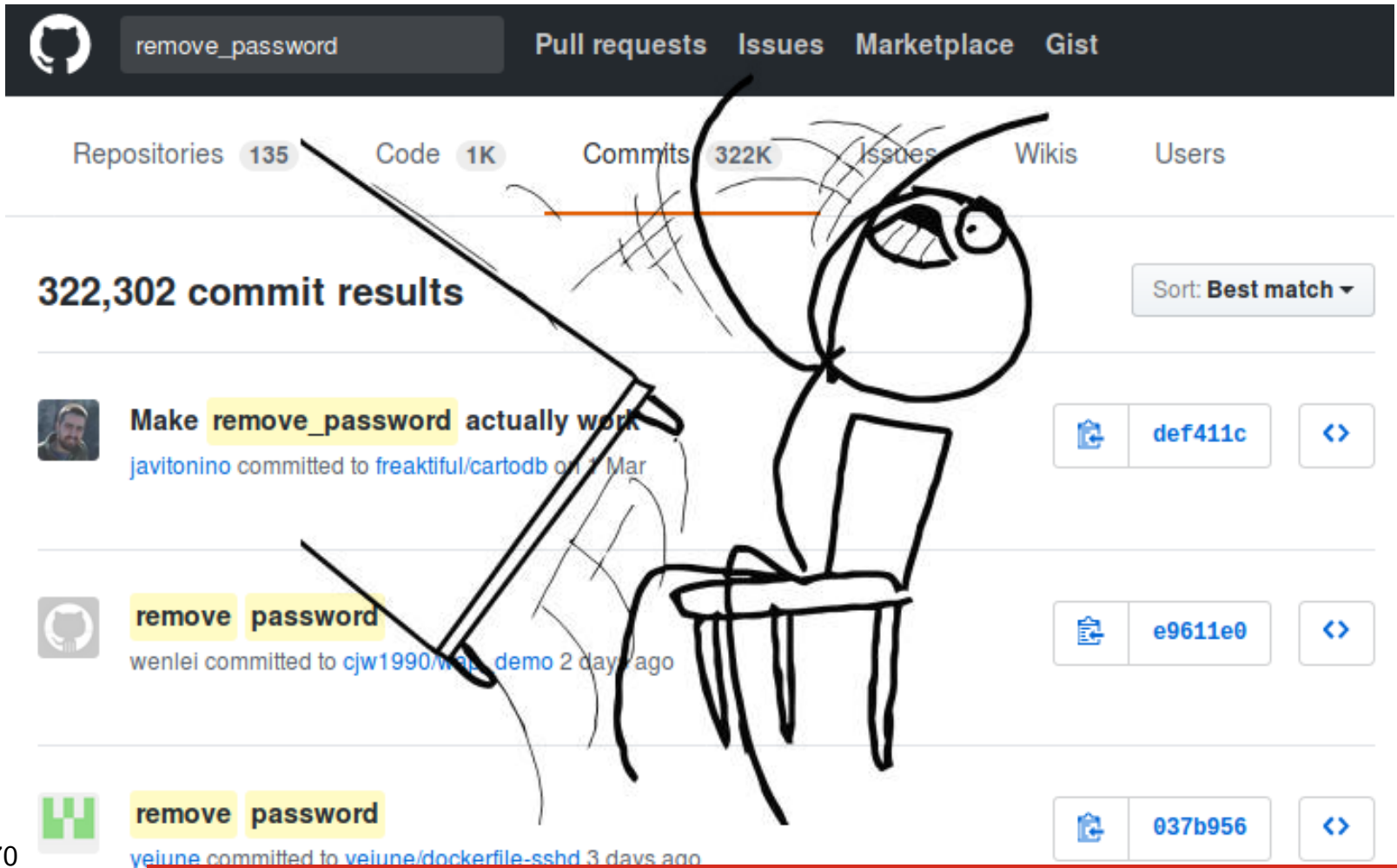
Sources of leaks: Others

- Mistakes from users
  - https://github.com/search?q=remove_password&type=Commits&ref=searchresults



Sources of leaks: Others

- Mistakes from users
  - https://github.com/search?q=remove_password&type=Commits&ref=searchresults



Why so many leaks?

- Economical interests (e.g. Adversaries promoting services)
- Political motives (e.g. Adversaries showing off)
- Collaboration (e.g. Criminals need to collaborate)
- Operational infrastructure (e.g. malware exfiltrating information on a pastie website)
- Mistakes and Errors

Are leaks frequent?

Yes!

and we have to deal with this as a CSIRT.

- **Contacting companies or organisations** who did specific accidental leaks
- **Discussing with media** about specific case of leaks and how to make it more practical/factual for everyone
- Evaluating the economical market for cyber criminals (e.g. DDoS booters[1] or reselling personal information - reality versus media coverage)
- Analysing collateral effects of malware, software vulnerabilities or exfiltration

→ And it's important to detect them automatically.

[1] https://github.com/D4-project/

Paste monitoring at CIRCL: Statistics

- Monitored paste sites: 27
  - *pastebin.com*
  - *ideone.com*
  - ...

	2016	2017	08.2018
Collected pastes	18,565,124	19,145,300	11,591,987
Incidents	244	266	208

Table: Pastes collected and incident[2] raised by CIRCL

[2] `http://www.circl.lu/pub/tr-46`

Privacy, AIL and GDPR

- Many modules in AIL can process personal data and even special categories of data as defined in GDPR (Art. 9).
- The data controller is often the operator of the AIL framework (limited to the organisation) and has to define **legal grounds for processing personal data**.
- To help users of AIL framework, a document is available which describe points of AIL in regards to the regulation[3].

[3] `https://www.circl.lu/assets/files/information-leaks-analysis-and-gdpr.pdf`

Potential legal grounds

- **Consent of the data subject** is in many cases not feasible in practice and often impossible or illogical to obtain (Art. 6(1)(a)).
- Legal obligation (Art. 6(1)(c)) - This legal ground applies mostly to CSIRTs, in accordance with the powers and responsibilities set out in CSIRTs mandate and with their constituency, as they may have the legal obligation to collect, analyse and share information leaks without having a prior consent of the data subject.
- Art. 6(1)(f) - Legitimate interest - Recital 49 explicitly refers to CSIRTs' right to process personal data provided that they have a legitimate interest but not colliding with fundamental rights and freedoms of data subject.

AIL Framework

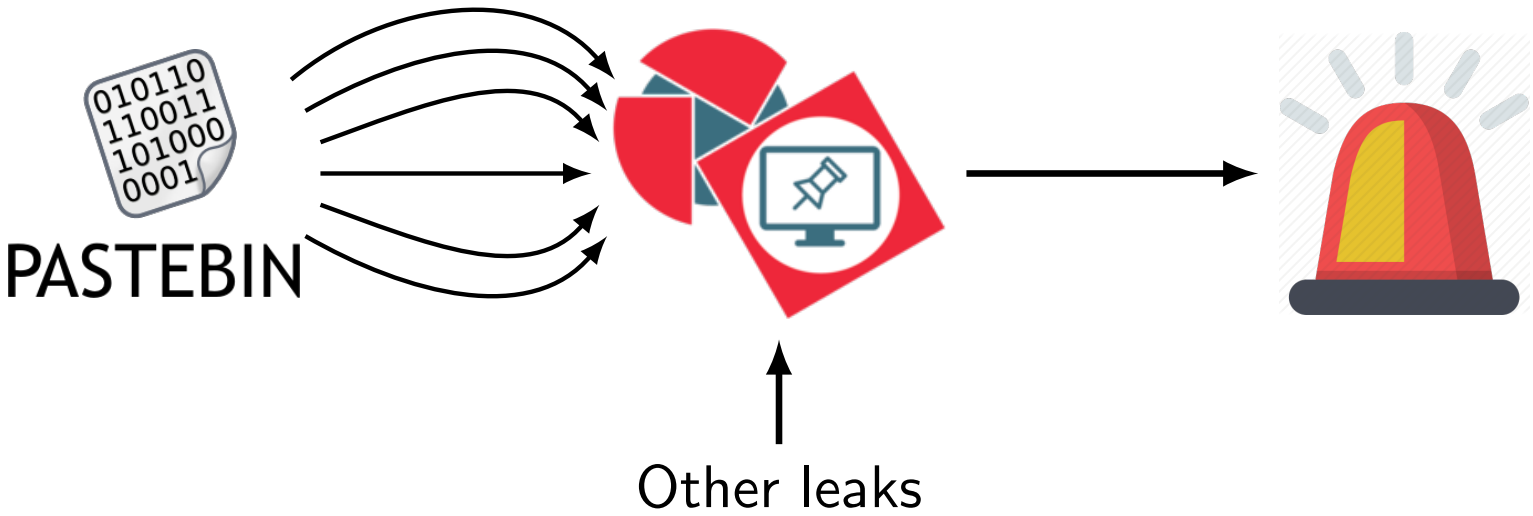
From a requirement to a solution: AIL Framework

History:

- AIL initially started as an **internship project** (2014) to evaluate the feasibility to automate the analysis of (un)structured information to find leaks.
- In 2018, AIL framework is an **open source software** in Python. The software is actively used (and maintained) by CIRCL.

AIL Framework: A framework for Analysis of Information Leaks

"AIL is a modular framework to analyse potential information leaks from unstructured data sources like pastes from Pastebin."



AIL Framework: Current capabilities

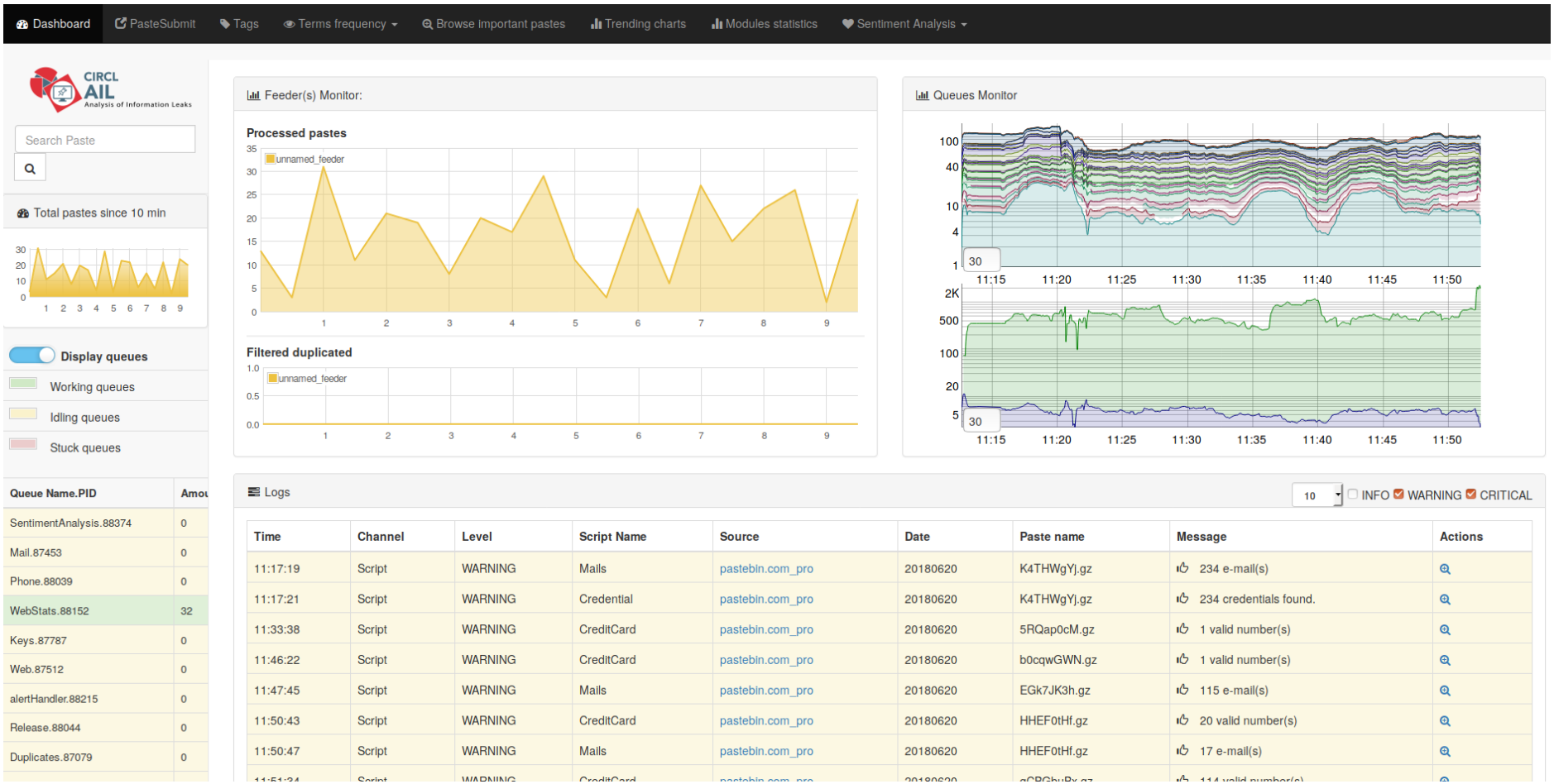
- Extending AIL to add a new **analysis module** can be done in 50 lines of Python
- The framework **supports multi-processors/cores by default**. Any analysis module can be started multiple times to support faster processing during peak times or bulk import
- **Multiple** concurrent **data input**

AIL Framework: Current features

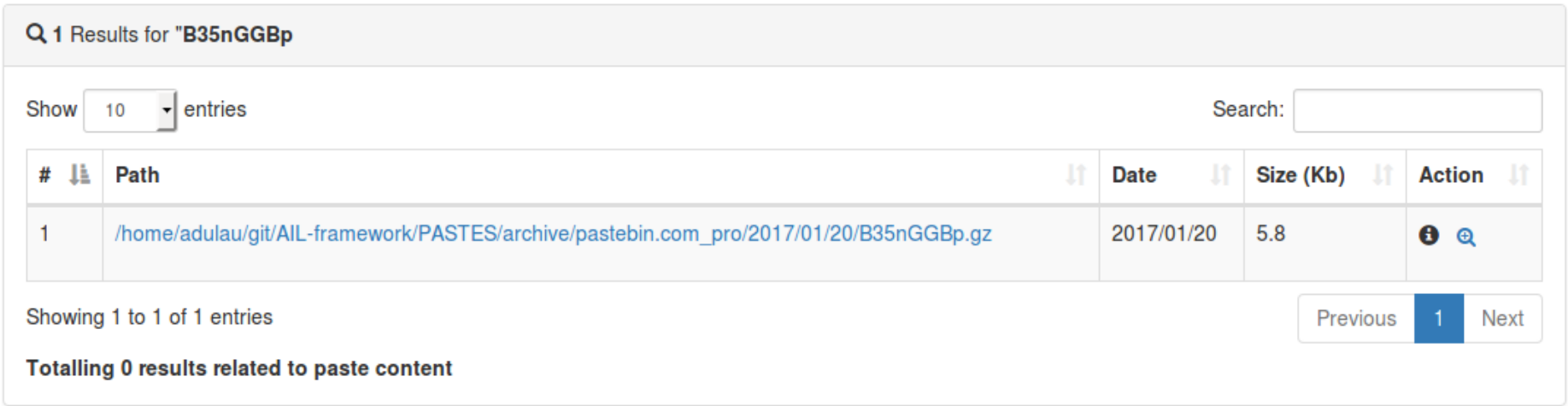
- Extracting **credit cards numbers, credentials, phone numbers, ...**
- Extracting and validating potential **hostnames**
- Keeps track of **duplicates**
- Submission to threat sharing and incident response platform (**MISP** and **TheHive**)
- **Full-text indexer** to index unstructured information
- **Tagging** for classification and searches
- Terms, sets and regex **tracking and occurences**
- Archives, files and raw **submission** from the UI
- **Sentiment/Mood analyser** for incoming data
- And many more

Live demo!

Example: Following a notification (0) - Dashboard



Example: Following a notification (1) - Searching



1 Results for "B35nGGBp

Show 10 entries

Search:

#	Path	Date	Size (Kb)	Action
1	/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2017/01/20/B35nGGBp.gz	2017/01/20	5.8	

Showing 1 to 1 of 1 entries

Previous 1 Next

Totalling 0 results related to paste content

Example: Following a notification (2) - Metadata

Date	Source	Encoding	Language	Size (Kb)	Mime	Number of lines	Max line length
20/01/2017	pastebin.com_pro	text/plain	('en', 1.0)	5.8	text/plain	510	336

Duplicate list:

Show 10 entries Search:

Hash type	Paste info	Date	Path
tlsh	Similarity: 93%	2017-01-12	/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2017/01/12/WeizLQUx.gz
tlsh	Similarity: 93%	2017-01-17	/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2017/01/17/Xqbx62vU.gz
tlsh	Similarity: 93%	2017-01-10	/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2017/01/10/iyfet4UM.gz
tlsh	Similarity: 92%	2017-01-14	/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2017/01/14/G7AB7q1m.gz
tlsh	Similarity: 92%	No date available	/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2016/12/31/CpDdkKbU.gz

Example: Following a notification (3) - Browsing content

Content:

```
http://members2.mofosnetwork.com/access/login/
somosextremos:buddy1990
brazzers_glenn:cocklick
brazzers61:braves01

http://members.naughtyamerica.com/index.php?m=login
gernblanston:3unc2352
Janhuss141200:310575
igetalliwant:1377zeph
pwilks89:mon22key
Bman1551:hockey

MoFos IKnowThatGirl PublicPickUps
http://members2.mofos.com
Chrismagg40884:loganm40
brando1:zzbrando1
aacoen:1q2w3e4r
1rstunkle23:my8self

BraZZers
http://ma.brazzers.com
gcjensen:gcj21pva
skycsc17:rbcdnd

                                ############################################################
                                    >| Get Daily Update Fresh Porn Password Here |<

                                          =>   http://www.erq.io/4mF1
```

Example: Following a notification (3) - Browsing content

Content:

```
Over 50000+ custom hacked xxx passwords by us! Thousands of free xxx passwords to the hottest paysites!

##########################################################
 >| Get Fresh New Premium XXX Site Password Here |<

  =>   http://www.erq.io/4mF1


##########################################################



http://ddfnetwork.com/home.html
eu172936:hCSBgKh
UecwB6zs:159X0$!r#6K78FuU

http://pornxn.stiffia.com/user/login
feldwWek8939:RObluJ8XtB
dabudka:17891789
brajits:brajits1

http://members.pornstarplatinum.com/sblogin/login.php/
gigiriveracom:xxxjay
jayx123:xxxjay69

http://members.vividceleb.com/
Rufio99:fairhaven
ScHiFRvi:102091
Chaos84:HOLE5244
Riptor705:blade7
Domi8
```

Setting up the framework

Setting up AIL-Framework from source or virtual machine

Setting up AIL-Framework from source

```
git clone https://github.com/CIRCL/AIL-framework.git
cd AIL-framework
./installing_deps.sh
cd var/www/
./update_thirdparty.sh
```

Using the virtual machine:

1. Download `https://www.circl.lu/assets/files/ail-training/AIL_v@4986352.ova`
2. Start virtualbox
3. File → import appliance → select AIL_June.ova
4. (for now) Prevent the automatic launch and `git pull` the changes

AIL ecosystem - Challenges and design

AIL ecosystem: Technologies used

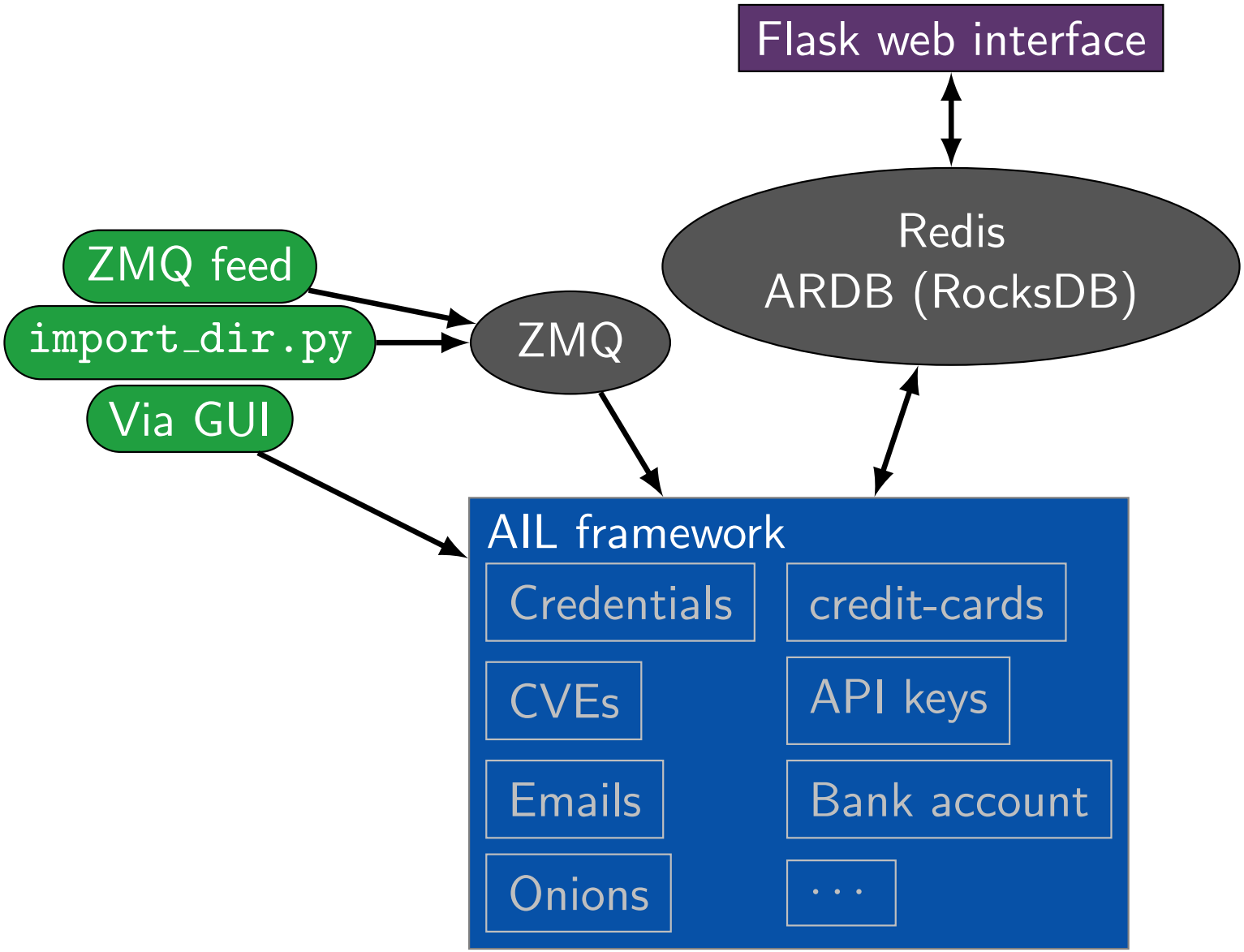
Programing language: Full python3

Databases: Redis and ARDB

Server: Flask

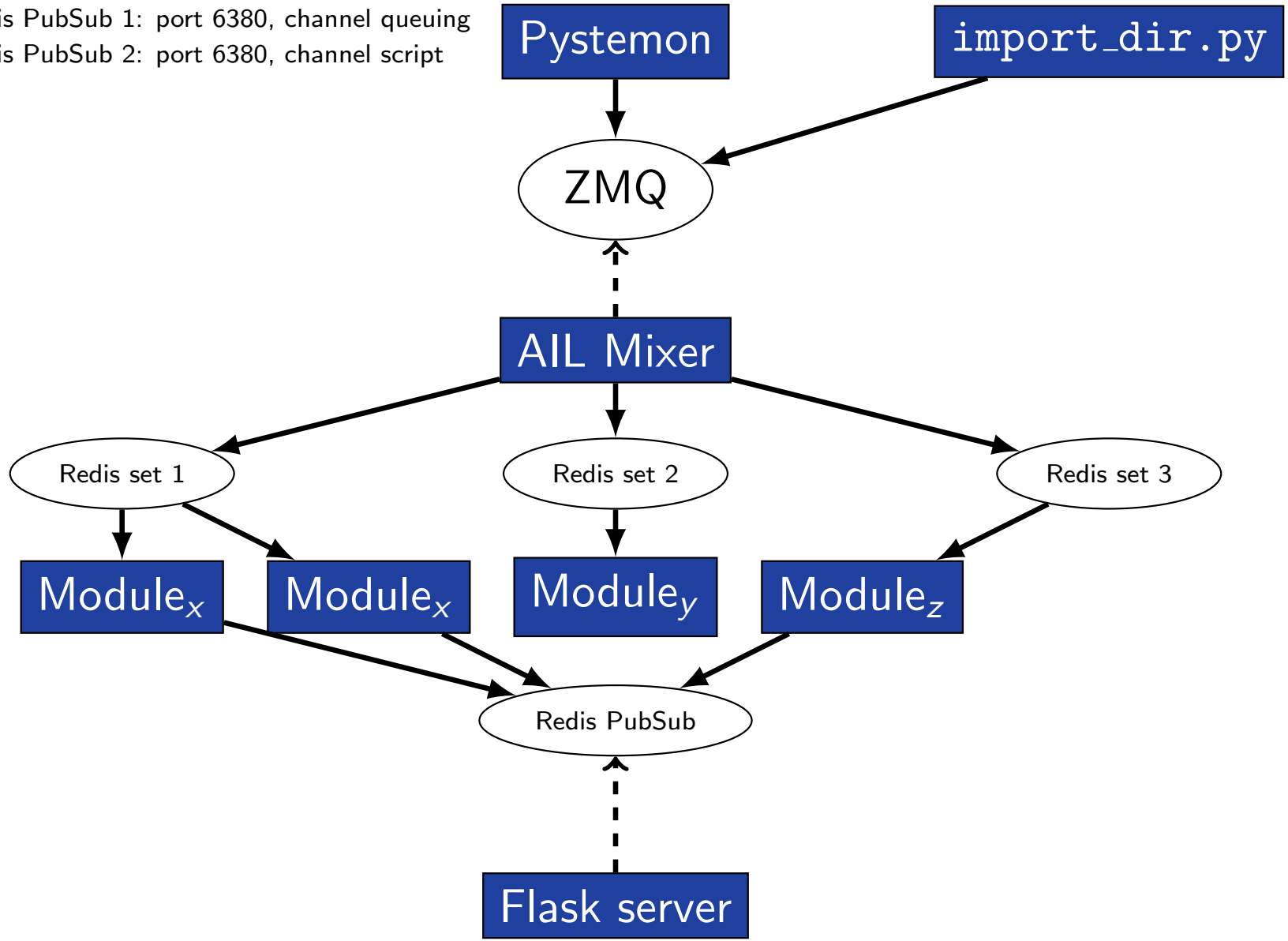
Data message passing: ZMQ, Redis list and Redis Publisher/Subscriber

AIL global architecture 1/2



AIL global architecture 2/2

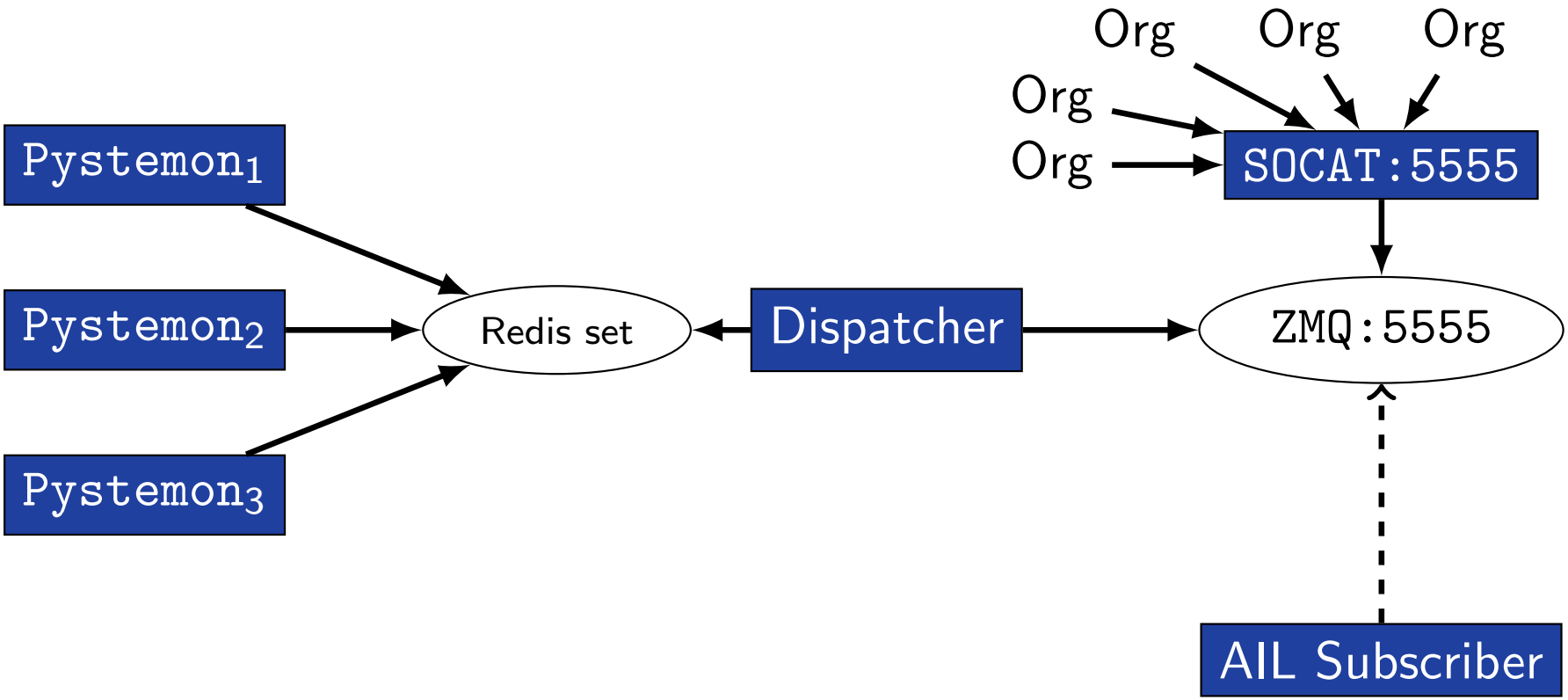
Redis PubSub 1: port 6380, channel queuing
Redis PubSub 2: port 6380, channel script



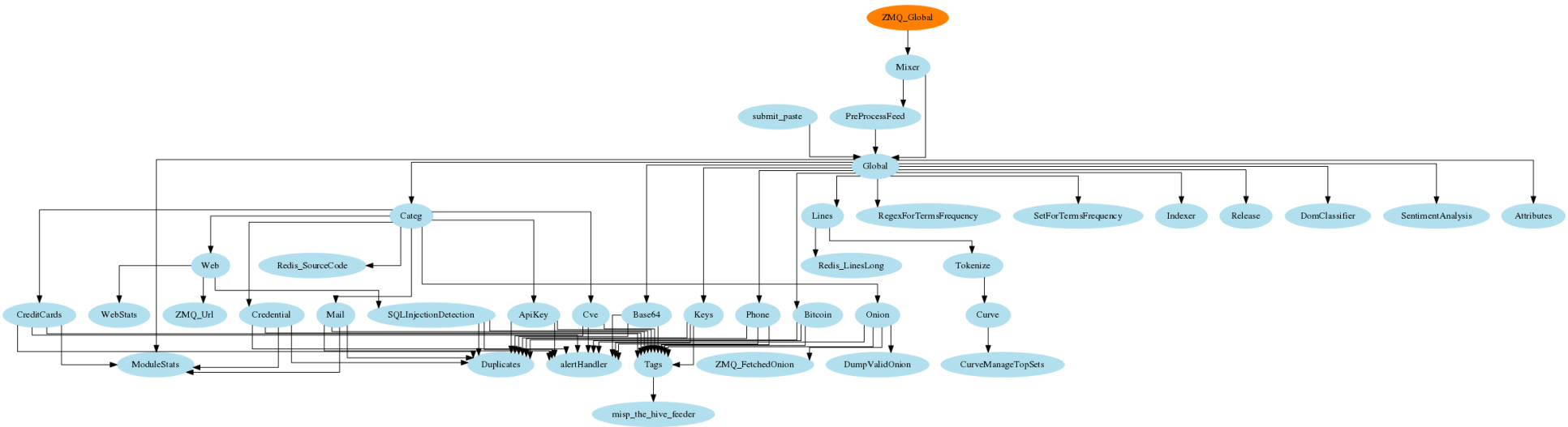
Data feeder: Gathering pastes with pystemon

Pystemon global architecture

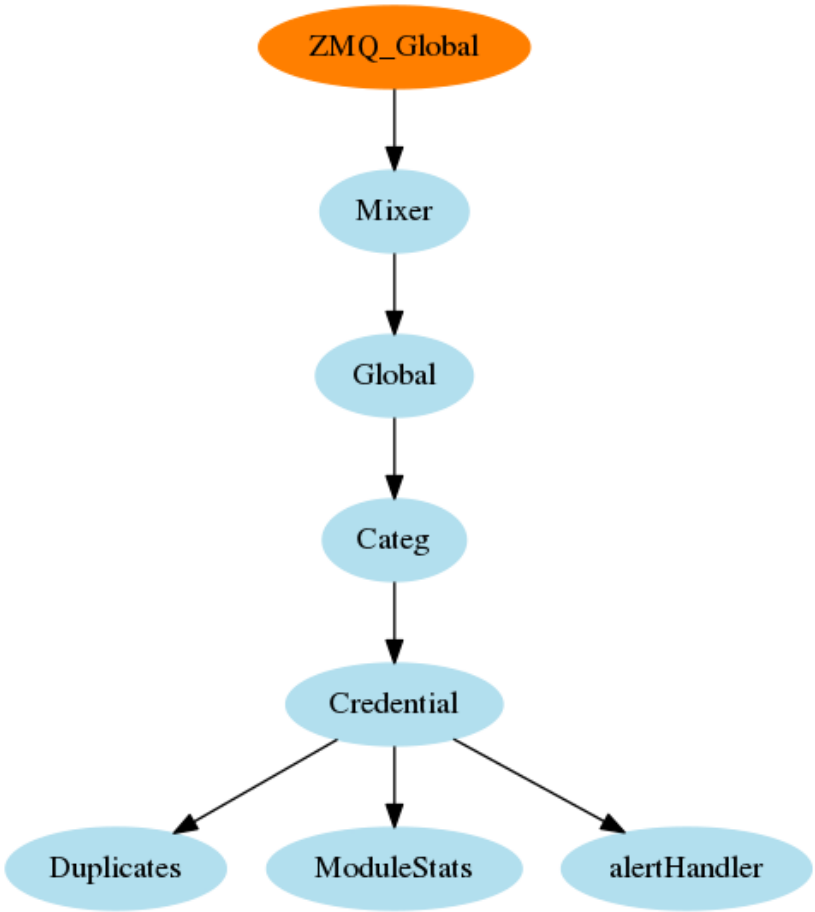
Redis PubSub 1: port 6380, channel queuing
Redis PubSub 2: port 6380, channel script



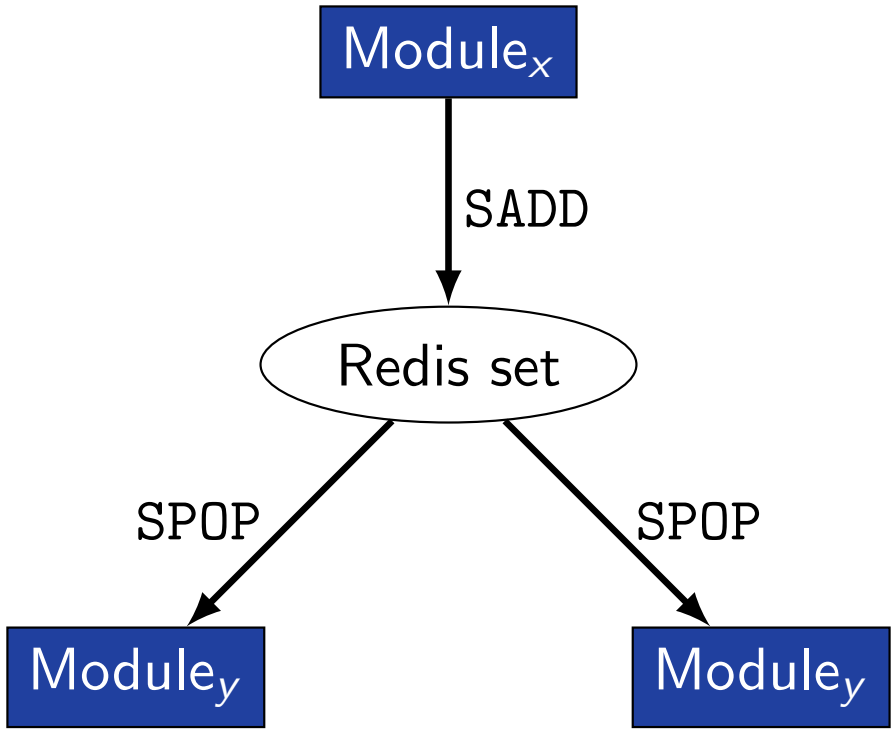
AIL global architecture: Data streaming between module



AIL global architecture: Data streaming between module (Credential example)



Message consuming



→ No message lost nor double processing

→ Multiprocessing!

Web crawler

- Web crawler is used to crawl regular website as well as .onion addresses
- Splash (scriptable browser) is rending the pages (including javascript) and produce screenshots (HAR archive too)

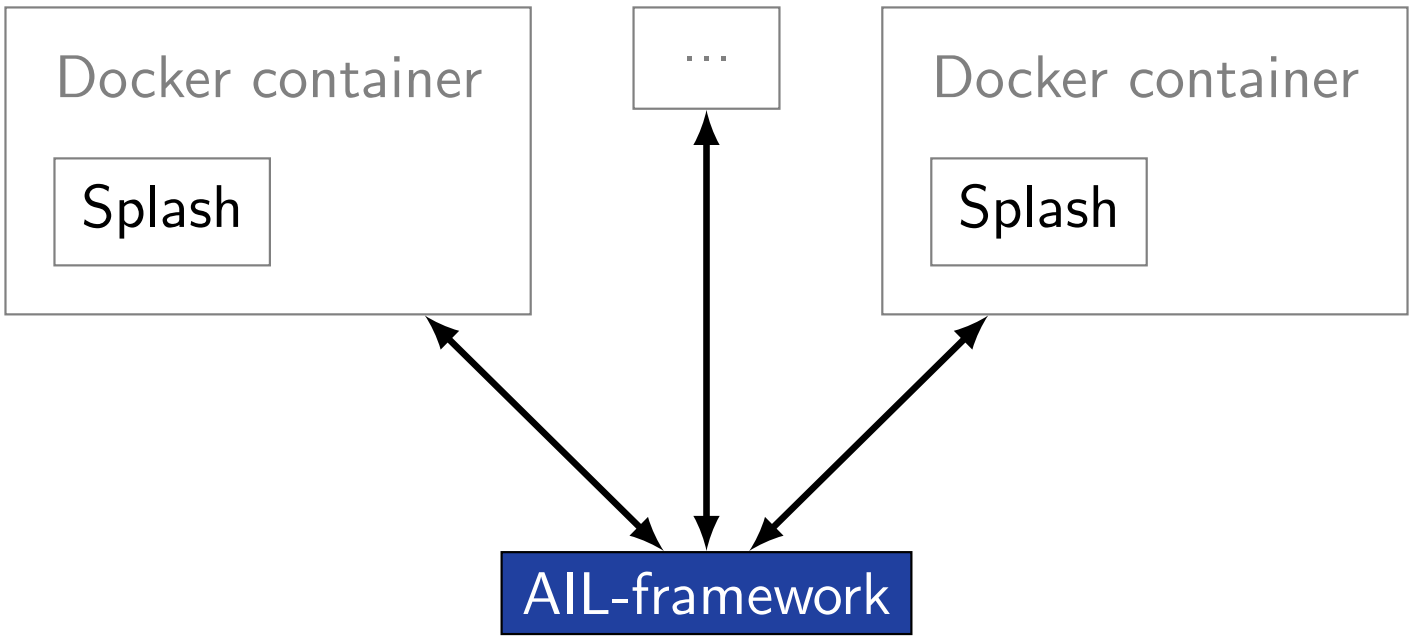


Figure: Architecture of AIL and its hidden services crawler

Starting the framework

Running your own instance from source

Make sure that ZMQ_Global→address = `tcp://crf.circl.lu:5556,tcp://127.0.0.1:5556` in bin/package/config.cfg

Accessing the environment and starting AIL

```
# Activate the virtualenv
. ./AILENV/bin/activate

# Launch the system
cd bin/
./LAUNCH -l

# Will also start the web interface
```

Running your own instance using the virtual machine

Login and passwords:

```
1 Web interface (default network settings):
2     http://192.168.56.51:7000/
3 Shell/SSH:
4     ail/Password1234
5
```

Feeding the framework

Feeding AIL

There are differents way to feed AIL with data:

1. Be a trusted partner with CIRCL and ask to get access to our feed

   info@circl.lu
2. Setup *pystemon* and use the custom feeder
   - *pystemon* will collect pastes for you
3. Feed your own data using the `import_dir.py` script
4. Feed your own file/text using the UI (`/PasteSubmit/`)

Feeding AIL

There are differents way to feed AIL with data:

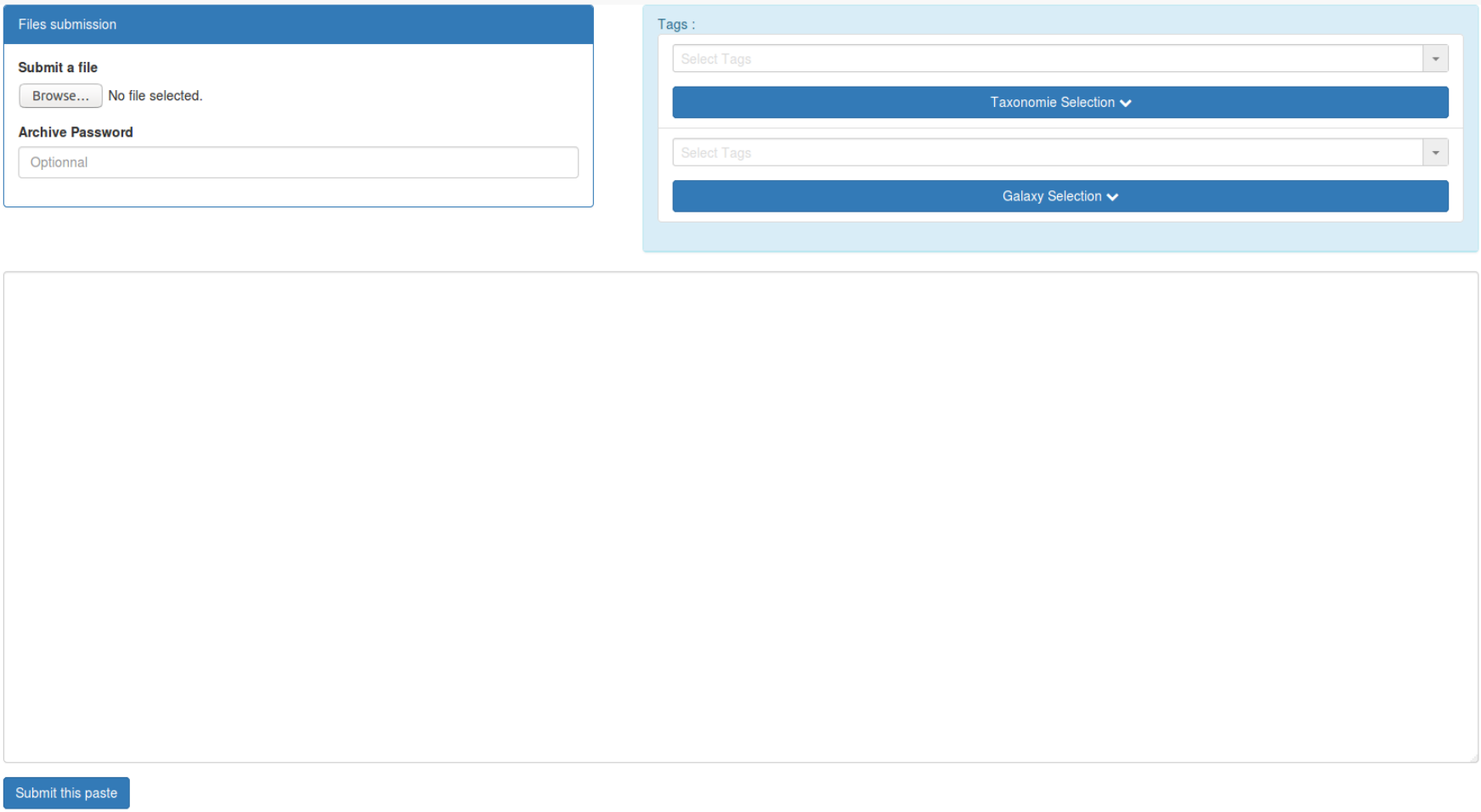
1. CIRCL trusted partners can ask to access our feed info@circl.lu
   - ▷ You already have access
2. ~~Setup *pystemon* and use the custom feeder~~
   - ~~*pystemon* will collect pastes for you~~
3. Feed your own file/text using the UI (`/PasteSubmit/`)
4. Feed your own data using the `import_dir.py` script

Plug-in AIL to the CIRCL feed

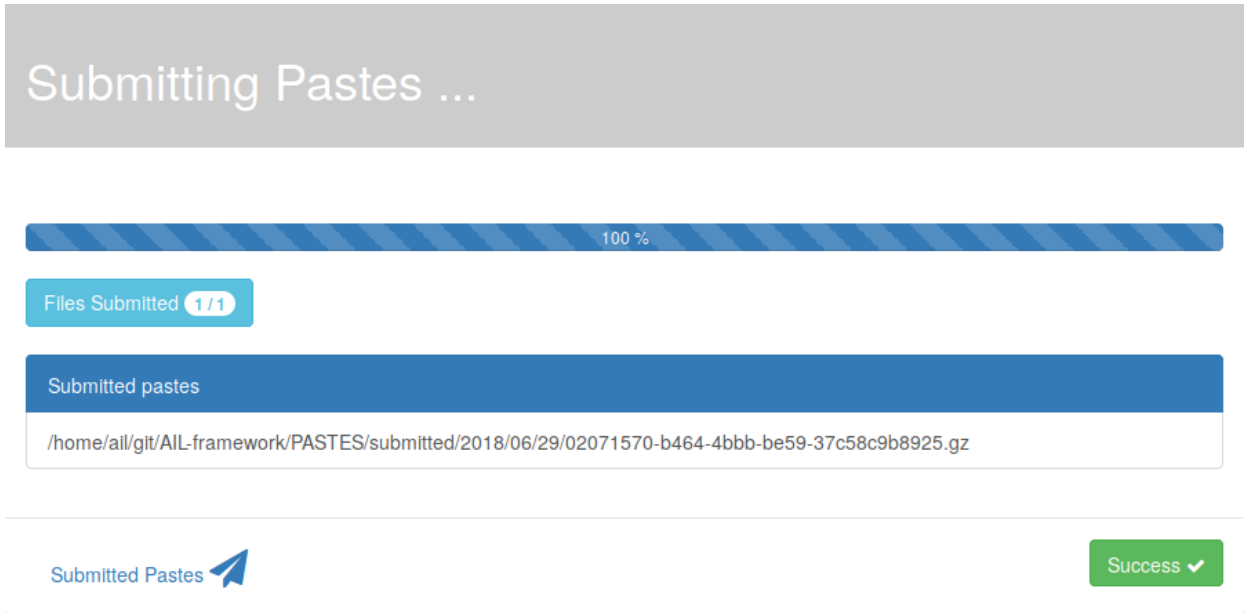
You can freely access the CIRCL feed during this workshop!

- In the file `bin/package/config.cfg`,
- Set `ZMQ_Global->address` to `tcp://crf.circl.lu:5556`

Via the UI (1)


Files submission
Submit a file
Browse...
No file selected.
Archive Password
Optionnal
Tags :
Select Tags
Taxonomie Selection
Select Tags
Galaxy Selection
Submit this paste

Via the UI (2)


Submitting Pastes ...
100 %
Files Submitted 1 / 1
Submitted pastes
/home/ail/git/AIL-framework/PASTES/submitted/2018/06/29/02071570-b464-4bbb-be59-37c58c9b8925.gz
Submitted Pastes
Success

Feeding AIL with your own data - `import_dir.py` (1)

/!\ 2 requirements:

1. Data to be fed must have the path hierarchy as the following:
   1.1 `year/month/day/(textfile/gzfile)`
   1.2 This is due to the inner representation of paste in AIL

2. Each file to be fed must be of a raisonable size:
   2.1 $\sim$ 3 `Mb` is already large
   2.2 This is because some modules are doing regex matching
   2.3 If you want to feed a large file, better split it in multiple ones

Feeding AIL with your own data - `import_dir.py` (2)

1. Check your local configuration `bin/package/config.cfg`
   - In the file `bin/package/config.cfg`,
   - Add `127.0.0.1:5556` in `ZMQ_Global`
   - (should already be set by default)

Feeding AIL with your own data - `import_dir.py` (2)

1. Check your local configuration `bin/package/config.cfg`
   - In the file `bin/package/config.cfg`,
   - Add `127.0.0.1:5556` in `ZMQ_Global`
   - (should already be set by default)
2. Launch `import_dir.py` with de directory you want to import
   - `import_dir.py -d dir_path`

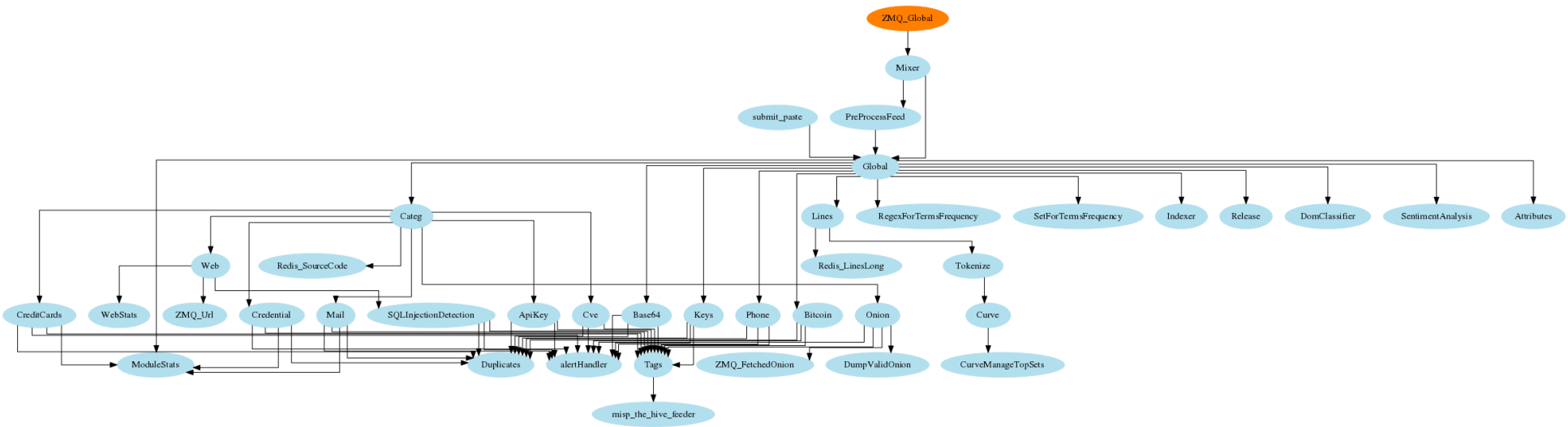
Feeding AIL with your own data - `import_dir.py` (2)

1. Check your local configuration `bin/package/config.cfg`
   - In the file `bin/package/config.cfg`,
   - Add `127.0.0.1:5556` in `ZMQ_Global`
   - (should already be set by default)
2. Launch `import_dir.py` with de directory you want to import
   - `import_dir.py -d dir_path`
3. Watch your data being feed to AIL

Creating new features

Developping new features: Plug-in a module in the system

Choose where to put your module in the data flow:



Then, modify `bin/package/modules.cfg` accordingly

Writing your own modules - /bin/template.py

```
1  import time
2  from pubsublogger import publisher
3  from Helper import Process
4  if __name__ == '__main__':
5      # Port of the redis instance used by pubsublogger
6      publisher.port = 6380
7      # Script is the default channel used for the modules.
8      publisher.channel = 'Script'
9      # Section name in bin/packages/modules.cfg
10     config_section = '<section name>'
11     # Setup the I/O queues
12     p = Process(config_section)
13     # Sent to the logging a description of the module
14     publisher.info("<description of the module>")
15     # Endless loop getting messages from the input queue
16     while True:
17         # Get one message from the input queue
18         message = p.get_from_set()
19         if message is None:
20             publisher.debug("{} queue is empty, waiting".format(config_section))
21             time.sleep(1)
22             continue
23         # Do something with the message from the queue
24         something_has_been_done = do_something(message)
25
```

AIL - Add your own web interface

1. Launch `var/www/create_new_web_module.py`
2. Enter the module's name
3. A template and flask skeleton has been created for your new webpage in `var/www/modules/`
4. You can start **coding** server-side in:

   `var/www/modules/`*`your_module_name`*`/Flask_`*`your_module_name.py`*
5. You can start **coding** client-side in:

   `var/www/modules/`*`your_module_name`*`/templates/`*`your_module_name.html`*

   `var/www/modules/`*`your_module_name`*`/templates/header_`*`your_module_name.html`*

Case study: Push alert to MISP

Push alert to MISP

Goal: push tags to MISP.

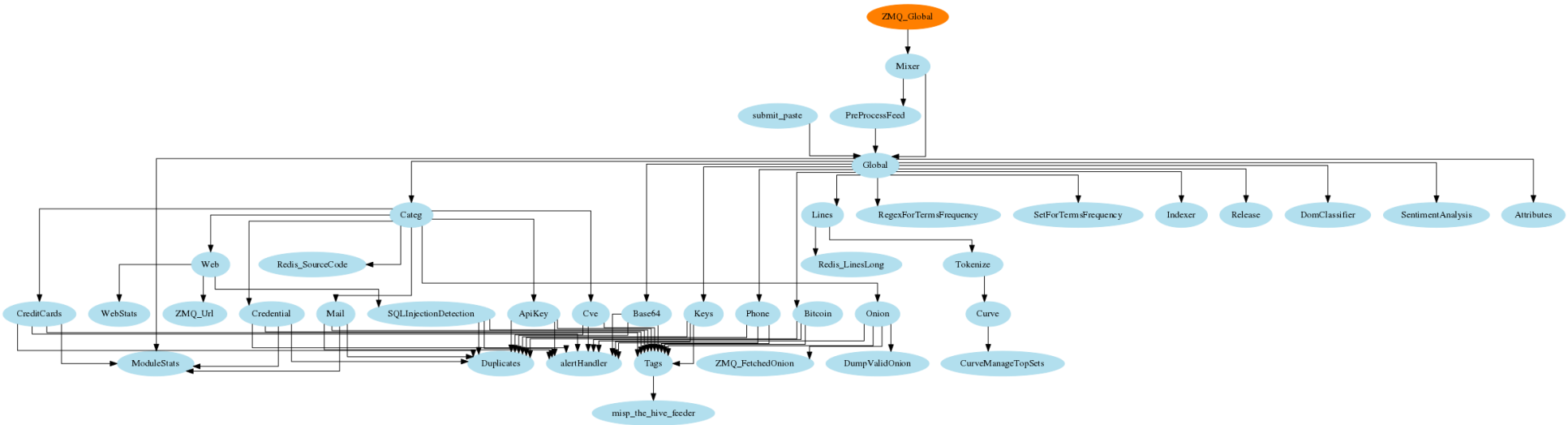
Push alert to MISP

1. Use infoleak taxonomy[4]
2. Add your own tags
3. Create an event on a paste

[4] https://www.misp-project.org/taxonomies.html

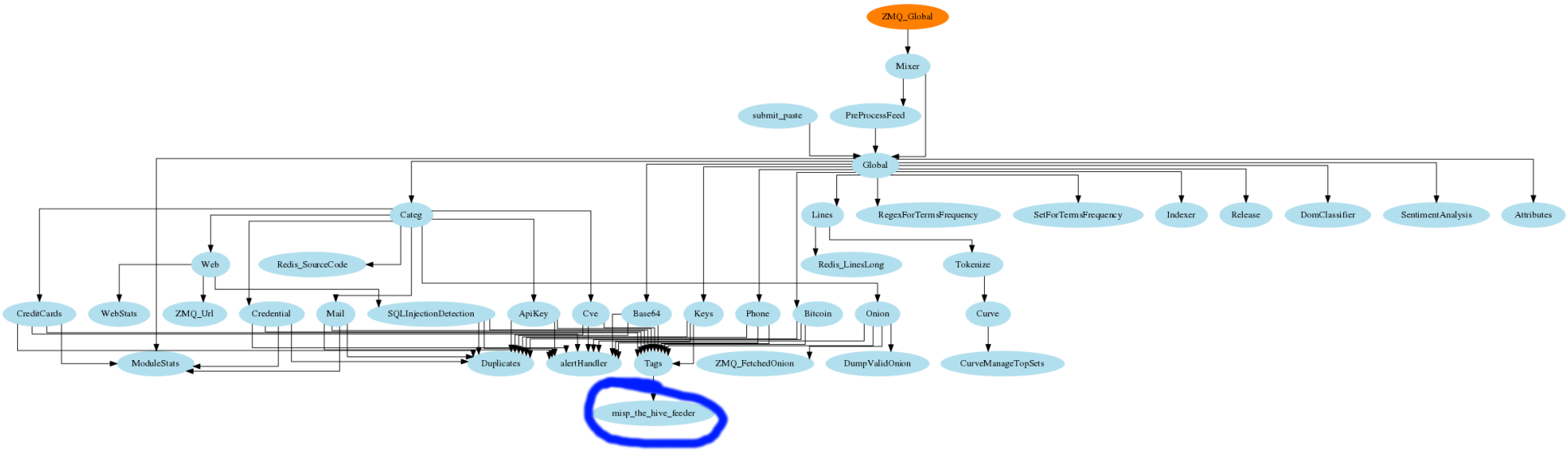
Case study: Finding the best place in the system

Best place to put it?



Case study: Finding the best place in the system

Best place to put it?

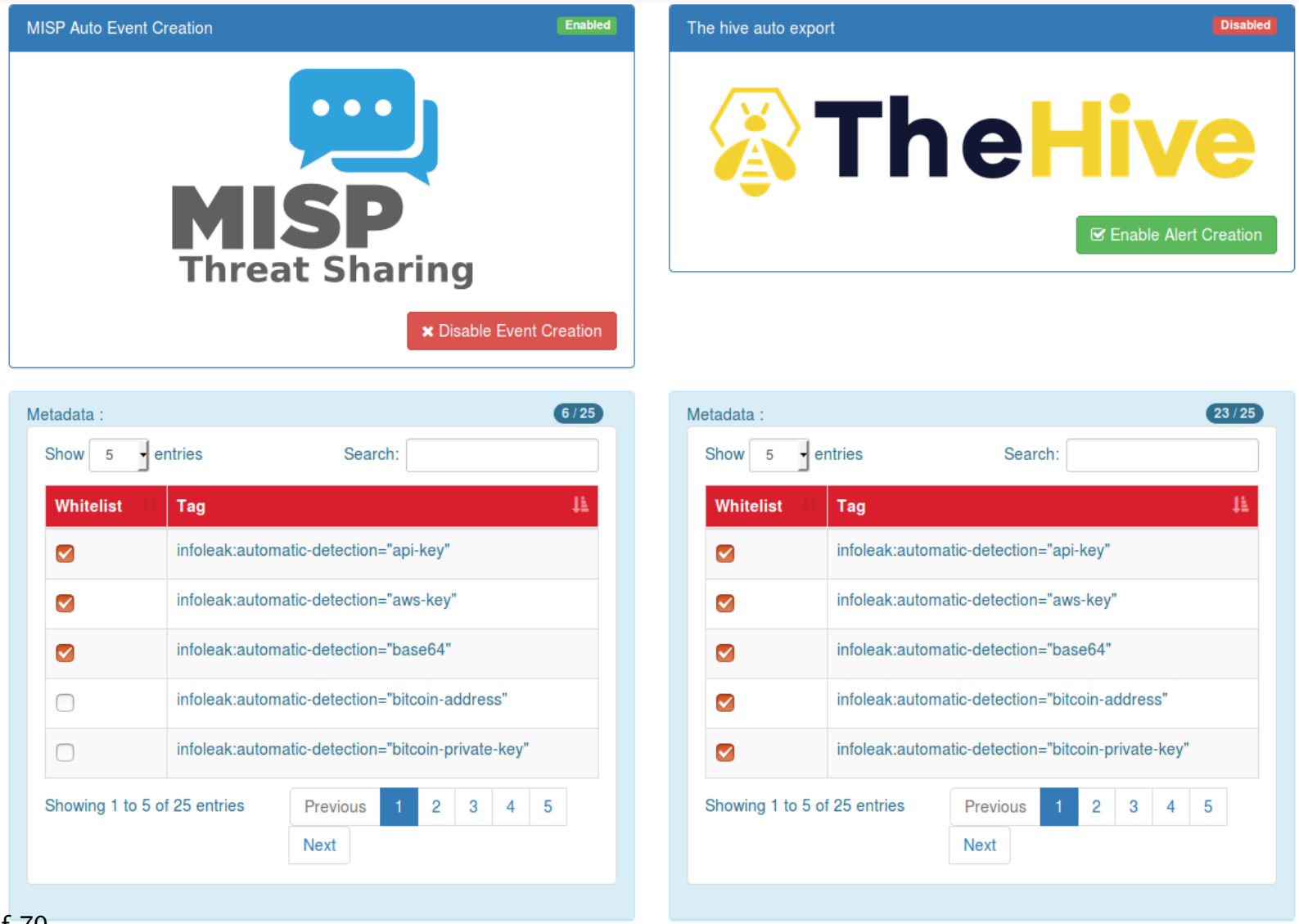


Case study: Updating `Flask server.py`

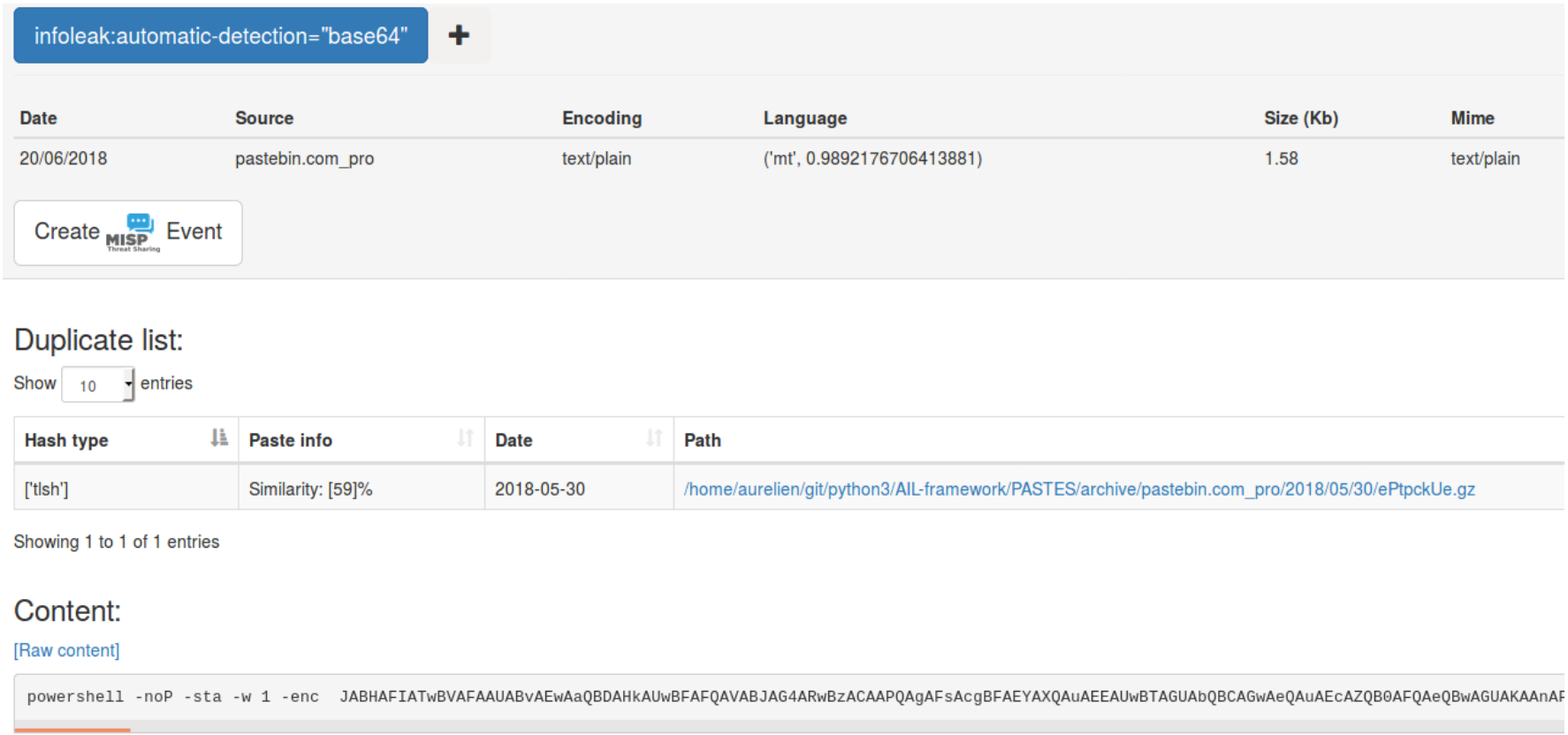
`Flask server.py`

```
1  [...]
2  # ========== INITIAL tags auto export ============
3  r_serv_db = redis.StrictRedis(
4      host=cfg.get("ARDB_DB", "host"),
5      port=cfg.getint("ARDB_DB", "port"),
6      db=cfg.getint("ARDB_DB", "db"),
7      decode_responses=True)
8  infoleak_tags = taxonomies.get('infoleak').machinetags()
9  infoleak_automatic_tags = []
10 for tag in taxonomies.get('infoleak').machinetags():
11     if tag.split('=')[0][:] == 'infoleak:automatic-detection':
12         r_serv_db.sadd('list_export_tags', tag)
13
14 r_serv_db.sadd('list_export_tags', 'infoleak:submission="manual"')
15 r_serv_db.sadd('list_export_tags', '<your_tag>')
16
```

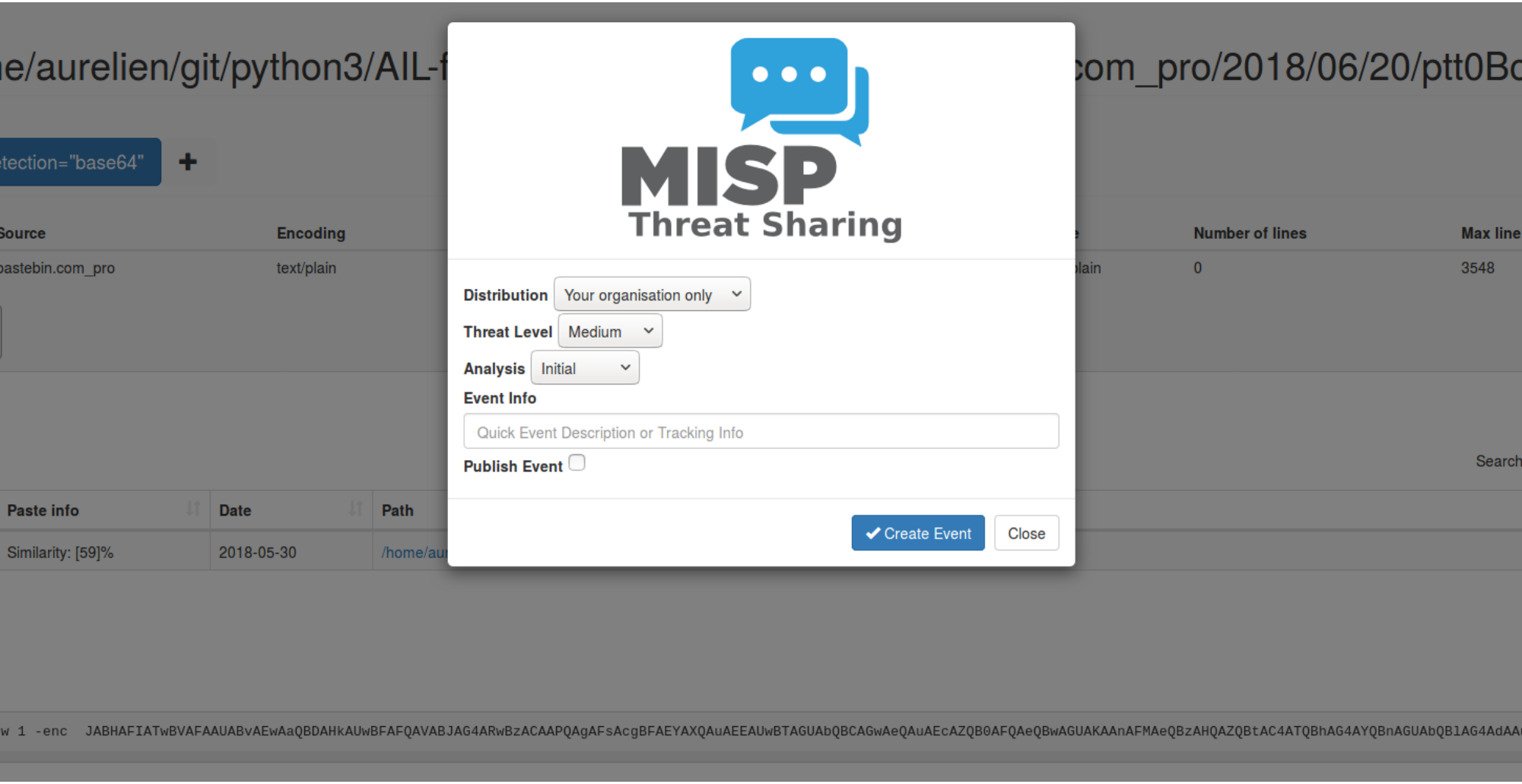
Auto Push Tags



Create an event



Create an event



Practical part

Practical part: Pick your choice

1. Update support of docker/ansible
2. Graph database on `Credential.py`
   - Top used passwords, most compromised user, ...
3. Webpage scrapper
   - Download html from URL found in pastes
   - Re-inject html as paste in AIL
4. Improvement of `Phone.py`
   - Way to much false positive as of now. Exploring new ways to validate phone numbers could be interesting
5. **Your custom feature**

Contribution rules

How to contribute

Glimpse of contributed features

- Docker
- Ansible
- Email alerting
- SQL injection detection
- Phone number detection

How to contribute

- Feel free to fork the code, play with it, make some patches or add additional analysis modules.

How to contribute

- Feel free to fork the code, play with it, make some patches or add additional analysis modules.
- Feel free to make a pull request for your contribution

How to contribute

- Feel free to fork the code, play with it, make some patches or add additional analysis modules.
- Feel free to make a pull request for your contribution
- That's it!

Final words

- Building AIL helped us to find additional leaks which cannot be found using manual analysis and **improve the time to detect duplicate/recycled leaks**.

  $\rightarrow$ Therefore quicker response time to assist and/or inform proactively affected constituents.

Annexes

Managing the framework

Managing AIL: Old fashion way

Access the script screen

```
1 screen -r Script
```

Table: GNU screen shortcuts

Shortcut	Action
C-a d	detach screen
C-a c	Create new window
C-a n	next window screen
C-a p	previous window screen

Managing your modules: Using the helper