

# 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](mailto:alexandre.dulaunoy@circl.lu)

Sami Mokaddem

[sami.mokaddem@circl.lu](mailto:sami.mokaddem@circl.lu)

Aurélien Thirion

[aurelien.thirion@circl.lu](mailto:aurelien.thirion@circl.lu)

[info@circl.lu](mailto:info@circl.lu)

December 20, 2018

## 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
    - 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
    - Source code & information about configurations
- Abused by attackers to store:
  - List of vulnerable/compromised sites
  - Software vulnerabilities (e.g. exploits)
  - Database dumps
    - User data
    - Credentials
    - Credit card details
  - More and more ...

# Examples of pastes

---

The image displays three overlapping screenshots of code pastes from a web interface. Each paste is shown in a light gray box with a title and size, and a list of line numbers on the left side.

**Top-left paste:** Title "text 4.41 KB". Line 1: "1. - - - - - Tool by Y3t1y3t ( u".

**Top-right paste:** Title "text 2.02 KB". Line 1: "1. KillerGram - Yuffie - Smoke The Big Dick [smkwhr] (Upload".

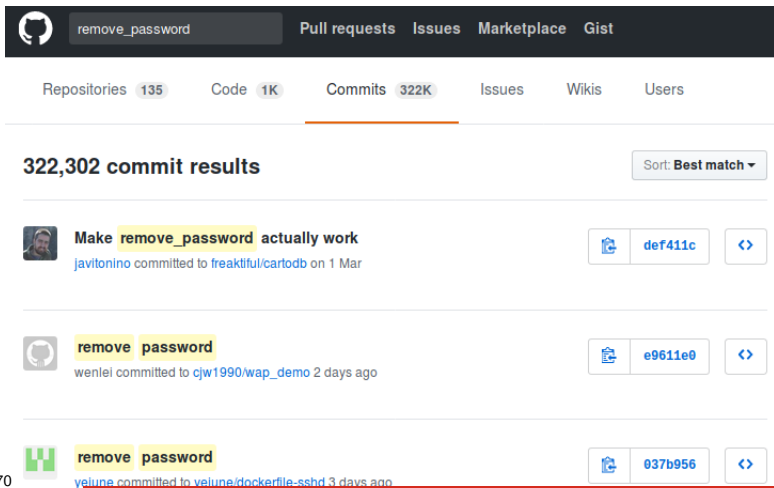
**Bottom-left paste:** Title "text 4.57 KB". Line 1: "1. #include "wejwyj.h"". Line 3: "3. int zapisz (FILE \*plik\_". Line 4: "4. int i, j;". Line 5: "5. if (obr->KOLOR==0) {". Line 7: "7. fprintf (plik\_wy, "P2". Line 8: "8. fprintf (plik\_wy, "%d". Line 9: "9. fprintf (plik\_wy, "%d". Line 10: "10. for (i=0; i<obr->wymy". Line 11: "11. for (j=0; j<obr->wymx; j++". Line 12: "12. fprintf (plik\_wy, "%d ",". Line 13: "13. }".

**Bottom-right paste:** Title "text 2.66 KB". Line 1: "1. <item name=\"%the\_component\_to\_be\_disabled%\" xsi:type=\"array\">". Line 2: "2. <item name=\"config\" xsi:type=\"array\">". Line 3: "3. <item name=\"componentDisabled\" xsi:type=\"boolean\">true</item>". Line 4: "4. </item>". Line 5: "5. </item>". Line 7: "7. <?xml version=\"1.0\"?>". Line 9: "9. <page xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" xsi:noNamespace". Line 10: "10. <body>". Line 11: "11. <referenceBlock name=\"checkout.root\">". Line 12: "12. <arguments>". Line 13: "13. <argument name=\"jsLayout\" xsi:type=\"array\">".

# Sources of leaks: Others

- Mistakes from users

- [https://github.com/search?q=remove\\_password&type=Commits&ref=searchresults](https://github.com/search?q=remove_password&type=Commits&ref=searchresults)



The screenshot shows the GitHub search interface for the query "remove\_password". The search results are filtered to "Commits" and show 322,302 results. The top three results are:

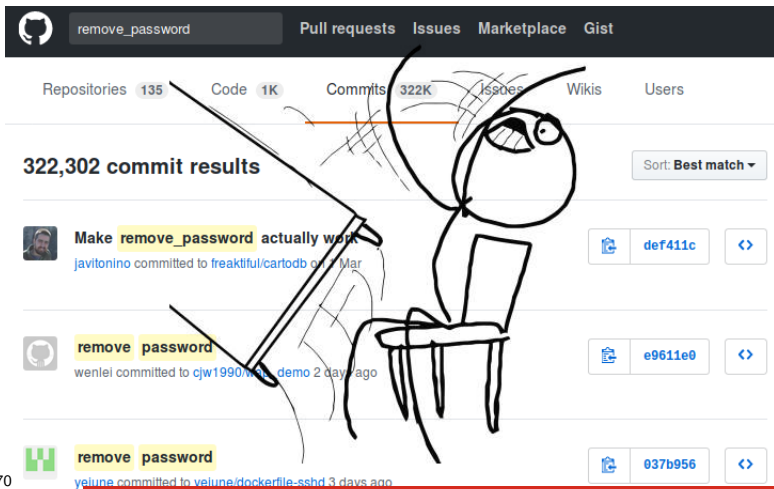
- Make remove\_password actually work** by javitonino, committed to freakiful/cartodb on 1 Mar. Commit hash: def411c.
- remove password** by wenlei, committed to cju1990/wap\_demo 2 days ago. Commit hash: e9611e0.
- remove password** by yelune, committed to yelune/dockerfile-sshd 3 days ago. Commit hash: 037b956.



# Sources of leaks: Others

- Mistakes from users

- [https://github.com/search?q=remove\\_password&type=Commits&ref=searchresults](https://github.com/search?q=remove_password&type=Commits&ref=searchresults)





The screenshot shows the GitHub search interface for the query "remove\_password". The search results are filtered to "Commits" and show 322,302 results. A hand-drawn stick figure is sitting on a chair, pointing a long pencil at the search results. The figure has a large, expressive face with wide eyes and a slightly open mouth, suggesting surprise or focus. The search results list several commits, with the first one being "Make remove\_password actually work" by javitonino. Other results include "remove password" by wenlei and "remove password" by yelune. The interface includes navigation tabs for Pull requests, Issues, Marketplace, and Gist, and a search bar with the query "remove\_password".


remove\_password   Pull requests   Issues   Marketplace   Gist

Repositories 135   Code 1K   Commits 322K   Issues   Wikis   Users

322,302 commit results   Sort: Best match ▾

 Make **remove\_password** actually work  
javitonino committed to freaktiful/cartodb on 7 Mar

 **remove password**  
wenlei committed to cju1990/w... demo 2 day / ago

 **remove password**  
yelune committed to yelune/dockerfile-ssh3 3 days ago

def411c   <>

e9611e0   <>

037b956   <>

## 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<sup>1</sup> 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.

## 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<sup>2</sup> raised by CIRCL

# 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<sup>3</sup>.

---

<sup>3</sup><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:

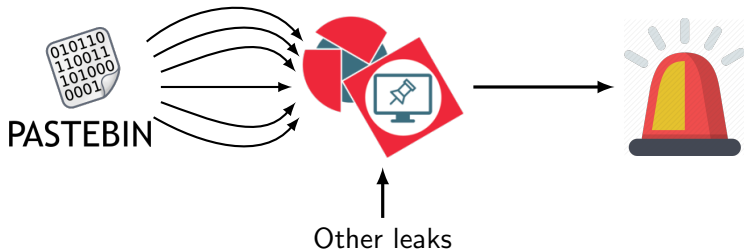
- AIL initially started as an **internship project** (2014) to evaluate the feasibility to automate the analysis of (un)structured information to find leaks.
- In 2018, AIL framework is an **open source software** in Python. The software is actively used (and maintained) by CIRCL.



# 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**

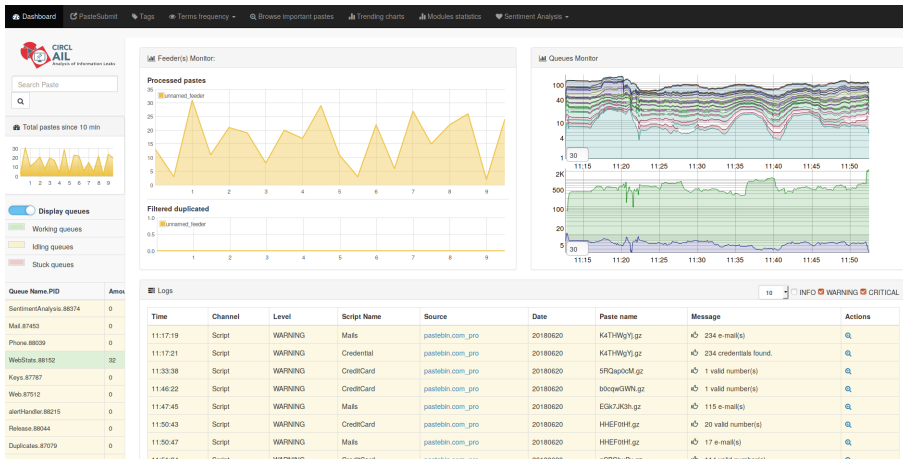
## AIL Framework: Current features

---

- 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 occurrences**
- 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

---

Q 1 Results for "B35nGGBp"

Show  entries Search:

#	Path	Date	Size (Kb)	Action
1	<a href="/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2017/01/20/B35nGGBp.gz">/home/adulau/git/AIL-framework/PASTES/archive/pastebin.com_pro/2017/01/20/B35nGGBp.gz</a>	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  entries Search:

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

# Example: Following a notification (3) - Browsing content

---

## Content:

```
http://members2.mofosnetwork.com/access/login/  
somoextremos: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:rbcndnd
```

```
#####
```

```
>| 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:R0bluJ8XtB  
dabudka:17891789  
brajits:brajits1  
  
http://members.pornstarplatinum.com/sblogin/login.php/  
gigiriveracom:xxxjay  
jayx123:xxxjay69  
  
http://members.vividceleb.com/  
Rufio99:fairhaven  
ScHiFRv1:102091  
Chaos84:HOLE5244  
Riptor705:blade7  
Dnm18
```

## Setting up the framework

# Setting up AIL-Framework from source or virtual machine

---

## Setting up AIL-Framework from source

```
1 git clone https://github.com/CIRCL/AIL-framework.git
2 cd AIL-framework
3 ./installing_deps.sh
4 cd var/www/
5 ./update_thirdparty.sh
```

Using the virtual machine:

1. Download [https://www.circl.lu/assets/files/ail-training/AIL\\_v@4986352.ova](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

## ALL ecosystem: Technologies used

---

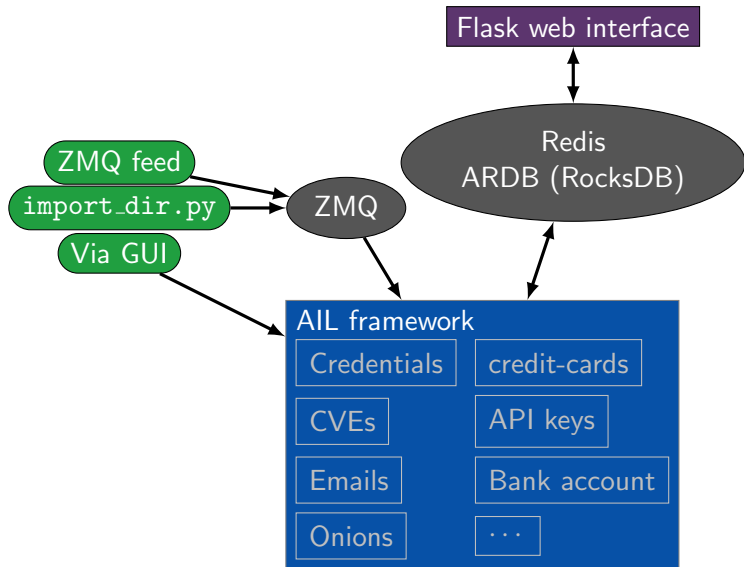
**Programing language:** Full python3

**Databases:** Redis and ARDB

**Server:** Flask

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

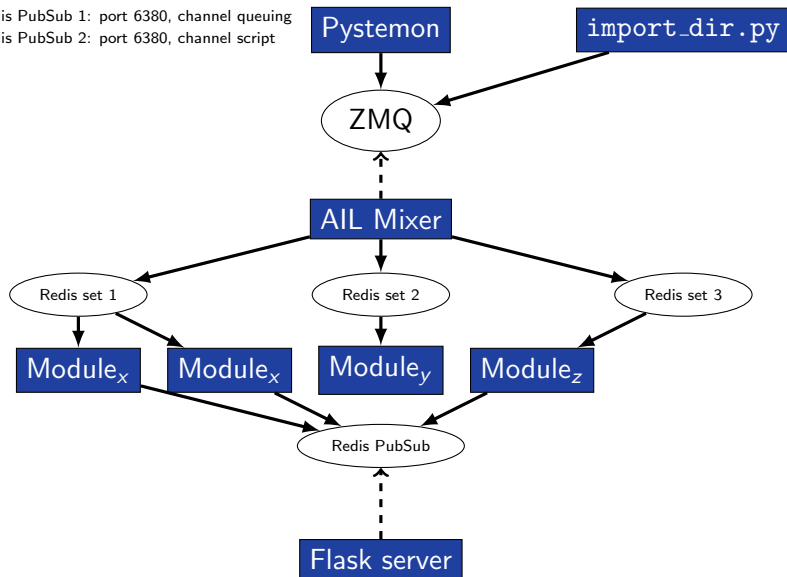
## AIL global architecture 1/2



## AIL global architecture 2/2

Redis PubSub 1: port 6380, channel queuing

Redis PubSub 2: port 6380, channel script

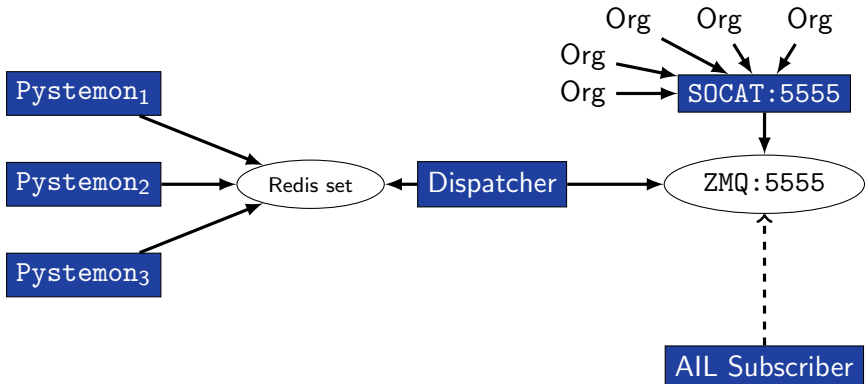


# Data feeder: Gathering pastes with pystemon

## Pystemon global architecture

Redis PubSub 1: port 6380, channel queuing

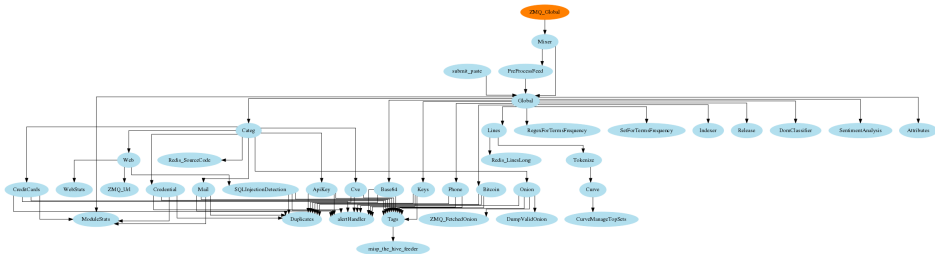
Redis PubSub 2: port 6380, channel script





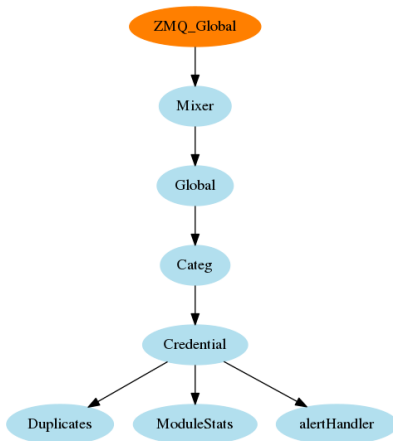
# AIL global architecture: Data streaming between module

---



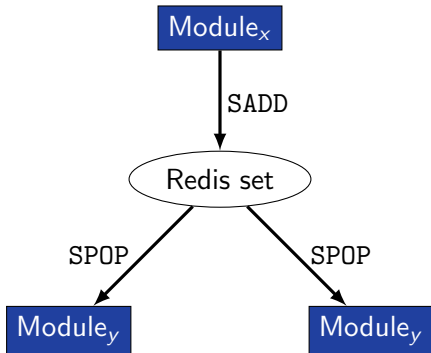
# ALL global architecture: Data streaming between module (Credential example)

---



## 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 rendering the pages (including javascript) and produce screenshots (HAR archive too)

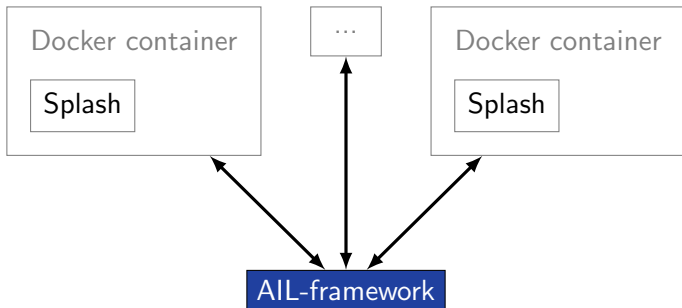


Figure: Architecture of AIL and its hidden services crawler

## 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

```
1 # Activate the virtualenv
2 . ./AILENV/bin/activate
3
4 # Launch the system
5 cd bin/
6 ./LAUNCH -l
7
8 # 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 different ways 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 different ways to feed AIL with data:

1. CIRCL trusted partners can ask to access our feed [info@circl.lu](mailto: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**

No file selected.

**Archive Password**

Tags :

## Via the UI (2)

---


Submitting Pastes ...

100 %

Files Submitted 1/1

Submitted pastes

```
/home/all/git/AIL.framework/PASTES/submitted/2018/06/29/02071570-b464-4bbb-be59-37c58c9b8925.gz
```

Submitted Pastes  Success ✓

## Feeding ALL 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 ALL
2. Each file to be fed must be of a reasonable size:
  - 2.1 ~ 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 ALL 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 ALL 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 the directory you want to import
  - `import_dir.py -d dir_path`



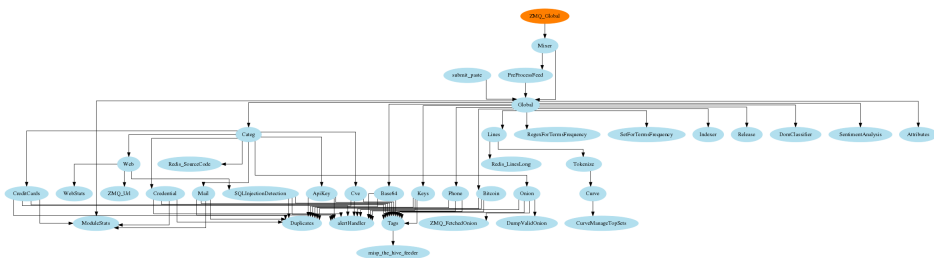
## Feeding ALL with your own data - import\_dir.py (2)

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 the directory you want to import
  - `import_dir.py -d dir_path`
3. Watch your data being feed to ALL

## Creating new features

# Developing 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)
```

## 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

---



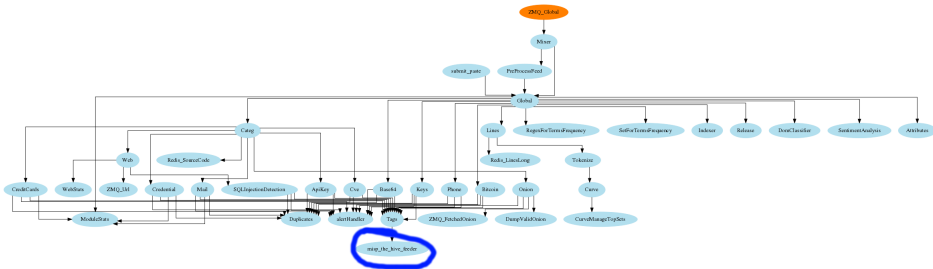
1. Use infoleak taxonomie
2. Add your own tags
3. Create an event on a paste





# 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

MISP Auto Event Creation Enabled



**MISP**  
Threat Sharing

✕ Disable Event Creation

The hive auto export Disabled



**TheHive**

Enable Alert Creation

Metadata : 6 / 25

Show  entries Search:

Whitelist	Tag	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="api-key"	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="aws-key"	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="base64"	
<input type="checkbox"/>	infoleak:automatic-detection="bitcoin-address"	
<input type="checkbox"/>	infoleak:automatic-detection="bitcoin-private-key"	

Showing 1 to 5 of 25 entries

Previous

Next

Metadata : 23 / 25

Show  entries Search:

Whitelist	Tag	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="api-key"	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="aws-key"	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="base64"	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="bitcoin-address"	
<input checked="" type="checkbox"/>	infoleak:automatic-detection="bitcoin-private-key"	

Showing 1 to 5 of 25 entries

Previous

Next

# Create an event

infoleak:automatic-detection="base64" +

Date	Source	Encoding	Language	Size (Kb)	Mime
20/06/2018	pastebin.com_pro	text/plain	('mt', 0.9892176706413881)	1.58	text/plain

Create  Event

## Duplicate list:

Show  entries

Hash type	Paste info	Date	Path
[t!sh]	Similarity: [59]%	2018-05-30	/home/aurelien/git/python3/AIL-framework/PASTES/archive/pastebin.com_pro/2018/05/30/ePtckUe.gz


Showing 1 to 1 of 1 entries

## Content:

[\[Raw content\]](#)

```
power shell -noP -sta -w 1 -enc JABHAFIATwBVAFAAUABvAEwAaQBDAHKAUwBFAFQAVABJAG4ARwBzACAAPQAgAFsAcgBFAEYAXQAUAEAAUwBTAGUAbQBCAGwAeQAuAECaZQB0AFQAEQBwAGUAKAAnAF
```

# Create an event



**MISP**  
Threat Sharing

**Distribution**

**Threat Level**

**Analysis**

**Event Info**

**Publish 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.**

→ 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

```
screen(1: ModuleInformation)

Running Queues
Action Queue name PID # S Time R Time Processed element CPU % Mem % Avg CPU%
<K> Attributes 31731 5 2017-08-03 00:24:03 0:00:01 G3rBPVqV 3.10% 1.56% 3.60%
<K> BrowseWarningPaste 31952 2 2017-08-03 00:23:55 0:00:09 yP3DaL03 0.00% 1.43% 0.00%
<K> Categ 31766 30 2017-08-03 00:23:58 0:00:06 Hs13zr6Y 6.70% 1.64% 17.40%
<K> Credential 31822 7 2017-08-03 00:24:04 0:00:00 yP3DaL03 3.50% 1.63% 3.50%
<K> CreditCards 31783 11 2017-08-03 00:24:04 0:00:00 q9q5sLnd 4.80% 1.66% 4.80%
<K> DomClassifier 31755 71 2017-08-03 00:23:52 0:00:12 Wz0FFBX 1.70% 1.64% 5.73%
<K> Indexer 31870 10 2017-08-03 00:24:03 0:00:01 0255zMLU 67.60% 1.93% 61.47%
<K> Lines 31744 5 2017-08-03 00:24:03 0:00:01 zLEpJfB 5.20% 1.57% 3.37%
<K> Mlxer 31704 2 2017-08-03 00:24:03 0:00:01 6GzeZ7zx 0.30% 0.43% 0.40%
<K> ModuleStats 31932 33 2017-08-03 00:23:57 0:00:07 7QCEJHTV 0.00% 1.64% 0.00%
<K> Phone 31888 2 2017-08-03 00:24:04 0:00:00 ghqFECHA 3.40% 1.59% 3.85%
<K> Release 31899 30 2017-08-03 00:23:57 0:00:07 3PwHXVtJ 1.80% 1.64% 0.55%
<K> SQLInjectionDetection 31941 1 2017-08-03 00:23:55 0:00:09 JNPO0wmj 0.00% 1.49% 0.10%
<K> Tokenize 31775 42 2017-08-03 00:24:03 0:00:01 WTSF5hgL 6.60% 1.57% 6.60%
<K> Web 31818 17 2017-08-03 00:23:45 0:00:19 JNPO0wmj 0.00% 1.74% 0.00%
<K> WebStats 31922 2 2017-08-03 00:23:14 0:00:50 JNPO0wmj 0.00% 0.51% 0.00%

Idle Queues
Action Queue Idle Time Last paste hash
<K> Global 31717 0:00:00 n0DwHkX
<K> Keys 31880 0:00:00 yCWJXRlp
<K> Mail 31805 0:00:01 rhn2F3Yt

Queues not running
Action Queue State
<S> Curve Stuck or idle, restarting disabled
<S> CurveManagementTopSets Not running by default
<S> Cve Stuck or idle, restarting disabled
<S> DumpValidOntion Not running by default
<S> Duplicates Stuck or idle, restarting disabled
<S> Ontion Stuck or idle, restarting disabled
<S> PreProcessFeed Not running by default
<S> RegexForTermsFrequency Stuck or idle, restarting disabled
<S> SentimentAnalysis Stuck or idle, restarting disabled
<S> SetForTermsFrequency Stuck or idle, restarting disabled

Logs
TTime Module PID Info
00:23:29 Duplicates 31725 Cleared invalid pid in MODULE_TYPE_Duplicates
00:23:29 SentimentAnalysis 31961 *invalid pid in MODULE_TYPE_SentimentAnalysis
00:23:29 RegexForTermsFrequency 31852 *id pid in MODULE_TYPE_RegexForTermsFrequency
00:23:29 Curve 31837 Cleared invalid pid in MODULE_TYPE_Curve
00:23:29 SetForTermsFrequency 31864 *valid pid in MODULE_TYPE_SetForTermsFrequency
00:23:11 * cleared redis module info

0:24 0$ bash [1 ModuleInformation] 2$ Mlxer 3$ Global 4$ Duplicates 5$ Attributes 6$ Lines 7$ DomClassifier 8$ Categ 9$ Tokenize 10$ CreditCards 11$ Ontion 12$ Mail 13$ Web 14$ Creden
```