

AIL Project

Open source framework to efficiently collect, crawl, dig, and analyze unstructured data



CIRCL

Computer Incident
Response Center
Luxembourg

Aurelien Thirion

aurelien.thirion@circl.lu

info@circl.lu

July 3, 2023

Links

- ALL project <https://github.com/ail-project> (**all components including feeders and crawler infrastructure**)
- ALL framework
<https://github.com/ail-project/ail-framework> (**analysis framework**)
- Training materials and slide deck
<https://github.com/ail-project/ail-training>
- Online chat <https://gitter.im/ail-project/community>



Legal and Ethics

Ethics in Information Security and Cybersecurity

- The materials and tools presented can open a significant numbers of questions regarding ethics;
- Our researches and tools are there for education, supporting the public good and improve incident response;
- We ask all users and participants to **follow ethical principles and act professionally**¹.

¹<https://www.acm.org/code-of-ethics>

<https://www.first.org/global/sigs/ethics/ethics-first>

Collecting, processing and analysing content - web pages

- Building a search engine on the web is a challenging task because:
 - it has to crawl webpages,
 - it has to make sense of **unstructured data**,
 - it has to **index** these data,
 - it has to provide a way to retrieve data and structure data (e.g. correlation).
- Doing so on Tor is even more challenging because:
 - services don't always want to be found,
 - parts of the dataset have to be discarded.
- in each case, it requires a lot of bandwidth, storage and computing power.

Collecting, processing and analysing content - structured data

- Some data are structured and are easy to process:
 - metadata!
 - API responses.
- Some even provide cryptographic evidences:
 - authentication mechanisms between peers,
 - OpenPGP can leak a lot of metadata
 - key ids,
 - subject of email in thunderbird,
 - Bitcoin's Blockchain is public,
 - pivoting on these data with external sources yields interesting results.

AIL Design Objectives

Session Objectives

- Demonstrate the practical usage and extensibility of an open source tool for monitoring web pages, pastes, forums, and hidden services
- Discuss the challenges involved and delve into the design principles of the AIL open source framework
- Explore various **collection mechanisms and sources utilized** by the AIL framework
- Gain knowledge on creating new modules within the AIL framework
- Acquire (quickly) proficiency in using, installing, and initializing AIL
- Understand the significance of integrating the AIL framework into the cyber threat intelligence life cycle, with notable tools such as MISP

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 2019, AIL framework is an **open source software** in Python. The software is actively used (and maintained) by CIRCL and many organisations.
- In 2020, AIL framework became a complete project called **ail project**².
- In 2023, AIL framework version 5.0 released with a new datastorage back-end.

²<https://github.com/ail-project/>

Capabilities Overview

Common usage

- **Check** if mail/password/other sensitive information (terms tracked) leaked
- **Detect** reconnaissance of your infrastructure
- **Search** for leaks inside large leak archive
- **Monitor** and crawl websites

Supporting CERT and Law Enforcement Activities


- Proactive Investigation: Detection of Leaks
 - Compilation of leaked emails and passwords
 - Analysis of leaked databases
 - Identification of exposed SaaS keys (AWS, Google,...)
 - Detection of compromised credit card information
 - Identification and analysis of compromised PGP private keys and certificate keys
- Contributing to Passive DNS and Metadata Collection Systems
- Sharing CVEs and Proof-of-Concepts (PoCs) for commonly exploited vulnerabilities
- Deanonymization of Hidden Services

Support CERT and Law Enforcement activities

- Website monitoring
 - Monitor booters, marketplaces, forums
 - Detect encoded exploits (WebShell, malware encoded in Base64,...)
 - SQL injections
- Automatic and manual submission to threat intelligence sharing and incident response platforms
 - MISP
 - TheHive
- Term/Regex/YARA monitoring for local companies/government keywords

Sources of leaks



Catching mistakes from users



 [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 1 Mar  **def411c** [↔](#)

 **remove password**
[wenlei](#) committed to [cjlw1990/wap_demo](#) 2 days ago  **e9611e0** [↔](#)

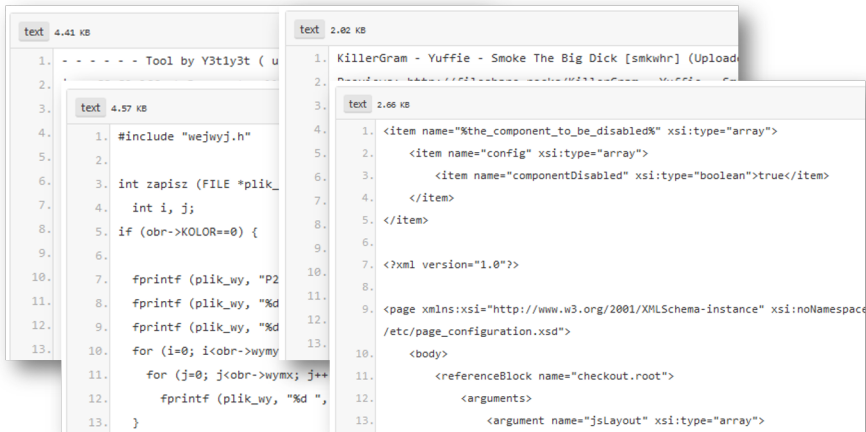
 **remove password**
[yejune](#) committed to [yejune/dockerfile-ssh](#) 3 days ago  **037b956** [↔](#)

 **Removed Passwords**

Example - Sources of leaks - paste monitoring

- Example: <https://gist.github.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 (items)



The image displays three overlapping text editor windows, each showing a different type of code paste:

- Top-left window:** A text file (4.41 KB) containing a single line of text: `-- -- Tool by Y3t1y3t (u`.
- Bottom-left window:** A text file (4.57 KB) containing C code for a file operation. The code includes a header `#include "wejwyj.h"`, declares `int zapisz (FILE *plik_`, `int i, j;`, and a function body with an `if (obr->KOLOR==0) {` block containing `fprintf` calls and a nested loop: `for (i=0; i<obr->wymy`, `for (j=0; j<obr->wymx; j++`, and `fprintf (plik_wy, "%d "`.
- Top-right window:** A text file (2.82 KB) containing the title `KillerGram - Yuffie - Smoke The Big Dick [smkwhr] (Upload`.
- Bottom-right window:** A text file (2.66 KB) containing XML code for a page configuration. It starts with `<?xml version="1.0"?>` and includes a `<page>` element with namespace declarations and a `<body>` element containing `<referenceBlock name="checkout.root">`, `<arguments>`, and `<argument name="jsLayout" xsi:type="array">`.

Purposes of Leaks

- **Economic Interests:** Adversaries may promote services for their own financial gain.
- **Ransom Model:** Leaks can be used to publicly pressure victims into meeting certain demands.
- **Political Motives:** Adversaries may leak information to showcase their power or influence.
- **Collaboration:** Criminals may need to collaborate and share leaked information for their operations.
- **Operational Infrastructure:** Examples include malware that exfiltrates information to pastie websites.
- **Mistakes and Errors:** Leaks can also occur due to unintentional mistakes or errors.

Objectives for SOC/CSIRT Teams

- **Contacting Companies or Organizations:** Reach out to companies or organizations responsible for specific accidental leaks to address the issue
- **Engaging with Media:** Collaborate with the media to discuss specific leak cases and find practical ways to increase factual information available to the public
- **Evaluate the Cybercriminal Economy:** Analyze the cybercriminal market, including activities such as DDoS booters³ and the reselling of personal information, in order to understand the disparity between reality and media coverage
- **Analyze the Collateral Effects:** Investigate the broader impact of malware, software vulnerabilities, or data exfiltration incidents

³<https://github.com/D4-project/>

Current capabilities

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**
- Automatic Tor Crawler and website crawling (handle cookies authentication) via Lacus⁴

⁴<https://github.com/ail-project/lacus>

AIL Framework - 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, regex and YARA **tracking and occurrences**
- Archives, files and raw **submission** from the UI
- PGP, Cryptocurrencies, Decoded (Base64, ...) and username Correlation
- And many more

Trackers - Retro Hunt

- Search and monitor specific keywords/patterns
 - Automatic Tagging
 - Email Notifications
- Track Word
 - ddos
- Track Set
 - booter,ddos,stresser;2
- Track Regex
 - circl\.lu
- Track Typo-squatting
- YARA rules
 - <https://github.com/ail-project/ail-yara-rules>

YARA Tracker

Certificate



Type 🔴 yara

Tracked all-yara-rules/rules/crypto/certificate.yar

Date 2023/05/12

Level Global

Creator admin@admin.test

First Seen 2023 / 05 / 12

Last Seen 2023 / 05 / 31

Tags

Mails

Webhook

Filters no filters

Objects Match decoded 0
item 00

[Edit Tracker](#) ✎ 🔴

Yara Rule:

```
rule certificates
{
  meta:
    author = "@kevthehermit"
    info = "Part of Pastehunter"
    reference = "https://github.com/kevthehermit/Pastehunter"

  strings:
    $ssh_priv = "BEGIN RSA PRIVATE KEY" wide ascii nocase
    $openssh_priv = "BEGIN OPENSSH PRIVATE KEY" wide ascii nocase
    $dsa_priv = "BEGIN DSA PRIVATE KEY" wide ascii nocase
    $ec_priv = "BEGIN EC PRIVATE KEY" wide ascii nocase
    $pgp_priv = "BEGIN PGP PRIVATE KEY" wide ascii nocase
    $pem_cert = "BEGIN CERTIFICATE" wide ascii nocase
    $pkcs7 = "BEGIN PKCS7"

  condition:
    any of them
}
```

📅 2023-05-12

📅 2023-05-31

🔍 Tracked Objects



Trackers - Practical part

- **Create and test your own tracker**

Create a new Tracker

E-Mails Notification (optional, space separated)

Webhook URL

Tracker Description (optional)

Show tracker to all Users

Objects to Track:

Decoded

Item

Filter Item by sources

PGP

Filter PGP by subtype:

name

mail

Tags

Select Tags

Taxonomie Selected

Select Tags

Galaxy Selected

Tracker Type:

Retro Hunt

test completed

Date 2023/05/10

Description None

Tags

Creator admin@admin.test

Filters {
 "item": {
 "date_from": "20230304",
 "date_to": "20230601"
 }
}

Objects Match item 3

Show Objects

```
rule certificates
{
  meta:
    author = "@KevTheHermal"
    info = "Part of PasteHunter"
    reference = "https://github.com/kevthehermal/PasteHunter"

  strings:
    $ssh_priv = "BEGIN RSA PRIVATE KEY" wide ascii nocase
    $openssh_priv = "BEGIN OPENSSH PRIVATE KEY" wide ascii nocase
    $dsa_priv = "BEGIN DSA PRIVATE KEY" wide ascii nocase
    $ec_priv = "BEGIN EC PRIVATE KEY" wide ascii nocase
    $pgp_priv = "BEGIN PGP PRIVATE KEY" wide ascii nocase
    $pem_cert = "BEGIN CERTIFICATE" wide ascii nocase
    $pkcs7 = "BEGIN PKCS7"

  condition:
    any of them
}
```

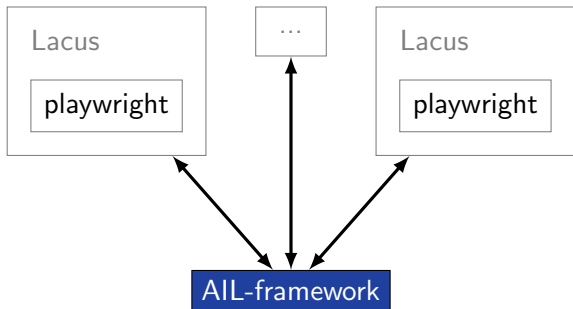
Show 10 entries

Search:

Type	Id	Tags
	archive/gist.github.com/2023/04/14/huzmiranda7_3b3d1133a3d3842092c5fc5fb39e84f2.gz	infoleak-automatic-detection:"private-key" test23 test12 infoleak-automatic-detection:"certificate"
	submitted/2023/04/20/submitted_cc9190ab-80d2-4d2b-9c9e-97c51e69a855.gz	infoleak-submission:"manual" test12 infoleak-automatic-detection:"rsa-private-key" infoleak-automatic-detection:"vpn-static-key" test23 infoleak-automatic-detection:"certificate" infoleak-automatic-detection:"onion"
	archive/gist.github.com/2023/04/13/chipzoller_dbd6d2d737d02ad4fe9d30a897170761.gz	test12 test23 infoleak-automatic-detection:"certificate"

Crawler

- Crawlers are used to navigate on regular website as well as .onion addresses (via automatic extraction of urls or manual submission)
- Lacus⁵ ("scriptable" browser) is rendering the pages (including javascript) and produce screenshots (HAR archive too)



⁵<https://github.com/ail-project/lacus>

Auto Crawler

How a domain is crawled by default

1. Fetch the first url
2. Render the **web page including javascript** (done by playwright via Lacus)
3. Extract all urls
4. Filter url: keep all url of this domain
5. crawl next url (max depth = 1)

Crawler: Cookiejar

Use your cookies to login and bypass captcha

Edit Cookiejar

Description	Date	UUID	User
3thxemke2x7hcibu.onion	2020/03/31	90674deb-38fb-4eba-a661-18899ccb3841	admin@admin.test

[Edit Description](#) [Add Cookies](#)

```
{
  "domain": ".3thxemke2x7hcibu.onion",
  "name": "mybb[lastactive]",
  "path": "/forum/",
  "value": "1583829465"
}
```


```
{
  "domain": ".3thxemke2x7hcibu.onion",
  "name": "loginattempts",
  "path": "/forum/",
  "value": "1"
}
```

```
{
  "domain": ".3thxemke2x7hcibu.onion",
  "name": "sid",
  "path": "/forum/",
  "value": "047ab0cd97ff5bcc77edb6a"
}
```

```
{
  "name": "remember_token",
  "value": "12|58cddd151d74d341f23"
}
```


```
{
  "domain": ".3thxemke2x7hcibu.onion",
  "name": "mybb[announcements]",
  "path": "/forum/",
  "value": "0"
}
```


Crawler: Cookiejar


3thxemke2x7hcibu.onion : 



First Seen	Last Check	Ports
2020/03/09	2020/03/30	[80]



`infoleak:automatic-detection="onion"` `infoleak:automatic-detection="base64"`

 manual

 Show Domain Correlations **139**



Add to  MISP Export



 Decoded **1** 

 Screenshot **134** 

Crawled Items Date: **2020/03/23 - 13:10:40** PORT: **80**

Show entries Search:

Crawled Pastes  

Shere Khan

Welcome back, [zulopari](#). You last visited: 03-20-2020, 01:35 PM [Log Out](#) [Search](#)

[User CP](#) [View New Posts](#) [View Today's Posts](#) [Private Messages \(Unread: 2, Total: 2\)](#)

You have 2 unread private messages. The most recent is from [Jack3](#) (50) **KEY FOR PRIVATE SECTIONS**

Shere Khan - Official Forum
Private Messages

Menu | [Inbox](#) | [Compose Message](#) | [Manage Folders](#) | [Empty Folders](#) | [Download Messages](#) | of PM space used.

Messenger

- Compose
- Inbox
- Unread
- Send Status
- Drafts
- Trash Can
- Tracking
- Edit Folder

Your Profile

- Edit Profile
- Change Password
- Change Email
- Change Avatar
- Change Signature
- Edit Options

Miscellaneous

- Group Memberships
- Buddy/Ignore List
- Manage Attachments
- Saved Drafts
- Subscribed Threads
- Forum Subscriptions
- View Profile

Inbox | | (Advanced Search)

Message Title	Sender	Date/Time Sent (see)
KEY FOR PRIVATE SECTIONS	Jack3	3 hours ago
Verification	Jack3	03-09-2020, 11:55 AM

Move To: or Delete the selected messages

Jump to Folder: [Go!](#)

Forum Team | [Contact Us](#) | [Shere Khan - hacking group](#) | [Return to Top](#) | [Lite \(Archive\) Mode](#) | [Mark all forums read](#) | [RSS Syndication](#)

Powered by **MyBB**, © 2002-2020 **MyBB Group**. Current Time: 03-23-2020, 01:51 PM

<http://3thxemke2x7hcibu.onion/forum/private.php>

Lacus: Web Capturing System

- Lacus⁶ is a web capturing system built on playwright.
- AIL utilizes Lacus for fetching and rendering domains.
 - Lacus can be installed and used independently from AIL.
 - Capture what you need by enqueueing requests.
 - Initiate the capture process.
 - Retrieve the capture results.

⁶<https://github.com/ail-project/lacus>

Crawler Settings - Lacus

AIL Lacus Crawler

 Connected

Lacus URL

http://lacus.circl.lu:7100

Edit 

Crawlers

 It works!

```
-----  
- TOR CRAWLER TEST OUTPUT: -  
-----
```

It works!

ReRun Test 

Number of Concurrent Crawlers to Launch: 15

Edit 

Crawler: DDoS Booter

UP

qy4n6ptiraa7mtfy73wcp6da2xrapmbanwfr5kei4zrq2va4uscvogid.onion :

First Seen	Last Check	Ports
2019/08/15	2019/10/06	[80]

[infoleak:automatic-detection="bitcoin-address"](#) [infoleak:automatic-detection="ethereum-address"](#)
[infoleak:automatic-detection="onion"](#) [infoleak:automatic-detection="credit-card"](#) [ddos](#)

⊞

Last Origin: [crawled/2019/10/05/mqbyxj4ladgz5cd.onion0aa31681-fa45-4fc3-8151-7a7c5ac7e906](#)

🔍 Show Domain Correlations 2


📁 Cryptocurrencies 2

Hide Full resolution

HOME ABOUT PROOF PRICE PAYMENT

DDOSTECH

WICKR. DDOS TECHNOLOGY




Reviews

April 25, 2019
I turned to this service on the recommendation of my friend, ordered an attack for a whole week, the work was done with high quality and responsibility.

September 21, 2018
I found this site through YAHOO, immediately contacted this service, and I had a free attack for almost ten minutes.

We accept:

Accept payments cryptocurrency. Cryptocurrency transfers guarantee your our security transaction. We accept BTC, ETH, DASH, LTC, ETC, XMP ...



Wallets Addresses

Recon and intelligence gathering tools

- **Attacker also share informations**
- Recon tools detected: 94
 - sqlmap
 - dnscan
 - whois
 - msfconsole (metasploit)
 - dnmap
 - nmap
 - ...

Recon and intelligence gathering tools

```
#####  
=====
```

Hostname	www.pabloquintanilla.cl	ISP	Wix.com Ltd.
Continent	North America	Flag	
US			
Country	United States	Country Code	US
Region	Unknown	Local time	19 Nov 2019 07:59 CST
City	Unknown	Postal Code	Unknown
IP Address	185.230.60.195	Latitude	37.751
	Longitude	-97.822	

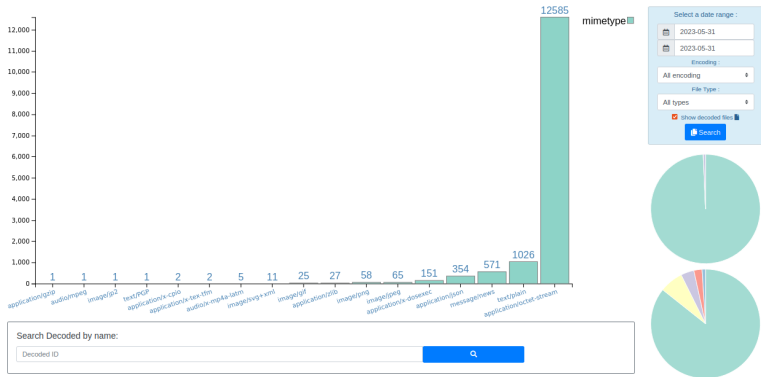
```
=====
```

```
#####  
> www.pabloquintanilla.cl  
Server:      38.132.106.139  
Address:     38.132.106.139#53  
  
Non-authoritative answer:  
www.pabloquintanilla.cl canonical name = www192.wixdns.net.  
www192.wixdns.net      canonical name = balancer.wixdns.net.  
Name:      balancer.wixdns.net  
Address: 185.230.60.211  
>  
#####  
Domain name: pabloquintanilla.cl  
Registrant name: SERGIO TORO  
Registrant organisation:  
Registrar name: NIC Chile  
Registrar URL: https://www.nic.cl
```

Decoder

- Search for encoded strings
 - Base64
 - Hexadecimal
 - Binary
- Guess Mime-type
- Items/Domains Correlation

Decoder:

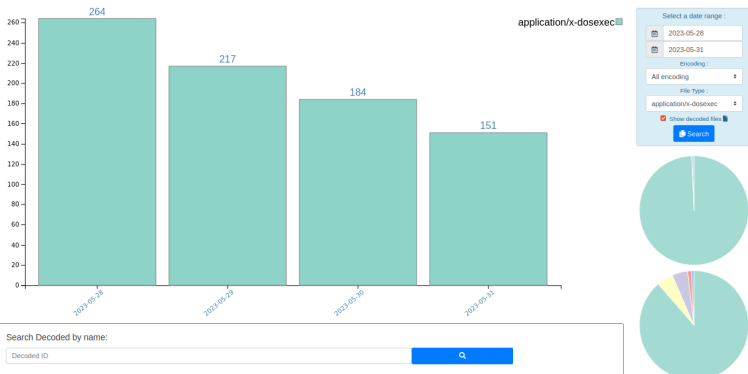


20230531 Decoded files:

Show 10 entries

estimated type	hash	first seen	last seen	nb item	size	Virus Total	Sparkline
image/gif	ee08cd7fe62be822c8ec1364wbf2d940dc3e05	20230404	20230531	214708	1108	Virus Total submission is disabled	
image/png	8003399c6a0e82086453aa04a887105ca2f6a4	20230404	20230531	8404	1054	Virus Total submission is disabled	
application/json	f9f918w60a395e2523bd844ec9c6842cda11f9	20230410	20230531	3947	44	Virus Total submission is disabled	

Decoder:



Search Decoded by name:















20230528 to 20230531 Decoded files:











Show 10 entries

Search:

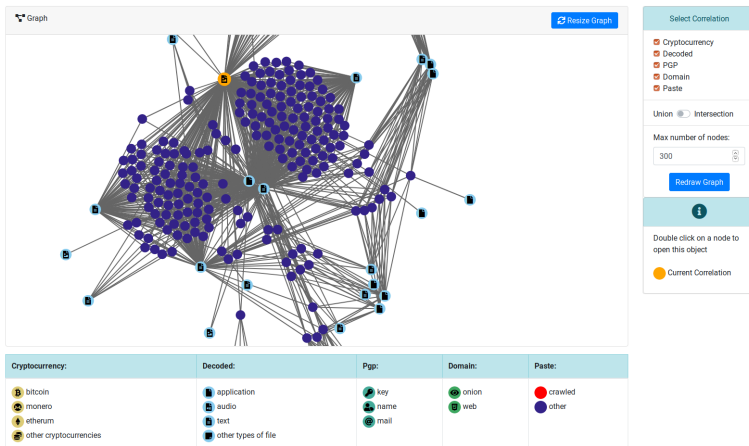
estimated type	hash	first seen	last seen	nb item	size	Virus Total	Sparkline
application/x-dosexec	c408501f702d8279704c380a61d329e611962	20230421	20230529	76	64	Virus Total submission is disabled	
application/x-dosexec	a9ecb74ce7d2b70f0cd0f5729931ce570161	20230405	20230530	56	55666	Virus Total submission is disabled	
application/x-dosexec	e5875aa6a6c6e013d5feb4bab4bc45b4c84127	20230529	20230531	4	32	Virus Total submission is disabled	

AIL Objects

Cryptocurrency:	Decoded:	Objects:
 bitcoin	 application	 cookie-name
 monero	 audio	 cve
 ethereum	 text	 screenshot
 other cryptocurrencies	 other types of file	 title

Pgp:	Username:	Domain:	Item:
 key	 telegram	 onion	 crawled
 name	 twitter	 web	 other
 mail	 jabber		

Correlations and relationship



Investigations

Tor Coin Mixer

UUID	9189d0e7c04c47a2985666e9507e0a5	Delete Edit Export as Event
Creator	admin@admin.test	
Tags	dark-web:topic:mixer	
Date	2023-05-31	
Threat Level	medium	
Analysis	initial	
Info	Tor Coin Mixer	
# Objects	6	
Timestamp	2023-05-31 12:50:45	
Last change	2023-05-31 12:54:20	

Objects

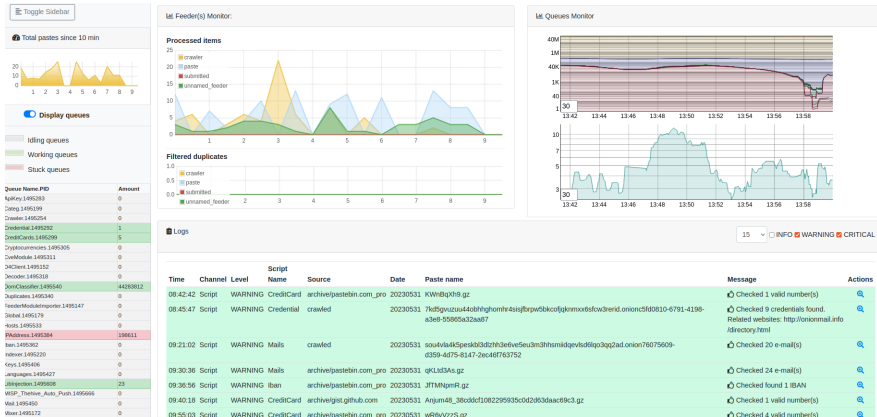
Show entries

Search:

Type	Id	Tags	
onion	jamblery7zqpknhjmg3mh6dajnyd0qxbu#f6voa32h5w4otux3crqdt.onion	dark-web:automatic-detection:mixer dark-web:automatic-detection:pgp-public-key:lock	Delete
onion	btmrxh#4cpcnclufwflussk23bvowsibe4#dree74a#jnz2vyqgd.onion	dark-web:automatic-detection:mixer	Delete
key	0xD3B280956F0E7CAF		Delete
mail	support@jambler.io		Delete
telegram	jambler		Delete
name	Jambler.io		Delete

Live demo!

Example: Dashboard



Example: Text search

Q 1 Results for "gandcrab"

Index: 2019-05-20 - 1365.328591 Mb


Show 10 entries Search:

#	Path	Date	Size (Kb)	Action
0	crawled/2019/05/17/vs5e7g245s3pxjoc.onion374a1a89-4b16-4c3f-a460-4be8898da140 crawler CVE	2019/05/17	15.44	i Q

Showing 1 to 1 of 1 entries Previous **1** Next

Totalling 1 results related to paste content

Example: Items Metadata (1)

infoleak:automatic-detection="phone-number"	infoleak:automatic-detection="mail"	infoleak:automatic-detection="base64"	+				
Date	Source	Encoding	Language	Size (Kb)	Mime	Number of lines	Max line length
04/05/2019	pastebin.com_pro	text/plain	None	6.12	text/plain	1650	100
Create  Event							

Duplicate list:

Show entries

Search:

Hash type	Paste info	Date	Path	Action
[tlsh]	Similarity: [19]%	2019-04-13	archive/pastebin.com_pro/2019/04/13/EbMVR87S.gz	<input type="checkbox"/>
[tlsh]	Similarity: [10]%	2019-04-11	archive/pastebin.com_pro/2019/04/11/2X5HFRVnX.gz	<input type="checkbox"/>
[tlsh]	Similarity: [23]%	2019-04-25	archive/pastebin.com_pro/2019/04/25/TS2b6M4c.gz	<input type="checkbox"/>
[tlsh]	Similarity: [14]%	2019-04-17	archive/pastebin.com_pro/2019/04/17/CuS93H7K.gz	<input type="checkbox"/>
[tlsh]	Similarity: [23]%	2019-04-20	archive/pastebin.com_pro/2019/04/20/AQ0qGVQ.gz	<input type="checkbox"/>
[tlsh]	Similarity: [20]%	2019-04-20	archive/pastebin.com_pro/2019/04/20/6DDc13b8.gz	<input type="checkbox"/>
[tlsh]	Similarity: [21]%	2019-05-05	alerts/pastebin.com_pro/2019/05/05/X8nJLzda.gz	<input type="checkbox"/>
[tlsh]	Similarity: [7]%	2019-04-13	archive/pastebin.com_pro/2019/04/13/Lyp4FVWW.gz	<input type="checkbox"/>

Showing 1 to 8 of 8 entries





Previous [1](#) Next

Example: Items Metadata (2)

Hash files:

Show entries



Search:

estimated type	hash	saved_path	Virus Total
 application/octet-stream	3975f058bb0d445b60c10a11f1a5d88e19e4fa84 (1)	HASHS/application/octet-stream /39/3975f058bb0d445b60c10a11f1a5d88e19e4fa84	Send this file to VT 
 application/octet-stream	fed93c1753270fc849a4db37027b569cdd9a6108 (1)	HASHS/application/octet-stream /fe/fed93c1753270fc849a4db37027b569cdd9a6108	Send this file to VT 

Showing 1 to 2 of 2 entries

Previous **1** Next


Example: Items Metadata (3)


 Crawled Item 

Domain [2gtyctckj2y5e3ln.onion:80](#)


Father [crawled/2019/05/20/2gtyctckj2y5e3ln.onion954e1b05-aaa-4586-a4bc-804bf27b54f7](#)

Url <http://2gtyctckj2y5e3ln.onion/index/forgot/password?tc=1>



 **Empire Market**

LOGIN REGISTER FORUMS VERIFY MIRROR

 **MNEMONIC VERIFICATION - PASSWORD/PIN RESET**

Please type your username and security mnemonic below that was provided to you at the time of registration.

Example: 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 IKnowThatGir1 PublicPickUps  
http://members2.mofos.com  
Chrismagg40884:loganm40  
brando1:zzbrando1  
aacoen:1q2w3e4r  
1rstunkle23:my8se1f
```

```
BraZZers  
http://ma.brazzers.com  
gcjensen:gcj21pva  
skjesc17:rbcndnd
```

```
#####
```

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

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

Example: 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  
feldwWek0939:R0bluJ8XtB  
dabudka:17891789  
brajits:brajits1  
  
http://members.pornstarlatinum.com/sblogin/login.php/  
gigiriveracom:xxxjay  
jayx123:xxxjay69  
  
http://members.vividceleb.com/  
Rufio99:fairhaven  
ScHiFRvi:102091  
Chaos84:HOLE5244  
Riptor795:blade7  
Dom180:harkonnen  
GaggedUK:a1k0chan  
  
http://www.ariellaferreira.com/
```

Example: Search by tags

Search Items by Tags :

2023-05-14 2023-05-27

infoleak-automatic-detection:"cve" infoleak-automatic-detection:"bitcoin-address"

Search Items

Show 10 entries Search:

Date	Item	Action
2023/05/16	archive/gist.github.com/2023/05/16/Vazgen788_c036ee7aad316d9038f2a3968abbbc5d.gz infoleak-automatic-detection:"searchsploit-tool" infoleak-automatic-detection:"cve" infoleak-automatic-detection:"ethereum-address" infoleak-automatic-detection:"base64" infoleak-automatic-detection:"bitcoin-address"	
2023/05/16	archive/gist.github.com/2023/05/16/Vijay922_d35cf2f5c9abe682140379e35d5cd935.gz infoleak-automatic-detection:"searchsploit-tool" infoleak-automatic-detection:"cve" infoleak-automatic-detection:"ethereum-address" infoleak-automatic-detection:"base64" infoleak-automatic-detection:"bitcoin-address"	
2023/05/16	archive/gist.github.com/2023/05/16/DmitriyLewen_930515cde810283b7804950efafe3273.gz infoleak-automatic-detection:"searchsploit-tool" infoleak-automatic-detection:"cve" infoleak-automatic-detection:"credential" infoleak-automatic-detection:"bitcoin-address"	
2023/05/19	archive/gist.github.com/2023/05/19/GrahamcOfBorg_46422a069e8b942352a65f3121a769c5.gz infoleak-automatic-detection:"cve" infoleak-automatic-detection:"credential" infoleak-automatic-detection:"bitcoin-address"	
2023/05/26	archive/pastebin.com_pro/2023/05/26/5ewhAH10.gz infoleak-automatic-detection:"ethereum-address" infoleak-automatic-detection:"cve" infoleak-automatic-detection:"bitcoin-address"	

Showing 1 to 5 of 5 entries

Previous 1 Next

Previous 1 Next

Items: 14

MISP

MISP Taxonomies

- **Tagging** is a simple way to attach a classification to an event or attribute.
- **Classification must be globally used to be efficient.**
- Provide a set of already defined classifications modeling estimative language
- Taxonomies are implemented in a simple JSON format ⁷.
- Can be easily cherry-picked or extended

⁷<https://github.com/MISP/misp-taxonomies>

Taxonomies useful in AIL

- **infoleak**: Information classified as being potential leak.
- **estimative-language**: Describe quality and credibility of underlying sources, data, and methodologies.
- **admiralty-scale**: Rank the reliability of a source and the credibility of an information
- **fpf**⁸: Evaluate the degree of identifiability of personal data and the types of pseudonymous data, de-identified data and anonymous data.

⁸Future of Privacy Forum

Taxonomies useful in AIL

- **tor**: Describe Tor network infrastructure.
- **dark-web**: Criminal motivation on the dark web.
- **copine-scale**⁹: Categorise the severity of images of child sex abuse.

⁹Combating Paedophile Information Networks in Europe

threat sharing and incident response platforms



Goal: submission to threat sharing and incident response platforms.

threat sharing and incident response platforms




1. Use infoleak taxonomy¹⁰
2. Add your own tags
3. Export AIL objects to MISP core format
4. Download it or Create a MISP Event¹¹

¹⁰<https://www.misp-project.org/taxonomies.html>

¹¹<https://www.misp-standard.org/rfc/misp-standard-core.txt>

MISP Export

1Gt545E48EPsyTC8voKQDCFpTkwuXduw :

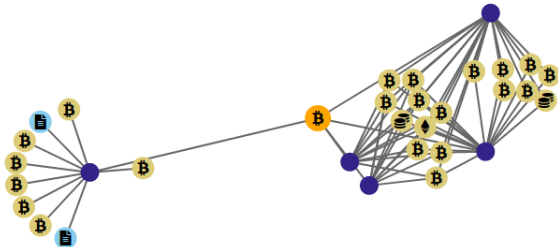
Object type	type	First seen	Last seen	Nb seen
cryptocurrency	 bitcoin	2020/01/17	2020/02/20	5

Expand Bitcoin address

Graph

Resize Graph

Add to  Export



MISP Export



nttfj36sp47cw2yecop572zvjjeazgazieunllouudplzqt2m
5h465yd.onion :

First Seen	Last Check	Ports
------------	------------	-------

2020/02/19	2020/02/19	[80]
------------	------------	------

infoleak:automatic-detection="onion"



Last Origin: [crawled/2020/02/19/dark.failc126d32a-3ed1-468f-ba24-f2e5956f4035](#)

🔍 Show Domain Correlations **4**

Add to Export

Hide

Empire Market

[LOGIN](#) [REGISTER](#) [FORUMS](#) [VE](#)

Login

LOGIN TO EMPIRE MARI

Welcome to Empire Market! Please log
Registrations are free and open to every


Username

Password

What's th

Login

MISP Export



MISP Exporter

Select a list of objects to export

Object Type	Object ID	Lvl	
Object type... ▾		0	
Object type... ▾	1Gt545E48EPsyTC8voKQDCfpTkwiuXduw	✓ 1	
Domain ▾	nttfj36sp47cw2yecop572zjvjeazgazieunllouudplzqt2m5h465yd.onion	✓ 0	

JSON Export Export to MISP Instance

Distribution: ▾

Threat Level: ▾

Analysis: ▾

Event Info:

Publish Event

[Export Objects](#)


Automatic MISP Export on tags

MISP Auto Event Creation Enabled



[✕ Disable Event Creation](#)

The hive Auto Alert Creation Disabled



[Enable Alert Creation](#)

MISP Tags To Push : 3 / 89

Show entries Search:

Enabled	Tag
<input checked="" type="checkbox"/>	infoleak:anlyst-detection="aws-key"
<input checked="" type="checkbox"/>	infoleak:automatic-detection="credit-card"
<input checked="" type="checkbox"/>	test_custom
<input type="checkbox"/>	infoleak:anlyst-detection="api-key"
<input type="checkbox"/>	infoleak:anlyst-detection="base64"

The Hive Tags To Push : 4 / 89

Show entries Search:

Enabled	Tag
<input type="checkbox"/>	infoleak:anlyst-detection="api-key"
<input type="checkbox"/>	infoleak:anlyst-detection="aws-key"
<input checked="" type="checkbox"/>	infoleak:anlyst-detection="base64"
<input checked="" type="checkbox"/>	infoleak:anlyst-detection="binary"
<input type="checkbox"/>	infoleak:anlyst-detection="bitcoin-address"

API

AIL exposes a ReST API which can be used to interact with the back-end¹².

```
1 curl https://127.0.0.1:7000/api/v1/get/item/default
2     --header "Authorization:
3     iHc1_ChZxj1aXmiFiF1mkxxQkzawwriEaZpPqyTQj "
4     -H "Content-Type: application/json"
5     --data @input.json -X POST
```

¹²https:

[//github.com/ail-project/ail-framework/blob/master/doc/README.md](https://github.com/ail-project/ail-framework/blob/master/doc/README.md)

Setting up the framework

Setting up AIL-Framework from source

Setting up AIL-Framework from source

```
1 git clone
   https://github.com/ail-project/ail-framework.git
2 cd AIL-framework
3 ./installing_deps.sh
```

Starting the framework

Running your own instance from source

Accessing the environment and starting AIL

```
1  
2 # Launch the system and the web interface  
3 cd bin/  
4 ./LAUNCH -l
```

Updating ALL

Launch the updater:

```
1 cd bin/  
2 # git pull and launch all updates:  
3 ./LAUNCH -u  
4  
5  
6 # PS:  
7 # The Updater is launched by default each time  
8 # you start the framework with  
9 # ./LAUNCH -l
```

Feeding the framework

Feeding Data to AIL

There are different ways to feed data into AIL:

1. AIL Importers:
 - Dir / Files
 - ZMQ
 - *pystemon*
2. AIL Feeders (discord, telegram, ActivityPub, ...)
3. Feed your own data using the API
4. Feed your own file/text using the UI (Submit section)

Feeding Data to AIL - Technical Considerations

- It is important to consider the size of each file being fed into AIL:
 - For optimal processing and efficiency, it is recommended to keep each file around 3 MB in size
 - This balance between processing capabilities and file size is crucial, as certain modules perform various computations, such as regexp matching, which has a default timeout of 30 seconds
 - If you need to process a large file, it is advisable to split it into multiple smaller files. The AIL leak feeder tool¹³ can assist you in this task.

¹³<https://github.com/ail-project/ail-feeder-leak>

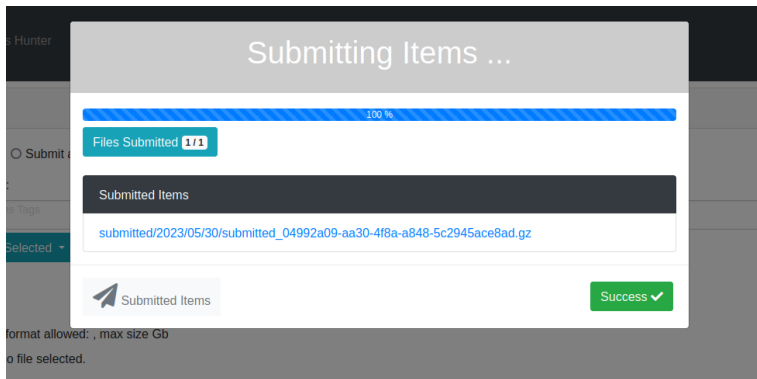
Via the UI (1)

The screenshot shows a web application interface for submitting items. At the top, there is a dark navigation bar with a logo on the left and a search bar on the right. The navigation bar contains the following items: Home, Submit, Tags, Leaks Hunter, Crawlers, Objects, Server Management, and Log Out. The search bar has the text 'Search' and a magnifying glass icon.

Below the navigation bar, there is a sidebar on the left with a 'Toggle Sidebar' button and a 'Submit Items' section. The main content area is titled 'Submit Item' and contains the following elements:

- Radio buttons for 'Submit a file' (unselected) and 'Submit a text' (selected).
- Section 'Optional Tags:' with two dropdown menus. The first is labeled 'Add Taxonomies Tags' and has a blue button below it that says 'Taxonomie Selected'. The second is labeled 'Add Galaxies Tags' and has a blue button below it that says 'Galaxy Selected'.
- Text: 'Submit a text, max size 1.0 Mb'.
- A 'Source' label above a text input field containing the text 'test text to submit'.
- A large blue button at the bottom labeled 'Submit Item'.

Via the UI (2)



API - Feeding AIL with your own data

api/v1/import/item

```
1 {  
2   "type": "text",  
3   "tags": [  
4     "infoleak:analyst-detection=\"private-key\""  
5   ],  
6   "text": "text to import"  
7 }
```

Importers

- Importers are located in the `/bin/importer` directory
- They are used to import different types of data into AIL
- Adding new Importers is straightforward.
- Available Importers:
 - AIL Feeders
 - ZMQ
 - pystemon
 - Files

File Importer

- importer/FileImporter.py

Import File

```
1 . ./AILENV/bin/activate
2 cd tools/
3 ./file_dir_importer.py -f MY_FILE_PATH
```

Import Dir

```
1 . ./AILENV/bin/activate
2 cd tools/
3 ./file_dir_importer.py -d MY_DIR_PATH
```

AIL feeders Importers

- **12+ feeders are available** for all AIL users to feed from external sources
- External feeders can run anywhere and are completely separated from AIL framework
- The feeder can use their **own internal logic** and even push JSON metadata
- Feeder are then pushing the generated JSON to AIL API

Certificate transparency feeder for AIL

- ail-feeder-cti¹⁴ is a generic software to extract information from a certstream server (certificate transparency)
- All metadata extracted will be processed by AIL
- Onion addresses crawled automatically by AIL if seen in a certificate

¹⁴<https://github.com/ail-project/ail-feeder-ct>

GitHub archive and GitHub repository

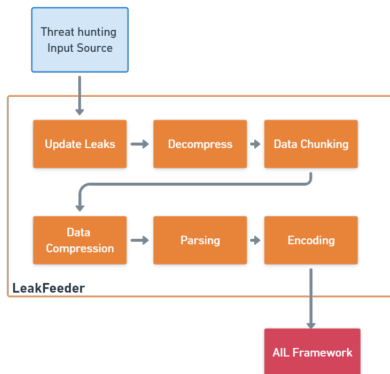
- ail-feeder-gharchive¹⁵ is a generic software to extract informations from **GHArchive**, collect and feed AIL via AIL ReST API
- ail-feeder-github-repo¹⁶ is collecting from a GitHub repository and push everything to AIL
- For monitoring a set of **suspicious git repositories** or finding leaks on existing or managed git repositories, it's a simple way to feed AIL with such source.

¹⁵<https://github.com/ail-project/ail-feeder-gharchive>

¹⁶<https://github.com/ail-project/ail-feeder-github-repo>

AIL LeakFeeder

- ail-feeder-leak¹⁷ automates the process to feed leaked large files automatically to AIL



¹⁷<https://github.com/ail-project/ail-feeder-leak>

AIL feeder ActivityPub

- ail-feeder-activity-pub¹⁸ is feeder for the ActivityPub standard used in distributed social networks (e.g. Mastodon)
- Accounts are required on the ActivityPub instance to get the stream

¹⁸<https://github.com/ail-project/ail-feeder-activity-pub>

AIL feeder telegram

- ail-feeder-telegram¹⁹ is a **Telegram feeder**
- An API ID/hash for Telegram is required and linked to your Telegram phone number

¹⁹<https://github.com/ail-project/ail-feeder-telegram>

More feeders

- ail-feeder-discord²⁰ is a generic **Discord** feeder for AIL
- ail-feeder-atom-rss²¹ is an **Atom and RSS reader** and feeder for AIL
- ail-feeder-jsonlogs²² is a **JSON aggregator** to submit generic JSON input into AIL

²⁰<https://github.com/ail-project/ail-feeder-discord>

²¹<https://github.com/ail-project/ail-feeder-atom-rss>

²²<https://github.com/ail-project/ail-feeder-jsonlogs>

Feeding AIL with custom JSON



conti leaks
@ContiLeaks



conti jabber leaks anonfiles.com/VeP6K6K5xc/1_t...

9:22 PM · 27 févr. 2022 · Twitter Web App

123 Retweets **23** Tweets cités **297** J'aime

```
{
  "ts": "2020-09-08T00:28:49.471678",
  "from": "ceram@q3mcco35auwcstmt.onion",
  "to": "stern@q3mcco35auwcstmt.onion",
  "body": "Проинструктируйте меня. Что делать?"
}
```

Feeding AIL with Conti leaks

- Conti jabber leaks are a good candidate for AIL analysis:
 - PGP keys
 - Bitcoin addresses, maybe others,
 - onion hidden services
- first we translated the files on english using deepl.com
- then we created a feeder to import json data in AIL
- Support added in AIL to correlate jabber usernames

Feeding AIL with Conti leaks

```
from pyail import PyAIL
#... imports
#... setup code
for content in sys.stdin:
    elm = json.loads(content)
    tmp = elm['body']
    tmpmt = {}
    tmpmt['jabber:to'] = elm['to']
    tmpmt['jabber:from'] = elm['from']
    tmpmt['jabber:ts'] = elm['ts']
    tmpmt['jabber:id'] = "{}".format(uuid.uuid4())
    pyail.feed_json_item(tmp, tmpmt, ailfeedertype,
        source_uuid)
```

feeder.py

```
$ cat ~/conti/* | jq . -c | python ./feeder.py
```

Feeding ALL with Conti leaks

- use grep to limit the noise on an instance by only sending interesting bits:
 - PGP keys

```
$ cat ~/conti/* | jq . -c | grep PGP | python ./feeder.py
```

- onion hidden services | grep http:// |
- telegram addresses | grep tg:// |
- bitcoins addresses | egrep
--regexp="[13][a-km-zA-HJ-NP-Z1-9]25,34" |

AIL ecosystem - Challenges and design

ALL ecosystem: Technologies used

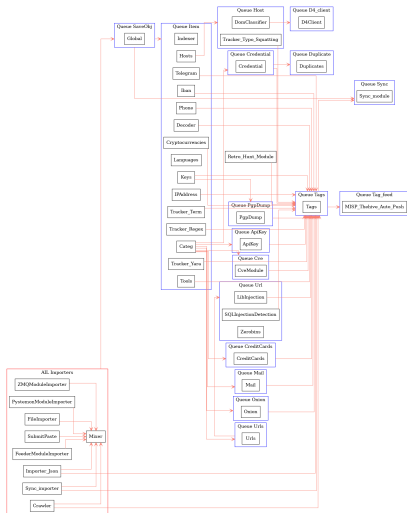
Programming language: Full python3

Databases: Redis and Kvrocks

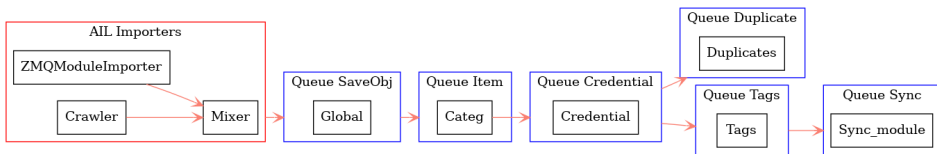
Server: Flask

Data message passing: Redis Set

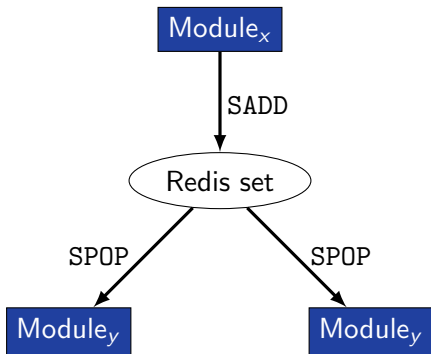
AIL global architecture: Data streaming between module



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



Message consuming

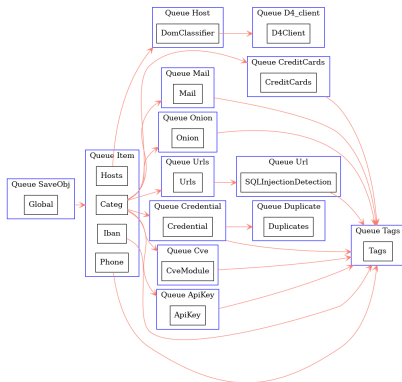


- No message lost nor double processing
- Multiprocessing!

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 configs/modules.cfg accordingly

Writing your own modules - /bin/modules/TemplateModule.py

```
1 from modules.abstract_module import AbstractModule
2
3 class NewModule(AbstractModule):
4
5     def __init__(self):
6         super().__init__()
7         self.logger.info(f'Module {self.module_name} initialized')
8
9     # Do something with the message from the queue
10    def compute(self, message):
11        # Process Message
12
13    # LAUNCH MODULE
14    if __name__ == '__main__':
15        module = NewModule()
16        module.run()
17
18
```

Writing your own Importer - /bin/importer/

```
1 from importer.abstract_importer import AbstractImporter
2 from modules.abstract_module import AbstractModule
3
4 class MyNewImporter(AbstractImporter):
5
6     def __init__(self):
7         super().__init__()
8         # super().__init__(queue=True) # if it's an one-time run importer
9         self.logger.info(f'Importer {self.name} initialized')
10
11     def importer(self, my_var): # import function
12         # Process my_var and get content to import
13         content = GET_MY_CONTENT_TO_IMPORT
14         # if content is not gzipped and/or not b64 encoded,
15         # set gzipped and/or b64 to False
16         message = self.create_message(item_id, content)
17         return message
18         # if it's an one-time run, otherwise create an AIL Module
19         # self.add_message_to_queue(message)
20
21 class MyNewModuleImporter(AbstractModule):
22     def __init__(self):
23         super().__init__() # init module ...
24         # init module ...
25         self.importer = MyNewImporter()
```


Writing your own Importer - /bin/importer/

```
1
2     def get_message(self):
3         return self.importer.importer()
4
5     def compute(self, message):
6         self.add_message_to_queue(message)
7
8 if __name__ == '__main__':
9     module = MyNewModuleImporter()
10    module.run()
11
12    # if it's an one-time run:
13    # importer = MyImporter()
14    # importer.importer(my_var)
15
16
```

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.
- 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.

Implementation Steps in AIL project

- **Gradual changes** in AIL to add required functionalities to support the objectives.
- **Time-memory trade-off** can be challenging to ensure a functional framework.
- Evaluation and integration of new modules in AIL based on time-memory comparisons.
- Semantic aspects are challenging due to the diverse data sources, unstructured data and languages seen.

Ongoing developments

- MISP Importer
- Bloom filter filtering
- Data retention and lifetime management of objects
- MISP modules expansion
- Auto classification of content by set of terms (semantic analysis)
- Improved export stream to third parties software
- Improved indexing relying on Solr, Lucene or other components

Contact

- CIRCL has developed a range of open-source tools for intelligence analysts and incident responders.
- We welcome partnerships and collaboration discussions. Feel free to contact us²³.

²³<mailto:info@circl.lu>

Annexes

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