Data mining Tor, social networks, OSINT with AIL Project

E.102

CIRCL Computer Incident Response Center Luxembourg

MISP Project
https://www.misp-project.org/

March 24, 2022

Introduction

Concepts - Deep Web

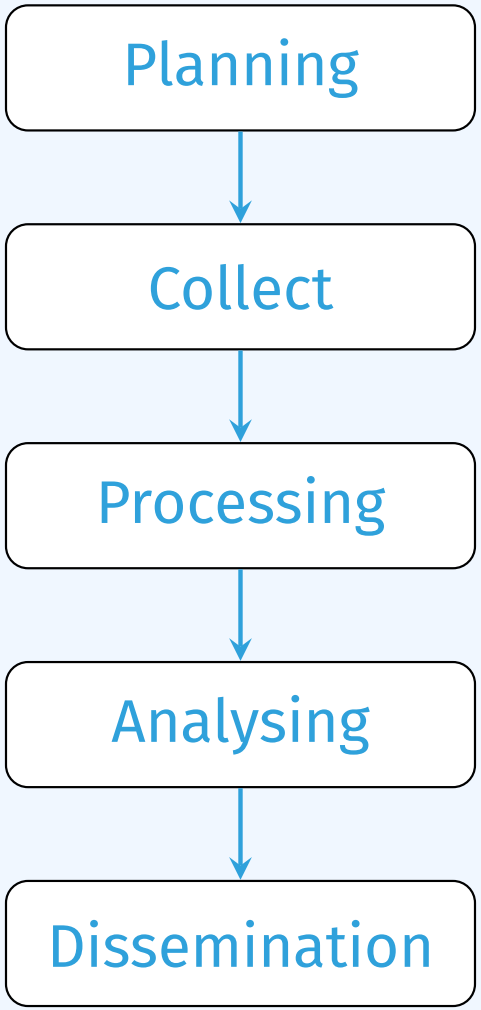
- **Deep Web** is the part of World Wide Web not indexed or directly accessible by standard web search-engines;
- This can be content hidden from **crawlers** by requiring a specific access and this can includes private social media, password-protected forums or content protected by different measures such as paywalls or specific security interface to access the information;
- A large portion of content accessible via Internet is part of the deep web[1].

[1] also called invisible web, hidden web or non-indexed web

Concepts - darknet

- **Darknet** is an overlay network running on top of Internet requiring specific software to access the network and its services;
- Tor, I2P and Freenet are the most commonly used ones. Many are used for hidden services access and some for proxy access to the Internet;
- There are **legitimate use-cases** for such network but also many **illegal or criminal usage**.

Lifecycle of collection and analysis



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 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,
  - OpenGPG 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

Objectives of the session

- Show how to use and extend an open source tool to monitor web pages, pastes, forums and hidden services
- Explain challenges and the design of the AIL open source framework
- Review different **collection mechanisms** and **sources**
- Learn how to create new modules
- Learn how to use, install and start AIL
- **Supporting investigation using the AIL framework** and including it in cyber threat intelligence lifecycle

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 is now a complete project called **ail project**[2].

[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 an archive
- **Monitor** and crawl websites

Support CERT/CSIRTs and Law Enforcement activities

- Proactive investigation: leaks detection
  - List of emails and passwords
  - Leaked database
  - AWS Keys
  - Credit-cards
  - PGP private keys
  - Certificate private keys
- Feed Passive DNS or any passive collection system
- CVE and PoC of vulnerabilities most used by attackers

Support CERT/CSIRTs and Law Enforcement activities

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

Sources of leaks: Paste monitoring

- Example: `https://gist.github.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: `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 ...

Why so many leaks?

- Economical interests (e.g. Adversaries promoting services)
- Ransom model (e.g. To publicly pressure the victims)
- 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[3] 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.

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

Paste monitoring at CIRCL: Statistics

- Monitored paste sites: 27
  - *gist.github.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[4] raised by CIRCL

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

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**
- Tor Crawler (handle cookies authentication)

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

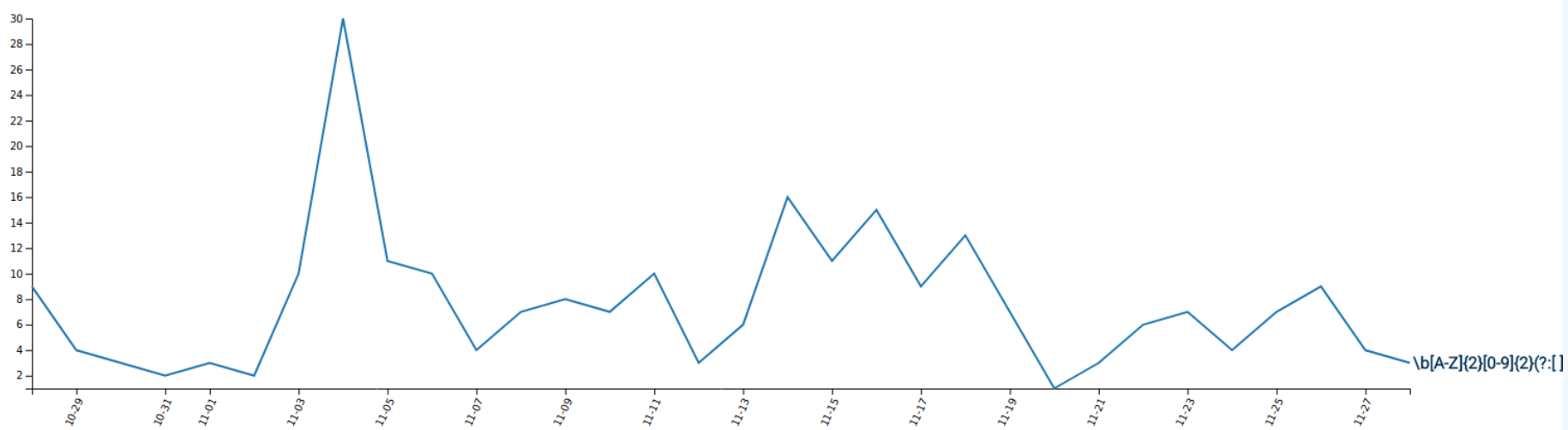
Terms Tracker

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

Terms Tracker

82a87a6a-88f1-4ab1-ba53-1bf15211b4b8

Type	Tracker	Date added	Level	Created by	First seen	Last seen	Tags	Email
regex	\b[A-Z]{2}[0-9]{2}(?:[ ]?[0-9]{4}){4}(?!(?:[ ]?[0-9]){3})(?:[ ]?[0-9]{1,2})?\b	2019/09/12	1	admin@admin.test	2018/08/31	2019/11/28		



yyyy-mm-dd

yyyy-mm-dd

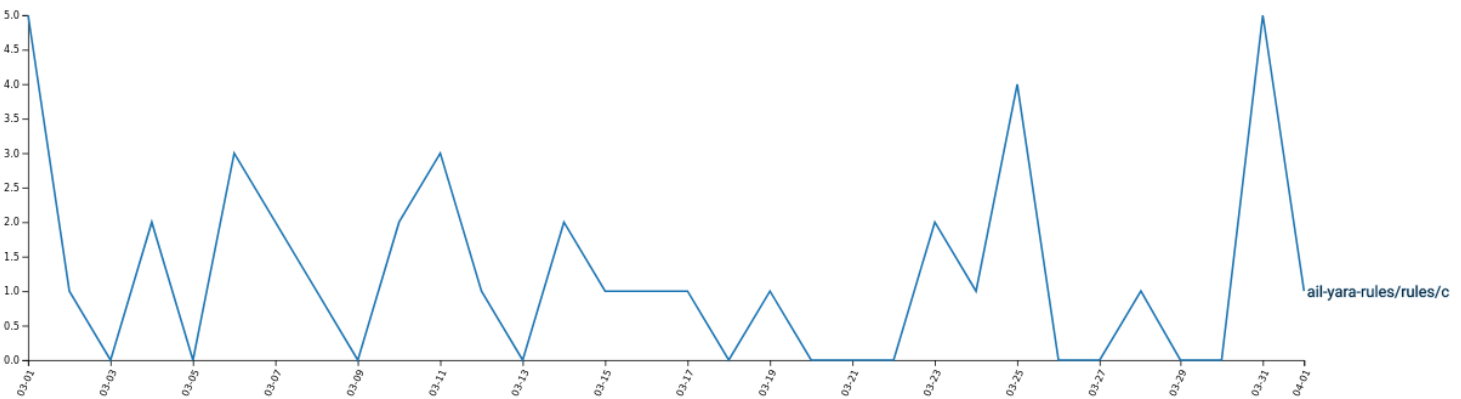
Search Tracked Items

YARA Tracker

Type	Tracker	Date added	Level	Created by	First seen	Last seen	Tags	Email
yara	ail-yara-rules/rules/code/vbscript.yar	2020/09/17	1	admin@admin.test	2020/09/17	2021/04/01		

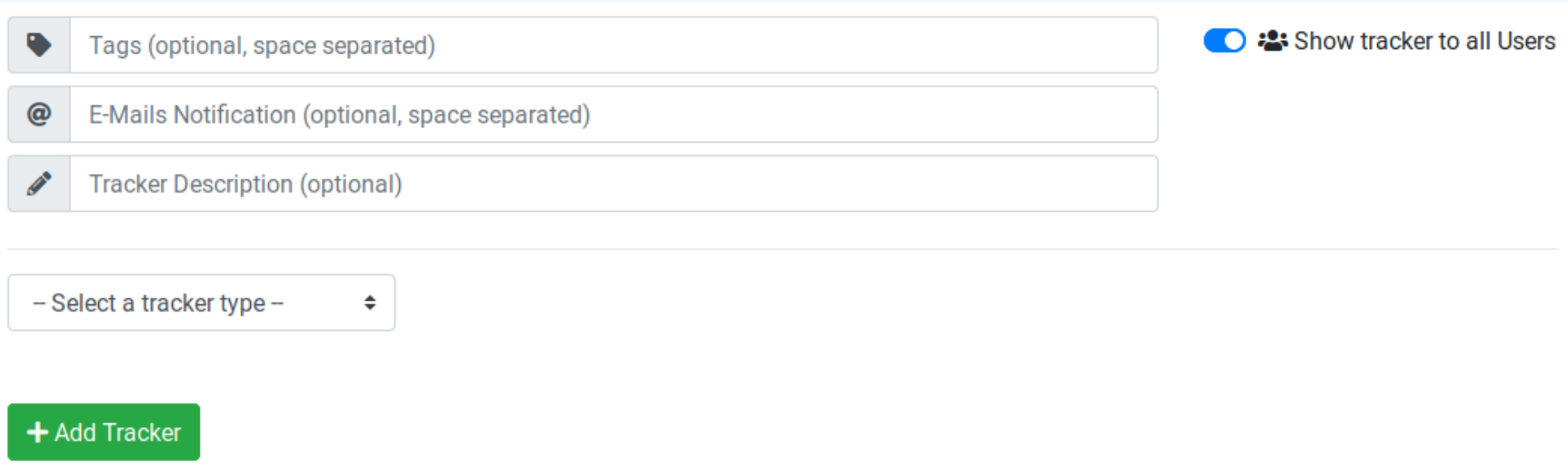
Edit Tracker

```
rule test_vbscript
{
    meta:
        author = "kevthehermit"
        info = "Part of PasteHunter"
        reference = "https://github.com/kevthehermit/PasteHunter"

    strings:
        $a = "Function" nocase wide ascii fullword
        $b = "CreateObject" nocase wide ascii fullword
        $c  = "Wscript" nocase wide ascii fullword
        $d = "As Long" nocase wide ascii fullword
        $e = "run" nocase wide ascii fullword
        $f = "for each" nocase wide ascii fullword
        $g = "end function" nocase wide ascii fullword
        $h = "NtAllocateVirtualMemory" nocase wide ascii fullword
        $i = "NtWriteVirtualMemory" nocase wide ascii fullword


    condition:
        5 of them
}
```



Terms Tracker - Practical part

- **Create and test** your own tracker



Recon and intelligence gathering tools

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

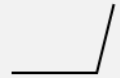
Recon and intelligence gathering tools

```
################################################################################
================================================================================
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
Creation date: 2018-11-21 14:34:34 CLST
                                :34 CLST
```

Decoder

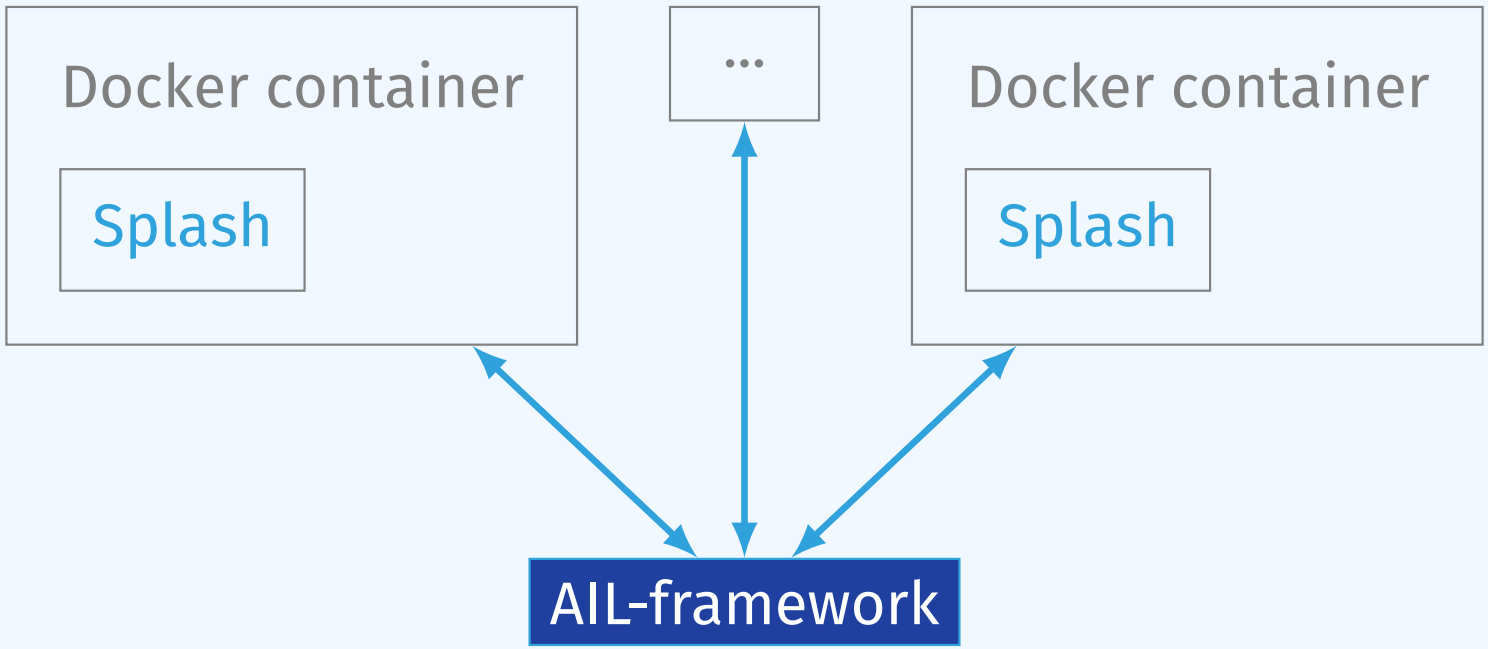
- Search for encoded strings
  - Base64
  - Hexadecimal
  - Binary
- Guess Mime-type
- Correlate paste with decoded items

Decoder:

estimated type	hash	first seen	last seen	nb item	size	Virus Total	Sparkline
application/x-dosexec	c11c2be8d9ba4e86c8effaa411aa6b867ba75abe	2019/11/28	2019/11/28	1	191	Send this file to VT	
application/x-dosexec	a50cba731204ecce193b40178399a250b5ce6f67	2019/11/28	2019/11/28	1	32768	Send this file to VT	
application/x-dosexec	cc5f2f0da71f443ec12ae1b3cb6ab8bad80f22c4	2019/11/28	2019/11/28	1	203	Send this file to VT	
application/x-dosexec	eed67e8fa9cb9a43fea21ae653983a8e0a174f63	2019/11/26	2019/11/28	6	83	Send this file to VT	

Crawler

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



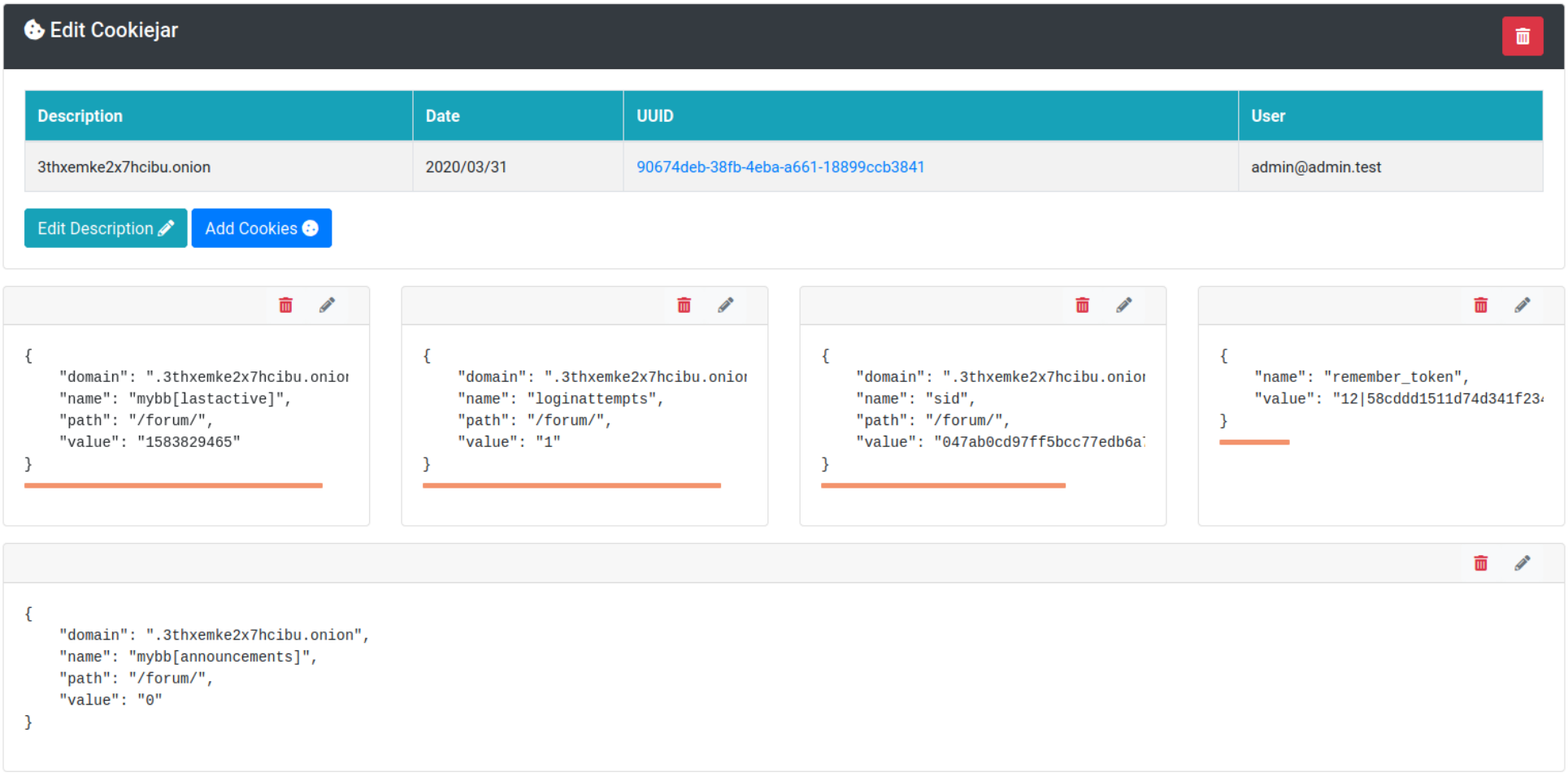
Crawler

How a domain is crawled by default

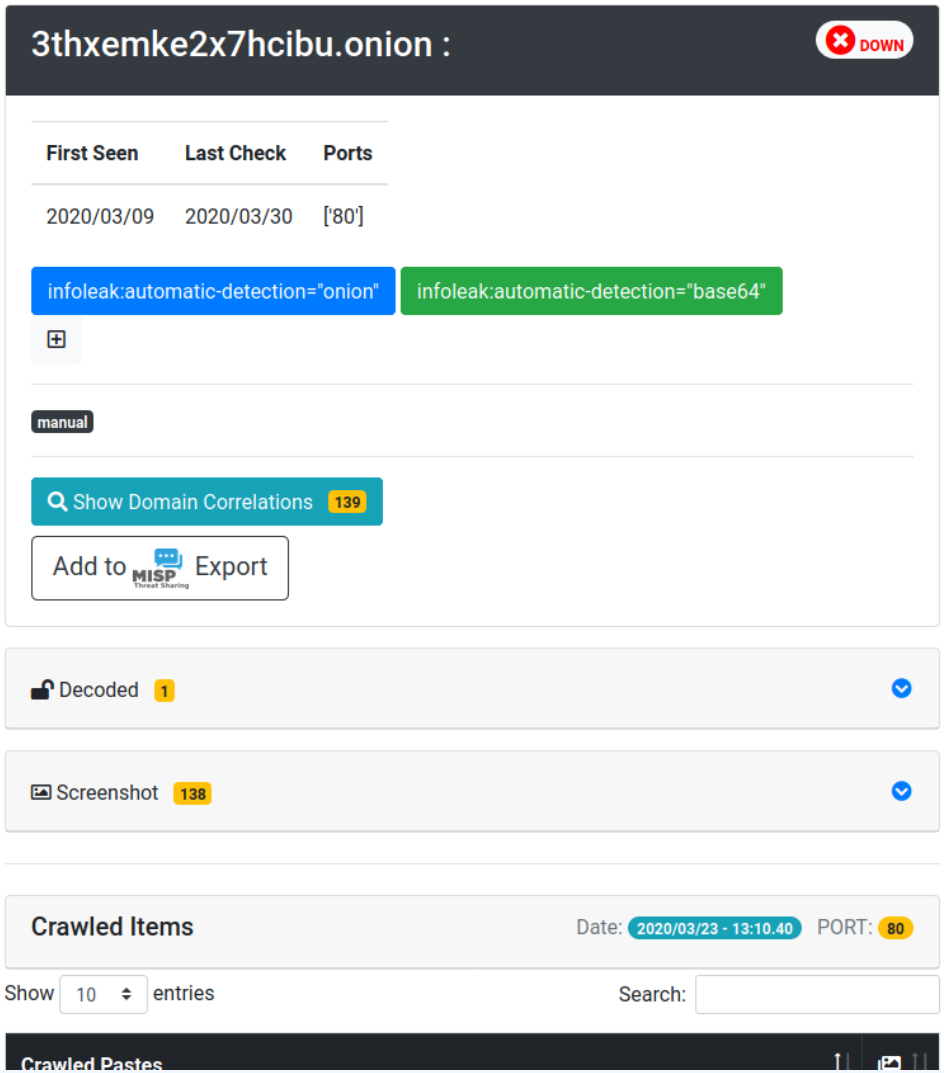
1. Fetch the first url
2. Render javascript (webkit browser)
3. Extract all urls
4. Filter url: keep all url of this domain
5. crawl next url (max depth = 1)

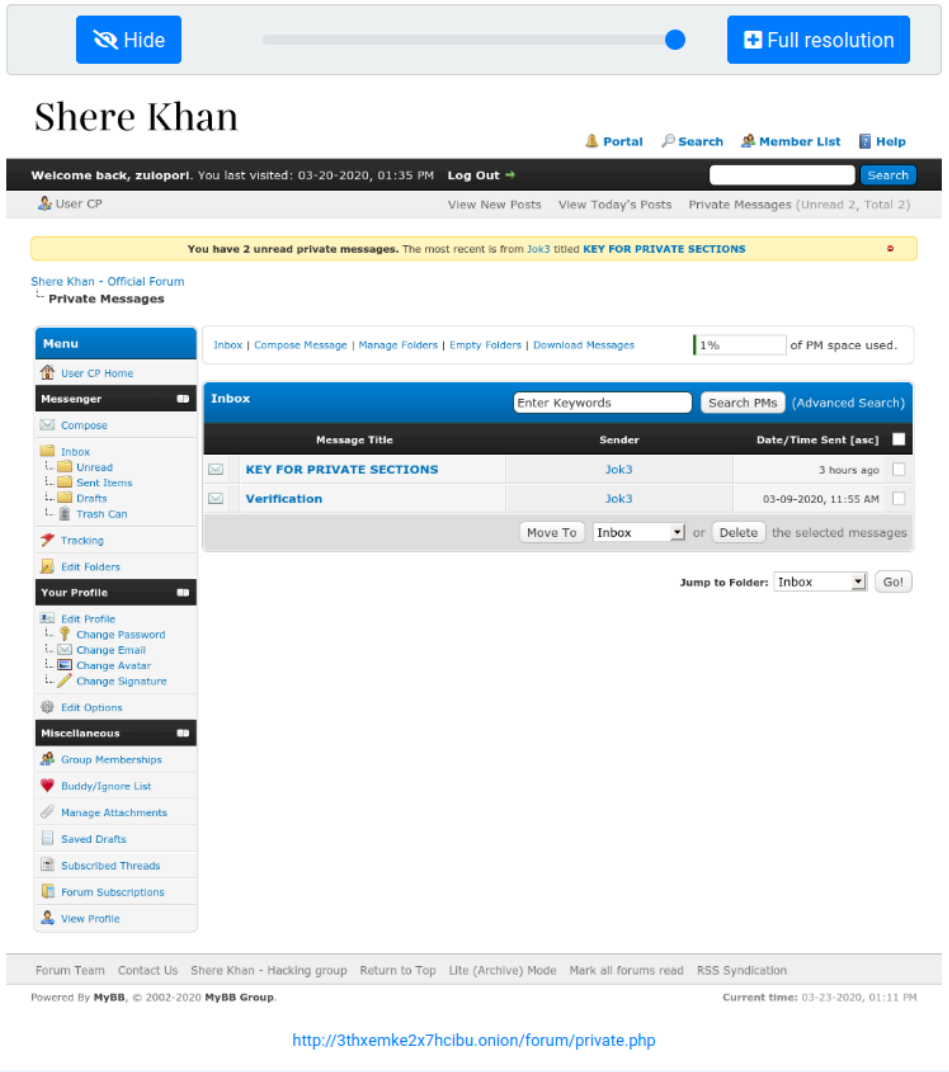
Crawler: Cookiejar

Use your cookies to login and bypass captcha



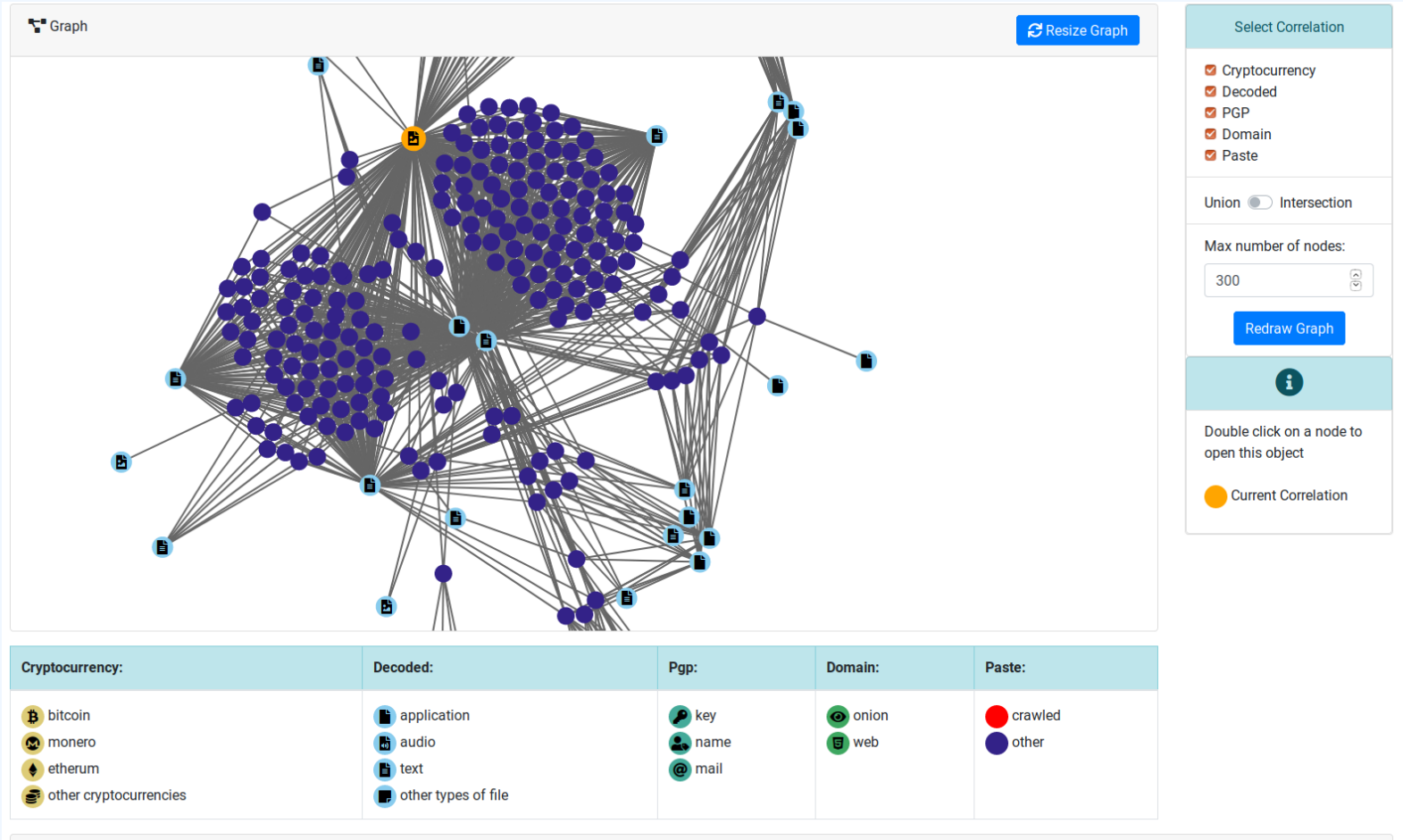
Crawler: Cookiejar





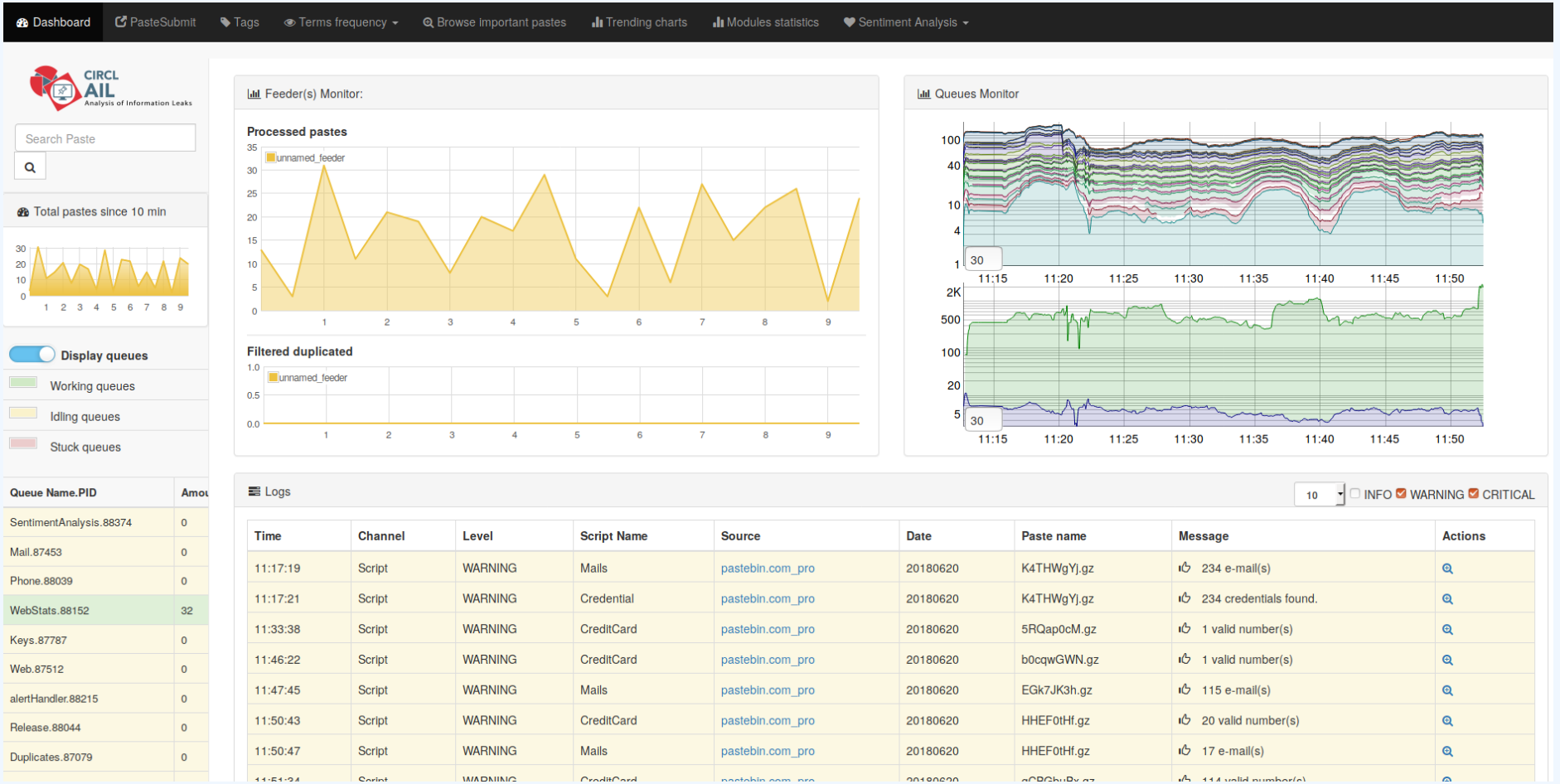
http://3thxemke2x7hcibu.onion/forum/private.php

Correlations and relationship

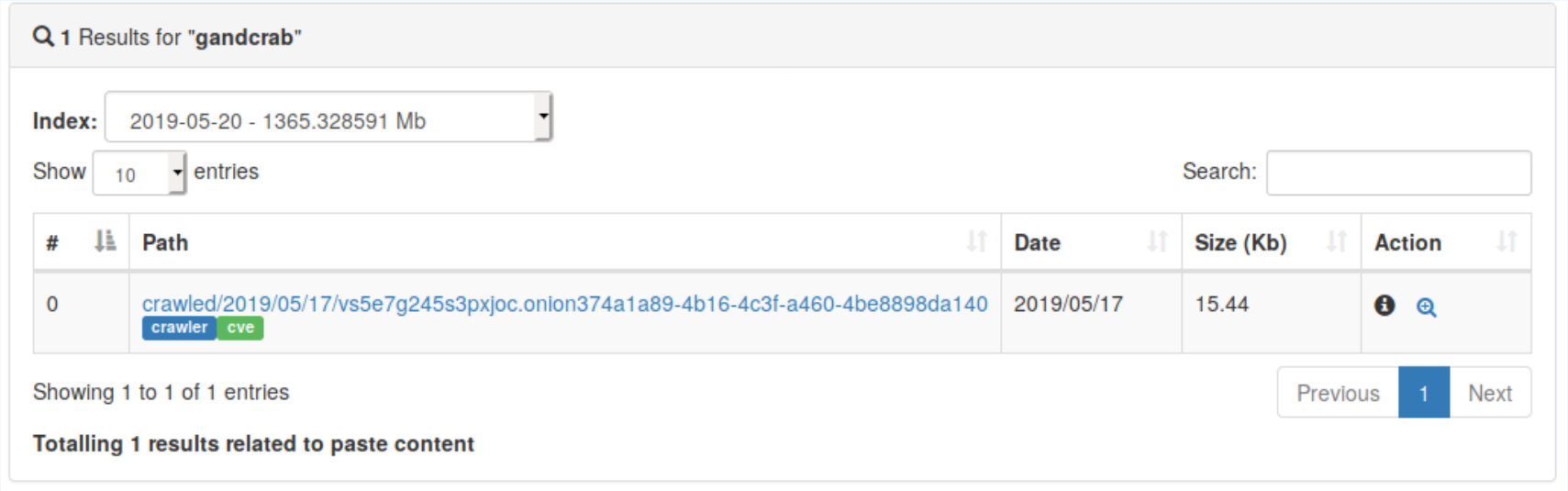


Live demo!

Example: Dashboard



Example: Text search



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 MISP Event

Duplicate list:

Show 10 entries Search:

Hash type	Paste info	Date	Path	Action
['tlsh']	Similarity: [19]%	2019-04-13	archive/pastebin.com_pro/2019/04/13/EbMVR87S.gz	
['tlsh']	Similarity: [10]%	2019-04-11	archive/pastebin.com_pro/2019/04/11/2X5HRVnX.gz	
['tlsh']	Similarity: [23]%	2019-04-25	archive/pastebin.com_pro/2019/04/25/TS2b6M4c.gz	
['tlsh']	Similarity: [14]%	2019-04-17	archive/pastebin.com_pro/2019/04/17/CuS93H7K.gz	
['tlsh']	Similarity: [23]%	2019-04-20	archive/pastebin.com_pro/2019/04/20/AQd0qGVQ.gz	
['tlsh']	Similarity: [20]%	2019-04-20	archive/pastebin.com_pro/2019/04/20/6DDc13b8.gz	
['tlsh']	Similarity: [21]%	2019-05-05	alerts/pastebin.com_pro/2019/05/05/X8nJLzda.gz	
['tlsh']	Similarity: [7]%	2019-04-13	archive/pastebin.com_pro/2019/04/13/Lyp4FVWW.gz	

Showing 1 to 8 of 8 entries Previous 1 Next

Example: Items Metadata (2)

Hash files:

Show 5 entries

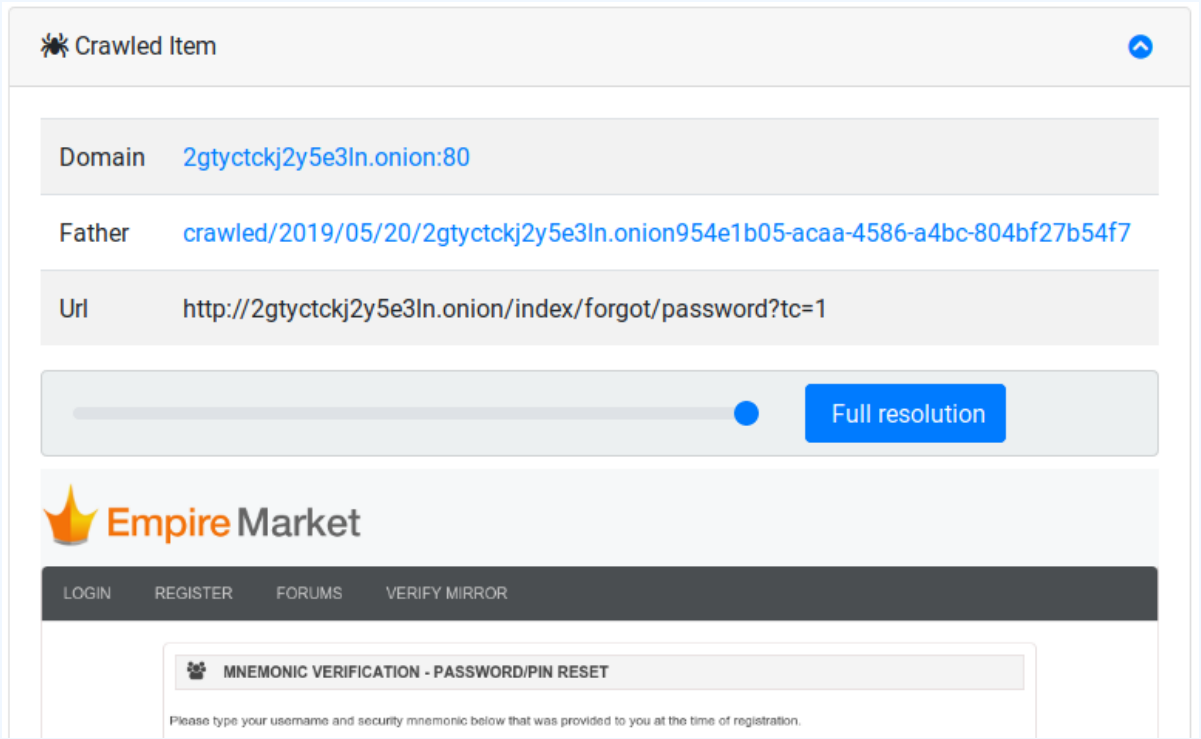
Search:

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)



Example: Browsing content

Content:

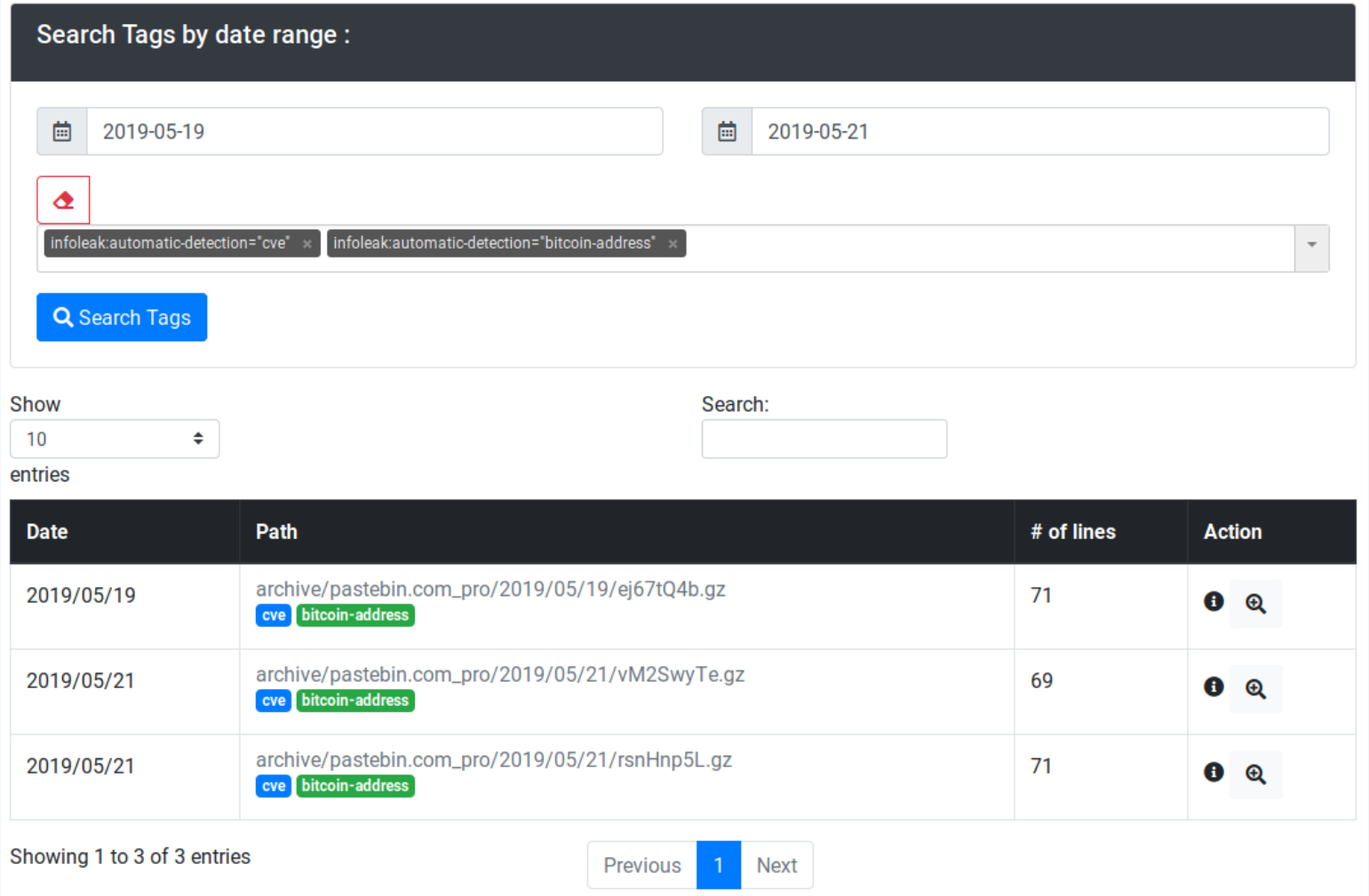
```
http://members2.mofosnetwork.com/access/login/
somosextremos:buddy1990
brazzers_glenn:cocklick
brazzers61:braves01

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: 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
Riptor795:blade7
Domi80:harkonnen
GaggedUK:a1k0chan

http://www.ariellaferrera.com/
```

Example: Search by tags



MISP

MISP Taxonomies

- **Tagging** is a simple way to attach a classification to an event or anattribute.
- **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 [5].
- Can be easily cherry-picked or extended

[5] 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**[6]: Evaluate the degree of identifiability of personal data and the types of pseudonymous data, de-identified data and anonymous data.

[6] Future of Privacy Forum

Taxonomies useful in AIL

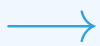
- **tor**: Describe Tor network infrastructure.
- **dark-web**: Criminal motivation on the dark web.
- **copine-scale**[7]: Categorise the severity of images of child sex abuse.

[7] Combating Paedophile Information Networks in Europe

THREAT SHARING AND INCIDENT RESPONSE PLATFORMS

→

Goal: submission to threat sharing and incident response platforms.



1. Use infoleak taxonomy[8]
2. Add your own tags
3. Export AIL objects to MISP core format
4. Download it or Create a MISP Event[9]

[8] https://www.misp-project.org/taxonomies.html
[9] https://www.misp-standard.org/rfc/misp-standard-core.txt

MISP Export

1Gt545E48EPsyTC8voKQDCFfpTkwiuXduw :

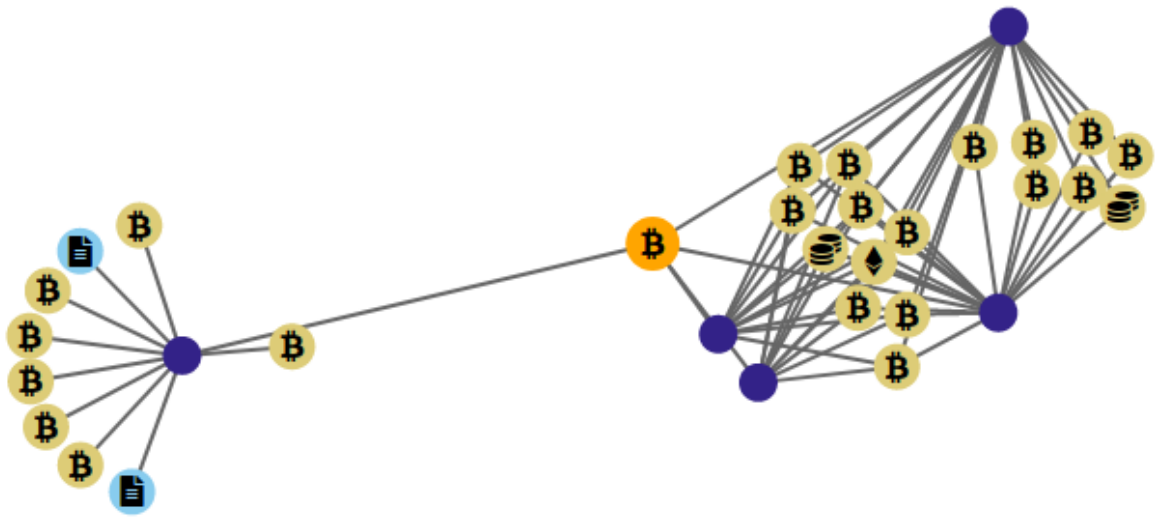
Object type	type	First seen	Last seen	Nb seen
cryptocurrency	bitcoin	2020/01/17	2020/02/20	5

Expand Bitcoin address

Graph

Resize Graph

Add to MISP Export



MISP Export

UP

nttfj36sp47cw2yecop572zjvjeazgazieunllouudplzqt2m5h465yd.onion :

First Seen	Last Check	Ports
2020/02/19	2020/02/19	['80']

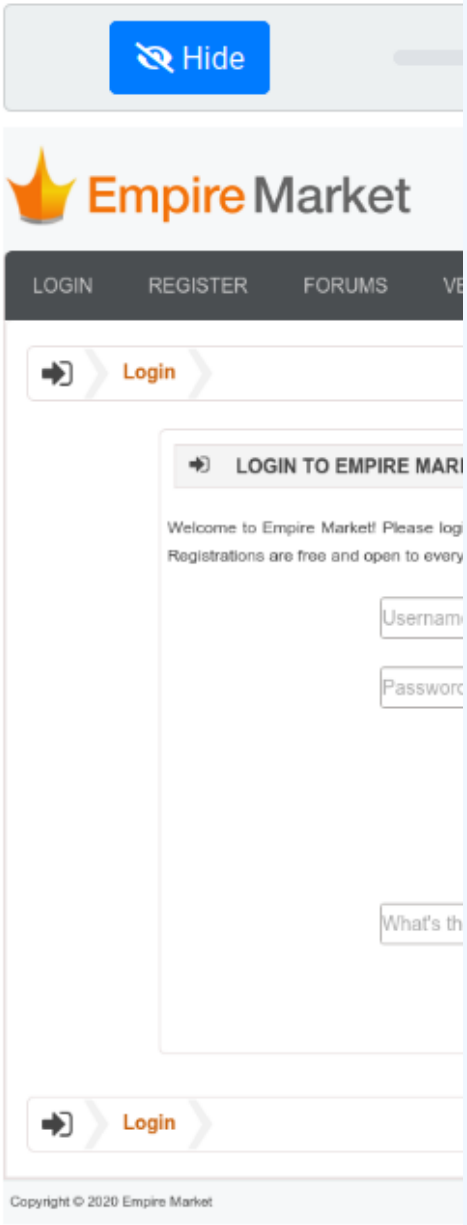
infoleak:automatic-detection="onion"

Last Origin: crawled/2020/02/19/dark.failc126d32a-3ed1-468f-ba24-f2e5956f4035

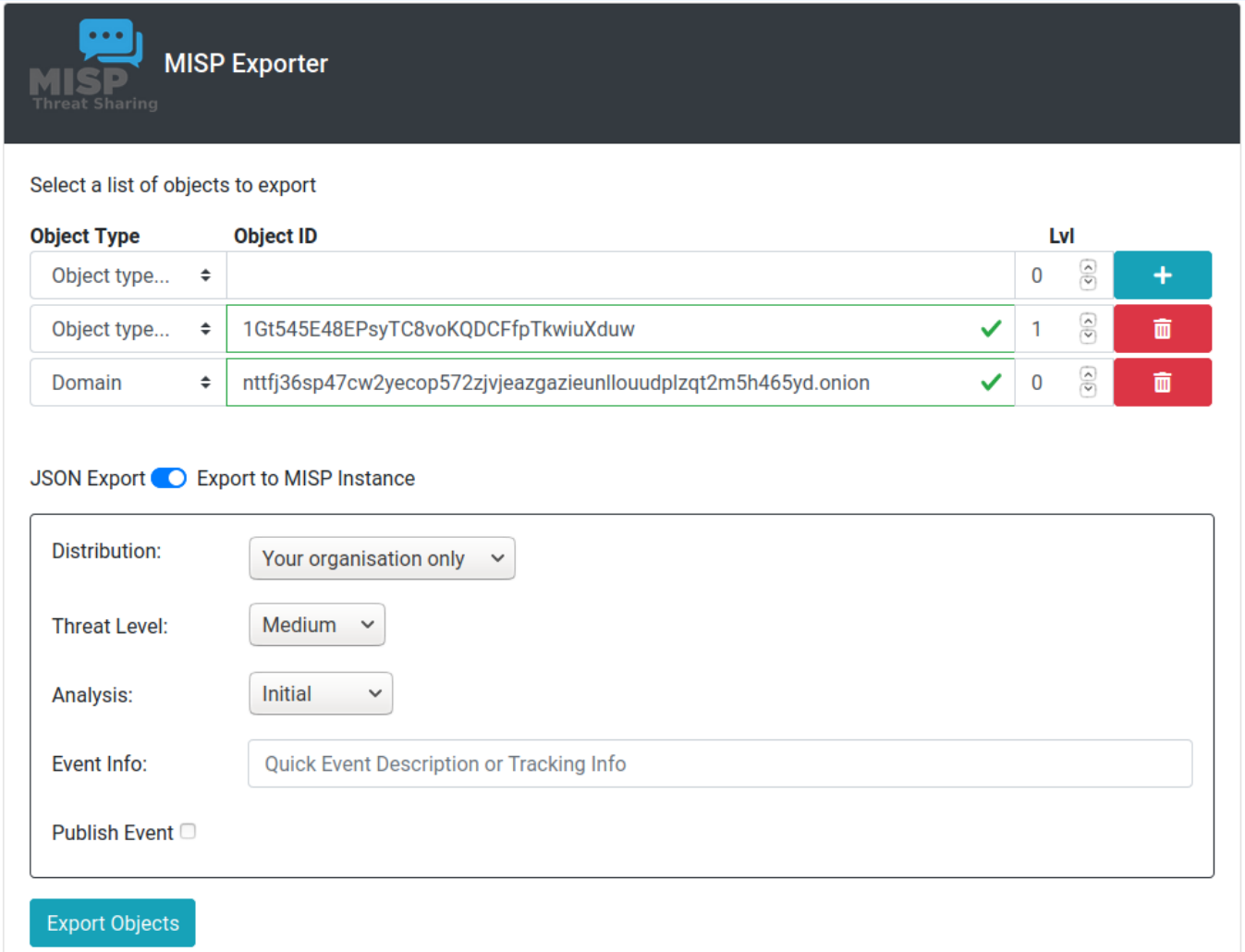
Show Domain Correlations 4

Add to MISP Export

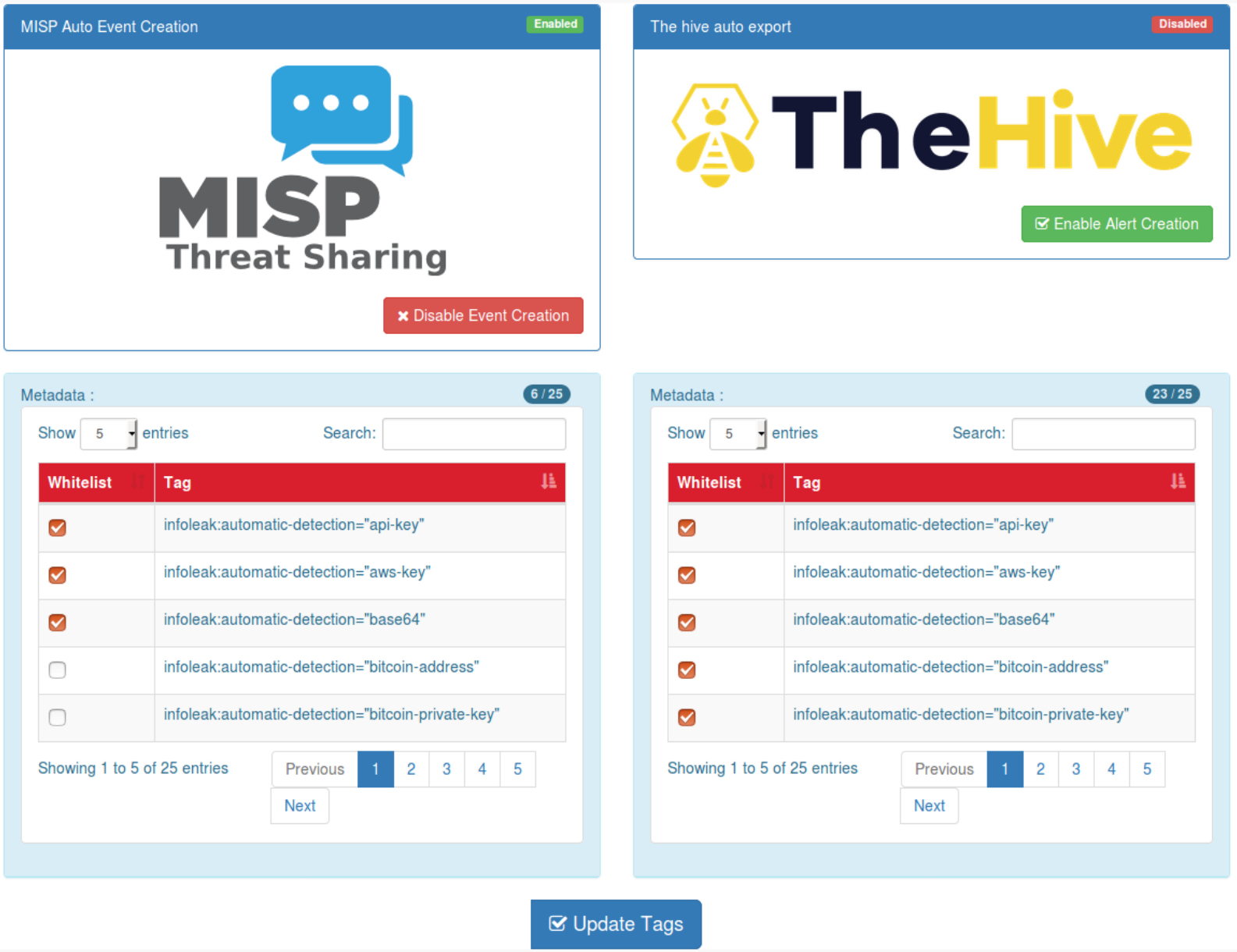
Screenshot 4



MISP Export


MISP Exporter
Threat Sharing
Select a list of objects to export
Object Type
Object ID
Lvl
Object type...
0
Object type...
1Gt545E48EPsyTC8voKQDCFfpTkwiuXduw
1
Domain
nttfj36sp47cw2yecop572zjvjeazgazieunllouudplzqt2m5h465yd.onion
0
JSON Export
Export to MISP Instance
Distribution:
Your organisation only
Threat Level:
Medium
Analysis:
Initial
Event Info:
Quick Event Description or Tracking Info
Publish Event
Export Objects

Automatic submission on tags



API

AIL exposes a ReST API which can be used to interact with the back-end[10].

```
curl https://127.0.0.1:7000/api/v1/get/item/default
        --header "Authorization: iHc1_ChZxj1aXmiFiF1m
        -H "Content-Type: application/json"
        --data @input.json -X POST
```

- AIL API is currently covering 60% of the functionality of back-end.

[10] 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
```

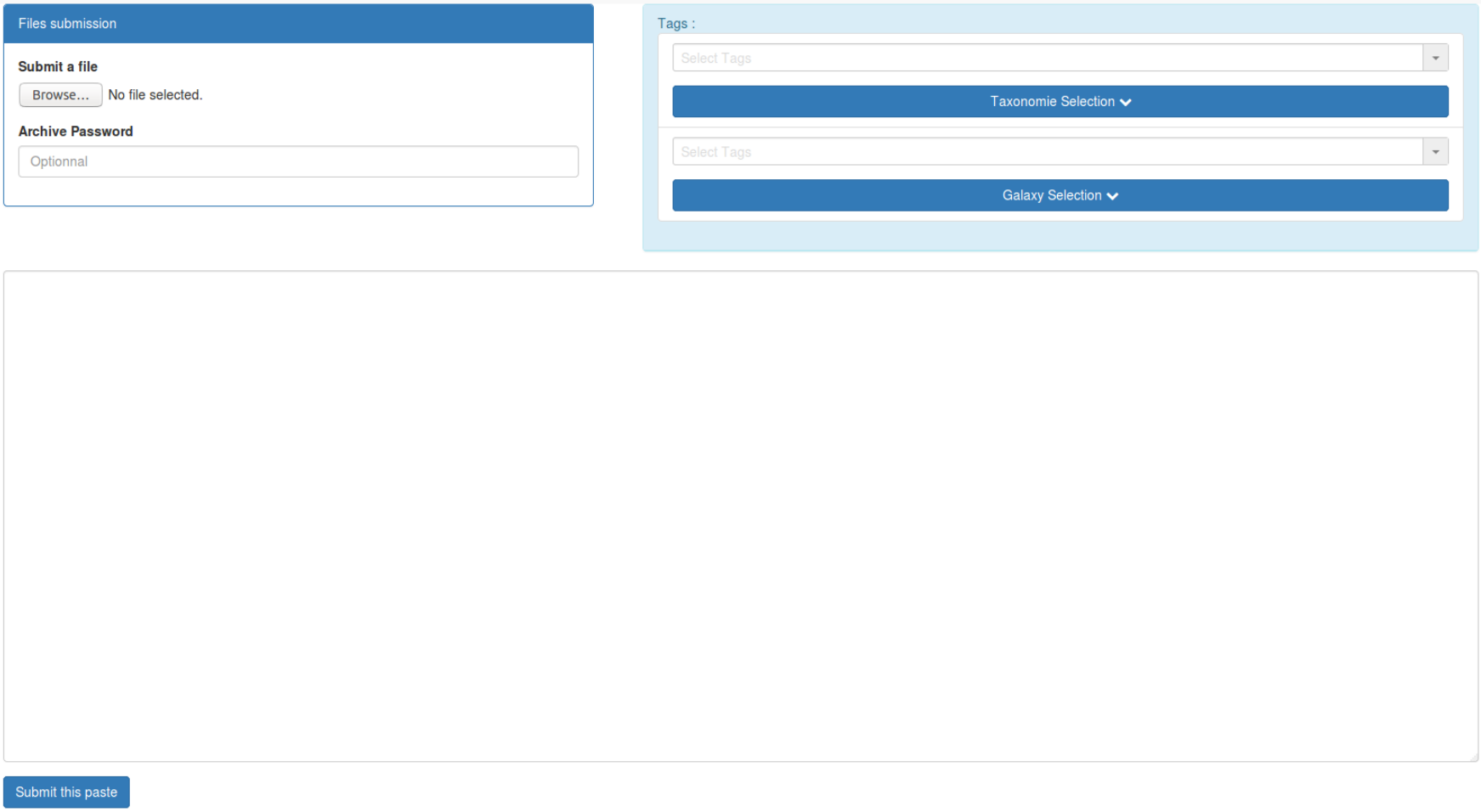
Feeding the framework

Feeding AIL

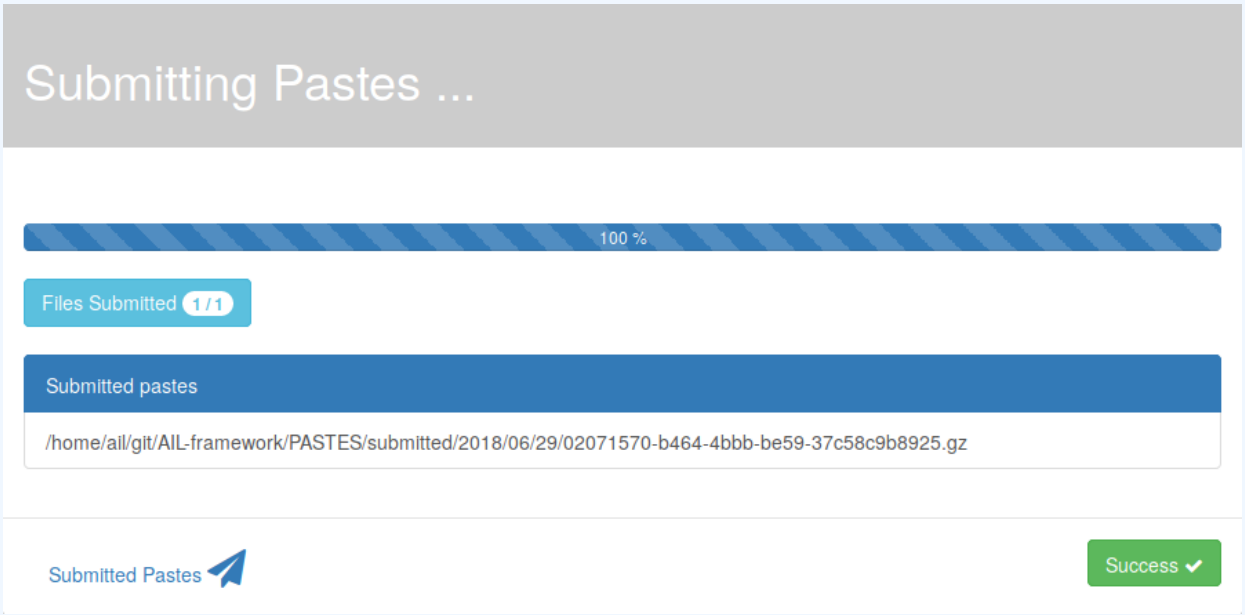
There are different way to feed AIL with data:

1. Setup *pystemon* and use the custom feeder
   - *pystemon* will collect items for you
2. Use the new JSON Feeder (twitter)
3. Feed your own data using the API or the `import_dir.py` script
4. Feed your own file/text using the UI (`Submit section`)

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 - API

api/v1/import/item

```
{
  "type": "text",
  "tags": [
    "infoleak:analyst-detection=\"private-key\""
  ],
  "text": "text to import"
}
```

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

/!\ requirements:

- Each file to be fed must be of a reasonable size:
  - $\sim$ 3 Mb / file is already large
  - This is because some modules are doing regex matching
  - 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 `configs/core.cfg`
   - In the file `configs/core.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`