

MISP FEEDS - A SIMPLE AND SECURE APPROACH TO GENERATE, SELECT AND COLLECT INTELLIGENCE

PROVIDING READY-TO-USE THREAT INTELLIGENCE IN

CIRCL / TEAM MISP PROJECT
TLP:WHITE

[HTTP://WWW.MISP-PROJECT.ORG/](http://www.misp-project.org/)
TWITTER: [@MISPPROJECT](https://twitter.com/MISPPROJECT)

MISP PROJECT

MISP feeds - A simple and secure approach to generate,
select and collect intelligence

2024-04-15

MISP FEEDS - A SIMPLE AND SECURE
APPROACH TO GENERATE, SELECT AND
COLLECT INTELLIGENCE

PROVIDING READY-TO-USE THREAT INTELLIGENCE IN

CIRCL / TEAM MISP PROJECT
TLP:WHITE

[HTTP://WWW.MISP-PROJECT.ORG/](http://www.misp-project.org/)
TWITTER: [@MISPPROJECT](https://twitter.com/MISPPROJECT)

MISP PROJECT

MISP Feeds provide a way to

- **Exchange information via any transports** (e.g. HTTP, TLS, USB keys)
- Preview events along with their attributes, objects
- Select and import events
- **Correlate attributes using caching**

MISP Feeds have the following advantages

- Feeds work without the need of MISP synchronisation (reducing attack surface and complexity to a static directory with the events)
- **Feeds can be produced without a MISP instance** (e.g. security devices, honeypot sensors)

MISP feeds - A simple and secure approach to generate, select and collect intelligence

MISP Feed - Basics

2024-04-15

- MISP Feeds provide a way to
- Exchange information via any transports (e.g. HTTP, TLS, USB keys)
 - Preview events along with their attributes, objects
 - Select and import events
 - Correlate attributes using caching
- MISP Feeds have the following advantages
- Feeds work without the need of MISP synchronisation (reducing attack surface and complexity to a static directory with the events)
 - Feeds can be produced without a MISP instance (e.g. security devices, honeypot sensors)

Feeds can be used to produce output from various security devices

- By default, MISP is bundled with ~50 default feeds (MISP feeds, CSV or freetext feeds) which are not enabled by default and described in a simple JSON file¹.
- The feeds include CIRCL OSINT feed but also feeds like abuse.ch, Tor exit nodes or many more².

Feeds

Generate feed lookup caches or fetch feed data (enabled feeds only)

Cache all feeds Cache freetext/CSV feeds Cache MISP feeds Fetch and store all feed data

← previous next →

Default feeds		Custom Feeds		All Feeds	Enabled Feeds											
Id	Enabled	Name	Feed Format	Provider	Input	Uri	Headers	Target	Publish	Delta Merge	Override IDS	Distribution	Tag	Lookup Visible	Caching	Actions
1	✓	CIRCL OSINT Feed MISP	MISP Feed	CIRCL	network	https://www.circl.lu/doc/misp/feed-osint						All communities	CIRCL OSINT Feed	✗	Age: 3m	🔍 🗑️ 🔄 📄
2	✓	The Botrij.eu Data MISP	MISP Feed	Botrij.eu	network	http://www.botrij.eu/data/feed-osint						All communities	FEED:KOEN	✗	Not cached	🔍 🗑️ 🔄 📄
18	✗	InThreat OSINT Feed MISP	MISP Feed	InThreat	network	https://feeds.inthreat.com/osint/misp/					Your organisation only	Your organisation only	osint:source-type="block-or-filter-list"	✗	Not cached	🔍 🗑️ 🔄 📄

¹<https://github.com/MISP/MISP/blob/2.4/app/files/feed-metadata/defaults.json>

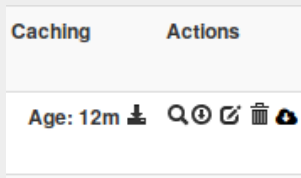
²<http://www.misp-project.org/feeds/>

2024-04-15

MISP feeds - A simple and secure approach to generate, select and collect intelligence

└─ Feed - Overview





- Cache feed attributes for correlation (not imported but visible in MISP)
- Disable feed
- Explore remote events
- Fetch all events (imported in MISP as event)
- Edit the feed configuration (e.g. authentication, URL,...)
- Remove feed
- Download feed metadata (to share feed details)

MISP feeds - A simple and secure approach to generate, select and collect intelligence

Feed - Operations

2024-04-15



- Cache feed attributes for correlation (not imported but visible in MISP)
- Disable feed
- Explore remote events
- Fetch all events (imported in MISP as event)
- Edit the feed configuration (e.g. authentication, URL,...)
- Remove feed
- Download feed metadata (to share feed details)

feed generator fetches events (matching some filtering) from a MISP instance and construct the manifest (defined in *MISP core format*) needed to export data.

Particularly,

- Used to generate the **CIRCL OSINT feed**
- Export events as json based on tags, organisation, events, ...
- Automatically update the dumps and the metadata file
- Comparable to a lightweight **TAXII interface**

MISP feeds - A simple and secure approach to generate, select and collect intelligence

2024-04-15

└─ Feed - Creation using PyMISP feed generator

Feed generator fetches events (matching some filtering) from a MISP instance and construct the manifest (defined in *MISP core format*) needed to export data.

Particularly,

- Used to generate the **CIRCL OSINT feed**
- Export events as json based on tags, organisation, events, ...
- Automatically update the dumps and the metadata file
- Comparable to a lightweight **TAXII interface**

Feed generator - CONFIGURATION FILE

```
1 url = 'your/misp/url'  
2 key = 'YourAPIKey'  
3 ssl = True  
4 outputdir = 'output_directory'  
5  
6 filters = {  
7     'tag': 'tlp:white|feed-export|!privint',  
8     'org': 'CIRCL'  
9 }  
10 # the above would generate a feed for all events created by CIRCL,  
    tagged tlp:white and/or feed-export but exclude anything  
    tagged privint  
11  
12 valid_attribute_distribution_levels = ['0', '1', '2', '3', '4', '5'  
    ,']  
13 # 0: Your Organisation Only  
14 # 4: Sharing Group  
15 # 5: Inherit Event  
16
```

2024-04-15
MISP feeds - A simple and secure approach to generate,
select and collect intelligence

└─ Feed generator - configuration file

```
url = "your/misp/url"  
key = "YourAPIKey"  
ssl = True  
outputdir = "output_directory"  
  
filters = {  
    'tag': 'tlp:white|feed-export|!privint',  
    'org': 'CIRCL'  
}  
  
# the above would generate a feed for all events created by CIRCL,  
tagged tlp:white and/or feed-export but exclude anything  
tagged privint  
  
valid_attribute_distribution_levels = ["0", "1", "2", "3", "4", "5"  
    ,]  
# 0: Your Organisation Only  
# 4: Sharing Group  
# 5: Inherit Event
```

The PyMISP feed generator is great but may be inadequate or inefficient:

- Batch import of attributes/objects
- Data producer doesn't have a MISP instance at hand and only wants to **produce a directly consumable feed**:



2024-04-15

MISP feeds - A simple and secure approach to generate, select and collect intelligence

└ Real-time Feed generator - Purpose

The PyMISP feed generator is great but may be inadequate or inefficient:

- Batch import of attributes/objects
- Data producer doesn't have a MISP instance at hand and only wants to **produce a directly consumable feed**:



- `generator.py` exposes a class allowing to generate a MISP feed in real-time
- Each items can be appended on daily generated events

Example:

```
1 # Init generator
2 generator = FeedGenerator()
3
4 # Adding an attribute to the daily event
5 attr_type = "ip-src"
6 attr_value = "8.8.8.8"
7 additional_data = {}
8 generator.add_attribute_to_event(attr_type,
9                                 attr_value,
10                                **additional_data)
```

MISP feeds - A simple and secure approach to generate, select and collect intelligence

└ Real-time Feed generator - Usage

- `generator.py` exposes a class allowing to generate a MISP feed in real-time
- Each items can be appended on daily generated events

Example:

```
# Init generator
generator = FeedGenerator()
# Adding an attribute to the daily event
attr_type = "ip-src"
attr_value = "8.8.8.8"
additional_data = {}
generator.add_attribute_to_event(attr_type,
                                attr_value,
                                **additional_data)
```

2024-04-15


```
1 # Adding a MISP object (cowrie) to the daily event
2 obj_name = "cowrie"
3 obj_data = {
4     "session": "session_id",
5     "username": "admin",
6     "password": "admin",
7     "protocol": "telnet"
8 }
9 generator.add_object_to_event(obj_name, **obj_data)
```

2024-04-15
MISP feeds - A simple and secure approach to generate, select and collect intelligence

└─ Real-time Feed generator - Usage (2)

```
# Adding a MISP object (cowrie) to the daily event
obj_name = "cowrie"
obj_data = {
    "session": "session_id",
    "username": "admin",
    "password": "admin",
    "protocol": "telnet"
}
generator.add_object_to_event(obj_name, **obj_data)
```

List Feeds

Add Feed

Import Feeds from JSON

Feed overlap analysis matrix

Export Feed settings

Add MISP Feed

Add a new MISP feed source.

Enabled

Lookup Visible

Name

Feed name

Provider

Name of the content provider

Source Format

Network

Url

URL of the feed

Source Format

MISP Feed

Any headers to be passed with requests (for example: Authorization)

Line break separated list of headers in the "headname: value" format

Add Basic Auth

Distribution

All communities

Default Tag

None

Filter rules:

Modify

Add

- Enabled
- Lookup visible
- Name
- Provider
- Source Format
- Url
- Source Format
- Headers
- Distribution
- Default Tag
- Filter rules

2024-04-15

MISP feeds - A simple and secure approach to generate, select and collect intelligence

└ Adding custom feed to MISP

ADD MISP Feed

Enabled

Lookup visible

Name

Provider

Source Format

Url

Source Format

Headers

Distribution

Default Tag

Filter rules



- <https://github.com/MISP/PyMISP>
- <https://github.com/MISP/>
- We welcome new functionalities and pull requests.

MISP feeds - A simple and secure approach to generate, select and collect intelligence

2024-04-15

└─ Q&A



- <https://github.com/MISP/PyMISP>
- <https://github.com/MISP/>
- We welcome new functionalities and pull requests.