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/
Twitter: *@MISPProject*

CIISI-IE Dublin 2024

MISP Feed - Basics

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

2024-07-08

└─MISP Feed - Basics

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 file[1].
- The feeds include CIRCL OSINT feed but also feeds like abuse.ch, Tor exit nodes or many more [2].

---

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

[2]http://www.misp-project.org/feeds/

MISP feeds - A simple and secure approach to generate, select and collect intelligence
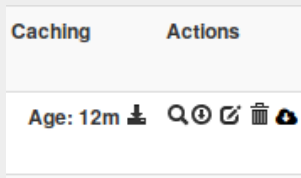
2024-07-08

└─Feed - Overview

Feed - Operations

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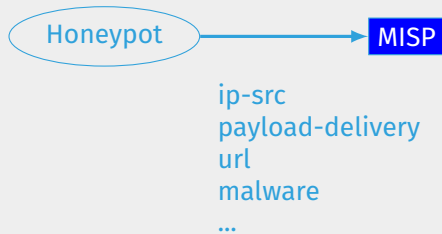
MISP feeds - A simple and secure approach to generate, select and collect intelligence

2024-07-08

└─Feed generator - configuration file

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

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

Honeypot → MISP

ip-src
payload-delivery
url
malware
...

- 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)
```

Real-time FEED GENERATOR - USAGE (2)

```python
# 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)
```

8

Adding custom feed to MISP

Add MISP Feed

List Feeds
Add Feed
Import Feeds from JSON
Feed overlap analysis matrix
Export Feed settings

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 "headername: 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

Q&A

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