1 MISP Training Slide Decks

MISP[1] is a threat intelligence platform for gathering, sharing, storing and correlating Indicators of Compromise of targeted attacks, threat intelligence, financial fraud information, vulnerability information or even counter-terrorism information.

This document includes the slides which are the support materials used for MISP trainings. The slides are licensed under CC-BY-SA license which allows you to freely use, remixes and share-alike the slides while still mentioning the contributors under the same conditions.

2 Contributors

- Steve Clement `https://github.com/SteveClement`
- Alexandre Dulaunoy `https://github.com/adulau`
- Andras Iklody `https://github.com/iglocska`
- Sami Mokaddem `https://github.com/mokaddem`
- Raphaël Vinot `https://github.com/rafiot`
- Gerard Wagener `https://github.com/haegardev`

3 Acknowledgment

The MISP project is co-financed and resource supported by CIRCL Computer Incident Response Center Luxembourg[2] and co-financed by a CEF (Connecting Europe Facility) funding under CEF-TC-2016-3 - Cyber Security as *Improving MISP as building blocks for next-generation information sharing.*

Co-financed by the Connecting Europe Facility of the European Union

CIRCL
Computer Incident
Response Center
Luxembourg

[1] `https://www.misp-project.org/`
[2] `https://www.circl.lu/`

An Introduction to Cybersecurity Information Sharing

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

Agenda

- (10:00 - 11:30) Introduction to Information Sharing with MISP
- (11:30 - 11:40) Coffee break
- (11:40 - 13:00) User perspective - diving into MISP functionalities and integration
- (13:00 - 14:00) **Lunch Break**
- (14:00 - 15:00) Administrating your MISP instance
- (15:00 - 15:45) Building your information sharing communities - CSIRT and financial sectors
- (15:45 - 16:45) Modules and extending MISP (taxonomies, objects and galaxies)
- (16:45 - 17:15) Future - Sharing Ideas

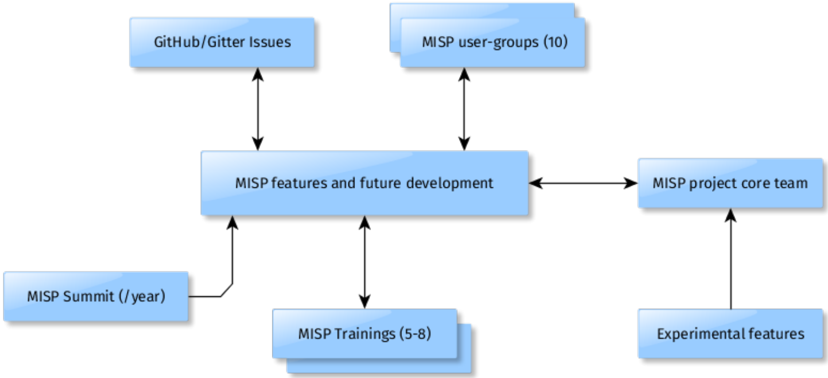
MISP and starting from a practical use-case

- During a malware analysis workgroup in 2012, we discovered that we worked on the analysis of the same malware.
- We wanted to share information in an easy and automated way **to avoid duplication of work**.
- Christophe Vandeplas (then working at the CERT for the Belgian MoD) showed us his work on a platform that later became MISP.
- A first version of the MISP Platform was used by the MALWG and **the increasing feedback of users** helped us to build an improved platform.
- MISP is now **a community-driven development**.

Development based on practical user feedback

- There are many different types of users of an information sharing platform like MISP:
    - **Malware reversers** willing to share indicators of analysis with respective colleagues.
    - **Security analysts** searching, validating and using indicators in operational security.
    - **Intelligence analysts** gathering information about specific adversary groups.
    - **Law-enforcement** relying on indicators to support or bootstrap their DFIR cases.
    - **Risk analysis teams** willing to know about the new threats, likelyhood and occurences.
    - **Fraud analysts** willing to share financial indicators to detect financial frauds.

MISP model of governance



Many objectives from different user-groups

- Sharing indicators for a **detection** matter.
  - 'Do I have infected systems in my infrastructure or the ones I operate?'
- Sharing indicators to **block**.
  - 'I use these attributes to block, sinkhole or divert traffic.'
- Sharing indicators to **perform intelligence**.
  - 'Gathering information about campaigns and attacks. Are they related? Who is targeting me? Who are the adversaries?'
- $\rightarrow$ These objectives can be conflicting (e.g. False-positives have different impacts)

Sharing Difficulties

- Sharing difficulties are not really technical issues but often it's a matter of **social interactions** (e.g. **trust**).
- Legal restriction[1]
  - "Our legal framework doesn't allow us to share information."
  - "Risk of information-leak is too high and it's too risky for our organization or partners."
- Practical restriction
  - "We don't have information to share."
  - "We don't have time to process or contribute indicators."
  - "Our model of classification doesn't fit your model."
  - "Tools for sharing information are tied to a specific format, we use a different one."

[1] `https://www.misp-project.org/compliance/`

MISP Project Overview

MISP Threat Sharing

Galaxy

warning-lists

Taxonomies

modules (import, export, enrichment)

- The **core project**[a] (PHP/Python3) supports the backend, API & UI.
- Modules (Python3) expand MISP functionalities.
- Taxonomies (JSON) to add categories & global tagging.
- Warning-lists (JSON) help analysts to detect potential false-positives.
- Galaxy (JSON) to add threat-actors, tools or "intelligence".
- Objects (JSON) to allow for templated composition of security related atomic points of information.

MISP features

- MISP[2] is a threat information sharing free & open source software.
- MISP has **a host of functionalities** that assist users in creating, collaborating & sharing threat information - e.g. flexible sharing groups, **automatic correlation**, free-text import helper, event distribution & proposals.
- Many export formats which support IDSes / IPSes (e.g. Suricata, Bro, Snort), SIEMs (eg CEF), Host scanners (e.g. OpenIOC, STIX, CSV, yara), analysis tools (e.g. Maltego), DNS policies (e.g. RPZ).
- A rich set of MISP modules[3] to add expansion, import and export functionalities.

[2] `https://github.com/MISP/MISP`

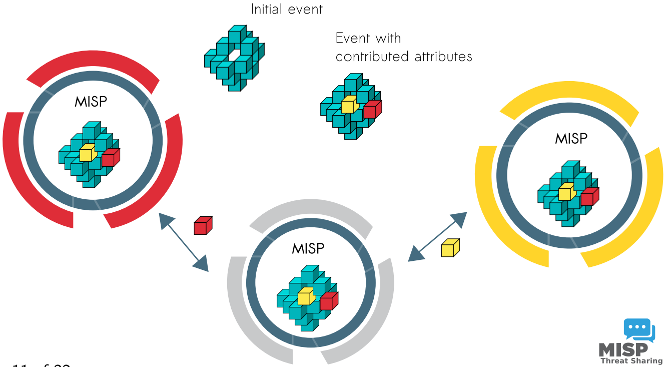
[3] `https://www.github.com/MISP/misp-modules`

Communities using MISP

- Communities are groups of users sharing within a set of common objectives/values.
- CIRCL operates multiple MISP instances with a significant user base (more than 950 organizations with more than 2400 users).
- **Trusted groups** running MISP communities in island mode (air gapped system) or partially connected mode.
- **Financial sector** (banks, ISACs, payment processing organizations) use MISP as a sharing mechanism.
- **Military and international organizations** (NATO, military CSIRTs, n/g CERTs,...).
- **Security vendors** running their own communities (e.g. Fidelis) or interfacing with MISP communities (e.g. OTX).

MISP core distributed sharing functionality

- MISPs' core functionality is sharing where everyone can be a consumer and/or a contributor/producer."
- Quick benefit without the obligation to contribute.
- Low barrier access to get acquainted to the system.



Events, Objects and Attributes in MISP

- MISP events are encapsulations for contextually linked information
- MISP attributes[4] initially started with a standard set of "cyber security" indicators.
- MISP attributes are purely **based on usage** (what people and organizations use daily).
- Evolution of MISP attributes is based on practical usage & users (e.g. the addition of **financial indicators** in 2.4).
- MISP objects are attribute compositions describing points of data using many facets, constructed along the lines of community and user defined templates.
- Galaxies granularly contextualise, classify & categorise data based on **threat actors**, **preventive measures**, tools used by adversaries.

[4] attributes can be anything that helps describe the intent of the event package from indicators, vulnerabilities or any relevant information

Terminology about Indicators

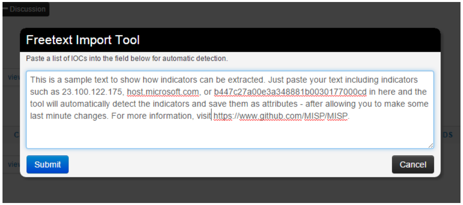
- Indicators[5]
  - Indicators contain a pattern that can be used to detect suspicious or malicious cyber activity.
- Attributes in MISP can be network indicators (e.g. IP address), system indicators (e.g. a string in memory) or even bank account details.
  - **A type (e.g. MD5, url) is how an attribute is described**.
  - An attribute is always in a category (e.g. Payload delivery) which puts it in a context.
    - **A category is what describes** an attribute.
  - An IDS flag on an attribute allows to determine if **an attribute can be automatically used for detection**.

[5] IoC (Indicator of Compromise) is a subset of indicators

Helping Contributors in MISP

- Contributors can use the UI, API or using the freetext import to add events and attributes.
    - Modules existing in Viper (a binary framework for malware reverser) to populate and use MISP from the vty or via your IDA.
- Contribution can be direct by creating an event but **users can propose attributes updates** to the event owner.
- **Users should not be forced to use a single interface to contribute**.

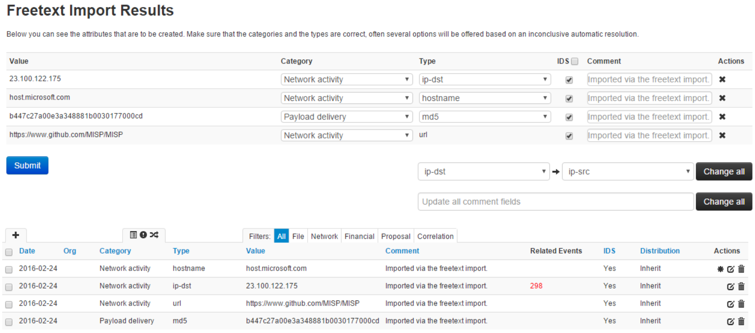
Example: Freetext import in MISP



Freetext Import Tool

Paste a list of IOCs into the field below for automatic detection.

This is a sample text to show how indicators can be extracted. Just paste your text including indicators such as 23.100.122.175, host.microsoft.com, or b447c27a00e3a348881b0030177000cd in here and the tool will automatically detect the indicators and save them as attributes - after allowing you to make some last minute changes. For more information, visit https://www.github.com/MISP/MISP.



Freetext Import Results

Below you can see the attributes that are to be created. Make sure that the categories and the types are correct, often several options will be offered based on an inconclusive automatic resolution.

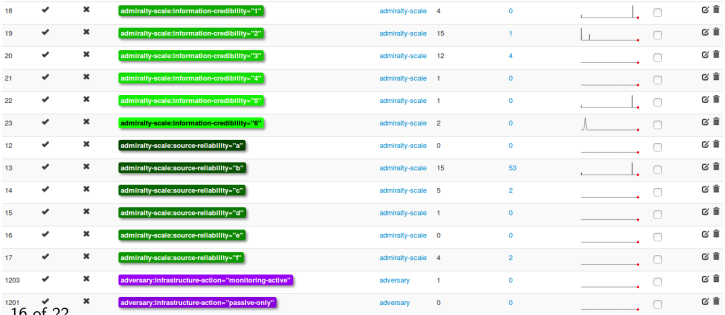
Value	Category	Type	IDS	Comment	Actions
23.100.122.175	Network activity	ip-dst	✓	Imported via the freetext import.	✖
host.microsoft.com	Network activity	hostname	✓	Imported via the freetext import.	✖
b447c27a00e3a348881b0030177000cd	Payload delivery	md5	✓	Imported via the freetext import.	✖
https://www.github.com/MISP/MISP	Network activity	url	✓	Imported via the freetext import.	✖

Submit
ip-dst → ip-src
Change all
Update all comment fields
Change all
Filters: All File Network Financial Proposal Correlation

Date	Org	Category	Type	Value	Comment	Related Events	IDS	Distribution	Actions
2016-02-24		Network activity	hostname	host.microsoft.com	Imported via the freetext import.		Yes	Inherit	
2016-02-24		Network activity	ip-dst	23.100.122.175	Imported via the freetext import.	298	Yes	Inherit	
2016-02-24		Network activity	url	https://www.github.com/MISP/MISP	Imported via the freetext import.		Yes	Inherit	
2016-02-24		Payload delivery	md5	b447c27a00e3a348881b0030177000cd	Imported via the freetext import.		Yes	Inherit	

Supporting Classification

- Tagging is a simple way to attach a classification to an event or an attribute.
- **Classification must be globally used to be efficient**.
- MISP includes a flexible tagging scheme where users can select from more than 42 existing taxonomies or create their own taxonomy.

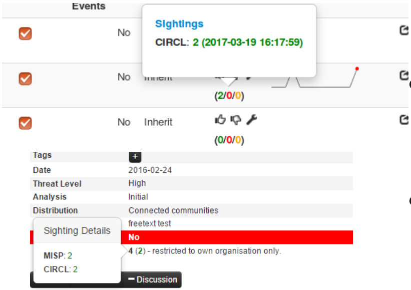


18	✓	✖	admiralty-scale:information-credibility="1"	admiralty-scale	4	0	☐
19	✓	✖	admiralty-scale:information-credibility="2"	admiralty-scale	15	1	☐
20	✓	✖	admiralty-scale:information-credibility="3"	admiralty-scale	12	4	☐
21	✓	✖	admiralty-scale:information-credibility="4"	admiralty-scale	1	0	☐
22	✓	✖	admiralty-scale:information-credibility="5"	admiralty-scale	1	0	☐
23	✓	✖	admiralty-scale:information-credibility="6"	admiralty-scale	2	0	☐
12	✓	✖	admiralty-scale:source-reliability="a"	admiralty-scale	0	0	☐
13	✓	✖	admiralty-scale:source-reliability="b"	admiralty-scale	15	53	☐
14	✓	✖	admiralty-scale:source-reliability="c"	admiralty-scale	5	2	☐
15	✓	✖	admiralty-scale:source-reliability="d"	admiralty-scale	1	0	☐
16	✓	✖	admiralty-scale:source-reliability="e"	admiralty-scale	0	0	☐
17	✓	✖	admiralty-scale:source-reliability="f"	admiralty-scale	4	2	☐
1203	✓	✖	adversary:infrastructure-action="monitoring-active"	adversary	1	0	☐
1201	✓	✖	adversary:infrastructure-action="passive-only"	adversary	0	0	☐

Supporting Sharing in MISP

- Delegate events publication to another organization (introduced in MISP 2.4.18).
    - The other organization can take over the ownership of an event and provide **pseudo-anonymity to initial organization**.
- Sharing groups allow custom sharing (introduced in MISP 2.4) per event or even at attribute level.
    - Sharing communities can be used locally or even cross MISP instances.
    - **Sharing groups** can be done at **event level or attributes level** (e.g. financial indicators shared to a financial sharing groups and cyber security indicators to CSIRT community).

Sightings support


Events

✓	No		
✓	No	Inherit	(2/0/0)
✓	No	Inherit	(0/0/0)

Sightings
CIRCL: 2 (2017-03-19 16:17:59)

Tags	+
Date	2016-02-24
Threat Level	High
Analysis	Initial
Distribution	Connected communities
	freetext test
	No
	4 (2) - restricted to own organisation only.

Sighting Details
MISP: 2
CIRCL: 2

- Sightings allow users to notify the community about the activities related to an indicator.
- In recent MISP versions, the sighting system supports negative sigthings (FP) and expiration sightings.
- Sightings can be performed via the API, and the UI, even including the import of STIX sighting documents.
- Many use-cases for scoring indicators based on users sighting.

Improving Information Sharing in MISP

- False-positives are a recurring challenge in information sharing.
- In MISP 2.4.39, we introduced the misp-warninglists[6] to help analysts in their day-to-day job.
- Predefined lists of well-known indicators which are often false-positives like RFC1918 networks, public DNS resolver are included by default.

[6] `https://github.com/MISP/misp-warninglists`

Improving support of sharing within and outside an organization

- Even in a single organization, multiple use-cases of MISP can appear (groups using it for dynamic malware analysis correlations, dispatching notification).
- In MISP 2.4.51, we introduced the ability to have **local MISP** servers connectivity to avoid changes in distribution level. This allows to have mixed synchronization setup within and outside an organization.
- Feed support was also introduced to support synchronization between untrusted and trusted networks.

Bootstrapping MISP with indicators

- We maintain the default CIRCL OSINT feeds (TLP:WHITE selected from our communities) in MISP to allow users to ease their bootstrapping.
- The format of the OSINT feed is based on standard MISP JSON output pulled from a remote TLS/HTTP server.
- Additional content providers can provide their own MISP feeds. (`https://botvrij.eu/`)
- Allows users to **test their MISP installations and synchronisation with a real dataset**.
- Opening contribution to other threat intel feeds but also allowing the analysis of overlapping data[7].

[7] A recurring challenge in information sharing

Conclusion

- **Information sharing practices come from usage** and by example (e.g. learning by imitation from the shared information).
- MISP is just a tool. What matters is your sharing practices. The tool should be as transparent as possible to support you.
- Enable users to customize MISP to meet their community's use-cases.

MISP Workbench - Manage your very own Cyber Threat Intelligence tool

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

Functionalities of MISP Workbench

- **Merging** of events (campaign, attacker, tool, victim, ...)
- **Comparing campaigns** composed of multiple events
- **Expanding** MISP in a timely manner (no Apache, MySQL & PHP)
- Extraction of **PE indicators** & correlation
- Reduce an investigation on a **subset of events**
- **Very fast lookups**, use the dataset in an **untrusted environment**

MISP Galaxy

- List of known keywords:
    - Adversary groups (with synonyms)
    - Threat actors tools (with synonyms)
- Used to automatically group related events

MISP galaxy - elements of threat actors

- An element list of threat actors included by default.

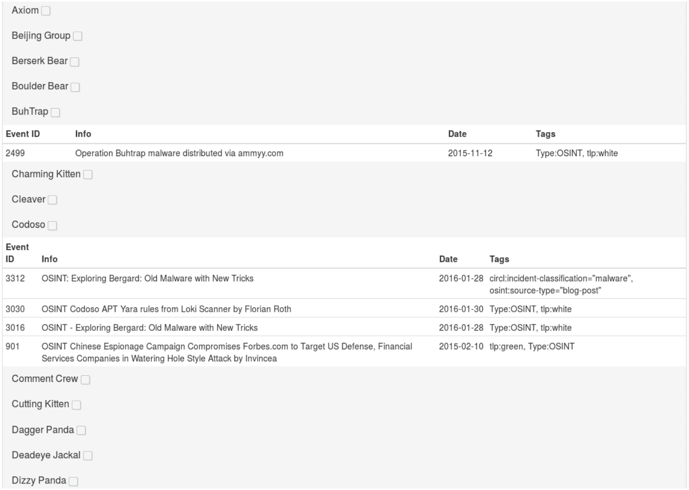
```
{
  "synonyms": [
    "PLA Unit 61486", "APT 2", "Group 36",
    "APT-2", "MSUpdater", "4HCrew", "SULPHUR"
  ],
  "country": "CN",
  "refs": [
    "http://cdn0.vox-cdn.com/assets/4589853/
     crowdstrike-intelligence-report-putter-panda.
         original.pdf"
  ],
  "description": "The CrowdStrike Intelligence team has
    been tracking this particular unit since 2012, under
  the codename PUTTER PANDA, and has documented activity
   dating back to 2007. The report identifies Chen Ping,
    aka cpyy, and the primary location of Unit 61486.",
  "group": "Putter Panda"
}
```

MISP galaxy - elements of threat actors tools

- An element list of tools used by various threat actors.
- The key-values can be freely combined.

```
{
  "value": "MSUpdater"
},
{
  "value": "Poison Ivy",
  "description": "Poison Ivy is a RAT which was freely
     available and first released in 2005.",
  "refs": ["https://www.fireeye.com/content/dam/fireeye-
      www/global/en/current-threats/pdfs/rpt-poison-ivy.
      pdf"]
},
{
  "value": "Elise Backdoor",
  "synonyms": ["Elise"]
}
```

Groups



Axiom ☐

Beijing Group ☐

Berserk Bear ☐

Boulder Bear ☐

BuhTrap ☐

Event ID	Info	Date	Tags
2499	Operation Buhtrap malware distributed via ammyy.com	2015-11-12	Type:OSINT, tlp:white

Charming Kitten ☐

Cleaver ☐

Codoso ☐

Event ID	Info	Date	Tags
3312	OSINT: Exploring Bergard: Old Malware with New Tricks	2016-01-28	circl:incident-classification="malware", osint:source-type="blog-post"
3030	OSINT Codoso APT Yara rules from Loki Scanner by Florian Roth	2016-01-30	Type:OSINT, tlp:white
3016	OSINT - Exploring Bergard: Old Malware with New Tricks	2016-01-28	Type:OSINT, tlp:white
901	OSINT Chinese Espionage Campaign Compromises Forbes.com to Target US Defense, Financial Services Companies in Watering Hole Style Attack by Invincea	2015-02-10	tlp:green, Type:OSINT

Comment Crew ☐

Cutting Kitten ☐

Dagger Panda ☐

Deadeye Jackal ☐

Dizzy Panda ☐

PE indicators

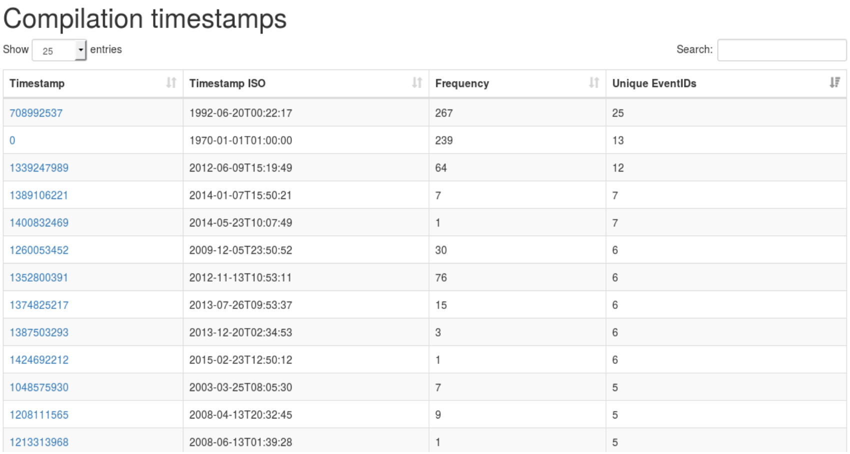
- Original filename
- Compilation timestamp
- Import hashes
- Number of sections
- Entry points
- Soon: API calls
- Soon: Entropy of the sections
- Soon: Fuzzy hashing on the import table

PE indicators

Samples

SHA256	entrypoint	ep_section	timestamp	nb_tls	is_pefile	originalfilename	imphash	timestamp_iso	secnumber	packed
0bc084fa55af0dad75a470f72004d5315b2e02c4f64c402d0a02e8ed3150c36a	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
5d42def31722ae8adb350b2982d91fb05a8568876b73b3061df3cd0639911b89	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
ed203079a9bd5300dc716f44d5cdf6d0f97f1dc51e0d8e39a44ecd81d6237d00	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
afdaa5e29a6fcb98a265c2917a81e9ef385e76d254b2e0c30c8950bc331159f6	9948	.text|0	1340713221	0	True	None	fb84e384d95a4cae125e8501af00a114	2012-06-26T14:20:21	6	0
2ed5c6852aeecb73d7f20c188fad4744217dfd388275e115df680c1383a5a814	587912	.itext|1	1339153947	0	True	MSRSAAP.EXE	8033c11f8a2fdfc317e8655120579933	2012-06-08T13:12:27	9	0
748d0d0eb211a736553bef4c457b53e99df845e25436448cd7f189351a9efdf8	None	None	None	None	None	None	None	None	None	None
44ecbaccfa804d2ea0d15c204351adee7fc3b84a1d9928794f2ca7f2341de42a	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
f56b1248cdecffd25dbf8a2895105fec38f0a4ce03241571c8eb8daafc9a168f	653744	INIT|1	1360325607	0	True	Ultra3.sys	b33e353364df75f474d68496ba228735	2013-02-08T13:13:27	5	90
3746fe21e75ecd84ae124f3b3b1f8cd4fd37945995134d289591983a2e592599	7286	.text|0	1402804315	0	True	Slen.exe	f45d33cfdbdacaa4f25d7c6d82ec1830	2014-06-15T05:51:55	4	0
955656f448c0c537afb99936da7a853fb6b12df0d9ff5228acf5887e754abebe	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
a262dc9e5855447ebd3052b06d714c76fc0656a5b426944e3b27b4a8a2eb2a7c	18563	.text|0	1361334819	0	True	None	10667bae2ab4615fa97a6d1160a88e87	2013-02-20T05:33:39	4	50
fe705f2cce140118c3b511f61604b54e37de376d42c082f3ce20e3081d647a35	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
a5c066f639dc9f8462bc7576e4f58442990d16a674304efd0d6cc3a819a1b41f	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
21457fb079fd0fc3c67ca0a16d59d0f2111a5dfb613cba5120463bda2a6b8c52	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	0
2d1c036a2cfe1f421850ebe46c1177bf890d87a724760bab8ac2f7a2c5832dd4	587912	.itext|1	1339084793	0	True	MSRSAAP.EXE	e5b4359a3773764a372173074ae9b6bd	2012-06-07T17:59:53	9	50
6c3c71c2694c74af96b6fd333e42d001c8bdf56641e32df5ab90429853dfdcfb	None	None	None	None	None	None	None	None	None	None
a9b30b928ebf9cda5136ee37053fa045f3a53d0706dcb2343c91013193de761e	13665	.text|0	1347598075	0	True	None	29d56d896bb10003da6c3fa464d55e96	2012-09-14T06:47:55	4	0
eeccb75ea447cd0e0445251f04458c7c6cf808da72a603e2e076d1a9455c2325	None	None	None	None	None	None	None	None	None	None
40a7bc2f5ba2da6d9a5cf0a66801990de27dd55267297475973293c7091da982	4096	.text|0	1105885028	0	True	None	2c6263c35bb577685ae76a0bc228b266	2005-01-16T15:17:08	4	0
3eca4278063b047dfa87e109c8b0ee7809f582d96e528c4db2f9596c897e92be	12847	.text|0	1353051322	0	True	None	63bf00403dae8328fff132b19e7e9b46	2012-11-16T08:35:22	3	0
e689ce6ee1391d090e20e295712d81457d26d75f816bc58d1e5658984b2bee1f	0	|10	1339153947	0	True	MSRSAAP.EXE	687bcdacb8c2d6718eba2097e5569f3b	2012-06-08T13:12:27	10	140
c42deea433b77119e198dfeae5bf041f5e5c476e6b0971617884b847ffb0fd49	None	None	None	None	None	None	None	None	None	None
065e22f727329160f3c3f4ab21722058121a53256e481b214d909365c925b4b9	717232	UPX1|1	1261071740	0	True	None	77b2e5e9b52fbef7638f64ab65f0c58c	2009-12-17T18:42:20	3	140

PE indicators - Compilation Timestamp



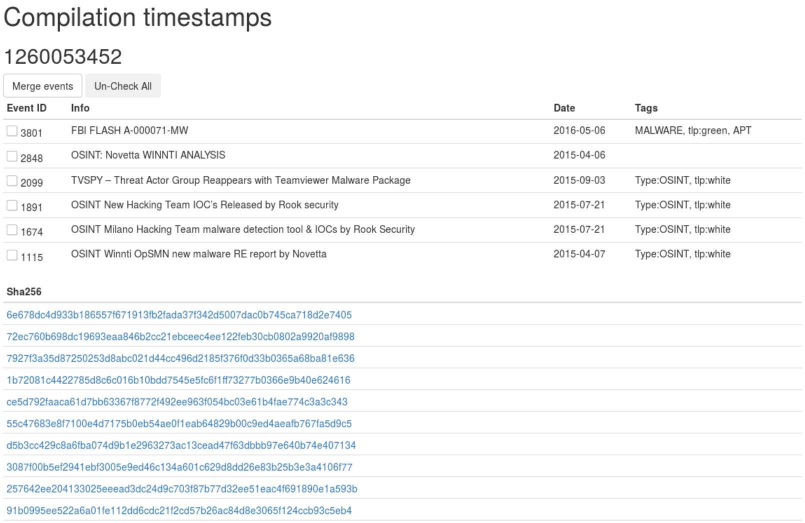
Compilation timestamps

Show 25 entries

Search:

Timestamp	Timestamp ISO	Frequency	Unique EventIDs
708992537	1992-06-20T00:22:17	267	25
0	1970-01-01T01:00:00	239	13
1339247989	2012-06-09T15:19:49	64	12
1389106221	2014-01-07T15:50:21	7	7
1400832469	2014-05-23T10:07:49	1	7
1260053452	2009-12-05T23:50:52	30	6
1352800391	2012-11-13T10:53:11	76	6
1374825217	2013-07-26T09:53:37	15	6
1387503293	2013-12-20T02:34:53	3	6
1424692212	2015-02-23T12:50:12	1	6
1048575930	2003-03-25T08:05:30	7	5
1208111565	2008-04-13T20:32:45	9	5
1213313968	2008-06-13T01:39:28	1	5

PE indicators - Compilation Timestamp



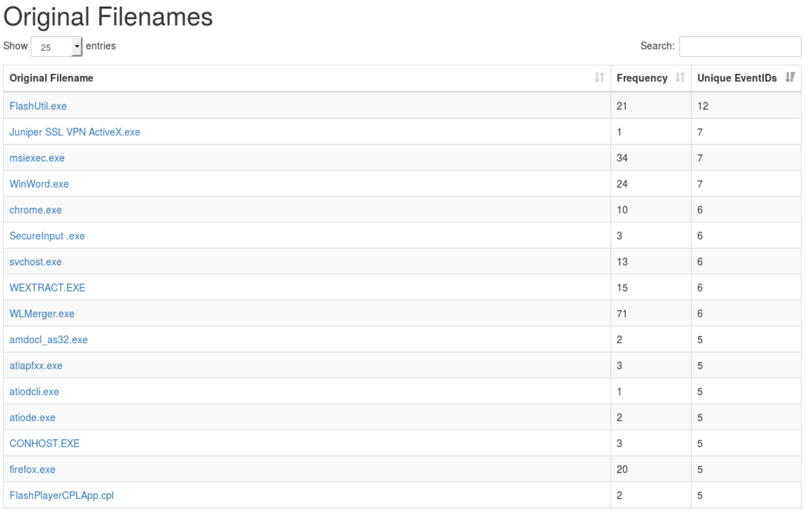
Compilation timestamps

1260053452

Event ID	Info	Date	Tags
☐ 3801	FBI FLASH A-000071-MW	2016-05-06	MALWARE, tlp:green, APT
☐ 2848	OSINT: Novetta WINNTI ANALYSIS	2015-04-06	
☐ 2099	TVSPY – Threat Actor Group Reappears with Teamviewer Malware Package	2015-09-03	Type:OSINT, tlp:white
☐ 1891	OSINT New Hacking Team IOC's Released by Rook security	2015-07-21	Type:OSINT, tlp:white
☐ 1674	OSINT Milano Hacking Team malware detection tool & IOCs by Rook Security	2015-07-21	Type:OSINT, tlp:white
☐ 1115	OSINT Winnti OpSMN new malware RE report by Novetta	2015-04-07	Type:OSINT, tlp:white

Sha256
6e678dc4d933b186557f671913fb2fada37f342d5007dac0b745ca718d2e7405
72ec760b698dc19693eaa846b2cc21ebceec4ee122feb30cb0802a9920af9898
7927f3a35d87250253d8abc021d44cc496d2185f376f0d33b0365a68ba81e636
1b72081c4422785d8c6c016b10bdd7545e5fc6f1ff73277b0366e9b40e624616
ce5d792faaca61d7bb63367f8772f492ee963f054bc03e61b4fae774c3a3c343
55c47683e8f7100e4d7175b0eb54ae0f1eab64829b00c9ed4aeafb767fa5d9c5
d5b3cc429c8a6fba074d9b1e2963273ac13cead47f63dbbb97e640b74e407134
3087f00b5ef2941ebf3005e9ed46c134a601c629d8dd26e83b25b3e3a4106f77
257642ee204133025eeead3dc24d9c703f87b77d32ee51eac4f691890e1a593b
91b0995ee522a6a01fe112dd6cdc21f2cd57b26ac84d8e3065f124ccb93c5eb4

PE indicators - Original Filenames

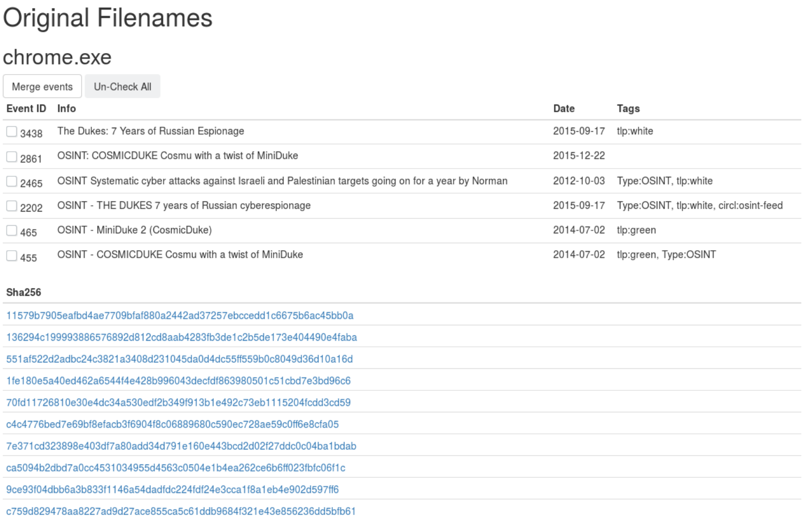


Original Filenames

Show 25 entries   Search:

Original Filename	Frequency	Unique EventIDs
FlashUtil.exe	21	12
Juniper SSL VPN ActiveX.exe	1	7
msiexec.exe	34	7
WinWord.exe	24	7
chrome.exe	10	6
SecureInput .exe	3	6
svchost.exe	13	6
WEXTRACT.EXE	15	6
WLMerger.exe	71	6
amdocl_as32.exe	2	5
atiapfxx.exe	3	5
atiodcli.exe	1	5
atiode.exe	2	5
CONHOST.EXE	3	5
firefox.exe	20	5
FlashPlayerCPLApp.cpl	2	5

PE indicators - Original Filenames



Original Filenames

chrome.exe

Merge events | Un-Check All

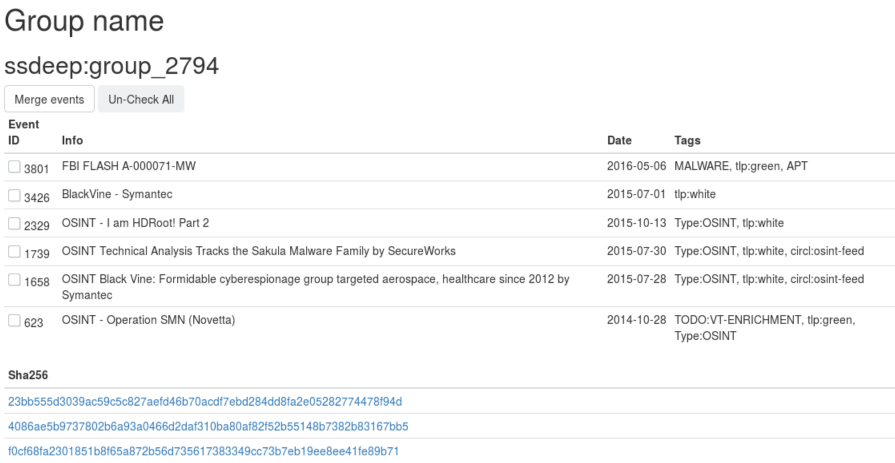
Event ID	Info	Date	Tags
☐ 3438	The Dukes: 7 Years of Russian Espionage	2015-09-17	tlp:white
☐ 2861	OSINT: COSMICDUKE Cosmu with a twist of MiniDuke	2015-12-22	
☐ 2465	OSINT Systematic cyber attacks against Israeli and Palestinian targets going on for a year by Norman	2012-10-03	Type:OSINT, tlp:white
☐ 2202	OSINT - THE DUKES 7 years of Russian cyberespionage	2015-09-17	Type:OSINT, tlp:white, circl:osint-feed
☐ 465	OSINT - MiniDuke 2 (CosmicDuke)	2014-07-02	tlp:green
☐ 455	OSINT - COSMICDUKE Cosmu with a twist of MiniDuke	2014-07-02	tlp:green, Type:OSINT

Sha256
11579b7905eafbd4ae7709bfaf880a2442ad37257ebccedd1c6675b6ac45bb0a
136294c199993886576892d812cd8aab4283fb3de1c2b5de173e404490e4faba
551af522d2adbc24c3821a3408d231045da0d4dc55ff559b0c8049d36d10a16d
1fe180e5a40ed462a6544f4e428b996043decfdf863980501c51cbd7e3bd96c6
70fd11726810e30e4dc34a530edf2b349f913b1e492c73eb1115204fcdd3cd59
c4c4776bed7e69bf8efacb3f6904f8c06889680c590ec728ae59c0ff6e8cfa05
7e371cd323898e403df7a80add34d791e160e443bcd2d02f27ddc0c04ba1bdab
ca5094b2dbd7a0cc4531034955d4563c0504e1b4ea262ce6b6ff023fbfc06f1c
9ce93f04dbb6a3b833f1146a54dadfdc224fdf24e3cca1f8a1eb4e902d597ff6
c759d829478aa8227ad9d27ace855ca5c61ddb9684f321e43e856236dd5bfb61

SSDeep Clustering

- Compute SSDeep hashes on big datasets
- Group samples by similarity
- Allow to pick groups with a certain level of similarities
- Especially interesting on targeted and/or unpaked samples

SSDeep



Group name

ssdeep:group_2794

Merge events | Un-Check All

Event ID	Info	Date	Tags
3801	FBI FLASH A-000071-MW	2016-05-06	MALWARE, tlp:green, APT
3426	BlackVine - Symantec	2015-07-01	tlp:white
2329	OSINT - I am HDRoot! Part 2	2015-10-13	Type:OSINT, tlp:white
1739	OSINT Technical Analysis Tracks the Sakula Malware Family by SecureWorks	2015-07-30	Type:OSINT, tlp:white, circl:osint-feed
1658	OSINT Black Vine: Formidable cyberespionage group targeted aerospace, healthcare since 2012 by Symantec	2015-07-28	Type:OSINT, tlp:white, circl:osint-feed
623	OSINT - Operation SMN (Novetta)	2014-10-28	TODO:VT-ENRICHMENT, tlp:green, Type:OSINT

Sha256
23bb555d3039ac59c5c827aefd46b70acdf7ebd284dd8fa2e05282774478f94d
4086ae5b9737802b6a93a0466d2daf310ba80af82f52b55148b7382b83167bb5
f0cf68fa2301851b8f65a872b56d735617383349cc73b7eb19ee8ee41fe89b71

MISP Hashstore

- Allow very **fast lookups** against big dataset.
- Only store hashed versions of the attributes.
- Can be used on untrusted or compromised systems (comparable to **bloom filter**).
- Hashstore can be used for forensic analysis (e.g. compare baseline
- Beta version available[1].

[1] `https://github.com/MISP/misp-workbench/tree/master/hashstore`

MISP Workbench

- Objective: bundle all the functionalities in one single tool
- Easily **enrich MISP dataset** with other fields (specially PE indicators)
- Simple connectors with other tools and datasets
- **Group events** using galaxies (adversaries and tools)
- **Full text indexing** and lookups for other keywords
- Display the amount of unique MISP events matching a PE attribute
- Single user **lightweitht interface**
- Stand-alone and offline

Implementation

- Full python 3
- Redis backend
- Whoosh full text indexer
- Pefile for the PE processing, radare2 will be used soon
- Flask + bootstrap web interface

Setup

- Export MySQL to Redis
  - Full snapshot for workbench
  - Partial snapshot for hashstore
- Doesn't respect MISP ACL
- Redis database can be moved to an other system
- Run full text indexing
- Import the PE indicators
- Run ssdeep correlation

Q&A

- Developed in collaboration with Marion Marschalek
- `https://github.com/MISP/misp-workbench`
- `https://github.com/MISP/misp-galaxy`
- `https://github.com/MISP/data-processing`
- `https://github.com/CIRCL/ssdc`
- We welcome new functionalities and pull requests.

What's next?

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

What's cooking?

MISP next features and work in progress

Tagging improvements

- Generating related tags (to show and propose similar tags for similar values)
- Special local tags to tags non-owned events
- "Tag everything project"
  - Gives us much more granularity.
  - **Convenient way to add features** without a database change.
- Tags exclusivity as expressed in Taxonomy (e.g. TLP:AMBER and TLP:GREEN tags are exclusive to an attribute or event)

Unified API and modules interface

- **Single search API** / scope (events, objects, attributes)
- Return in **any format** supported by the internal converters and export module
- **Consistent filters** for all output formats
- Open up export modules for bulk exports (framing system)

Graphing improvements

- Highly used but a currently underdeveloped feature
- Open up the **correlation graph to the enrichment module functionality**
- Allow adding attributes directly from the correlation graph
- Allow tagging / attaching clusters directly from the correlation graph
- Advanced correlation where correlations are proposed based on fuzzy matching
- Persistent / shareable graphs (on correlation - already available in graph event)
- Gephi export/integration (on correlation - already available in graph event)

MISP objects improvements

- In application object template editor
- Object level tagging and galaxies
- **Share the object designs within partners on-demand** (e.g. remotely browse shared templates of a partner and import them).
- Closer integration of the objects into the various exports
- MISP-modules upgrade for tighter object integration

MISP galaxy 2.0

- Currently galaxy clusters are static and based on the shared repository / an out of bound created local repository
- 2.0 will allow the interactive creation / editing of galaxies and clusters
- Sharing these across instances will happen purely in MISP instead of just sharing the tags

MISP Darwin

- MISP events are great for more technical analysts or staff familiar with MISP
- The goal is to consolidate the information and automatically **generate natural language reports out of these events**
- Upcoming new project on GitHub
- Python code for managing the creation based on triggers and conversion mechanisms
- Using a list of pre-defined strings from customiseable libraries
- Similar approach as warninglists, taxonomies or galaxies. Just create your own JSON

MISP Hashstore

- Allow very **fast lookups** against big dataset.
- Only store hashed versions of the attributes.
- Can be used on untrusted or compromised systems (comparable to **bloom filter**).
- Hashstore can be used for forensic analysis (e.g. compare baseline
- Beta version available[1].

[1]https://github.com/MISP/misp-workbench/tree/master/hashstore

MISP privacy-aware exchange

- A privacy-aware exchange module to securely and privately share your indicators.
- The basic idea is to transform MISP attributes into something sharable which does not leak any information.
- A first prototype is accessible[2].

[2] `https://github.com/MISP/misp-privacy-aware-exchange`

MISP dashboard 2.0

- Tighter integration with MISP
- 2 way communication with MISP
- Authenticated / ACL enabled version

MISP Gamification

- Goal is to encourage users to contribute by offering recognition for their efforts.
- Profiles with various metrics tracking contribution.
- Opt-in system since it requires a loss of anonymity.
- Gain points by
  - Entering events
  - Proposing changes (that have to be accepted to get credit)
  - Reviewing events and pointing out false positives
- Based on the existing work in misp-dashboard (MISP up vote on usefulness on information will be added).

Conclusion

- **Information sharing practices come from usage** and by example (e.g. learning by imitation from the shared information).
- MISP is just a tool. What matters is your sharing practices. The tool should be as transparent as possible to support you.
- Enable users to customize MISP to meet their community's use-cases.
- MISP is evolving into a modular tool for information sharing and "CTI".
- **Contributions and ideas originate from the community of users**.
- Co-funding of new features or projects around MISP are welcome.

Q&A

MISP
Threat Sharing

- https://github.com/MISP/MISP
- https://github.com/MISP/ for misp-modules, misp-objects, misp-taxonomies and misp-galaxy.
- Feel free to open an issue or make a pull-request on GitHub.

Contributing to MISP Project

Become part of the community to design, develop and improve information sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

Code of Conduct

- The MISP project has a Contributor Covenant Code of Conduct[1].
- The goal of the code of conduct is to foster an **open, fun and welcoming environment**.
- Another important aspect of the MISP projects is to welcome different areas of expertise in information sharing and analysis. The **diversity of the MISP community** is important to make the project useful for everyone.

[1] `https://github.com/MISP/MISP/code_of_conduct.md`

Reporting a bug, an issue or suggesting features

- The most common way to contribute to the MISP project is to report a bug, issues or suggesting features.
- Each project (MISP core, misp-modules, misp-book, misp-taxonomies, misp-galaxy, misp-object or PyMISP) has their **own issue management**.
- Don't forget that you can **cross-reference issues** from other sub-projects.
- If you know an answer or could help on a specific issue, we welcome all contributions including **useful comments to reach a resolution**.

Reporting security vulnerabilities

- **If you find security vulnerabilities (even minor ones) in MISP project, send an encrypted email** (info@circl.lu) with the details and especially how to reproduce the issues. Avoid to share publicly the vulnerability before a fix is available in MISP. PGP key fingerprint: CA57 2205 C002 4E06 BA70 BE89 EAAD CFFC 22BD 4CD5.
- We usually fix reported and confirmed security vulnerabilities in less than 48 hours.
- **We will request a CVE number** if the reporters didn't ask for one (don't forget to mention how you want to be credited).

Automatic integration and testing

- The majority of the repositories within the MISP GitHub organisation includes automatic integration with TravisCI.
- If you contribute and make a pull-request, **verify if your changes affect the result of the tests**.
- Automatic integration is not perfect including Travis but it's a quick win to catch new bugs or major issues in contribution.
- When you do a pull-request, TravisCI is automatically called[2].
    - If this fails, no worries, **review the output at Travis** (it's not always you).
- We are working on additional automatic tests including unit testing for the MISP core software (contributors are welcome).

[2] `https://travis-ci.org/MISP`

JSON validation for MISP libraries

- All JSON format (**galaxy, taxonomies, objects or warning-lists**) are described in a JSON Schema[3].
- The TravisCI tests are including JSON validation (via *jq*) and validated with the associated JSON schema.
- How to contribute a JSON library (objects, taxonomies, galaxy or warning-list):
  - If you update a JSON library, don't forget to run *jq_all_the_things.sh*. It's fast and easy. If it fails, review your JSON.
  - Commit your code and make a pull-request.
- Documentations (in PDF and HTML format) for the librairies are automatically generated from the JSON via asciidoctor[4].

[3] schema_name.json

[4] example `https://github.com/MISP/misp-galaxy/blob/master/tools/adoc_galaxy.py`

Documentation

- In addition to the automatic generation of documentations from JSON files, we maintain **misp-book**[5] which is a generic documentation for MISP including usage, API documentation, best practices and specific configuration settings.
- The book is generated in HTML, PDF, epub and mobi using GitBook[6] which is a framework to write documentation in MarkDown format.
- TravisCI is included in misp-book and **the book generation is tested at each commit**.
- The MISP book is regularly published on misp-project.org and circl.lu website.
- Contributors are welcome especially for new topics[7] and also fixing our broken english.

[5] `https://github.com/MISP/misp-book`

[6] `https://github.com/GitbookIO`

[7] Topics of interest are analysts best-practices,

Internet-Draft - IETF for MISP formats

- If you want to contribute to our IETF Internet-Draft for the MISP standard, misp-rfc[8] is the repository where to contribute.
- **Update only the markdown file**, the XML and ASCII for the IETF I-D are automatically generated.
- If a major release or updates happen in the format, we will publish the I-D to the IETF[9].
- The process is always MISP implementation $\rightarrow$ IETF I-D updates.

[8] `https://github.com/MISP/misp-rfc`

[9] `https://datatracker.ietf.org/doc/search/?name=misp&activedrafts=on&rfcs=on`

MISP core development crash course

How I learned to stop worrying and love the PHP

Team CIRCL

MISP Training @ Prague
20180917

Some things to know in advance...

- MISP is based on PHP 5.6+
- Using the MVC framework CakePHP 2.x
- What we'll look at now will be a quick glance at the structuring / layout of the code

MVC frameworks in general

- separation of business logic and views, interconnected by controllers
- main advantage is clear separation of the various components
- lean controllers, fat models (kinda...)
- domain based code reuse
- No interaction between Model and Views, ever

Structure of MISP Core app directories

- Config: general configuration files
- Console: command line tools
- Controller: Code dealing with requests/responses, generating data for views based on interactions with the models
- Lib: Generic reusable code / libraries
- Model: Business logic, data gathering and modification
- Plugin: Alternative location for plugin specific codes, ordered into controller, model, view files
- View: UI views, populated by the controller

Controllers - scope

- Each public function in a controller is exposed as an API action
- request routing (admin routing)
- multi-use functions (POST/GET)
- request/response objects
- contains the action code, telling the application what data fetching/modifying calls to make, preparing the resulting data for the resulting view
- grouped into controller files based on model actions
- Accessed via UI, API, AJAX calls directly by users
- For code reuse: behaviours
- Each controller bound to a model

Controllers - functionalities of controllers

- pagination functionality
- logging functionality
- Controllers actions can access functionality / variables of Models
- Controllers cannot access code of other controller actions (kind of...)
- Access to the authenticated user's data
- beforeFilter(), afterFilter() methods
- Inherited code in AppController

Controllers - components

- Components = reusable code for Controllers
  - Authentication components
  - RestResponse component
  - ACL component
  - Cidr component
  - IOCImport component (should be moved)

Controllers - additional functionalities

- code handling API requests
- auth/session management
- ACL management
- API management
- Security component
- important: quertString/PyMISP versions, MISP version handler
- future improvements to the export mechanisms

Models - scope

- Controls anything that has to do with:
  - finding subsets of data
  - altering existing data
  - inherited model: AppModel
  - reusable code for models: Behaviours
  - regex, trim

Models - hooking system

- Versatile hooking system
  - manipulate the data at certain stages of execution
  - code can be located in 3 places: Model hook, AppModel hook, behaviour

Model - hooking pipeline (add/edit)

- Hooks / model pipeline for data creation / edits
  - beforeValidate() (lowercase all hashes)
  - validate() (check hash format)
  - afterValidate() (we never use it
  - could be interesting if we ever validated without saving)
  - beforeSave() (purge existing correlations for an attribute)
  - afterSave() (create new correlations for an attribute / zmq)

Models - hooking pipeline (delete/read)

- Hooks for deletions
  - beforeDelete() (purge correlations for an attribute)
  - afterDelete() (zmq)
- Hooks for retrieving data
  - beforeFind() (modify the find parameters before execution, we don't use it)
  - afterFind() (json decode json fields)

Models - misc

- code to handle version upgrades contained in AppModel
- generic cleanup/data migration tools
- centralised redis/pubsub handlers
- (Show example of adding an attribute with trace)

Views - scope and structure

- templates for views
- layouts
- reusable template code: elements
  - attribute list, rows (if reused)
- reusable code: helpers
  - commandhelper (for discussion boards), highlighter for searches, tag colour helper
- views per controller

Views - Types of views and helpers

- ajax views vs normal views
- data views vs normal views vs serialisation in the controller
- sanitisation h()
- creating forms
  - sanitisation
  - CSRF

Distribution

- algorithm for checking if a user has access to an attribute
- creator vs owner organisation
- distribution levels and inheritance (events -¿ objects -¿ attributes)
- shorthand inherit level
- sharing groups (org list, instance list)
- correlation distribution
- algorithms for safe data fetching (fetchEvents(), fetchAttributes(),...)

Testing your code

- funtional testing
- impact scope
  - view code changes: only impacts request type based views
  - controller code changes: Should only affect given action
  - model code changes: can have impact on entire application
  - lib changes: can have affect on the entire application
- Don't forget: queryACL, change querystring

Deep-dive into PyMISP

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

Context

- MISP is a large project
- Your production environment is even more complex
- 3rd party services are even worse
- Querying MISP via CURL is doable, but get's painful fast
- Talking to MySQL directly can be dangerous
- POST a JSON blob, receive a JSON blob. You can do it manually(-ish)

Big picture

- Core goal: providing stable access to APIs, respect access control
- Simplifying handling & automation of indicators in 3rd party tools
- Hiding complexity of the JSON blobs
- Providing pre-cooked examples for commonly used operations
- Helping integration with existing infrastructure

Common queries: Recent changes on a timeframe

There are 4 main cases here:

- Metadata of the events that have been modified
  - **search_index** ⇒ timestamp (1h, 1d, 7d, ...), returns list of all the modified events
- Full events (metadata + attributes)
  - **search** ⇒ timestamp (1h, 1d, 7d, ...)
- Modified attributes
  - **search** ⇒ controller = attributes and timestamp (1h, 1d, 7d, ...)
- Other use case: get last **published** events by using the last parameter in the **search** method.

Common queries: Search things

There are 3 main cases here:

- Easy, but slow: full text search with **search_all**
- Faster: use the **search** method and search by tag, type, enforce the warning lists, with(-out) attachments, dates interval, ...
- Get malware samples (if available on the instance).

Common queries: create things

There are 3 main cases here:

- Add Event, edit its metadata
- Add attributes or objects to event
- (un)Tag event or attribute (soon object)
- Edit Attributes medatada
- Upload malware sample (and automatically expand it)

Administrative tasks

Assyming you have the right to do it on the instance.

- Managing users
- Managing organisations
- Managing sync servers

Other Capabilities

- Upload/download samples
- **Proposals**: add, edit, accept, discard
- **Sightings**: Get, set, update
- Export **statistics**
- Manage **feeds**
- Get MISP server version, recommended PyMISP version
- And more, look at the api file

MISPEvent - Usecase

```
from pymisp import MISPEvent, EncodeUpdate

# Create a new event with default values
event = MISPEvent()

# Load an existing JSON dump (optional)
event.load_file('Path/to/event.json')
event.info = 'My cool event'  # Duh.

# Add an attribute of type ip-dst
event.add_attribute('ip-dst', '8.8.8.8')

# Mark an attribute as deleted (From 2.4.60)
event.delete_attribute('<Attribute UUID>')

# Dump as json
event_as_jsondump = json.dumps(event, cls=EncodeUpdate)
```

Basics

- Python 3.5+ is recommended
- PyMISP is always inline with current version (pip3 install pymisp)
- Dev version: pip3 install git+https://github.com/MISP/PyMISP.git
- Get your auth key from: `https://misppriv.circl.lu/events/automation`
  - Not available: you don't have "Auth key access" role. Contact your instance admin.
- Source available here: git clone `https://github.com/MISP/PyMISP.git`

Examples

- **PyMISP needs to be installed (duh)**
- Usage:
    - Create examples/keys.py with the following content

```
misp_url = "https://url-to-your-misp"
misp_key = "<API_KEY>"
misp_verifycert = True
```

- Proxy support:

```
proxies = {
        'http': 'http://127.0.0.1:8123',
        'https': 'http://127.0.0.1:8123',
}
PyMISP(misp_url, misp_key, misp_verifycert, proxies=proxies)
```

Examples

- Lots of ideas on how to use the API
- You may also want to look at the tests directory
- All the examples use argparse. Help usage is available: **script.py -h**
    - **add_file_object.py**: Attach a file (PE/ELF/Mach-O) object to an event
    - **upload.py**: Upload a malware sample (use advanced expansion is available on the server)
    - **last.py**: Returns all the most recent events (on a timeframe)
    - **add_named_attribute.py**: Add attribute to an event
    - **sighting.py**: Update sightings on an attribute
    - **stats.py**: Returns the stats of a MISP instance
    - {**add,edit,create**}_**user.py** : Add, Edit, Create a user on MISP

Usage

- Basic example

```
from pymisp import PyMISP
api = PyMISP(url, apikey, verifycert=True, debug=False, proxies=None)
response = api.<function>
if response['error']:
        # <something went wrong>
else:
        # <do something with the output>
```

Concept behind AbstractMISP

- JSON blobs are python dictionaries
- ... Accessing content can be a pain
- **AbstractMISP inherits collections.MutableMapping**, they are all dictionaries!
- ... Has helpers to load, dump, and edit JSON blobs
- **Important**: All the public attributes (not starting with a _) defined in a class are dumped to JSON
- **Tags**: Events and Attributes have tags, soon Objects. Tag handling is defined in this class.
- **edited**: When pushing a full MISPEvent, only the objects without a timestamp, or with a newer timestamp will be updated. This method recursively finds updated events, and removes the timestamp key from the object.

MISPEvent, MISPAttribute, MISPObject, MISPSighting...

- **Pythonic** representation of MISP elements
- **Easy manipulation**
  - Load an existing event
  - Update te metadata, add attributes, objects, tags, mark an attribute as deleted, ...
  - Set relations between objects
  - Load and add attachments or malware samples as pseudo files
- **Dump** to JSON

MISPEvent - Main entrypoints

- load_file(event_path)
- load(json_event)
- add_attribute(type, value, **kwargs)
- add_object(obj=None, **kwargs)
- add_attribute_tag(tag, attribute_identifier)
- get_attribute_tag(attribute_identifier)
- add_tag(tag=None, **kwargs)
- objects[], attributes[], tags[]
- edited, all other paramaters of the MISPEvent element (info, date, ...)
- to_json()

MISPObject - Main entrypoints

- add_attribute(object_relation, **value)
- add_reference(referenced_uuid, relationship_type, comment=None, **kwargs)
- has_attributes_by_relation(list_of_relations)
- get_attributes_by_relation(object_relation)
- attributes[], relations[]
- edited, all other paramaters of the MISPObject element (name, comment, ...)
- to_json()
- Can be validated against their template
- Can have default parameters applied to all attributes (i.e. distribution, category, ...)

MISPAttribute - Main entrypoints

- add_tag(tag=None, **kwargs)
- delete()
- malware_binary (if relevant)
- tags[]
- edited, all other paramaters of the MISPObject element (value, comment, ...)
- to_json()

PyMISP - Tools

- Libraries requiring specfic 3rd party dependencies
- Callable via PyMISP for specific usecases
- Curently implemented:
  - **OpenIOC** to MISP Event
  - MISP to **Neo4J**

PyMISP - Default objects generators

- File - PE/ELF/MachO - Sections
- VirusTotal
- Generic object generator

PyMISP - Logging / Debugging

- debug=True passed to the constructor enable debug to stdout
- Configurable using the standard logging module
- Show everything send to the server and received by the client

```
import pymisp
import logging

logger = logging.getLogger('pymisp')
logger.setLevel(logging.DEBUG)  # enable debug to stdout

logging.basicConfig(level=logging.DEBUG,    # Enable debug to file
                    filename="debug.log",
                    filemode='w',
                    format=pymisp.FORMAT)
```

Q&A

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

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

Providing ready-to-use threat intelligence in MISP standard format

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL
TLP:WHITE

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

MISP Feed - Basics

MISP Feeds provide a way to

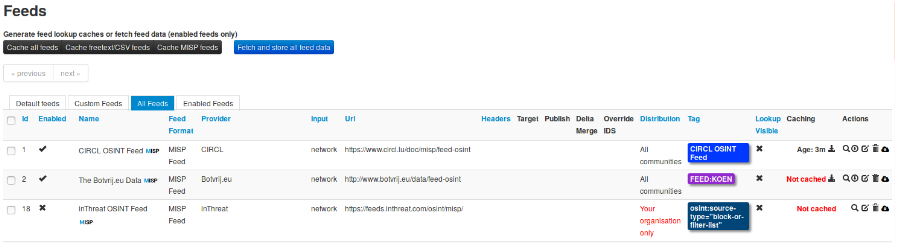
- **Exchange information via any transports** (e.g. HTTP, TLS, USB keys)
- Preview events along with their attributes, objects
- Select and import events
- **Correlate attributes using caching**

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)

Feed - Overview

- 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].



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 »

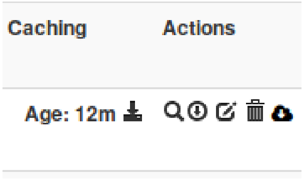
Default feeds | Custom Feeds | All Feeds | Enabled Feeds

	Id	Enabled	Name	Feed Format	Provider	Input	Url	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 Botvrij.eu Data MISP	MISP Feed	Botvrij.eu	network	http://www.botvrij.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	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/

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 - 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 lighweight **TAXII interface**

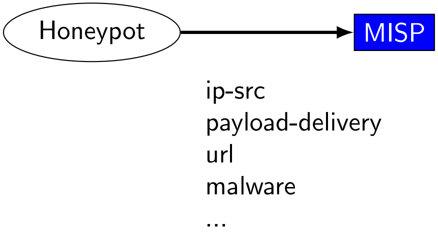
Feed generator - configuration file

```
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

```

Real-time Feed generator - Purpose

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**:



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

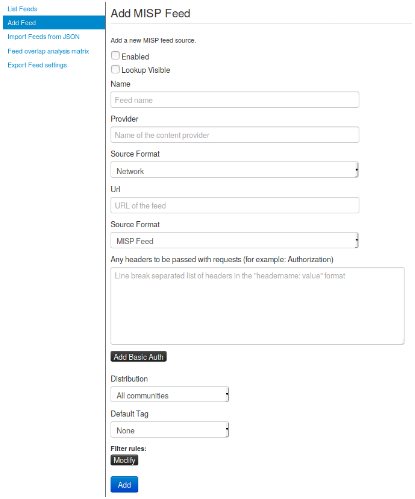
Example:

```
#  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)
```

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)
```

Adding custom feed to MISP


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 "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.

MISP workshop

Introduction into Information Sharing using MISP for CSIRTs

CIRCL
Computer Incident
Response Center
Luxembourg

MISP
Threat Sharing

Team CIRCL
TLP:WHITE

MISP Training @ Prague
20180917

Plan for this session

- Explanation of the CSIRT use case for information sharing and what CIRCL does
- Building an information sharing community and best practices

Communities operated by CIRCL

- As a CSIRT, CIRCL operates a wide range of communities
- We use it as an **internal tool** to cover various day-to-day activities
- Whilst being the main driving force behind the development, we're also one of the largest consumers
- Different communities have different needs and restrictions

Communities operated by CIRCL

- Private sector community
  - Our largest sharing community
  - Over **900 organisations**
  - **2000 users**
  - Functions as a central hub for a lot of sharing communities
  - Private organisations, Researchers, Various SoCs, some CSIRTs, etc
- CSIRT community
  - Tighter community
  - National CSIRTs, connections to international organisations, etc

Communities operated by CIRCL

- Financial sector community
  - Banks, payment processors, etc.
  - Sharing of **mule accounts** and **non-cyber threat infomartion**
- X-ISAC
  - **Bridging the gap** between the various sectorial and georgraphical ISACs
  - New, but ambitious initiative
  - Goal is to **bootstrap the cross-sectorial sharing** along with building the infrastructure to enable sharing when needed

Communities operated by CIRCL

- Coming up - the ATT&CK EU community
  - Work on attacker modelling
  - With the assistance of Mitre themselves
  - Unique opportunity to **standardise on TTPs**
  - Looking for organisations that want to get involved!

Communities supported by CIRCL

- FIRST.org's MISP community
- Telecom and Mobile operators' community
- Various ad-hoc communities for exercises for example
  - Most recently for example for the ENISA exercise a few weeks ago

Sharing Scenarios in MISP

- Sharing can happen for **many different reasons**. Let's see what we believe are the typical CSIRT scenarios
- We can generally split these activities into 4 main groups when we're talking about traditional CSIRT tasks:
  - Core services
  - Proactive services
  - Advanced services
  - Sharing communities managed by CSIRTs for various tasks

CSIRT core services

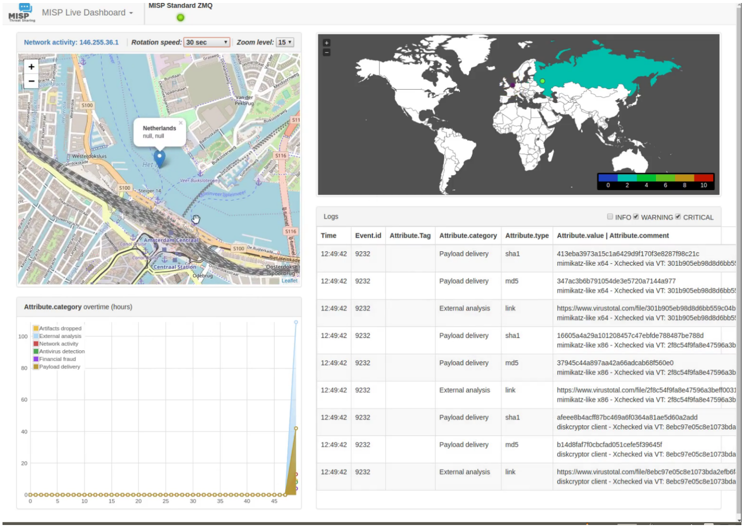
- Incident response
  - **Internal storage** of incident response data
  - Sharing of indicators **derived from incident response**
  - **Correlating data** derived and using the built in analysis tools
  - **Enrichment** services
  - **Collaboration** with affected parties via MISP during IR
  - **Co-ordination** and collaboration
  - **Takedown** requests
- Alerting of information leaks (integration with **AIL**[1])

[1] `https://github.com/CIRCL/AIL-framework`

CSIRT proactive services

- **Contextualising** both internal and external data
- **Collection** and **dissimination** of data from various sources (including OSINT)
- Storing, correlating and sharing own manual research (**reversing, behavioural analysis**)
- Aggregating automated collection (**sandboxing, honeypots, spamtraps, sensors**)
    - MISP allows for the creation of **internal MISP "clouds"**
    - Store **large specialised datasets** (for example honeypot data)
    - MISP has **interactions with** a large set of such **tools** (Cuckoo, Mail2MISP, etc)
- **Situational awareness** tools to monitor trends and adversary TTPs within my sector/geographical region (MISP-dashboard, built in statistics)

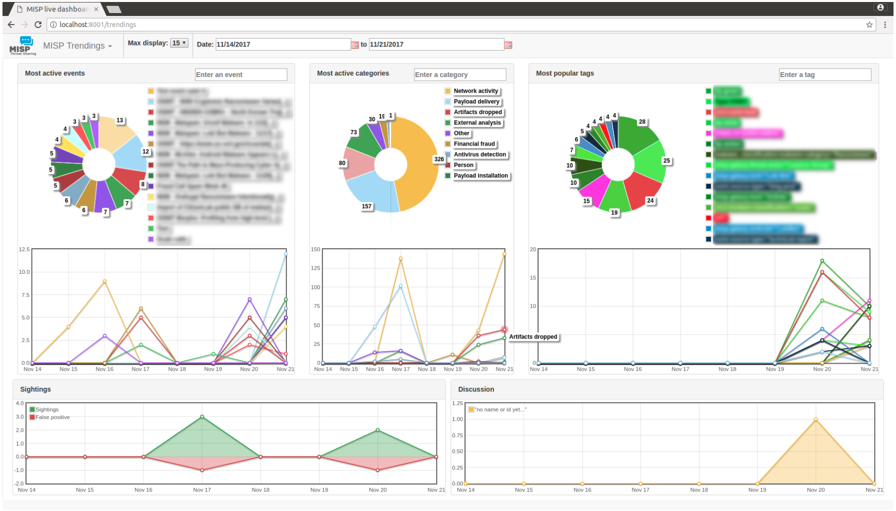
CSIRT proactive services - MISP dashboard



Logs ☐ INFO ☑ WARNING ☑ CRITICAL

Time	Event.id	Attribute.Tag	Attribute.category	Attribute.type	Attribute.value | Attribute.comment
12:49:42	9232		Payload delivery	sha1	413eba3973a15c1a6429d9f170f3e8287f98c21c mimikatz-like x64 - Xchecked via VT: 301b905eb98d8d6bb5
12:49:42	9232		Payload delivery	md5	347ac3b6b791054de3e5720a7144a977 mimikatz-like x64 - Xchecked via VT: 301b905eb98d8d6bb5
12:49:42	9232		External analysis	link	https://www.virustotal.com/file/301b905eb98d8d6bb559c04b mimikatz-like x64 - Xchecked via VT: 301b905eb98d8d6bb5
12:49:42	9232		Payload delivery	sha1	16605a4a29a101208457c47ebfde788487be788d mimikatz-like x86 - Xchecked via VT: 2f8c54f9fa8e47596a3b
12:49:42	9232		Payload delivery	md5	37945c44a897aa42a66adcab68f560e0 mimikatz-like x86 - Xchecked via VT: 2f8c54f9fa8e47596a3b
12:49:42	9232		External analysis	link	https://www.virustotal.com/file/2f8c54f9fa8e47596a3beff0031 mimikatz-like x86 - Xchecked via VT: 2f8c54f9fa8e47596a3b
12:49:42	9232		Payload delivery	sha1	afeee8b4acff87bc469a6f0364a81ae5d60a2add diskcryptor client - Xchecked via VT: 8ebc97e05c8e1073bda
12:49:42	9232		Payload delivery	md5	b14d8faf7f0cbcfad051cefe5f39645f diskcryptor client - Xchecked via VT: 8ebc97e05c8e1073bda
12:49:42	9232		External analysis	link	https://www.virustotal.com/file/8ebc97e05c8e1073bda2efb6f diskcryptor client - Xchecked via VT: 8ebc97e05c8e1073bda

CSIRT proactive services - MISP dashboard



MISP live dashboard

localhost:8001/trendings

MISP Trendings

Max display: 15

Date: 11/14/2017 to 11/21/2017

Most active events — Enter an event

[redacted]

Most active categories — Enter a category

- 🟧 Network activity
- 🟦 Payload delivery
- 🟥 Artifacts dropped
- 🟩 External analysis
- 🟪 Other
- 🟫 Financial fraud
- 🟦 Antivirus detection
- 🟥 Person
- 🟩 Payload installation

Most popular tags — Enter a tag

[redacted]

Sightings

- 🟩 Sightings
- 🟥 False positive

Discussion

- 🟧 "no name or id yet..."

CSIRT advanced services

- Supporting **forensic analysts**
- Collaboration with **law enforcement**
- **Vulnerability** information sharing
  - **Notifications** to the constituency about relevant vulnerabilities
  - **Co-ordinating** with vendors for notifications (*)
  - Internal / closed community sharing of pentest results
  - We're planning on starting a series of hackathons to find

CSIRTs' management of sharing communities for constituent actions:

- **Reporting** non-identifying information about incidents (such as outlined in NISD)
- **Seeking** and engaging in **collaboration** with CSIRT or other parties during an incident
- Pre-sharing information to **request for help** / additional information from the community
- **Pseudo-anonymised sharing** through 3rd parties to **avoid attribution** of a potential target
- Building processes for **other types of sharing** to get the community engaged and acquainted with the methodologies of sharing (mule account information, border control, etc)

A quick note on compliance...

- Collaboration with Deloitte as part of a CEF project for creating compliance documents
  - Information sharing and cooperation **enabled by GDPR**
  - How MISP enables stakeholders identified by the **NISD** to perform key activities
  - **AIL** and MISP
- For more information: https://github.com/CIRCL/compliance

Bringing different sharing communities together

- We generally all **end up sharing with peers that face similar threats**
- Division is either **sectorial or geographical**
- So why even bother with trying to bridge these communities?

Advantages of cross sectorial sharing

- **Reuse of TTPs** across sectors
- Being hit by something that **another sector has faced before**
- **Hybrid threats** - how seemingly unrelated things may be interesting to correlate
- Prepare other communities for the capability and **culture of sharing** for when the need arises for them to reach out to CSIRT
- Generally our field is ahead of several other sectors when it comes to information sharing, might as well **spread the love**

Getting started with building your own sharing community

- Starting a sharing community is **both easy and difficult** at the same time
- Many moving parts and most importantly, you'll be dealing with a diverse group of people
- Understanding and working with your constituents to help them face their challenges is key

Getting started with building your own sharing community

- When you are starting out - you are in a unique position to drive the community and set best practices...

Running a sharing community using MISP - How to get going?

- Different models for constituents
  - Connecting to a MISP instance hosted by a CSIRT
  - Hosting their own instance and connecting to CSIRT's MISP
  - Becoming member of a sectorial MISP community that is connected to CSIRT's community
- Planning ahead for future growth
  - Estimating requirements
  - Deciding early on common vocabularies
  - Offering services through MISP

Rely on our instincts to immitate over expecting adherence to rules

- Lead by example - the power of immitation
- Encourage improving by doing instead of blocking sharing with unrealistic quality controls
    - What should the information look like?
    - How should it be contextualise
    - What do you consider as useful information?
    - What tools did you use to get your conclusions?
- Side effect is that you will end up raising the capabilities of your constituents

What counts as valuable data?

- Sharing comes in many shapes and sizes
  - Sharing results / reports is the classical example
  - Sharing enhancements to existing data
  - Validating data / flagging false positives
  - Asking for support from the community
- Embrace all of them. Even the ones that don't do either, you'll never know when they change their minds...

How to deal with organisations that only "leech"?

- From our own communities, only about 30% of the organisations actively share data
- We have come across some communities with sharing requirements
- In our experience, this sets you up for failure because:
    - Organisations will lose protection who would possibily benefit the most from it
    - Organisations that want to stay above the thresholds will start sharing junk / fake data
    - You lose organisations that might turn into valuable contributors in the future

So how does one convert the passive organisations into actively sharing ones?

- Rely on organic growth
- Help them increase their capabilities
- As mentioned before, lead by example
- Rely on the inherent value to one's self when sharing information (validation, enrichments, correlations)
- Give credit where credit is due, never steal the accolades of your community (that is incredibly demotivating)

Dispelling the myths around blockers when it comes to information sharing

- Sharing difficulties are not really technical issues but often it's a matter of **social interactions** (e.g. **trust**).
  - You can play a role here: organise regular workshops, conferences, have face to face meetings
- Legal restrictions
  - "Our legal framework doesn't allow us to share information."
  - "Risk of information leak is too high and it's too risky for our organization or partners."
- Practical restrictions
  - "We don't have information to share."
  - "We don't have time to process or contribute indicators."
  - "Our model of classification doesn't fit your model."
  - "Tools for sharing information are tied to a specific format, we use a different one."

Contextualising the information

- Sharing technical information is a great start
- However, to truly create valueable information for your community, always consider the context:
  - Your IDS might not care why it should alert on a rule
  - But your analysts will be interested in the threat landscape and the "big picture"
- Classify data to make sure your partners understand why it is important for them
- Massively important once an organisation has the maturity to filter the most critical subsets of information for their own defense

Choice of vocabularies

- MISP has a verify versatile system (taxonomies) for classifying and marking data
- However, this includes different vocabularies with obvious overlaps
- MISP allows you to pick and choose vocabularies to use and enforce in a community
- Good idea to start with this process early
- If you don't find what you're looking for:
  - Create your own (JSON format, no coding skills required)
  - If it makes sense, share it with us via a pull request for redistribution

Shared libraries of meta-information (Galaxies)

- The MISPProject in co-operation with partners provides a curated list of galaxy information
- Can include information packages of different types, for example:
  - Threat actor information
  - Specialised information such as Ransomware, Exploit kits, etc
  - Methodology information such as preventative actions
  - Classification systems for methodologies used by adversaries - ATT&CK
- Consider improving the default libraries or contributing your own (simple JSON format)
- If there is something you cannot share, run your own galaxies and share it out of bound with partners
- Pull requests are always welcome

False-positive handling

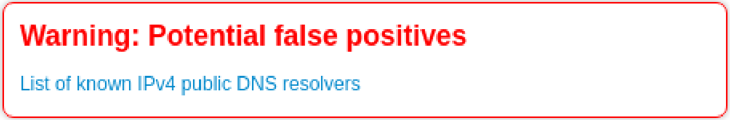
- You might often fall into the trap of discarding seemingly "junk" data
- Besides volume limitations (which are absolutely valid, fear of false-positives is the most common reason why people discard data) - Our recommendation:
  - Be lenient when considering what to keep
  - Be strict when you are feeding tools
- MISP allows you to filter out the relevant data on demand when feeding protective tools
- What may seem like junk to you may be absolutely critical to other users

Many objectives from different user-groups

- Sharing indicators for a **detection** matter.
  - 'Do I have infected systems in my infrastructure or the ones I operate?'
- Sharing indicators to **block**.
  - 'I use these attributes to block, sinkhole or divert traffic.'
- Sharing indicators to **perform intelligence**.
  - 'Gathering information about campaigns and attacks. Are they related? Who is targeting me? Who are the adversaries?'
- $\rightarrow$ These objectives can be conflicting (e.g. False-positives have different impacts)

False-positive handling

- Analysts will often be interested in the modus operandi of threat actors over long periods of time
- Even cleaned up infected hosts might become interesting again (embedded in code, recurring reuse)
- Use the tools provided to eliminate obvious false positives instead and limit your data-set to the most relevant sets


Warning: Potential false positives

List of known IPv4 public DNS resolvers

Managing sub-communities

- Often within a community smaller bubbles of information sharing will form
- For example: Within a national private sector sharing community, specific community for financial institutions
- Sharing groups serve this purpose mainly
- As a CSIRT running a national community, consider bootstraping these sharing communities
- Organisations can of course self-organise, but you are the ones with the know-how to get them started

Managing sub-communities

- Consider compartmentalisation - does it make sense to move a secret squirrel club to their own sharing hub to avoid accidental leaks?
- Use your best judgement to decide which communities should be separated from one another
- Create sharing hubs with manual data transfer
- Some organisations will even have their data air-gapped - Feed system
- Create guidance on what should be shared outside of their bubbles - organisations often lack the insight / experience to decide how to get going. Take the initiative!

Get in touch if you need some help to get started

- Getting started with building a new community can be daunting. Feel free to get in touch with us if you have any questions!
- Contact: info@circl.lu
- `https://www.circl.lu/`
- `https://github.com/MISP` - `https://gitter.im/MISP/MISP` - `https://twitter.com/MISPProject`

MISP User Training - General usage of MISP

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

MISP - VM

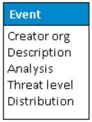
- Credentials
  - MISP admin: admin@admin.test/admin
  - SSH: misp/Password1234
- Available at the following location (VirtualBox and VMWare):
  - `https://www.circl.lu/misp-images/latest/`

MISP - General Usage

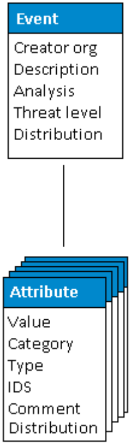
Plan for this part of the training

- Data model
- Viewing data
- Creating data
- Co-operation
- Distribution
- Exports

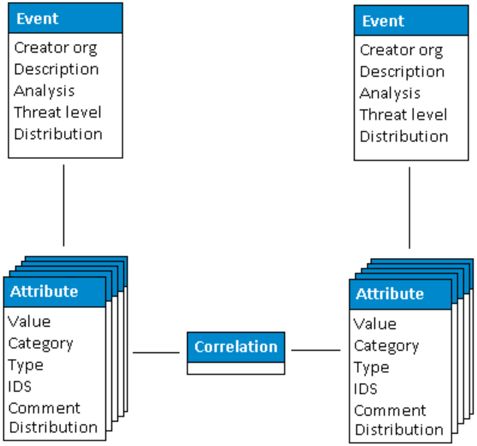
MISP - Event (MISP's basic building block)



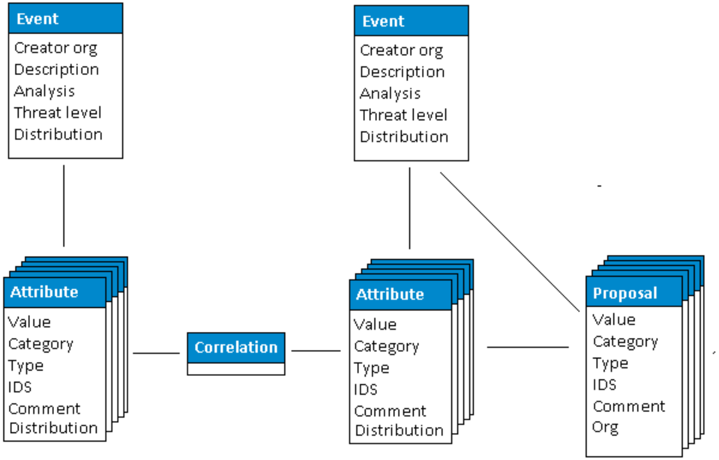
MISP - Event (Attributes, giving meaning to events)



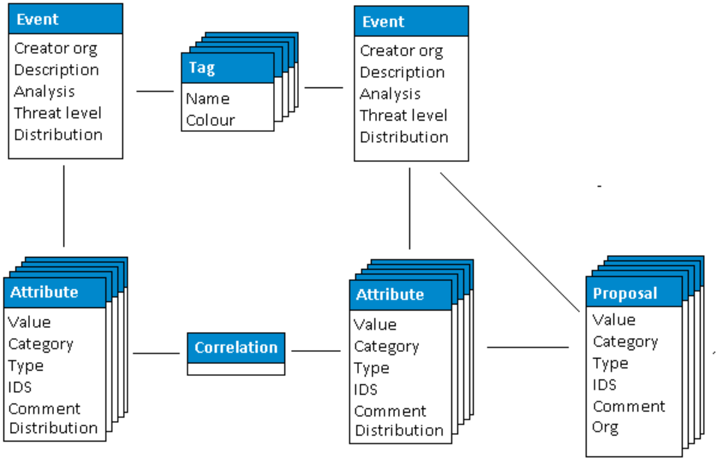
MISP - Event (Correlations on similar attributes)



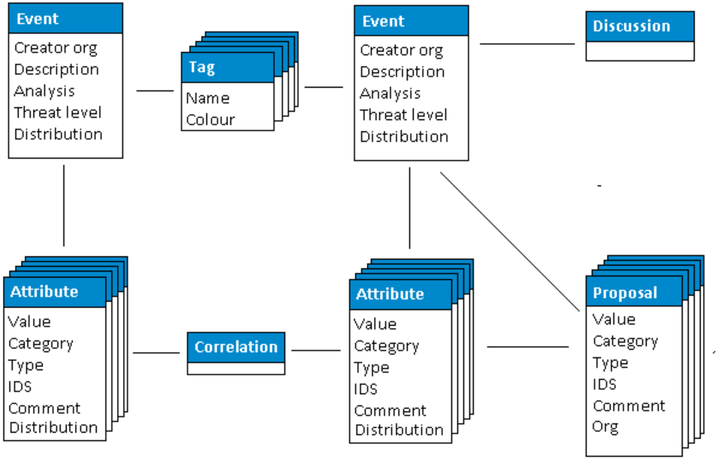
MISP - Event (Proposals)



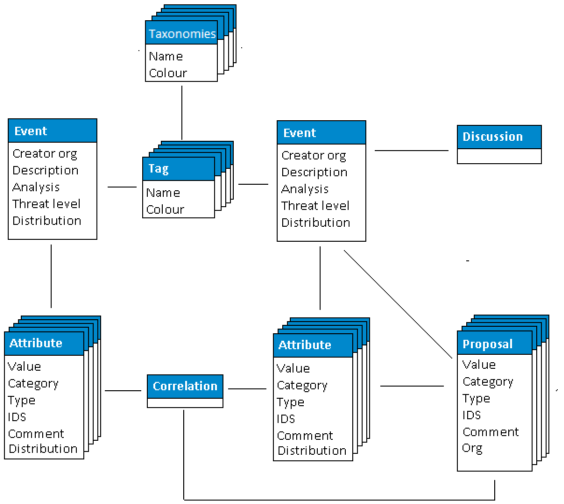
MISP - Event (Tags)



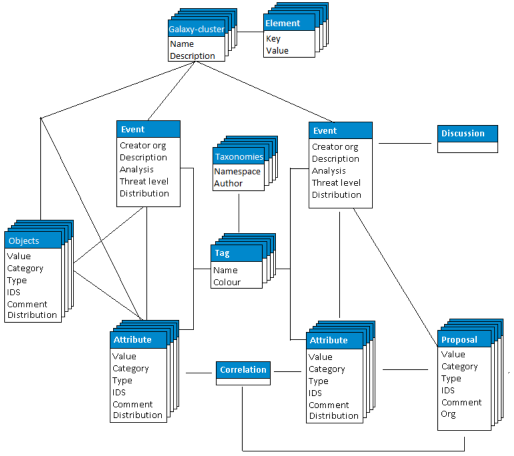
MISP - Event (Discussions)



MISP - Event (Taxonomies and proposal correlations)



MISP - Event (The state of the art MISP datamodel)



MISP - Viewing the Event Index

- Event Index
  - Event context
  - Tags
  - Distribution
  - Correlations
- Filters

MISP - Viewing an Event

- Event View
  - Event context
  - Attributes
    - Category/type, IDS, Correlations
  - Objects
  - Galaxies
  - Proposals
  - Discussions
- Tools to find what you are looking for
- Correlation graphs

MISP - Creating and populating events in various ways (demo)

- The main tools to populate an event
    - Adding attributes / batch add
    - Adding objects and how the object templates work
    - Freetext import
    - Import
    - Templates
    - Adding attachments / screenshots
    - API

MISP - Various features while adding data

- What happens automatically when adding data?
  - Automatic correlation
  - Input modification via validation and filters (regex)
  - Tagging / Galaxy Clusters
- Various ways to publish data
  - Publish with/without e-mail
  - Publishing via the API
  - Delegation

MISP - Using the data

- Correlation graphs
- Downloading the data in various formats
- Cached exports
- API (explained later)
- Collaborating with users (proposals, discussions, emails)

MISP - Sync explained (if no admin training)

- Sync connections
- Pull/push model
- Previewing instances
- Filtering the sync
- Connection test tool
- Cherry pick mode

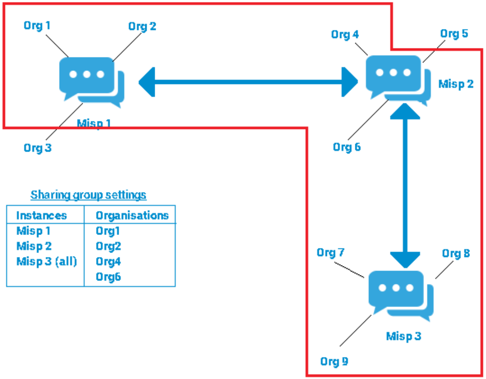
MISP - Feeds explained (if no admin training)

- Feed types (MISP, Freetext, CSV)
- Adding/editing feeds
- Previewing feeds
- Local vs Network feeds

MISP - Distributions explained

- Your Organisation Only
- This Community Only
- Connected Communities
- All Communities
- Sharing Group

MISP - Distribution and Topology



Sharing group settings

Instances	Organisations
Misp 1	Org1
Misp 2	Org2
Misp 3 (all)	Org4
	Org6

MISP - Exports and API

- Download an event
- Quick glance at the APIs
- Download search results
- Cached exports

MISP - Shorthand admin (if no admin training)

- Settings
- Troubleshooting
- Workers
- Logs

Viper - Using MISP from your terminal

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

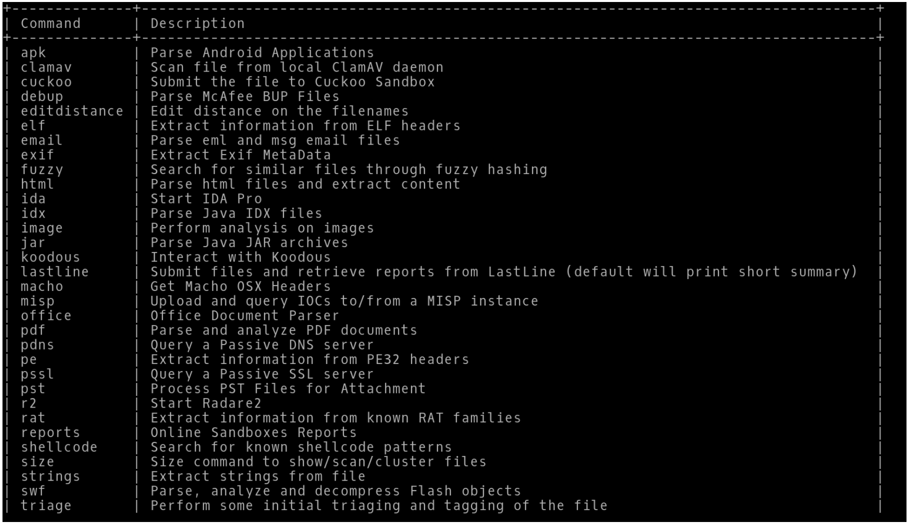
Viper - Main ideas

Viper is a **binary analysis and management framework**. *Its fundamental objective is to provide a solution to* **easily organize** *your collection of* **malware** *and* **exploit samples** *as well as your collection of* **scripts** *you created or found over the time to facilitate your daily research. Think of it as a* **Metasploit for malware researchers***: it provides a terminal interface that you can use to* **store**, **search** *and* **analyze** *arbitrary files with and a framework to* **easily create plugins** *of any sort.*

Viper

- **Solid CLI**
- Plenty of modules (PE files, *office, ELF, APK, ...)
- Connection to **3rd party services** (MISP, VirusTotal, cuckoo)
- Connectors to **3rd party tools** (IDA, radare)
- **Locale storage** of your own zoo
- Django interface is available (I've been told)

Viper

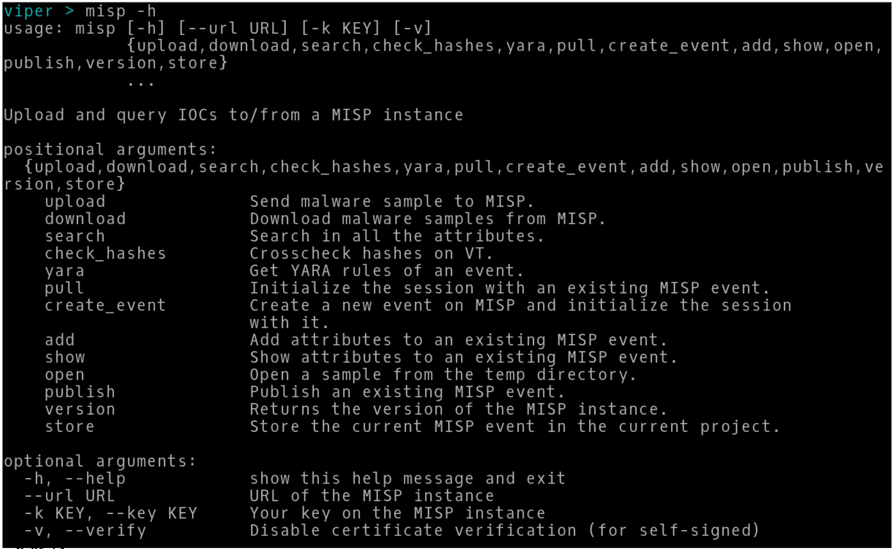


```
+--------------+-----------------------------------------------------------------------------------------+
| Command      | Description                                                                             |
+--------------+-----------------------------------------------------------------------------------------+
| apk          | Parse Android Applications                                                              |
| clamav       | Scan file from local ClamAV daemon                                                      |
| cuckoo       | Submit the file to Cuckoo Sandbox                                                       |
| debup        | Parse McAfee BUP Files                                                                  |
| editdistance | Edit distance on the filenames                                                          |
| elf          | Extract information from ELF headers                                                    |
| email        | Parse eml and msg email files                                                           |
| exif         | Extract Exif MetaData                                                                   |
| fuzzy        | Search for similar files through fuzzy hashing                                          |
| html         | Parse html files and extract content                                                    |
| ida          | Start IDA Pro                                                                           |
| idx          | Parse Java IDX files                                                                    |
| image        | Perform analysis on images                                                              |
| jar          | Parse Java JAR archives                                                                 |
| koodous      | Interact with Koodous                                                                   |
| lastline     | Submit files and retrieve reports from LastLine (default will print short summary)      |
| macho        | Get Macho OSX Headers                                                                   |
| misp         | Upload and query IOCs to/from a MISP instance                                           |
| office       | Office Document Parser                                                                  |
| pdf          | Parse and analyze PDF documents                                                         |
| pdns         | Query a Passive DNS server                                                              |
| pe           | Extract information from PE32 headers                                                   |
| pssl         | Query a Passive SSL server                                                              |
| pst          | Process PST Files for Attachment                                                        |
| r2           | Start Radare2                                                                           |
| rat          | Extract information from known RAT families                                             |
| reports      | Online Sandboxes Reports                                                                |
| shellcode    | Search for known shellcode patterns                                                     |
| size         | Size command to show/scan/cluster files                                                 |
| strings      | Extract strings from file                                                               |
| swf          | Parse, analyze and decompress Flash objects                                             |
| triage       | Perform some initial triaging and tagging of the file                                   |
```

PyMISP & Viper

- Full featured **CLI for MISP**
- **Remote storage** of your zoo
- Search / **Cross check with VirusTotal**
- Create / Update / Show / Publish Event
- Download / Upload Samples
- Mass export / Upload / Download
- Get Yara rules

MISP Module

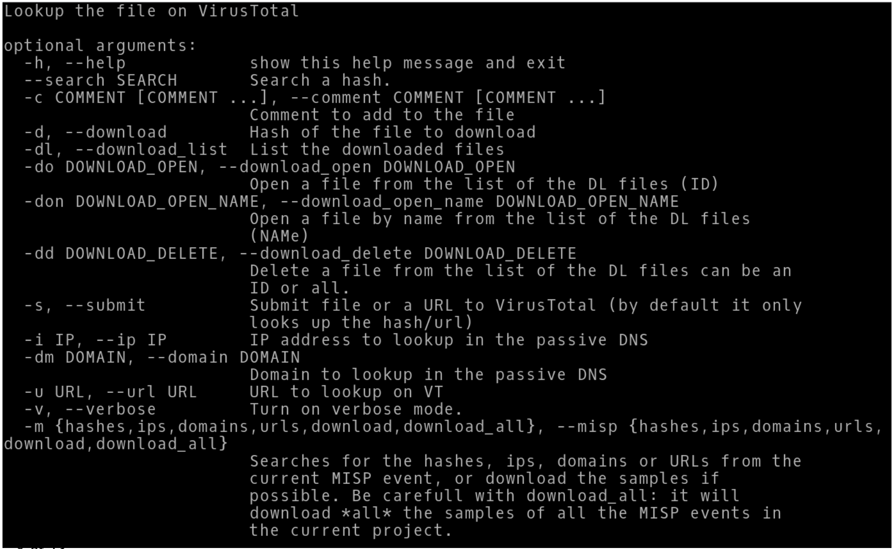


```
viper > misp -h
usage: misp [-h] [--url URL] [-k KEY] [-v]
            {upload,download,search,check_hashes,yara,pull,create_event,add,show,open,
publish,version,store}
            ...

Upload and query IOCs to/from a MISP instance

positional arguments:
  {upload,download,search,check_hashes,yara,pull,create_event,add,show,open,publish,ve
rsion,store}
    upload              Send malware sample to MISP.
    download            Download malware samples from MISP.
    search              Search in all the attributes.
    check_hashes        Crosscheck hashes on VT.
    yara                Get YARA rules of an event.
    pull                Initialize the session with an existing MISP event.
    create_event        Create a new event on MISP and initialize the session
                        with it.
    add                 Add attributes to an existing MISP event.
    show                Show attributes to an existing MISP event.
    open                Open a sample from the temp directory.
    publish             Publish an existing MISP event.
    version             Returns the version of the MISP instance.
    store               Store the current MISP event in the current project.

optional arguments:
  -h, --help            show this help message and exit
  --url URL             URL of the MISP instance
  -k KEY, --key KEY     Your key on the MISP instance
  -v, --verify          Disable certificate verification (for self-signed)
```

Viper & VT

- Searches for hashes/ips/domains/URLs from the current MISP event, or download the samples
- Download samples from current MISP event
- Download all samples from all the MISP events of the current session

VirusTotal Module



```
Lookup the file on VirusTotal

optional arguments:
  -h, --help            show this help message and exit
  --search SEARCH       Search a hash.
  -c COMMENT [COMMENT ...], --comment COMMENT [COMMENT ...]
                        Comment to add to the file
  -d, --download        Hash of the file to download
  -dl, --download_list  List the downloaded files
  -do DOWNLOAD_OPEN, --download_open DOWNLOAD_OPEN
                        Open a file from the list of the DL files (ID)
  -don DOWNLOAD_OPEN_NAME, --download_open_name DOWNLOAD_OPEN_NAME
                        Open a file by name from the list of the DL files
                        (NAMe)
  -dd DOWNLOAD_DELETE, --download_delete DOWNLOAD_DELETE
                        Delete a file from the list of the DL files can be an
                        ID or all.
  -s, --submit          Submit file or a URL to VirusTotal (by default it only
                        looks up the hash/url)
  -i IP, --ip IP        IP address to lookup in the passive DNS
  -dm DOMAIN, --domain DOMAIN
                        Domain to lookup in the passive DNS
  -u URL, --url URL     URL to lookup on VT
  -v, --verbose         Turn on verbose mode.
  -m {hashes,ips,domains,urls,download,download_all}, --misp {hashes,ips,domains,urls,
download,download_all}
                        Searches for the hashes, ips, domains or URLs from the
                        current MISP event, or download the samples if
                        possible. Be carefull with download_all: it will
                        download *all* the samples of all the MISP events in
                        the current project.
```

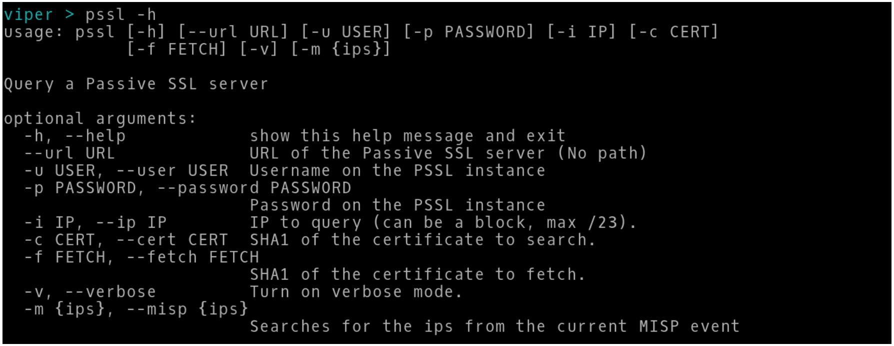
Extra features

- Link to a MISP event
- Local storage of the MISP event
- On the fly cross-check of MISP atributes with 3rd party services
- Never leaving your CLI!

Other modules

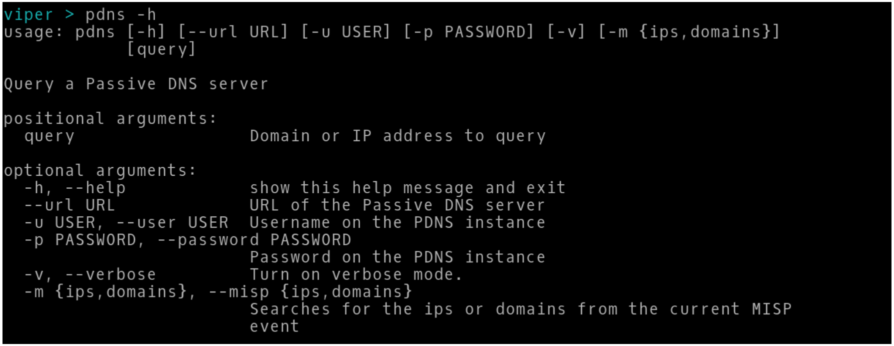
- Fully featured CLI for **Passive SSL**
- Fully featured CLI for **Passive DNS**
- Can launch Radare2 or IDA

Passive SSL



```
viper > pssl -h
usage: pssl [-h] [--url URL] [-u USER] [-p PASSWORD] [-i IP] [-c CERT]
            [-f FETCH] [-v] [-m {ips}]

Query a Passive SSL server

optional arguments:
  -h, --help            show this help message and exit
  --url URL             URL of the Passive SSL server (No path)
  -u USER, --user USER  Username on the PSSL instance
  -p PASSWORD, --password PASSWORD
                        Password on the PSSL instance
  -i IP, --ip IP        IP to query (can be a block, max /23).
  -c CERT, --cert CERT  SHA1 of the certificate to search.
  -f FETCH, --fetch FETCH
                        SHA1 of the certificate to fetch.
  -v, --verbose         Turn on verbose mode.
  -m {ips}, --misp {ips}
                        Searches for the ips from the current MISP event
```

Passive DNS



```
viper > pdns -h
usage: pdns [-h] [--url URL] [-u USER] [-p PASSWORD] [-v] [-m {ips,domains}]
            [query]

Query a Passive DNS server

positional arguments:
  query                 Domain or IP address to query

optional arguments:
  -h, --help            show this help message and exit
  --url URL             URL of the Passive DNS server
  -u USER, --user USER  Username on the PDNS instance
  -p PASSWORD, --password PASSWORD
                        Password on the PDNS instance
  -v, --verbose         Turn on verbose mode.
  -m {ips,domains}, --misp {ips,domains}
                        Searches for the ips or domains from the current MISP
                        event
```

Q&A

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

MISP Training: MISP Deployment and Integration

CIRCL
Computer Incident
Response Center
Luxembourg

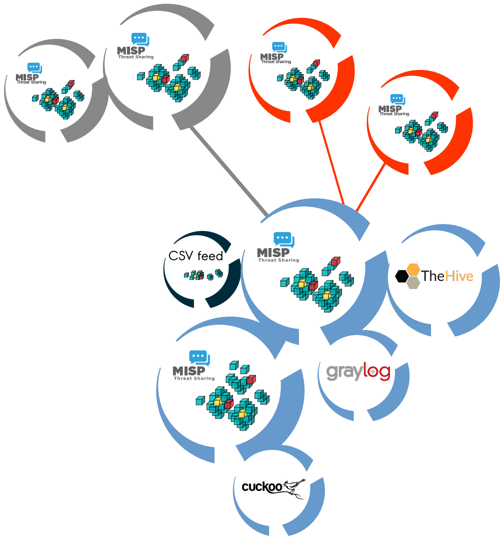
MISP
Threat Sharing

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

A Common Integration



Recommended MISP Setup

- Provisioning your MISP infrastructure depends heavily on the **number of attributes/events** (whether your dataset is below or above 50 million attributes).
- Number of MISP instances and the overall design depends on the following factors:
  - Is your community private? Are you gathering MISP events from other communities? Are you **publishing events to external** (trusted/untrusted) communities.
  - Do you plan to have **automatic tools** (e.g. sandbox analysis or low-value information needing correlation or an analyst workbench) feeding MISP?

Vendors and Formats

- There is **a jungle of formats** with some vendors having little to no interest in keeping their users autonomous.
- Attacks and threats require a **dynamic format** to be efficiently shared (e.g. from financial indicators to personal information).
- **Review your current list of formats/vendors** to ensure a limited loss of information, especially when exporting from MISP to other formats (e.g. STIX not supporting financial indicators or taxonomies/galaxies).

Use case: Normalizing OSINT and Private Feeds

- Normalizing external input and feed into MISP (e.g. feed importer).
- Comparing feeds before import (how many similarities? false-positives?).
- Evaluating quality of information before import (warning-list lookup at feed evaluation).

Connecting Devices and Tools to MISP

- One of the main goals of MISP is to feed protective or detection tools with data
  - IDSes / IPSes (e.g. Suricata, Bro, Snort format as included in Cisco products)
  - SIEMs (e.g. CEF, CSV or real-time ZMQ pub-sub or Sigma)
  - Host scanners (e.g. OpenIOC, STIX, yara rule-set, CSV)
  - Various analysis tools (e.g. Maltego)
  - DNS policies (e.g. RPZ)
- Various ways of exporting this data (downloads of the selected data, full exports, APIs)
- The idea was to leave the selection process of the subset of data to be pushed to these up to the user using APIs.

SIEM and MISP Integration

- SIEMs and MISP can be integrated with different techniques depending on the processes at your SOC or IR:
  - Pulling events (via the API) or indicator lists at **regular intervals** in a given time frame to perform lookups.
  - Subscribing to the MISP ZMQ **pub-sub channel** to directly get the published events and use these in a lookup process.
  - **Lookup expansion module** in MISP towards the SIEM to have a direct view of the attributes matched against the SIEM.
- The above options can be combined, depending on your organisation or requirements to increase coverage and detection.

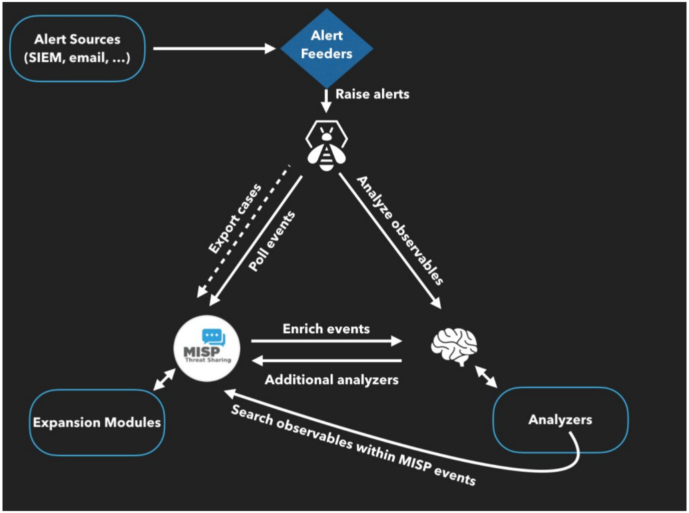
ZMQ integration: misp-dashboard

- A dashboard showing live data and statistics from the ZMQ pub-sub of one or more MISP instances.
- Building **low-latency software** by consuming pub-sub channel provides significant advantages over standard API use.
- Process information in **real-time** when it's updated, created, published or gathered in MISP.
- Demo!

New integrations: IR and threat hunting using MISP

- Close co-operation with **the Hive project** for IR
  - Interact with MISP directly from the Hive
  - Use both the MISP modules and the **Cortex** analysers in MISP or the Hive directly
- Using MISP to support your threat hunting via **McAfee OpenDXL**
- (`https://securingtomorrow.mcafee.com/business/optimize-operations/expanding-automated-threat-hunting-response-open-dxl`)

The Hive integration



Reporting Back from your Devices, Tools or Processes

As **Sightings** can be positive, negative or even based on expiration, different use cases are possible:

- **Sightings** allow users to notify a MISP instance about the activities related to an indicator.
- Activities can be from a SIEM (e.g. Splunk lookup validation or **false-positive feedback**), a NIDS or honeypot devices[1].
- Sighting can affect the API to limit the NIDS exports and improve the NIDS rule-set directly.

[1] `https://www.github.com/MISP/misp-sighting-tools`

Q&A

- info@circl.lu (if you want to join the CIRCL MISP sharing community)
- `https://github.com/MISP/` - `http://www.misp-project.org/`
- We welcome any contributions to the project, be it pull requests, ideas, github issues,...

MISP-Dashboard

Real-time overview of threat intelligence from MISP instances

Team CIRCL

info@circl.lu

September 19, 2018

MISP ZeroMQ

MISP ZeroMQ

MISP includes a flexible publish-subscribe model to allow real-time integration of the MISP activities:

- Event publication
- Attribute creation or removal
- Sighting
- User login

$\rightarrow$ Operates at global level in MISP

MISP ZeroMQ

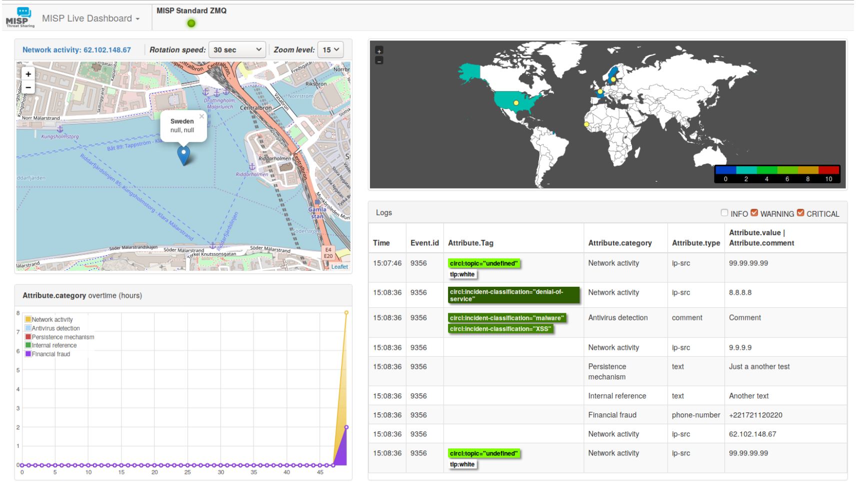
MISP ZeroMQ functionality can be used for various model of integration or to extend MISP functionalities:

- Real-time search of indicators into a SIEM[1]
- Dashboard activities
- Logging mechanisms
- Continuous indexing
- Custom software or scripting

[1] Security Information & Event Management

MISP-Dashboard: An introduction

MISP-Dashboard - Realtime activities and threat intelligence



MISP Live Dashboard

MISP Standard ZMQ

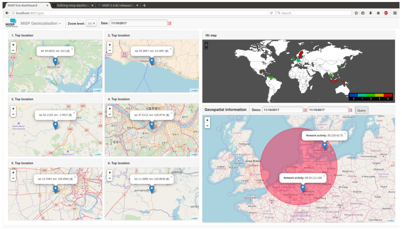
Network activity: 62.102.148.67 | Rotation speed: 30 sec | Zoom level: 15

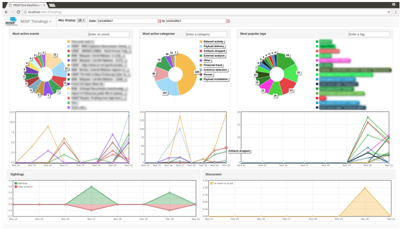
Attribute.category overtime (hours)

Logs ☐ INFO ☑ WARNING ☑ CRITICAL

Time	Event.id	Attribute.Tag	Attribute.category	Attribute.type	Attribute.value | Attribute.comment
15:07:46	9356	circl:topic="undefined" tlp:white	Network activity	ip-src	99.99.99.99
15:08:36	9356	circl:incident-classification="denial-of-service"	Network activity	ip-src	8.8.8.8
15:08:36	9356	circl:incident-classification="malware" circl:incident-classification="XSS"	Antivirus detection	comment	Comment
15:08:36	9356		Network activity	ip-src	9.9.9.9
15:08:36	9356		Persistence mechanism	text	Just a another test
15:08:36	9356		Internal reference	text	Another text
15:08:36	9356		Financial fraud	phone-number	+221721120220
15:08:36	9356		Network activity	ip-src	62.102.148.67
15:08:36	9356	circl:topic="undefined" tlp:white	Network activity	ip-src	99.99.99.99

MISP-Dashboard - Features





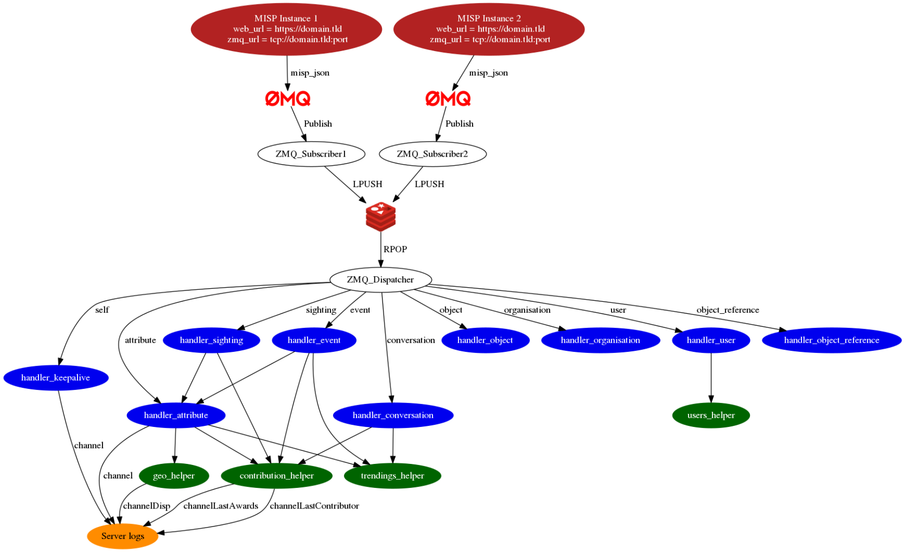
- Subscribe to multiple **ZMQ** MISP instances
- Provides historical geolocalised information
- Present an experimental **Gamification of the platform**
- Shows when and how MISP is used
- Provides real time information showing current threats and activity

MISP-Dashboard: Architecture and development

Setting up the dashboard

1. Be sure to have a running redis server: e.g.
   - `redis-server -p 6250`
2. Update your configuration in `config.cfg`
3. Activate your virtualenv:
   - `.  ./DASHENV/bin/activate`
4. Listen to the MISP feed by starting the zmq_subscriber:
   - `./zmq_subscriber.py`
5. Start the dispatcher to process received messages:
   - `./zmq_dispatcher.py`
6. Start the Flask server:
   - `./server.py`
7. Access the interface at `http://localhost:8001/`

MISP-Dashboard architecture



Writing your handler

```
# Register your handler
dico_action = {
        "misp_json":                        handler_dispatcher,
        "misp_json_event":                  handler_event,
        "misp_json_self":                   handler_keepalive,
        "misp_json_attribute":              handler_attribute,
        "misp_json_object":                 handler_object,
        "misp_json_sighting":               YOUR_CUSTOM_SIGHTINGS_HANDLER,
        "misp_json_organisation":           handler_log,
        "misp_json_user":                   handler_user,
        "misp_json_conversation":           handler_conversation,
        "misp_json_object_reference":       handler_log,
}

```

```
# Implement your handler

# e.g. user handler
def handler_user(zmq_name, jsondata):
    # json action performed by the user
    action = jsondata['action']
    # user json data
    json_user = jsondata['User']
    # organisation json data
    json_org = jsondata['Organisation']
    # organisation name
    org = json_org['name']
    # only consider user login
    if action == 'login':
        timestamp = time.time()
        # users_helper is a class to interact with the DB
        users_helper.add_user_login(timestamp, org)

```

Future development

Optimizing contribution scoring and model to encourage sharing and contributions enrichment

Increasing geolocation coverage

Global filtering capabilities

- Geolocation: Showing wanted attribute or only on specific region
- Trendings: Showing only specified taxonomies

Tighter integration with MISP

- Present in MISP by default
- Authenticated / ACL enabled version

Conclusion

MISP-Dashboard can provides realtime information to support security teams, CSIRTs or SOC showing current threats and activity by providing:

- Historical geolocalised information
- Geospatial information from specific regions
- The most active events, categories, tags, attributes, ...

It also propose a prototype of gamification of the platform providing incentive to share and contribute to the community

MISP User Training - Administration of MISP 2.4

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

MISP - VM

- VM can be downloaded at `https://www.circl.lu/misp-training/`
- Credentials
  - MISP admin: admin@admin.test/admin
  - SSH: misp/Password1234
- 2 network interfaces
  - NAT
  - Host only adapter
- Start the enrichment system by typing:
  - cd /home/misp/misp-modules/bin
  - python3 misp-modules.py

MISP - Administration

- Plan for this part of the training
  - User and Organisaton administration
  - Sharing group creation
  - Templates
  - Tags and Taxonomy
  - Whitelisting and Regexp entries
  - Setting up the synchronisation
  - Scheduled tasks
  - Feeds
  - Settings and diagnostics
  - Logging
  - Troubleshooting and updating

MISP - Creating Users

- Add new user (andras.iklody@circl.lu)
- NIDS SID, Organisation, disable user
- Fetch the PGP key
- Roles
  - Re-using standard roles
  - Creating a new custom role
- Send out credentials

MISP - Creating Organisations

- Adding a new organisation
- UUID
- Local vs External organisation
- Making an organisation self sustaining with Org Admins
- Creating a sync user

MISP - Sharing groups

- The concept of a sharing group
- Creating a sharing group
- Adding extending rights to an organisation
- Include all organisations of an instance
- Not specifying an instance
- Making a sharing group active
- Reviewing the sharing group

MISP - Templates

- Why templating?
- Create a basic template
- Text fields
- Attribute fields
- Attachment fields
- Automatic tagging

MISP - Tags and Taxonomies

- git submodule init && git submodule update
- Loading taxonomies
- Enabling taxonomies and associated tags
- Tag management
- Exportable tags

MISP - Object Templates

- git submodule init && git submodule update
- Enabling objects (and what about versioning)

MISP - Whitelisting, Regexp entries, Warninglists

- Block from exports - whitelisting
- Block from imports - blacklisting via regexp
- Modify on import - modification via regexp
- Maintaining the warninglists

MISP - Setting up the synchronisation

- Requirements - versions
- Pull/Push
- One way vs Two way synchronisation
- Exchanging sync users
- Certificates
- Filtering
- Connection test tool
- Previewing an instance
- Cherry picking and keeping the list updated

MISP - Scheduled tasks

- How to schedule the next execution
- Frequency, next execution
- What happens if a job fails?

MISP - Setting up the synchronisation

- MISP Feeds and their generation
- PyMISP
- Default free feeds
- Enabling a feed
- Previewing a feed and cherry picking
- Feed filters
- Auto tagging

MISP - Settings and diagnostics

- Settings
  - Settings interface
  - The tabs explained at a glance
  - Issues and their severity
  - Setting guidance and how to best use it

MISP - Settings and diagnostics continued

- Basic instance setup
- Additional features released as hotfixes
- Customise the look and feel of your MISP
- Default behaviour (encryption, e-mailing, default distributions)
- Maintenance mode
- Disabling the e-mail alerts for an initial sync

MISP - Settings and diagnostics continued

- Plugins
  - Enrichment Modules
  - RPZ
  - ZeroMQ

MISP - Settings and diagnostics continued

- Diagnostics
    - Updating MISP
    - Writeable Directories
    - PHP settings
    - Dependency diagnostics

MISP - Settings and diagnostics continued

- Workers
  - What do the background workers do?
  - Queues
  - Restarting workers, adding workers, removing workers
  - Worker diagnostics (queue size, jobs page)
  - Clearing worker queues
  - Worker and background job debugging

MISP - Settings and diagnostics continued

- Seeking help
  - Dump your settings to a file!
  - Make sure to sanitise it
  - Send it to us together with your issue to make our lives easier
  - Ask Github (https://github.com/MISP/MISP)
  - Have a chat with us on gitter (https://gitter.im/MISP/MISP)
  - Ask the MISP mailing list
  - If this is security related, drop us a PGP encrypted email to `mailto:info@circl.lu`

MISP - Logging

- Audit logs in MISP
- Enable IP logging / API logging
- Search the logs, the fields explained
- External logs
  - /var/www/MISP/app/tmp/logs/error.log
  - /var/www/MISP/app/tmp/logs/resque-worker-error.log
  - /var/www/MISP/app/tmp/logs/resque-scheduler-error.log
  - /var/www/MISP/app/tmp/logs/resque-[date].log
  - /var/www/MISP/app/tmp/logs/error.log
  - apache access logs

MISP - Updating MISP

- git pull
- git submodule init && git submodule update
- reset the permissions if it goes wrong according to the INSTALL.txt
- when MISP complains about missing fields, make sure to clear the caches
    - in /var/www/MISP/app/tmp/cache/models remove myapp*
    - in /var/www/MISP/app/tmp/cache/persistent remove myapp*
- No additional action required on hotfix level
- Read the migration guide for major and minor version changes

MISP - Administrative tools

- Upgrade scripts for minor / major versions
- Maintenance scripts

Information Sharing and Taxonomies

Practical Classification of Threat Indicators using MISP

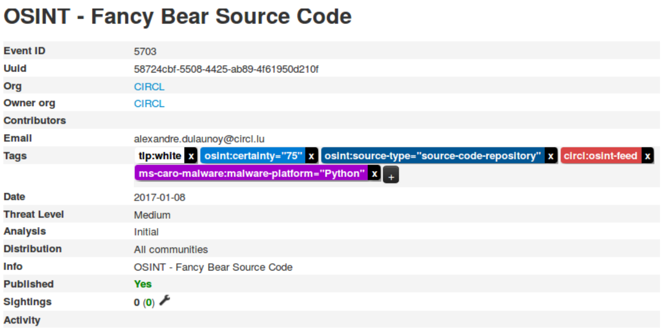
CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

From Tagging to Flexible Taxonomies



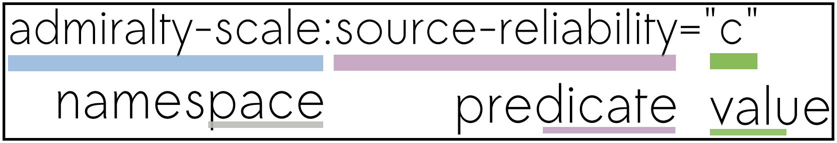
OSINT - Fancy Bear Source Code

Event ID	5703
Uuid	58724cbf-5508-4425-ab89-4f61950d210f
Org	CIRCL
Owner org	CIRCL
Contributors	
Email	alexandre.dulaunoy@circl.lu
Tags	tlp:white x osint:certainty="75" x osint:source-type="source-code-repository" x circl:osint-feed x ms-caro-malware:malware-platform="Python" x +
Date	2017-01-08
Threat Level	Medium
Analysis	Initial
Distribution	All communities
Info	OSINT - Fancy Bear Source Code
Published	Yes
Sightings	0 (0)
Activity	

- Tagging is a simple way to attach a classification to an event or an attribute.
- In the early version of MISP, tagging was local to an instance.
- **Classification must be globally used to be efficient**.
- After evaluating different solutions of classification, we build a new scheme using the concept of machine tags.

Machine Tags

- Triple tag or machine tag was introduced in 2004 to extend geotagging on images.



- A machine tag is just a tag expressed in way that allows systems to parse and interpret it.
- Still have a human-readable version:
  - admiralty-scale:Source Reliability="Fairly reliable"

MISP Taxonomies

- Taxonomies are implemented in a simple JSON format.
- Anyone can create their own taxonomy or reuse an existing one.
- The taxonomies are in an independent git repository[1].
- These can be freely reused and integrated in other threat intel tools.
- Taxonomies are licensed under CC0 (public domain) except if the taxonomy author decided to use another license.

[1] `https://www.github.com/MISP/misp-taxonomies/`

Existing Taxonomies

- NATO - **Admiralty Scale**
- CIRCL Taxonomy - **Schemes of Classification in Incident Response and Detection**
- eCSIRT and IntelMQ incident classification
- EUCI **EU classified information marking**
- Information Security Marking Metadata from DNI (Director of National Intelligence - US)
- NATO Classification Marking
- OSINT **Open Source Intelligence - Classification**
- TLP - **Traffic Light Protocol**
- Vocabulary for Event Recording and Incident Sharing - **VERIS**
- and many more like ENISA, Europol, or the draft FIRST SIG Information Exchange Policy.

Want to write your own taxonomy? 1/2

```
{
  "namespace": "admiralty-scale",
  "description": "The Admiralty Scale (also called the NATO
      System) is used to rank the reliability of a source and
      the credibility of an information.",
  "version": 1,
  "predicates": [
    {
      "value": "source-reliability",
      "expanded": "Source Reliability"
    },
    {
      "value": "information-credibility",
      "expanded": "Information Credibility"
    }
  ],
....
```

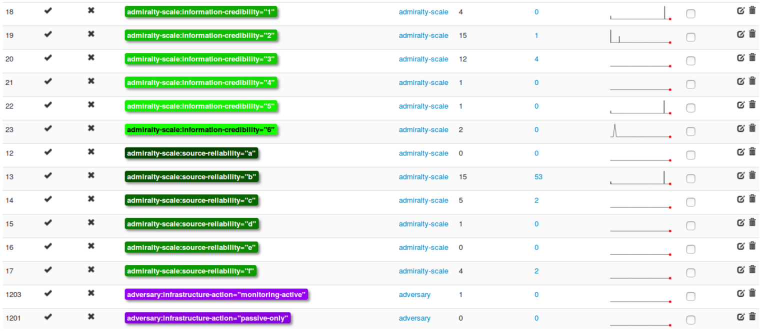
Want to write your own taxonomy? 2/2

```
{
  "values": [
    {
      "predicate": "source-reliability",
      "entry": [
        {
          "value": "a",
          "expanded": "Completely reliable"
        },
....
```

- Publishing your taxonomy is as easy as a simple git pull request on misp-taxonomies[2].

[2] https://github.com/MISP/misp-taxonomies

How are taxonomies integrated in MISP?

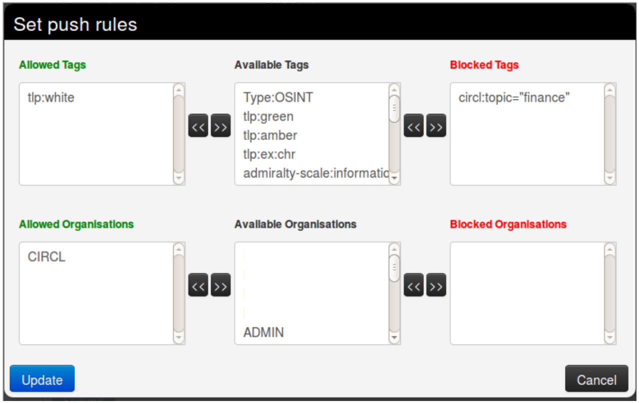


18	✓	✗	admiralty-scale:information-credibility="1"	admiralty-scale	4	0	☐	
19	✓	✗	admiralty-scale:information-credibility="2"	admiralty-scale	15	1	☐	
20	✓	✗	admiralty-scale:information-credibility="3"	admiralty-scale	12	4	☐	
21	✓	✗	admiralty-scale:information-credibility="4"	admiralty-scale	1	0	☐	
22	✓	✗	admiralty-scale:information-credibility="5"	admiralty-scale	1	0	☐	
23	✓	✗	admiralty-scale:information-credibility="6"	admiralty-scale	2	0	☐	
12	✓	✗	admiralty-scale:source-reliability="a"	admiralty-scale	0	0	☐	
13	✓	✗	admiralty-scale:source-reliability="b"	admiralty-scale	15	53	☐	
14	✓	✗	admiralty-scale:source-reliability="c"	admiralty-scale	5	2	☐	
15	✓	✗	admiralty-scale:source-reliability="d"	admiralty-scale	1	0	☐	
16	✓	✗	admiralty-scale:source-reliability="e"	admiralty-scale	0	0	☐	
17	✓	✗	admiralty-scale:source-reliability="f"	admiralty-scale	4	2	☐	
1203	✓	✗	adversary:infrastructure-action="monitoring-active"	adversary	1	0	☐	
1201	✓	✗	adversary:infrastructure-action="passive-only"	adversary	0	0	☐	

- MISP administrator can just import (or even cherry pick) the namespace or predicates they want to use as tag.
- Tags can be exported to other instances.
- Tags are also accessible via the MISP REST API.

Filtering the distribution of events among MISP instances

- Applying rules for distribution based on tags:


Set push rules

Allowed Tags	Available Tags	Blocked Tags
tlp:white	Type:OSINT tlp:green tlp:amber tlp:ex:chr admiralty-scale:informatic	circl:topic="finance"

Allowed Organisations	Available Organisations	Blocked Organisations
CIRCL	ADMIN	

Other use cases using MISP taxonomies

- Tags can be used to set events or attributes for **further processing by external tools** (e.g. VirusTotal auto-expansion using Viper).
- Ensuring a classification manager **classies the events before release** (e.g. release of information from air-gapped/classified networks).
- **Enriching IDS export** with tags to fit your NIDS deployment.
- Using **IntelMQ** and MISP together to process events (tags limited per organization introduced in MISP 2.4.49).

Future functionalities related to MISP taxonomies

- **Sighting** support (thanks to NCSC-NL) is integrated in MISP allowing to auto expire IOC based on user detection.
- Adjusting taxonomies (adding/removing tags) based on their score or visibility via sighting.
- Simple taxonomy editors to **help non-technical users** to create their taxonomies.
- **Filtering mechanisms** in MISP to rename or replace taxonomies/tags at pull and push synchronisation.
- More public taxonomies to be included.

PyTaxonomies

- **Python module** to handle the taxonomies
- **Offline** and online mode (fetch the newest taxonomies from GitHub)
- Simple **search** to make tagging easy
- Totally independant from MISP
- **No external dependencies** in offline mode
- Python3 only
- Can be used to create & **dump a new taxonomy**

PyTaxonomies

```
from pytaxonomies import Taxonomies
taxonomies = Taxonomies()
taxonomies.version
# => '20160725'
taxonomies.description
# => 'Manifest file of MISP taxonomies available.'
list(taxonomies.keys())
# => ['tlp', 'eu-critical-sectors', 'de-vs', 'osint', 'circl', 'veris',
#         'ecsirt', 'dhs-ciip-sectors', 'fr-classif', 'misp', 'admiralty-scale', ...]
taxonomies.get('enisa').description
# 'The present threat taxonomy is an initial version that has been developed on
# the basis of available ENISA material. This material has been used as an ENISA-internal
# structuring aid for information collection and threat  consolidation purposes.
# It emerged in the time period 2012-2015.'
print(taxonomies.get('circl'))
# circl:incident-classification="vulnerability"
# circl:incident-classification="malware"
# circl:incident-classification="fastflux"
# circl:incident-classification="system-compromise"
# circl:incident-classification="sql-injection"
# ....
print(taxonomies.get('circl').machinetags_expanded())
# circl:incident-classification="Phishing"
# circl:incident-classification="Malware"
# circl:incident-classification="XSS"
# circl:incident-classification="Copyright issue"
# circl:incident-classification="Spam"
# circl:incident-classification="SQL Injection"
```

The dilemma of false-positive

- False-positive is a **common issue** in threat intelligence sharing.
- It's often a contextual issue:
  - false-positive might be different per community of users sharing information.
  - organization might have their **own view** on false-positive.
- Based on the success of the MISP taxonomy model, we build misp-warninglists.

MISP warning lists

- misp-warninglists are lists of well-known indicators that can be associated to potential false positives, errors or mistakes.
- Simple JSON files

```
{
  "name": "List of known public DNS resolvers",
  "version": 2,
  "description": "Event contains one or more public DNS
      resolvers as attribute with an IDS flag set",
  "matching_attributes": [
    "ip-src",
    "ip-dst"
  ],
  "list": [
    "8.8.8.8",
    "8.8.4.4", ...]
}
```

MISP warning lists

- The warning lists are integrated in MISP to display an info/warning box at the event and attribute level.
- Enforceable via the API where all attributes that have a hit on a warninglist will be excluded.
- This can be enabled at MISP instance level.
- Default warning lists can be enabled or disabled like **known public resolver**, **multicast IP addresses**, **hashes for empty values**, **rfc1918**, **TLDs** or **known google domains**.
- The warning lists can be expanded or added in JSON locally or via pull requests.
- Warning lists can be also used for **critical or core infrastructure warning**, **personally identifiable information**...

Q&A

- `https://github.com/MISP/MISP`
- `https://github.com/MISP/misp-taxonomies`
- `https://github.com/MISP/PyTaxonomies`
- `https://github.com/MISP/misp-warninglists`
- info@circl.lu (if you want to join one of the MISP community operated by CIRCL)
- PGP key fingerprint: CA57 2205 C002 4E06 BA70 BE89 EAAD CFFC 22BD 4CD5

Extending MISP with Python modules

MISP - Malware Information Sharing Platform & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

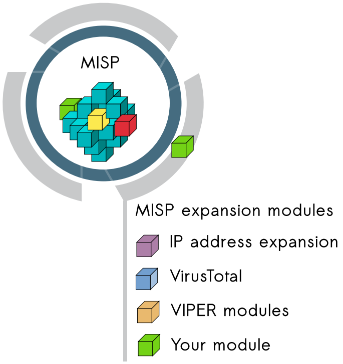
Why we want to go more modular...

- Ways to extend MISP before modules
  - APIs (PyMISP, MISP API)
    - Works really well
    - **No integration with the UI**
  - Change the core code
    - Have to change the core of MISP, diverge from upstream
    - Needs a deep understanding of MISP internals
    - Let's not beat around the bush: **Everyone hates PHP**

Goals for the module system

- Have a way to extend MISP without altering the core
- Get started **quickly** without a need to study the internals
- Make the **modules as light weight as possible**
    - Module developers should only have to worry about the data transformation
    - Modules should have a simple and clean skeleton
- In a friendlier language - **Python**

MISP modules - extending MISP with Python scripts



- Extending MISP with expansion modules with zero customization in MISP.
- A simple ReST API between the modules and MISP allowing auto-discovery of new modules with their features.
- Benefit from existing Python modules in Viper or any other tools.
- MISP modules functionnality introduced in MISP 2.4.28.
- MISP import/export modules introduced in MISP 2.4.50.

MISP modules - installation

- MISP modules can be run on the same system or on a remote server.
- Python 3 is required to run MISP modules.
  - sudo apt-get install python3-dev python3-pip libpq5
  - cd /usr/local/src/
  - sudo git clone https://github.com/MISP/misp-modules.git
  - cd misp-modules
  - sudo pip3 install -I -r REQUIREMENTS
  - sudo pip3 install -I .
  - sudo vi /etc/rc.local, add this line: ‘sudo -u www-data misp-modules -s &‘

MISP modules - Simple REST API mechanism

- http://127.0.0.1:6666/modules - introspection interface to get **all modules available**
  - returns a JSON with a description of each module
- http://127.0.0.1:6666/query - interface to **query a specific module**
  - to send a JSON to query the module
- **MISP autodiscovers** the available modules and the MISP site administrator can enable modules as they wish.
- If a configuration is required for a module, **MISP adds automatically the option** in the server settings.

Finding available MISP modules

- curl -s http://127.0.0.1:6666/modules

```
{
"type": "expansion",
"name": "dns",
"meta": {
  "module-type": [
    "expansion",
    "hover"
  ],
  "description": "Simple DNS expansion
     service to resolve IP address from
     MISP attributes",
  "author": "Alexandre Dulaunoy",
  "version": "0.1"
},
"mispattributes": {
  "output": [
    "ip-src",
    "ip-dst"
  ],
  "input": [
    "hostname",
    "domain"
  ]
}
```

Querying a module

- curl -s http://127.0.0.1:6666/query -H "Content-Type: application/json" –data @body.json -X POST

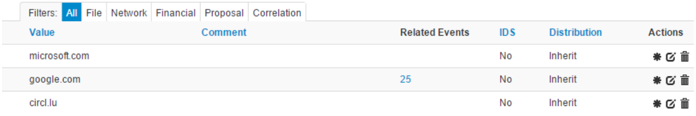
body.json

```
{"module": "dns", "hostname": "www.circl.lu"}
```

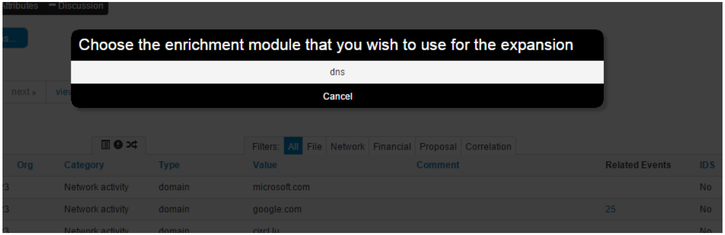
- and the response of the dns module:

```
{"results": [{"values": ["149.13.33.14"],
 "types": ["ip-src", "ip-dst"]}]}
```

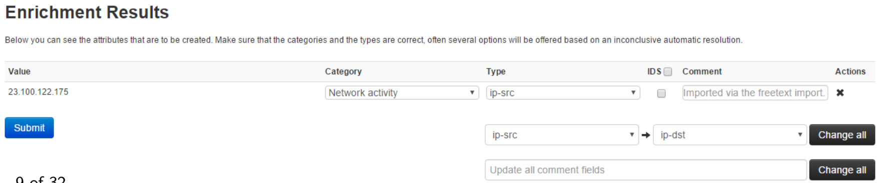
MISP modules - How it's integrated in the UI?



Value	Comment	Related Events	IDS	Distribution	Actions
microsoft.com			No	Inherit	
google.com		25	No	Inherit	
circl.lu			No	Inherit	


Choose the enrichment module that you wish to use for the expansion

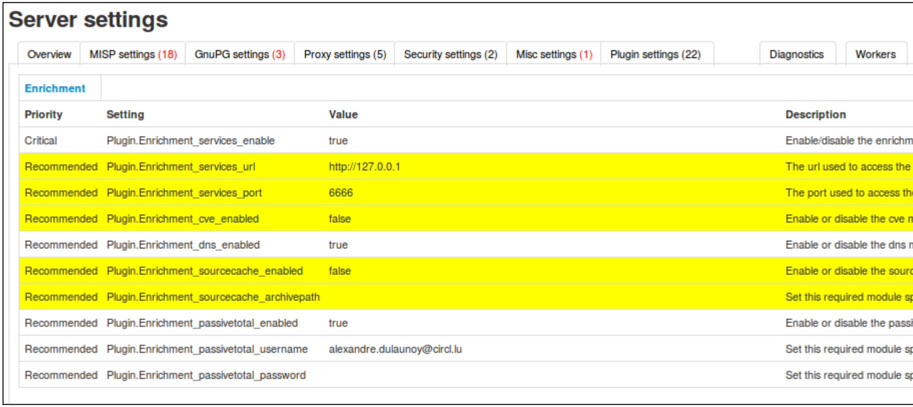
Org	Category	Type	Value	Comment	Related Events	IDS
3	Network activity	domain	microsoft.com			No
3	Network activity	domain	google.com		25	No
3	Network activity	domain	circl.lu			No


Enrichment Results

Below you can see the attributes that are to be created. Make sure that the categories and the types are correct, often several options will be offered based on an inconclusive automatic resolution.

Value	Category	Type	IDS	Comment	Actions
23.100.122.175	Network activity	ip-src	☐	Imported via the freetext import.	✖

MISP modules - configuration in the UI



Server settings

Overview | MISP settings (18) | GnuPG settings (3) | Proxy settings (5) | Security settings (2) | Misc settings (1) | Plugin settings (22) | Diagnostics | Workers

Enrichment

Priority	Setting	Value	Description
Critical	Plugin.Enrichment_services_enable	true	Enable/disable the enrichm
Recommended	Plugin.Enrichment_services_url	http://127.0.0.1	The url used to access the
Recommended	Plugin.Enrichment_services_port	6666	The port used to access th
Recommended	Plugin.Enrichment_cve_enabled	false	Enable or disable the cve m
Recommended	Plugin.Enrichment_dns_enabled	true	Enable or disable the dns m
Recommended	Plugin.Enrichment_sourcecache_enabled	false	Enable or disable the sourc
Recommended	Plugin.Enrichment_sourcecache_archivepath		Set this required module sp
Recommended	Plugin.Enrichment_passivetotal_enabled	true	Enable or disable the passi
Recommended	Plugin.Enrichment_passivetotal_username	alexandre.dulaunoy@circl.lu	Set this required module sp
Recommended	Plugin.Enrichment_passivetotal_password		Set this required module sp

MISP modules - main types of modules

- Expansion modules - enrich data that is in MISP
  - Hover type - showing the expanded values directly on the attributes
  - Expansion type - showing and adding the expanded values via a proposal form
- Import modules - import new data into MISP
- Export modules - export existing data from MISP

Creating your Expansion module (Skeleton)

```
import json
import dns.resolver

misperrors = {'error' : 'Error'}
mispattributes = {'input': [], 'output': []}
moduleinfo = {'version': '', 'author': '',
              'description': '', 'module-type': []}

def handler(q=False):
    if q is False:
        return False
    request = json.loads(q)
    r = {'results': [{'types': [], 'values':[]}]}
    return r
def introspection():
    return mispattributes
def version():
    return moduleinfo
```

Creating your Expansion module (metadata 1)

```
misperrors = {'error' : 'Error'}
mispattributes = {'input': ['hostname', 'domain'], 'output': ['ip-src', 'ip-dst']}
moduleinfo = {'version': '', 'author': '',
              'description': '', 'module-type': []}
```

Creating your Expansion module (metadata 2)

```
misperrors = {'error' : 'Error'}
mispattributes = {'input': ['hostname', 'domain'], 'output': ['ip-src', 'ip-dst']}
moduleinfo = {'version': '0.1', 'author': 'Alexandre Dulaunoy',
              'description': 'Simple DNS expansion service to
      resolve IP address from MISP attributes', 'module-type': ['expansion','hover']}
```

Creating your Expansion module (handler 1)

```
def handler(q=False):
    if q is False:
        return False
    request = json.loads(q)
    # MAGIC
    # MORE MAGIC
    r = {'results': [
        {'types': output_types, 'values':values},
        {'types': output_types2, 'values':values2}
    ]}
    return r
```

Creating your Expansion module (handler 2)

```
if request.get('hostname'):
    toquery = request['hostname']
elif request.get('domain'):
    toquery = request['domain']
else:
    return False
r = dns.resolver.Resolver()
r.timeout = 2
r.lifetime = 2
r.nameservers = ['8.8.8.8']
try:
    answer = r.query(toquery, 'A')
except dns.resolver.NXDOMAIN:
    misperrors['error'] = "NXDOMAIN"
    return misperrors
except dns.exception.Timeout:
    misperrors['error'] = "Timeout"
    return misperrors
except:
    misperrors['error'] = "DNS␣resolving␣error"
    return misperrors
r = {'results': [{'types': mispattributes['output'], 'values':[str(answer[0])]}]}
return r
```

Creating your module - finished DNS module

```
import json
import dns.resolver
misperrors = {'error' : 'Error'}
mispattributes = {'input': ['hostname', 'domain'], 'output': ['ip-src', 'ip-dst']}
moduleinfo = {'version': '0.1', 'author': 'Alexandre Dulaunoy',
              'description': 'Simple DNS expansion service to resolve IP address from MISP attributes', 'module-type': ['expansion','hover']}
def handler(q=False):
    if q is False:
        return False
    request = json.loads(q)
    if request.get('hostname'):
        toquery = request['hostname']
    elif request.get('domain'):
        toquery = request['domain']
    else:
        return False
    r = dns.resolver.Resolver()
    r.timeout = 2
    r.lifetime = 2
    r.nameservers = ['8.8.8.8']
    try:
        answer = r.query(toquery, 'A')
    except dns.resolver.NXDOMAIN:
        misperrors['error'] = "NXDOMAIN"
        return misperrors
    except dns.exception.Timeout:
        misperrors['error'] = "Timeout"
        return misperrors
    except:
        misperrors['error'] = "DNS resolving error"
        return misperrors
    r = {'results': [{'types': mispattributes['output'], 'values':[str(answer[0])]}]}
    return r

def introspection():
    return mispattributes

def version():
    return moduleinfo
```

Testing your module

- Copy your module dns.py in modules/expansion/
- Restart the server misp-modules.py

```
[adulau:~/git/misp-modules/bin]$ python3 misp-modules.py
2016-03-20 19:25:43,748 - misp-modules - INFO - MISP modules passivetotal imported
2016-03-20 19:25:43,787 - misp-modules - INFO - MISP modules sourcecache imported
2016-03-20 19:25:43,789 - misp-modules - INFO - MISP modules cve imported
2016-03-20 19:25:43,790 - misp-modules - INFO - MISP modules dns imported
2016-03-20 19:25:43,797 - misp-modules - INFO - MISP modules server started on TCP port 6666
```

- Check if your module is present in the introspection
- curl -s http://127.0.0.1:6666/modules
- If yes, test it directly with MISP or via curl

Code samples (Configuration)

```
# Configuration at the top
moduleconfig = ['username', 'password']
# Code block in the handler
    if request.get('config'):
        if (request['config'].get('username') is None) or (request['config'].get('password') is None):
            misperrors['error'] = 'CIRCL Passive SSL authentication is missing'
            return misperrors

    x = pypssl.PyPSSL(basic_auth=(request['config']['username'], request['config']['password']))
```

Default expansion module set

- asn history
- CIRCL Passive DNS
- CIRCL Passive SSL
- Country code lookup
- CVE information expansion
- DNS resolver
- DomainTools
- eupi (checking url in phishing database)
- IntelMQ (experimental)
- ipasn
- PassiveTotal - http://blog.passivetotal.org/misp-sharing-done-differently
- sourcecache
- Virustotal
- Whois

Import modules

- Similar to expansion modules
- Input is a file upload or a text paste
- Output is a list of parsed attributes to be editend and verified by the user
- System is still new but some modules already exist
  - Cuckoo JSON import
  - email import
  - OCR module
  - Simple STIX import module
- Many ideas for future modules (OpenIOC import, connector to sandboxes, STIX 2.0, etc)

Creating your Import module (Skeleton)

```
import json

misperrors = {'error' : 'Error'}
userConfig = {
                'number1': {
                    'type': 'Integer',
                    'regex': '/^[0-4]$/i',
                    'errorMessage': 'Expected␣a␣number␣in␣range␣[0-4]',
                    'message': 'Column␣number␣used␣for␣value'
                }
            };
inputSource = ['file', 'paste']
moduleinfo = {'version': '', 'author': '',
              'description': '', 'module-type': ['import']}
moduleconfig=[]

def handler(q=False):
    if q is False:
        return False
    request = json.loads(q)
    request["data"] = base64.b64decode(request["data"])
    r = {'results': [{'categories': [], 'types': [], 'values':[]}]}
    return r

def introspection():
    return {'userConfig': userConfig, 'inputSource': inputSource, 'moduleConfig': moduleConfig}

def version():
    return moduleinfo
```

Creating your import module (userConfig and inputSource)

```
userConfig = {
    'number1': {
        'type': 'Integer',
        'regex': '/^[0-4]$/i',
        'errorMessage': 'Expected␣a␣number␣in␣range␣[0-4]',
        'message': 'Column␣number␣used␣for␣value'
    }
};
inputSource = ['file', 'paste']
```

Creating your import module (Handler)

```
def handler(q=False):
    if q is False:
        return False
    request = json.loads(q)
    request["data"] = base64.b64decode(request["data"])
    r = {'results': [{'categories': [], 'types': [], 'values':[]}]}
    return r
```

Creating your import module (Introspection)

```
def introspection():
    modulesetup = {}
    try:
        userConfig
        modulesetup['userConfig'] = userConfig
    except NameError:
        pass
    try:
        moduleConfig
        modulesetup['moduleConfig'] = moduleConfig
    except NameError:
        pass
    try:
        inputSource
        modulesetup['inputSource'] = inputSource
    except NameError:
        pass
    return modulesetup
```

Export modules

- Input is currently only a single event
- Dynamic settings
- Later on to be expanded to event collections / attribute collections
- Output is a file in the export format served back to the user
- Export modules was recently introduced but a CEF export module already available
- Lots of ideas for upcoming modules and including interaction with misp-darwin

Creating your Export module (Skeleton)

```
import json
inputSource = ['event']
outputFileExtension = 'txt'
responseType = 'application/txt'
moduleinfo = {'version': '0.1', 'author': 'Andras Iklody',
              'description': 'Skeleton export module',
              'module-type': ['export']}

def handler(q=False):
    if q is False:
        return False
    request = json.loads(q)
    # insert your magic here!
    output = my_magic(request["data"])
    r = {"data":base64.b64encode(output.encode('utf-8')).decode('utf-8')}
    return r

def introspection():
    return {'userConfig': userConfig, 'inputSource': inputSource, 'moduleConfig': moduleConfig, 'outputFileExtension': outputFileExtension}

def version():
    return moduleinfo
```

Creating your export module (settings)

```
inputSource = ['event']
outputFileExtension = 'txt'
responseType = 'application/txt'
```

Creating your export module (handler)

```
def handler(q=False):
    if q is False:
        return False
    request = json.loads(q)
    # insert your magic here!
    output = my_magic(request["data"])
    r = {"data":base64.b64encode(output.encode('utf-8')).decode('utf-8')}
    return r
```

Creating your export module (introspection)

```
def introspection():
    modulesetup = {}
    try:
        responseType
        modulesetup['responseType'] = responseType
    except NameError:
        pass
    try:
        userConfig
        modulesetup['userConfig'] = userConfig
    except NameError:
        pass
    try:
        moduleConfig
        modulesetup['moduleConfig'] = moduleConfig
    except NameError:
        pass
    try:
        outputFileExtension
        modulesetup['outputFileExtension'] = outputFileExtension
    except NameError:
        pass
    try:
        inputSource
```

Upcoming additions to the module system - General

- Expose the modules to the APIs
- Move the modules to background processes with a messaging system
- Difficulty is dealing with uncertain results on import (without the user having final say)

Q&A

MISP
Threat Sharing

- https://github.com/MISP/misp-modules
- https://github.com/MISP/
- We welcome new modules and pull requests.
- MISP modules can be designed as standalone application.

MISP Training: Galaxies

CIRCL
Computer Incident
Response Center
Luxembourg

MISP
Threat Sharing

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

MISP Galaxies

- MISP started out as a platform for technical indicator sharing
- The need for a way to describe threat actors, tools and other commonalities became more and more pressing
- **Taxonomies quickly became essential for classifying events**
- The weakness of the tagging aproach is that it's not very descriptive
- We needed a way to attach **more complex structures to data**
- Also, with the different naming conventions for the same "thing" attribution was a mess
- This is where the Galaxy concept came in

Solution

- Pre-crafted galaxy "clusters" via GitHub project
- Attach them to an event and attribute(s)
- The main design principle was that these higher level informations are meant for human consumption
- This means flexibility - key value pairs, describe them dynamically
- Technical indicators remain strongly typed and validated, galaxies are loose key value lists

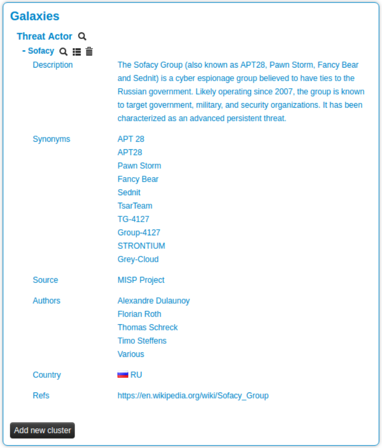
The galaxy object stack

- **Galaxy**: The type of data described (Threat actor, Tool, ...)
- **Cluster**: An individual instance of the galaxy (Sofacy, Turla, ...)
- **Element**: Key value pairs describing the cluster (Country: RU, Synonym: APT28, Fancy Bear)
- **Reference**: Referenced galaxy cluster (Such as a threat actor using a specific tool)

(some) Existing galaxies

- **Exploit-Kit**: An enumeration of known exploitation kits used by adversaries
- **Microsoft activity group**: Adversary groups as defined by Microsoft
- **Preventive measure**: Potential preventive measures against threats
- **Ransomware**: List of known ransomwares
- **TDS**: Traffic Direction System used by adversaries
- **Threat-Actor**: Known or estimated adversary groups
- **Tool**: Tools used by adversaries (from Malware to common tools)
- **MITRE ATT&CK**: Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK^{TM})

What a cluster looks like


Galaxies

Threat Actor

- Sofacy

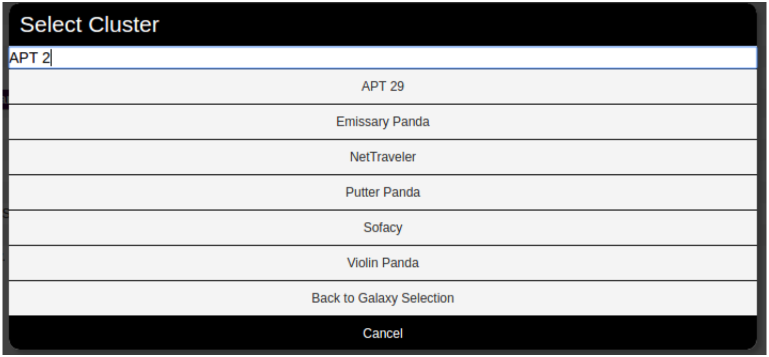
Description	The Sofacy Group (also known as APT28, Pawn Storm, Fancy Bear and Sednit) is a cyber espionage group believed to have ties to the Russian government. Likely operating since 2007, the group is known to target government, military, and security organizations. It has been characterized as an advanced persistent threat.
Synonyms	APT 28 APT28 Pawn Storm Fancy Bear Sednit TsarTeam TG-4127 Group-4127 STRONTIUM Grey-Cloud
Source	MISP Project
Authors	Alexandre Dulaunoy Florian Roth Thomas Schreck Timo Steffens Various
Country	RU
Refs	https://en.wikipedia.org/wiki/Sofacy_Group

Attaching clusters to events

- Internally simply using a taxonomy-like tag to attach them to events
- Example: misp-galaxy:threat-actor="Sofacy"
- **Synchronisation works out of the box** with older instances too. They will simply see the tags until they upgrade.
- Currently, as mentioned we rely on the community's contribution of galaxies

Attaching clusters

- Use a searchable synonym database to find what you're after



Creating your own galaxy

- Creating galaxy clusters has to be straightforward to get the community to contribute
- Building on the prior success of the taxonomies and warninglists
- Simple JSON format in similar fashion
- Just drop the JSON in the proper directory and let MISP ingest it
- We always look forward to contributions to our galaxies repository

Galaxy JSON

- If you want to create a completely new galaxy instead of enriching an existing one

```
{
    "name" : "Threat Actor",
    "type" : "threat-actor",
    "description": "Threat actors are characteristics of
        malicious actors (or adversaries) representing a cyber
        attack threat including presumed intent and
        historically observed behaviour.",
    "version": 1,
    "uuid": "698774c7-8022-42c4-917f-8d6e4f06ada3"
}
```

Cluster JSON

- Clusters contain the meat of the data
- Skeleton structure as follows

```
{
  "values": [
    {
      "meta": {},
      "description": "",
      "value": "",
      "related_clusters": [{}],
    }
  ]
}
```

Cluster JSON value example

```
    {
      "meta": {
        "synonyms": [
            "APT 28", "APT28", "Pawn Storm", "Fancy Bear",
            "Sednit", "TsarTeam", "TG-4127", "Group-4127",
            "STRONTIUM", "Grey-Cloud"
        ],
        "country": "RU",
        "refs": [
          "https://en.wikipedia.org/wiki/Sofacy_Group"
        ]
      },
      "description": "The Sofacy Group (also known as APT28,
          Pawn Storm, Fancy Bear and Sednit) is a cyber
          espionage group believed to have ties to the
          Russian government. Likely operating since 2007,
          the group is known to target government, military,
          and security organizations. It has been
          characterized as an advanced persistent threat.",
      "value": "Sofacy"
    },
```

meta best practices

- Reusing existing values such as **properties, complexity, effectiveness, country, possible_issues, colour, motive, impact, refs, synonyms, derivated_from, status, date, encryption, extensions, ransomnotes, cfr-suspected-victims, cfr-suspected-state-sponsor, cfr-type-of-incident, cfr-target-category**.
- Or adding your own meta fields.

meta best practices - a sample

```
{
    "description": "Putter Panda were the subject of an
        extensive report by CrowdStrike, which stated: 'The
        CrowdStrike Intelligence team has been tracking this
        particular unit since 2012, under the codename
        PUTTER PANDA, and has documented activity dating
        back to 2007. The report identifies Chen Ping, aka
        cpyy, and the primary location of Unit 61486.'",
    "meta": {
      "cfr-suspected-state-sponsor": "China",
      "cfr-suspected-victims": [
        "U.S. satellite and aerospace sector"
      ],
      "cfr-target-category": [
        "Private sector",
        "Government"
      ],
      "cfr-type-of-incident": "Espionage",
      "country": "CN",
      "refs": [
        "http://cdn0.vox-cdn.com/assets/4589853/crowdstrike-
            intelligence-report-putter-panda.original.pdf",
```

Expressing relation between clusters

- Cluster can be related to one or more clusters using default relationships from MISP objects and a list of tags to classify the relation.

```
        "related": [
        {
          "dest-uuid": "5ce5392a-3a6c-4e07-9df3-9b6a9159ac45",
          "tags": [
            "estimative-language:likelihood-probability=\"
                likely\""
          ],
          "type": "similar"
        }
      ],
      "uuid": "0ca45163-e223-4167-b1af-f088ed14a93d",
      "value": "Putter Panda"
```

PyMISPGalaxies

```
from pymispgalaxies import Clusters
c = Clusters()
list(g.keys())
# ['threat-actor', 'ransomware', 'exploit-kit', 'tds', 'tool', 'rat', 'mitre-attack-patter
#  'mitre-tool', 'microsoft-activity-group', 'mitre-course-of-action', 'mitre-malware',
#  'mitre-intrusion-set', 'preventive-measure']
print(c.get("rat"))
# misp-galaxy:rat="Brat"
# misp-galaxy:rat="Loki RAT"
# misp-galaxy:rat="join.me"
# misp-galaxy:rat="Setro"
# misp-galaxy:rat="drat"
# misp-galaxy:rat="Plasma RAT"
# misp-galaxy:rat="NanoCore"
# misp-galaxy:rat="DarkTrack"
# misp-galaxy:rat="Theef"
# misp-galaxy:rat="Greame"
# misp-galaxy:rat="Nuclear RAT"
# misp-galaxy:rat="DameWare Mini Remote Control"
# misp-galaxy:rat="ProRat"
# misp-galaxy:rat="death"
# misp-galaxy:rat="Dark DDoSeR"
# ....
print(c.get("rat").description)
# remote administration tool or remote access tool (RAT), also called sometimes remote
# access trojan, is a piece of software or programming that allows a remote "operator"
# to control a system as if they have physical access to that system.
```

Q&A

- info@circl.lu (if you want to join the CIRCL MISP sharing community)
- OpenPGP fingerprint: 3B12 DCC2 82FA 2931 2F5B 709A 09E2 CD49 44E6 CBCD
- `https://github.com/MISP/` - `http://www.misp-project.org/`
- We welcome any contributions to the project, be it pull requests, ideas, github issues,...

Information Sharing and Taxonomies

Practical Classification of Threat Indicators using MISP

CIRCL
Computer Incident
Response Center
Luxembourg

Team CIRCL

http://www.misp-project.org/
Twitter: *@MISPProject*

MISP Training @ Prague
20180917

Objects - or How We Learned to Stop Worrying and Love the Templates

- Attributes are a simple but powerful tool to describe data
- Lacking the capability to create containers around attributes describing a common concept
- The goal was to develop something semi-standardised, with the option to **dynamically build templates**
- We have considered a list of different solutions such as simple boolean operators, but found that the current implementation was superior.
- The result is a simple template that uses the basic attriubte types as building blocks along with some meta data
- The template does **not have to be known** in order to use the constructed objects
- What we maintain now is a set of common objects, but similarly to our other JSON formats, users can extend it with their own ideas

MISP Object Templates

- Using a similar JSON format as the taxonomies, galaxies, warninglists.
- You can find the default set of object templates in the git repository[1].
- Some of the object templates capture objects from other standards or mimic the output of tools
- We tried to capture the most common use-cases coming from our own use-case as well as those of various partners that got involved
- Improvements or pull requests for new object templates are of course always welcome

[1] `https://www.github.com/MISP/misp-objects/`

Existing Object examples

- AIL-leak - **AIL object, an example for an object catering to the output of another tool**
- Android permission - **An object used to further contextualise another object**
- Bank account
- File **Generic object to describe a file**
- Passive DNS
- Regex
- Sandbox report
- Vulnerability **Enabling new use-cases such as pre-sharing of vulnerability information**
- x509
- Yara **Verbatim sharing of rule sets along with meta-data**

Object Template skeleton

```
{
  "requiredOneOf": [],
  "required": [],
  "attributes": {},
  "version": 1,
  "description": "My description",
  "meta-category": "Chosen meta category",
  "uuid": "Object template uuid",
  "name": "Object template name"
}
```

Adding elements to an object template

```
"regexp-type": {
  "description": "Type of the regular expression syntax.",
  "disable_correlation": true,
  "ui-priority": 0,
  "misp-attribute": "text",
  "values_list": [
    "PCRE",
    "PCRE2",
    "POSIX BRE",
    "POSIX ERE"
  ]
},
```

Attribute keys

- Primary key: Object relation
- description: A description of the attribute in relation to the object
- disable_correlation: You can disable correlations for attributes in the resulting object
- ui-priority: Not implemented yet, but the idea is to have a "quick view" of objects only showing certain prio levels
- misp-attribute: The misp attribute type used as as the building block
- values_list: an optional list of values from which the user **must** choose instead of entering a value manually
- sane_defaults: an optional list of values from which the user **may** choose instead of entering a value
- multiple: Allow the user to add **more** than one of this attribute

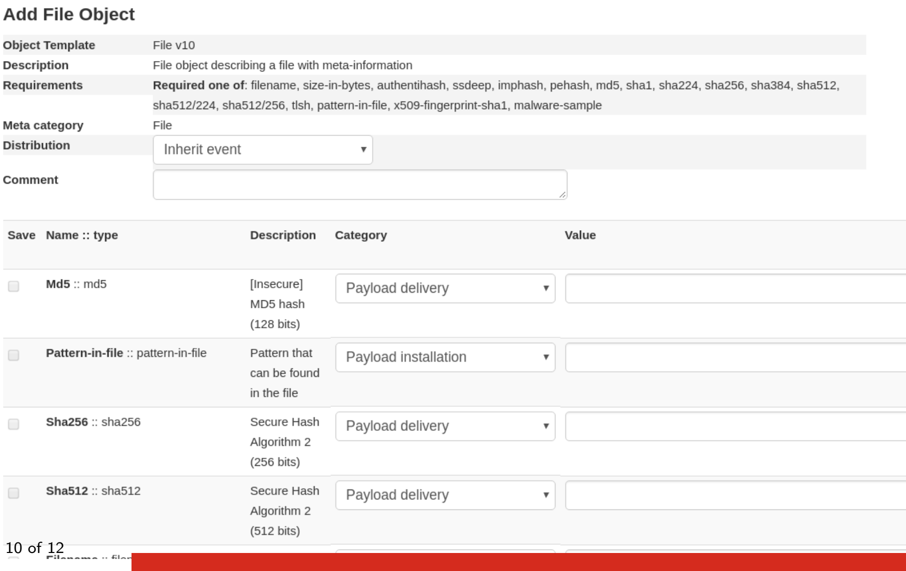
Enforcement of certain keys

- The template also defines which of the added attributes are mandatory
- Requirements are pointed to via their **object relations names**
- We differentiate between two types of rule sets:
  - Required: Everything in this list has to be set in order for the object to validate
  - Required One Of: Any of the attributes in this list will satisfy the requirements

What will the the template actually do?

- Templates create a form that can be used to populate an event
- When using templates, MISP will enforce everything according to the template rules
- However, these are only optional, users can avoid using the templates when creating events via the API
- The reason for this is that you do not need to have the template in order to create an object
- The limitation of this system: You **cannot modify** objects that were created with unknown templates

Templates as rendered in the UI

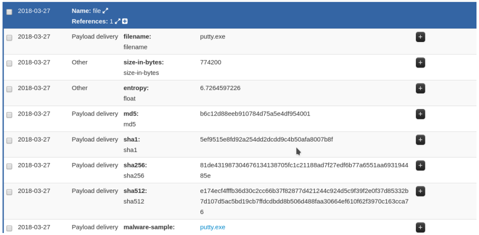


Add File Object

Object Template	File v10
Description	File object describing a file with meta-information
Requirements	**Required one of**: filename, size-in-bytes, authentihash, ssdeep, imphash, pehash, md5, sha1, sha224, sha256, sha384, sha512, sha512/224, sha512/256, tlsh, pattern-in-file, x509-fingerprint-sha1, malware-sample
Meta category	File
Distribution	Inherit event
Comment	

Save	Name :: type	Description	Category	Value
☐	**Md5** :: md5	[Insecure] MD5 hash (128 bits)	Payload delivery	
☐	**Pattern-in-file** :: pattern-in-file	Pattern that can be found in the file	Payload installation	
☐	**Sha256** :: sha256	Secure Hash Algorithm 2 (256 bits)	Payload delivery	
☐	**Sha512** :: sha512	Secure Hash Algorithm 2 (512 bits)	Payload delivery	

Templates as rendered in the UI



2018-03-27	**Name:** file **References:** 1			
2018-03-27	Payload delivery	**filename:** filename	putty.exe	+
2018-03-27	Other	**size-in-bytes:** size-in-bytes	774200	+
2018-03-27	Other	**entropy:** float	6.7264597226	+
2018-03-27	Payload delivery	**md5:** md5	b6c12d88eeb910784d75a5e4df954001	+
2018-03-27	Payload delivery	**sha1:** sha1	5ef9515e8fd92a254dd2dcdd9c4b50afa8007b8f	+
2018-03-27	Payload delivery	**sha256:** sha256	81de431987304676134138705fc1c21188ad7f27edf6b77a6551aa693194485e	+
2018-03-27	Payload delivery	**sha512:** sha512	e174ecf4fffb36d30c2cc66b37f82877d421244c924d5c9f39f2e0f37d85332b7d107d5ac5bd19cb7ffdcdbdd8b506d488faa30664ef610f62f3970c163cca76	+
2018-03-27	Payload delivery	**malware-sample:**	putty.exe	+

Q&A

- https://github.com/MISP/MISP
- https://github.com/MISP/misp-objects
- info@circl.lu (if you want to join one of the MISP community operated by CIRCL)
- PGP key fingerprint: CA57 2205 C002 4E06 BA70 BE89 EAAD CFFC 22BD 4CD5