

MISP OBJECT TEMPLATE

BUILDING CUSTOM AND OPEN DATA MODELS

CIRCL / TEAM MISP PROJECT

[HTTP://WWW.MISP-PROJECT.ORG/](http://www.misp-project.org/)

TWITTER: [@MISPPROJECT](https://twitter.com/MISPPROJECT)

FIRST.ORG/AFRICA CERT



MISP
Threat Sharing

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 attribute 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.

- Using a similar JSON format as the taxonomies, galaxies, warninglists.
- You can find the default set of object templates in the git repository¹.
- 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

- All-leak - **All 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

```
1 {
2   "requiredOneOf": [],
3   "required": [],
4   "attributes": {},
5   "version": 1,
6   "description": "My description",
7   "meta-category": "Chosen meta category",
8   "uuid": "Object template uuid",
9   "name": "Object template name"
10 }
```

ADDING ELEMENTS TO AN OBJECT TEMPLATE

```
1 "regexp-type": {
2   "description": "Type of the regular expression syntax.",
3   "disable_correlation": true,
4   "ui-priority": 0,
5   "misp-attribute": "text",
6   "values_list": [
7     "PCRE",
8     "PCRE2",
9     "POSIX BRE",
10    "POSIX ERE"
11  ]
12 },
```

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

- 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 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	<input type="text"/>

Save	Name :: type	Description	Category	Value
<input type="checkbox"/>	Md5 :: md5	[Insecure] MD5 hash (128 bits)	Payload delivery ▾	<input type="text"/>
<input type="checkbox"/>	Pattern-in-file :: pattern-in-file	Pattern that can be found in the file	Payload installation ▾	<input type="text"/>
<input type="checkbox"/>	Sha256 :: sha256	Secure Hash Algorithm 2 (256 bits)	Payload delivery ▾	<input type="text"/>
<input type="checkbox"/>	Sha512 :: sha512	Secure Hash Algorithm 2 (512 bits)	Payload delivery ▾	<input type="text"/>
<input type="checkbox"/>	Filename :: filename	Filename	Payload delivery ▾	<input type="text"/>

TEMPLATES AS RENDERED IN THE UI

2018-03-27		Name: 5c	References: 1
2018-03-27	Payload delivery	filename: putty.exe	-
2018-03-27	Other	size-in-bytes: 774200	-
2018-03-27	Other	entropy: 6.7264597226	-
2018-03-27	Payload delivery	md5: d6c12d98eeb910784f75d5e4d954001	-
2018-03-27	Payload delivery	sha1: 5ef9515e8f92a254d52dcd9c4b50afab007b8f	-
2018-03-27	Payload delivery	sha256: 81de431987304676134138709c1c21188a7727edf6b77af6551aa6931944	-
2018-03-27	Payload delivery	sha512: e174ec4fffb36d93c2cc66b3782877d421244c324d5c9f992e0f37d85332b	-
2018-03-27	Payload delivery	malware-sample: putty.exe	-



- <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