MISP format

MISP & Threat Sharing

CIRCL
Computer Incident
Response Center
Luxembourg

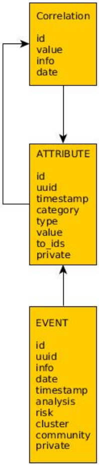
Andras Iklody - *TLP:WHITE*

MISP Summit III - 10/16/2017

The MISP core format

- Historically MISP has always used its own format internally
- We generally had two main design goals when it comes to the format:
  - Design the format to be the least complex we can come up with to map whatever information we wanted to convey
  - Enhance the format when it's needed instead of planning ahead - code is law
- Started out extremely minimalistic to fit our needs back then

The MISP format as of MISP 1



Correlation

id
value
info
date

ATTRIBUTE

id
uuid
timestamp
category
type
value
to_ids
private

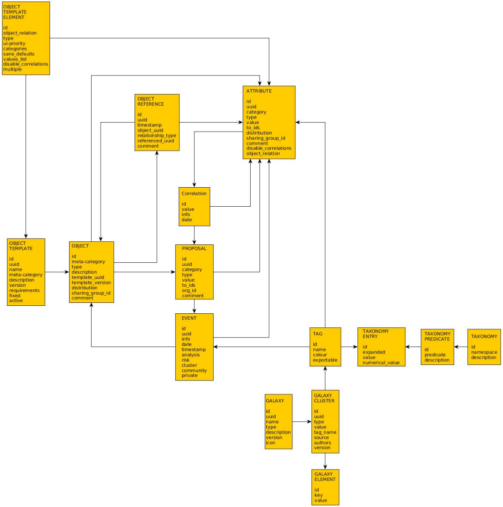
EVENT

id
uuid
info
date
timestamp
analysis
risk
cluster
community
private

The MISP core format evolved

- Over time as we have interacted with new communities and added new features, the format grew
- It has been enhanced gradually over the past 5 years
- Relied on small gradual changes instead of sweeping redesigns
- Current MISP internal format's complexity is much higher than the original

The MISP format as of MISP 2.4.80



OBJECT TEMPLATE ELEMENT
id
object_relation
type
ui-priority
categories
sane_defaults
values_list
disable_correlations
multiple

OBJECT REFERENCE
id
uuid
timestamp
object_uuid
relationship_type
referenced_uuid
comment

ATTRIBUTE
id
uuid
category
type
value
to_ids
distribution
sharing_group_id
comment
disable_correlations
object_relation

Correlation
id
value
info
date

OBJECT TEMPLATE
id
uuid
name
meta-category
description
version
requirements
fixed
active

OBJECT
id
meta-category
type
description
template_uuid
template_version
distribution
sharing_group_id
comment

PROPOSAL
id
uuid
category
type
value
to_ids
org_id
comment

EVENT
id
uuid
info
date
timestamp
analysis
risk
cluster
community
private

TAG
id
name
colour
exportable

TAXONOMY ENTRY
id
expanded
value
numerical_value

TAXONOMY PREDICATE
id
predicate
description

TAXONOMY
id
namespace
description

GALAXY
id
uuid
name
type
description
version
icon

GALAXY CLUSTER
id
uuid
type
value
tag_name
source
authors
version

GALAXY ELEMENT
id
key
value

Why standardise at all?

- More and more requests from other tools / vendors to integrate with us
  - Complaints about having to go through a jungle of PHP code to figure out how to do it
- Validation from 3rd parties
- We believe that specialised formats are best for specialised tasks (suricata/bro/snort, Yara, Sigma...)

Why standardise at all?

- However, for information exchange we found the offerings out there to be lacking
  - OpenIOC is limited to indicators only
  - STIX is a massive beast with a long list of flaws (that we are hoping to somewhat steer in the right direction)
  - None of them allow us to exchange the scope of data that we exchange with MISP in general
- For best interoperability we recommend our own format, therefore we need to standardise it
- Over time as we have interacted with new communities and added new features,

Ongoing effort to standardise MISP

- IETF draft document for the MISP core format
- IETF draft documents for the MISP supporting formats
- Available at `https://github.com/MISP/misp-rfc`

A list of the currently described MISP formats

- MISP core format: basically the exchange format of MISP (Events, Attributes, Objects, Tags, Sharing Groups, Proposals...)
- MISP JSON formats:
  - MISP taxonomies
  - MISP galaxies
  - MISP object-templates

The MISP core format

- Describes the format used to exchange information between MISP instances
- Includes descriptions of all structures that get exchanged between MISPs
  - Events
  - Objects, Object References
  - Attributes, Proposals
  - Tags, Galaxies
  - Organisations

The MISP taxonomy and galaxy formats

- Describes the formats used to create the JSON structures for the respective objects
- Due to the wealth of categorisation/contextual information, used by more and more organisations even outside of MISP (such as Alienvault OTX)
- The standards aim to make life for content creators easier

The MISP object template format

- Since the release of MISP objects, users have started building their own object templates
- These templates are then used to create individual objects based on the pre-defined patterns
- Also includes a vocabulary containing the default relationships to be used for object references and soon galaxy referenced

Governance of the MISP standards

- Get input from wherever we can (github, trainings, twitter,...)
- Use the input to add/remove/shift priorities wherever it makes sense
- The goal was to mimic how STIX 1.x was handled by Mitre, though with (hopefully) greater resistance to vendor pressure
- But ultimately it's a dictatorship
-

Why a dictatorship?

- Our efforts as part of OASIS have made us jaded...
  - Due to the cumbersome bureaucratic process, a painful lack of flexibility / agility
  - Mistakes seen as failure with any suggestion to rectify it being taboo (partially because of the high investment of agreeing on any changes in the first place)
  - Vendors pushing non-sensical requirements fit for their products only
  - The same vendors being over-represented compared to your average users/implementers as they're paid to work on the standardisation effort full time
- Turns into a dishonest dictatorship

Quick summary

- Though we are a dictatorship, our success is based on your feedback!
- If you are implementing MISP data or want to enhance the repositories out there: `https://github.com/MISP/misp-rfc`
- Feel free to open up issues or pull requests if you find errors in the documents