## MISP and Decaying of Indicators

AN INDICATOR SCORING METHOD AND ONGOING IMPLE-

TFAM CIRCL

INFO@CIRCL.LU

**SEPTEMBER 11, 2024** 



## **EXPIRING IOCS: WHY AND HOW?**

#### INDICATORS - PROBLEM STATEMENT

- Sharing information about threats is crucial
- Organisations are sharing more and more

## Contribution by unique organisation (Orgc.name) on MISPPriv:

Date	Unique Org			
2013	17			
2014	43			
2015	82			
2016	105			
2017	118			
2018	125			
2019-10	135			

```
{
    "distribution": [1, 2, 3]
}
```

#### INDICATORS - PROBLEM STATEMENT

- Various users and organisations can share data via MISP, multiple parties can be involved
  - ► Trust, data quality and time-to-live issues
  - ► Each user/organisation has **different use-cases** and interests
    - Conflicting interests such as operational security, attribution,... (depends on the user)
  - → Can be partially solved with *Taxonomies*

#### INDICATORS - PROBLEM STATEMENT

- Various users and organisations can share data via MISP, multiple parties can be involved
  - ► Trust, data quality and time-to-live issues
  - ► Each user/organisation has **different use-cases** and interests
    - Conflicting interests such as operational security, attribution,... (depends on the user)
  - → Can be partially solved with *Taxonomies*
- Attributes can be shared in large quantities (more than 7.3 million on MISPPRIV)
  - Partial info about their freshness (Sightings)
  - Partial info about their validity (last update)
  - ightarrow Can be partially solved with our *Decaying model*

## REQUIREMENTS TO ENJOY THE DECAYING FEATURE IN MISP

- Starting from MISP 2.4.116, the decaying feature is available
- Don't forget to update the decay models and enable the ones you want
- The decaying feature has no impact on the information in MISP, it's just an overlay to be used in the user-interface and API
- Decay strongly relies on Taxonomies and Sightings, don't forget to review their configuration

#### SIGHTINGS - REFRESHER

Sightings add temporal context to indicators. A user, script or an IDS can extend the information related to indicators by reporting back to MISP that an indicator has been seen, or that an indicator can be considered as a false-positive

- Sightings give more credibility/visibility to indicators
- This information can be used to **prioritise and decay** indicators

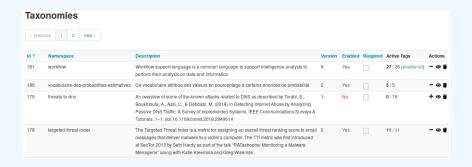


# ORGANISATIONS OPT-IN - SETTING A LEVEL OF CONFIDENCE

MISP is a peer-to-peer system, information passes through multiple instances.

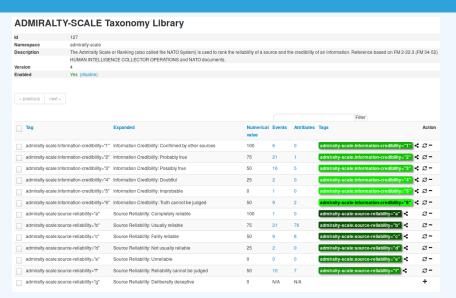
- **Producers can add context** (such as tags from *Taxonomies*, *Galaxies*) about their asserted confidence or the reliability of the data
- Consumers can have different levels of trust in the producers and/or analysts themselves
- Users might have other contextual needs
  - → Achieved thanks to *Taxonomies*

## TAXONOMIES - REFRESHER (1)



- Tagging is a simple way to attach a classification to an *Event* or an *Attribute*
- Classification must be globally used to be efficient

## TAXONOMIES - REFRESHER (2)



## → Cherry-pick allowed Tags

## TAXONOMIES - REFRESHER (3)

- Some taxonomies have numerical\_value
  - $\rightarrow$  Can be used to prioritise Attributes

Description	Value
Completely reliable	100
Usually reliable	75
Fairly reliable	50
Not usually reliable	25
Unreliable	0
Reliability cannot be judged	50 ?
Deliberatly deceptive	0?

Description	Value
Confirmed by other sources	100
Probably true	75
Possibly true	50
Doubtful	25
Improbable	0
Truth cannot be judged	50 ?

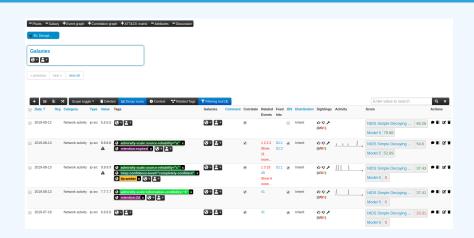
### Scoring Indicators: Our solution

## Where,

- $score \in [0, +\infty]$
- base\_score  $\in$  [0,100]
- decay is a function defined by model's parameters controlling decay speed
- Attribute Contains Attribute's values and metadata (Taxonomies, Galaxies, ...)
- Model Contains the Model's configuration

# CURRENT IMPLEMENTATION IN MISP

## IMPLEMENTATION IN MISP: Event/view



- Decay score toggle button
  - ► Shows Score for each Models associated to the Attribute type

#### IMPLEMENTATION IN MISP: API RESULT

#### /attributes/restSearch

```
"Attribute": [
2
       "category": "Network activity",
       "type": "ip-src",
       "to ids": true,
       "timestamp": "1565703507",
       [...]
7
       "value": "8.8.8.8",
8
      "decay score": [
10
           "score": 54.475223849544456,
11
           "decayed": false,
12
           "DecayingModel": {
13
             "id": "85",
14
             "name": "NIDS Simple Decaying Model"
15
16
17
18
```

## IMPLEMENTATION IN MISP: PLAYING WITH MODELS

- Automatic scoring based on default values
- **User-friendly UI** to manually set *Model* configuration (lifetime, decay, etc.)
- **Simulation** tool
- Interaction through the API
- Opportunity to create your own formula or algorithm

## **DECAYING MODELS IN DEPTH**

## Scoring Indicators: base\_score (1)

When scoring indicators<sup>1</sup>, multiple parameters<sup>2</sup> can be taken into account. The **base score** is calculated with the following in mind:

- Data reliability, credibility, analyst skills, custom prioritisation tags (economical-impact), etc.
- Trust in the source

$$base\_score = \omega_{tg} \cdot tags + \omega_{sc} \cdot source\_confidence$$

### Where,

$$\omega_{sc} + \omega_{tg} = \mathbf{1}$$

<sup>&</sup>lt;sup>1</sup>Paper available: https://arxiv.org/pdf/1803.11052

<sup>&</sup>lt;sup>2</sup>at a variable extent as required

## Scoring Indicators: base\_score (2)

Current implentation ignores source\_confidence:

$$\rightarrow$$
 base\_score = tags

	Computation				
Tag	Eff. Ratio		numerical_value	Result	
admiralty-scale:source-reliability="Completely reliable"	0.50	*	100.00	50.00	
phishing:psychological-acceptability="high"	0.50	*	75.00	37.50	
				87.50	

ightarrow The base\_score can be use to prioritize attribute based on their attached context and source

## SCORING INDICATORS: DECAY SPEED (1)

$$\texttt{score}(\texttt{Attribute}) = \texttt{base\_score}(\texttt{Attribute}, \, \texttt{Model}) \quad \bullet \quad \texttt{decay}(\texttt{Model}, \, \texttt{time})$$

## The decay is calculated using:

- The lifetime of the indicator
  - May vary depending on the indicator type
  - short for an IP, long for an hash
- The decay rate, or speed at which an attribute loses score over time
- The time elapsed since the latest update or sighting

#### SCORING INDICATORS: PUTTING IT ALL TOGHETHER

 $\rightarrow$  decay rate is **re-initialized upon sighting** addition, or said differently, the score is reset to its base score as new *sightings* are applied.

$$score = base\_score \cdot \left(1 - \left(\frac{t}{\tau}\right)^{\frac{1}{\delta}}\right)$$

- $\blacksquare \tau =$ lifetime
- lacksquare  $\delta = \operatorname{decay}$  speed

## IMPLEMENTATION IN MISP: MODELS DEFINITION

$$\Rightarrow score = base\_score \cdot \left(1 - \left(\frac{t}{\tau}\right)^{\frac{1}{\delta}}\right)$$

Models are an instanciation of the formula where elements can be defined:

- Parameters: lifetime, decay\_rate, threshold
- base\_score
- default base score
- formula
- associate Attribute types
- creator organisation

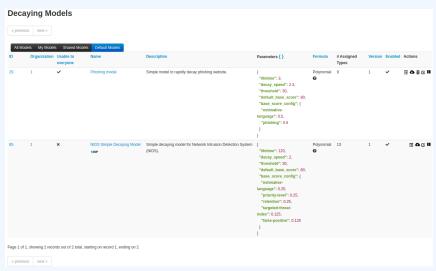
#### IMPLEMENTATION IN MISP: MODELS TYPES

## Multiple model types are available

- **Default Models**: Models created and shared by the community. Available from misp-decaying-models repository<sup>3</sup>.
  - ► → Not editable
- **Organisation Models**: Models created by a user belonging to an organisation
  - ► These models can be hidden or shared to other organisation
  - ► → Editable

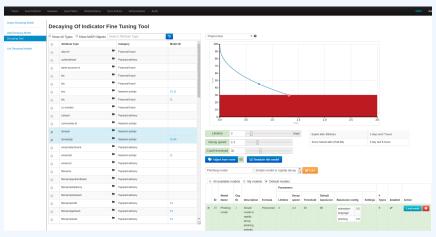
<sup>3</sup>https://github.com/MISP/misp-decaying-models.git

### IMPLEMENTATION IN MISP: INDEX



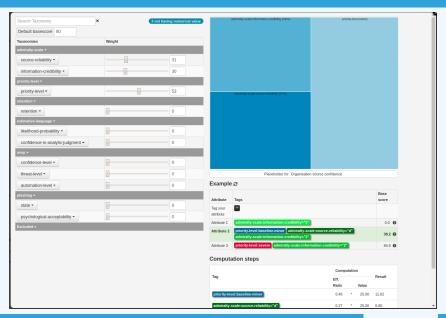
View, update, add, create, delete, enable, export, import

## IMPLEMENTATION IN MISP: FINE TUNING TOOL



Create, modify, visualise, perform mapping

## Implementation in MISP: base\_score tool



## IMPLEMENTATION IN MISP: SIMULATION TOOL



Simulate Attributes with different Models

## IMPLEMENTATION IN MISP: API QUERY BODY

#### /attributes/restSearch

```
"includeDecayScore": 1,
"includeFullModel": 0,
"excludeDecayed": 0,
"decayingModel": [85],
"modelOverrides": {
    "threshold": 30
}
"score": 30,
}
```

## CREATING A NEW DECAY ALGORITHM (1)

The current architecture allows users to create their **own** formulae.

- Create a new file \$filename in app/Model/DecayingModelsFormulas/
- 2. Extend the Base class as defined in DecayingModelBase
- Implement the two mandatory functions computeScore and isDecayed using your own formula/algorithm
- 4. Create a Model and set the formula field to \$filename Use cases:
  - Add support for **more feature** (expiration taxonomy)
  - Query external services then influence the score
  - Completely **different approach** (i.e streaming algorithm)
  - ...

## CREATING A NEW DECAY ALGORITHM (2)

```
<?php
include_once 'Base.php';
3
  class Polynomial extends DecayingModelBase
5
       public const DESCRIPTION = 'The description of your new
       decaying algorithm':
7
       public function computeScore($model, $attribute, $base_score,
8
       $elapsed_time)
          // algorithm returning a numerical score
10
11
12
       public function isDecayed($model, $attribute, $score)
13
14
           // algorithm returning a boolean stating
15
           // if the attribute is expired or not
16
17
18
19
20
```

#### **DECAYING MODELS 2.0**

- Improved support of Sightings
  - ► False positive Sightings should somehow reduce the score
  - Expiration Sightings should mark the attribute as decayed
- Potential *Model* improvements
  - ► Instead of resetting the score to base\_score once a Sighting is set, the score should be increased additively (based on a defined coefficient); thus **prioritizing surges** rather than infrequent Sightings
  - Take into account related Tags or Correlations when computing score
- Increase Taxonomy coverage
  - ► Users should be able to manually override the numerical\_value of *Tags*
- For specific type, take into account data from other services
  - Could fetch data from BGP ranking, Virus Total, Passive X for IP/domain/... and adapt the score