MISP 3 - Teaching an Old Dog New Tricks

Paving the way forward

Andras Iklody & Sami Mokaddem

MISP Project
https://www.misp-project.org/
$ whoarewe

Andras Iklody
@iglocska

Sami Mokaddem
@mokaddem_sami

THE MISP COMMUNITY

@mokaddem & @iglocska
Get ready everyone, he’s about to do something stupid.
Why MISP 3?
The plan
Considerations
Why MISP 3?
MISP is based on CakePHP 2.x
- End of Security Support in **June 2021**
- Maintained fork github.com:MISP/cakephp.git

CakePHP supports PHP version **<=7.4**
- End of Security Support in **November 2022**
MISP supports a wide range of use cases...

... meaning loads of feature-clutter the interface

All options visible regardless of the user profile

Lack of coherent page navigation
To list a few..

- Sub-optimal database structure
- Start with something small, build it out has its disadvantages
  - Attribute type, value not a first-class citizen
  - Logs all in one place
  - Indexing rework (performance and moving validation to the DB)
- Confusing mess of multiple graphing interfaces
- Files - Especially tricky with dockerised and load balanced setups
- Tagging
Port of the codebase to a new stack

- CakePHP 2.x → CakePHP 5

Rework of old baggage

- Database updates
- Front-end libraries (Bootstrap, Graphing, ...)
- Background jobs & Scheduled tasks
- Purging old libraries
> Pruning unused / dead end functionalities

- Populate using the templating system
- Deprecated export functionalities
- Discussion / Posts
- ...

WE WANT YOU TO GIVE US FEEDBACK
Step I - Preparing the grounds
Refactoring the codebase for improved portability using factories

- Framework-agnostic
- Reusable code for front and back-end
- Extracting and encapsulating specialised functionalities into libraries
Setting the stage with Cerebrate

- Dev started in May 2020, built on MISP3’s stack
- Application built on top of ported MISP libraries
- New UI laying the foundation for MISP 3
- Streamlined integration of new features into MISP3
  - Tagging, Inbox system, Settings, …
Migrate least connected part first
Step II - Porting the Codebase
Step II - Roadmap for a 3-Wave Porting
Wave 1  Least complex/inter-connected models
  ➤  E.g. Blocklist, Warninglist, Object-template, User

Wave 2  More glue relying on component already migrated
  ➤  E.g. Authkey, *-Tag, Taxonomy

Wave 3  The actual meat of the application
  ➤  E.g. Attribute, Event, Workflow
Step II - Test driven development

$ composer test
$ sh ./tests/Helper/wiremock/start.sh
WireMock 1 started on port 8080
$ phpunit
[*] Running DB migrations, it may take some time ...

The WireMock server is started ...... 
port: 8080 
enable-browser-proxying: false 
disable-banner: true 
no-request-journal: false 
verbose: false

PHPUnit 8.5.22 by Sebastian Bergmann and contributors.

- Complementary to PyMISP test
- In-framework **Unit Tests** and **Endpoint Tests**
- Improved CI pipeline and enforced code standard
Migration officially started in January 2023

- Around **27 tables** have been moved
- Some partially, others completely
Migration speed ramping up. The more we port, the faster we go.

This branch is 333 commits ahead, 914 commits behind 2.4.

Even while supporting and improving 2.4.
Look and Feel
Most of the changes are **invisible**

Some user interfaces can still be displayed
## Codebase Migration: Look and Feel II

![Image of Codebase Migration: Look and Feel II interface](image)

### Organisation View

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>UUID</th>
<th>URL</th>
<th>Nationality</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ORGNAME</td>
<td>7e9251cb-3b15-417a-9e92-a508474c354d</td>
<td>Luxembourg</td>
<td>ADMIN</td>
<td></td>
</tr>
</tbody>
</table>

### User index

<table>
<thead>
<tr>
<th>ID</th>
<th>Org</th>
<th>Role</th>
<th>Email</th>
<th>SID</th>
<th>Last Login</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ORGNAME</td>
<td>admin</td>
<td><a href="mailto:admin@admin.test">admin@admin.test</a></td>
<td>400000</td>
<td>2023-03-08 07:15:14</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ORGNAME</td>
<td>Sync user</td>
<td><a href="mailto:sync@admin.test">sync@admin.test</a></td>
<td>3567891</td>
<td>2022-03-17 10:19:40</td>
<td></td>
</tr>
</tbody>
</table>
**Codebase Migration: Look and Feel II**

![User Interface Screenshot]

A screenshot of a user interface showing a user profile with various details such as user ID, email, organization, role, and notification settings. The interface contains elements for managing user accounts and privacy settings, indicating a focus on security and access control.

<table>
<thead>
<tr>
<th>ID</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td><a href="mailto:admin@admin.test">admin@admin.test</a></td>
</tr>
<tr>
<td>Organisation</td>
<td>OFFICIA</td>
</tr>
<tr>
<td>Role</td>
<td>admin</td>
</tr>
</tbody>
</table>

**Email notifications**
- Event published notification
- Daily notifications
- Weekly notifications
- Monthly notifications

**Contact alert enabled**

**NIDS Start SID**
- 4000000

**Terms accepted**

**Must change password**

**PGP key**
- N/A

**S/MIME Public certificate**

**Disabled**

**Authentication keys**

**Events**


two/OSF/three/OSF/five/OSF

CODEBASE MIGRATION: LOOK AND FEEL II
Updating Bootstrap greatly improves aesthetics
And allow us to integrate themes seamlessly
CODEBASE MIGRATION: LOOK AND FEEL II
Step III - The TODOs
Indicator centric perspective

- Alternative to the Event centric view
- Unified view of everything we know about a given Indicator
- Allows us to take better decisions
- Enable users to manage their IoC working set
- Start an investigation more easily from a single indicator
Redefine How We Interact with Data II

- Unified search mechanics
  - Code deduplication
  - Streamlined way to search for data
  - Opening up the full power of the API searches to UI users
  - Translation layer for the deprecated endpoints
Refactor the Event view

- Key Elements at first glance
- Emphasis on the context (Insights, Taxonomies, Galaxies, Correlation, ·)
- Massive performance gains by moving to the composition of separate atomic endpoints
- Unified graph interface
- Sneak peak 😊
SNEAK PEAK OF THE NEW EVENT VIEW - WiP
CONSIDERATIONS
Created in **2012**, Officially became a standard in 2016

**No breaking changes** since its birth, And we’ll maintain this streak

Format will keep evolving to support new functionalities
The aim is to achieve a near **100% compatibility** with the old API.

"Near" only due to the functionalities removed as a result of deprecation.

**Strategy:** Mapping with a translation layer
API Compatibility means Synchronisation compatibility

MISP 3 servers will be able to sync with MISP 2.4 and vice versa

**BUT**

- **MISP 2.4 → 3**
  - Full support
- **MISP 3 → 2.4**
  - Lossy when sharing new types of datapoints
  - E.g: Tags on Objects
- **MISP 2.4** will be supported for a limited time
- **6 months** support post MISP 3 release
  - Potential changes/improvements on 2.4 to better support MISP 3 interactions
No one-click update; manual script execution required

- Migration tools will be included in MISP 3 to help you
- This allows us to make underlaying changes such as
  - Database changes
  - Libraries changes (e.g. supervisor in favour of cake-resque)
- **Simplified** installation based on package managers
- Upstream Docker installer
- OS targets: **Ubuntu** and RHEL
Reworked UX/UI

Alternative, **Analyst centric** in addition to the data centric approach

Improved **search and trend** monitoring tools

**Improved performance** and resilience

Want to get involved?

Removal of the main painpoints of MISP 2.x’s limitations across the board
We will list features marked for culling
  ▶ If you’re using any of them, please let us know!
We will be launching a beta phase in the future
  ▶ Feedback & improvements are more than welcome!
Want to get involved?
  ▶ 3-x branch - MISP/MISP/tree/3.x
  ▶ Project for migration - github.com/orgs/MISP/projects/2