HORIZON EUROPE PROGRAMME
HORIZON-CL3-2021-CS-01-01

A EUROPEAN CYBER RESILIENCE FRAMEWORK WITH ARTIFICIAL INTELLIGENCE -ASSISTED ORCHESTRATION & AUTOMATION FOR BUSINESS CONTINUITY, INCIDENT RESPONSE & INFORMATION EXCHANGE

D3.2: AI-assisted Situational Awareness, Prediction & Response Enablers v2

Abstract: This document reports the work conducted in the PHOENI2X Work Package 3 (WP3) and more specifically in Task 3.1 (User & Entity Behaviour Analytics), Task 3.2 (CTI Discovery, Analytics & Threat Hunting), Task 3.3 (Attack Prediction, Response Recommendation & Adaptation) and Task 3.4 (Risk Impact Assessment, Alert Triage & Response Prioritisation). The main goal of this document is to describe the design and development done for each one of the AI-assisted Situational Awareness, Prediction & Response enablers. The developed enablers have been integrated in the PHOENI2X framework in the context of Task 5.2 (Framework Integration & Testing) and their functionalities demonstrated in the energy, transport and health sectors via the PHOENI2X Use Cases.

Contractual Date of Delivery	31/05/2025
Actual Date of Delivery	02/06/2025
Deliverable Security Class	PU
Editor	ATOS
Contributors	SEA, SANL, UiO, UPC, ATOS
Quality Assurance	UPC, UiO

This project has received funding from the Horizon Europe Research and Innovation programme under Grant Agreement No 101070586

Document Revisions & Quality Assurance

Internal Reviewers

Reviewer #1: #3 UNIVERSITAT POLITECNICA DE CATALUNYA (UPC)

Reviewer #2: #14 UNIVERSITETET I OSLO (UiO)

Revisions

Version	Date	By	Overview
01	07/04/2025	Editor	Initial draft, with D3.1 as basis and assignments
02	11/04/2025	Editor	Previous contents from D3.1 are highlighted
03	24/04/2025	Editor	• Addition of abbreviations contributed by SPHYNX • Updated section 4.1 by SPHYNX • Updated chapter 5 by UPC
04	30/04/2025	Editor	• Addition of Abbreviations • Updated Introduction by ATOS • Updated chapter 2 by ATOS • Updated section 3.1 by SPHYNX • Updated section 4.1 by SPHYNX • Updated section 4.2 by ATOS • Updated chapter 5 by UPC • Updated chapter 6 by ATOS • Updated conclusions by UPC
05	12/05/2025	Editor	• Removal of previous section 3.2 (User Security Behaviour Scales) • Updated 4.3 by UiO • Updated 5.3.1 by UPC • Updated 6 by ATOS • Updates of Introduction and Conclusions by ATOS
06	21/05/2025	Editor	Updates of section 3.1 (UEBA) by SPHYNX: • Updated 3.1.1 (Overview) • Updated 3.1.3.2 (Initial Results Exploration)
07	22/05/2025	Editor	New section 4.2 (SPA CTI Assessment) added by SPHYNX
08	27/05/2025	Editor	Review by UPC: several suggestions, correction of spelling errors, some improvement of wording
09	28/05/2025	Editor	• Updated chapter 6 (CERCA) by ATOS • Suggestions by internal reviewer regarding section 3.1.3.2 (UEBA / Initial Results Exploration) have been addressed by SPHYNX
10	29/05/2025	Editors	General review of format of document by ATOS
11	31/05/2025	Editors	New section 4.5 ‘Pushing CACAO to the Next Level’ by UiO

Contents

List of Tables

List of Figures

List of Abbreviations

ABIS: Abbreviated Impulsiveness Scale

AI: Artificial Intelligence

AMI: Advanced Metering Infrastructure

API: Application Programming Interface

ATC-IB: Attitudes Towards Cybersecurity and cybercrime In Business

AutoML: Automated Machine Learning

CBS: Conservative Behavior Scale

CERCA: Cyber Risk Assessment Calculator

CLI: Command Line Interface

CPE: Common Platform Enumeration

CRC: Cyber Resilience Centre

CTI: Cyber Threat Intelligence

DDoS: Distributed Denial of Service

DLMS: Device Language Message Specification

DoS: Denial of Service

EOS: Exposure to Offence Scale

EPS: Electric Power System

EWS: Early Warning Systems

EXM: Entity eXtraction Module

GDPR: General Data Protection Regulation

HAIS-Q: Human Aspects of Information Security Questionnaire

HTML: HyperText Markup Language

HTTP: HyperText Transfer Protocol

IDS: Intrusion Detection Systems

IoCs: Collecting Indicators of Compromise

IoR: Internet of Railways

IoT: Internet of Things

IPS: Intrusion Prevention Systems

JSON: JavaScript Object Notation

KAB: Knowledge, Attitude, and Behaviour

LightGBM: Light Gradient-Boosting Machine

LoRaWAN: Long Range Wide Area Network

LSTM: Long Short-Term Memory

MISP: Malware Information Sharing Platform

ML: Machine Learning

MQTT: Message Queuing Telemetry Transport

MVP: Minimum Viable Product

NLP: Natural Language Processing

NVD: National Vulnerability Database

OCS: Online Cognition Scale

OSINT: Open-Source Intelligence

PMEM: Predictive Maintenance

PoC: Proof-of-Concept

PSCC: Power System Control Centers

RBS: Risky Behaviour Scale

RNNs: Recurrent neural networks

ROAR: Resilience Automation, Orchestration, and Response

RPS: Risk Perception Scale

RScB: Risky cybersecurity Behaviours scale

RSS: Really Simple Syndication

SBOM: Software Bill of Materials

SDO: STIX Domain Object

SeBIS: Security Behaviour Intentions Scale

SQL: Structured Query Language

SRO: STIX Relationship Object

STIX: Structured Threat Information Expression

TAC: Threat Actor Context

TAXII: Trusted Automated eXchange of Intelligence Information

TII: Threat Intelligence Integrator

TIPs: Threat Intelligence Platforms

TPS: Traction Power Systems

UEBA: User & Entity Behaviour Analytics

UML: Unified Modelling Language

VM: Virtual Machine

1 Introduction

1.1 Purpose of the Document

This document (D3.2) presents the final status of the AI-assisted Situational Awareness, Prediction & Response enablers of the PHOENI2X framework. It details the technical work since the initial iteration of the deliverable (D3.1 [19]). This work includes development of functionality and integration efforts into the project use cases.

The enablers described represent the work of the WP3 (AI-assisted Situational Awareness, Prediction & Response Enablers) tasks, as described in the project Grant Agreement:

- Task 3.1 - User & Entity Behaviour Analytics: This task designs, develops, and delivers the UEBA enablers.
- Task 3.2 - CTI Discovery, Analytics & Threat Hunting; This task designs, develops and delivers the CTI Discovery, Analytics & Threat Hunting enablers.
- Task 3.3 - Attack Prediction, Response Recommendation & Adaptation: This task designs, develops and delivers the Attack Prediction, Response Recommendation & Adaptation enablers.
- Task 3.4 - Risk Impact Assessment, Alert Triage & Response Prioritisation: This task designs, develops and delivers the Risk Impact Assessment, Alert Triage & Response Prioritisation enablers

This document is structured in several chapters. Chapter 2 'AI-assisted Situational Awareness, Prediction & Response,' provides an overview of all the involved enablers, their roles in the PHOENI2X framework, as well as their interactions with the other components. Chapters 3 to 6 provide details of the final design and implementation for each group of enablers, covering tasks 3.1 to 3.4. Chapter 7 presents the Conclusions.

1.2 Overall Methodology

Initially, D2.1 "PHOENI2X Requirements & Architecture" detailed the technologies and components to be provided by WP3 partners. D2.1 identified implementation details, including hardware requirements, input and output data specifications, interfacing, interactions, and functionalities, which comprise the PHOENI2X architecture.

Next, the interactions between the enablers and the 'Baseline prevention, detection, and response' toolset were defined, as well as the UML diagrams for three use cases of project in D5.1 "PHOENI2X framework - MVP." This work helped to develop and integrate the components that comprised the initial version of the 'AI-assisted Situational Awareness, Prediction & Response' enablers reported in D3.1.

Finally, the latest implementation efforts and enhancements of the enablers were accomplished, mainly driven by their integration into the different use cases.

1.3 Relationship with other PHOENI2X deliverables

Deliverable D3.2 builds on the work made in D3.1 and provides an update on the final implementation status, integration, and enhancements of the enablers. The objective of D3.2 is to assess the operational maturity and deployment status of each enabler, as well as to explain how each enabler contributes to the overall architecture.

In addition to the relationship with deliverables 2.1 and 5.1 mentioned in the previous section, as illustrated in the overall architecture Figure 2, the components of 'AI-assisted Situational Awareness, Prediction & Response' are related to the baseline toolset, which is defined in D2.2 'Baseline Enablers', and with the 'Coordinated Response & Preparedness' enablers, whose components are defined in

D4.2 ‘Coordinated Response & Preparedness Enablers v2’. The final version of the AI-assisted Situational Awareness, Prediction & Response enablers has been integrated into the overall solution through the use cases, as described in D5.3 ‘PHOENI2X framework – Final’.

2 AI-assisted Situational Awareness, Prediction & Response

Within WP3, four enablers have been developed (as illustrated in Figure 1) to support situational awareness, risk impact assessment, prioritization, recommendations, and adaptive responses:

- **User & Entity Behavior Analysis (UEBA)**: This component helps detect and better respond to threats. It does this by analyzing data using models developed with Machine Learning (ML), statistical methods, and tools that help automate ML (AutoML). It can detect threats from user actions and make people more cybersecurity aware.
- **CTI discovery, extraction, analytics and threat hunting**: this is done through the Threat Intelligence Integrator (TII), which improves the CTI by providing more context. The TII works by: (i) collecting Indicators of Compromise (IoCs) from different Threat Intelligence Platforms (TIPs), which use Open Source Intelligence (OSINT) from various sources; (ii) cleaning the data by removing duplicate or unnecessary elements and piecing them together to create a consolidated IoC; and (iii) obtaining real-time and static data from the system and using it, along with IoCs, to determine the score of a threat. TII also helps share this information more securely by adding encryption to MISP, allowing information of IoCs to be hidden and controlling who can access it.
- **Attack Prediction, Response Recommendation & Adaptation**: this component helps to prevent cyberattacks. It uses supervised and unsupervised learning methods. It performs three main tasks: (i) classify potential attacks into groups; (ii) detect and predict what attacks might occur by training models to detect unexpected actions; and (iii) suggesting ways to react and adapt after detecting or predicting an attack.
- **Risk Impact Assessment & Prioritisation**: this component evaluates risk by considering the probability of an incident and its potential impact. The assessment includes security, economic, confidentiality, integrity and availability factors. It also takes into account the organization's business profile, the results of the scan, and the real-time system monitoring. Risk scores are calculated using qualitative and quantitative rules, which then support better decisions by prioritizing countermeasures based on CTI data.

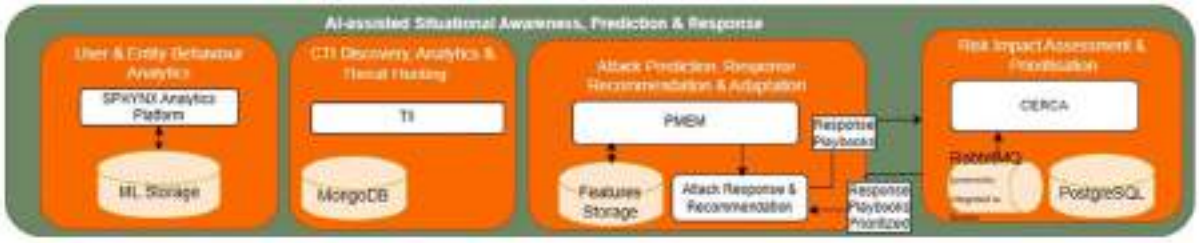

Figure 1. AI-assisted Situational Awareness, Prediction & Response Enablers (Source: D2.1 [21])

Figure 2 shows how the AI-assisted Situational Awareness, Prediction, and Response enablers take part of the wider PHOENI2X framework. These enablers support the continuous monitoring, analysis, and understanding of the threat risks. To do this, the four enablers work together with other modules of the PHOENI2X framework, following the architecture defined in D2.1 "PHOENI2X Requirements & Architecture" and the UML sequence diagrams created for each use case in D5.1 "PHOENI2X Framework - MVP".

The final version of these enablers, as described in this deliverable, will be integrated into the final PHOENI2X framework and will be reported in D5.3 "PHOENI2X framework - Final".

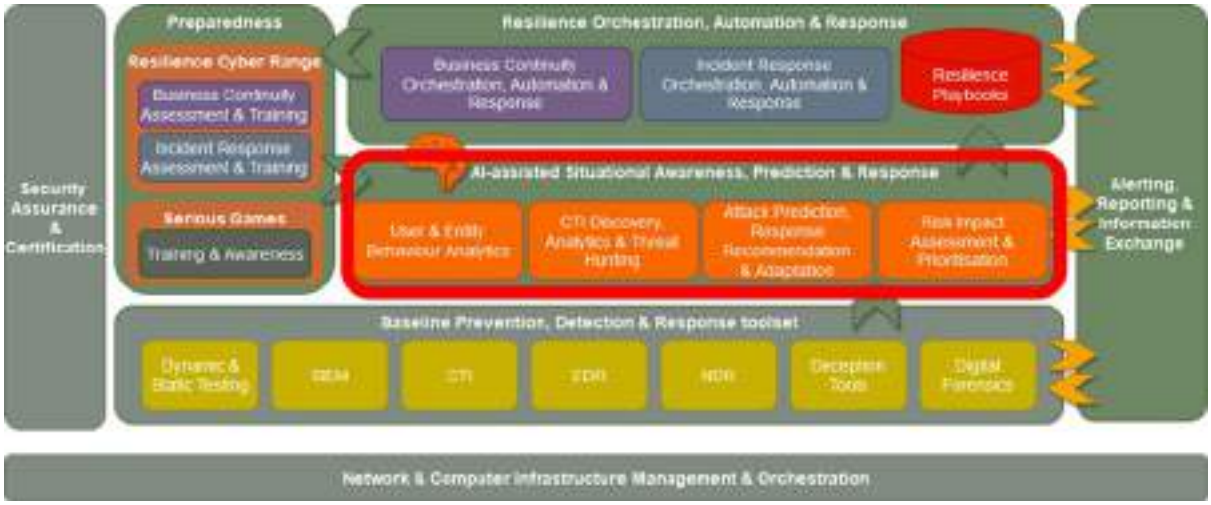

Figure 2. AI-assisted enablers in the PHOENI2X architecture (Source: DoA [20]).

3 User & Entity Behaviour Analytics Enabler

This section presents the work done in Task 3.1 “User & Entity Behaviour Analytics”. In this task, the AutoML-based UEBA enabler, which analyses the behaviour of users to detect insider threats, compromised accounts, and other malicious activities, has been developed.

3.1 AutoML-based UEBA

3.1.1 Overview

User and Entity Behavior Analytics (UEBA) is a cybersecurity strategy designed to detect insider threats, compromised accounts, and other suspicious activities by monitoring and analyzing the behavior of users and entities within a network. It leverages machine learning algorithms and statistical models to establish a baseline of typical behavior for each user or entity and then identifies anomalies or deviations that may signal potential security threats.

UEBA systems gather and analyze data from multiple sources across an organization’s IT infrastructure, such as network traffic, logs, endpoints, and applications. By correlating these diverse data points, UEBA solutions can reveal behavioral patterns that suggest malicious activity like unauthorized access, data theft, or unusual file transfers.

Figure 3. UEBA Main Panel

For more details about UEBA you can refer to section 3.1.1 of D3.1 [19].

3.1.2 Design Details

SphynxML UEBA uses advanced machine learning techniques to proactively identify potential security threats by analyzing historical user and entity data within the network. By developing and applying tailored UEBA use cases, SphynxML builds customized machine learning models that can detect unusual or potentially malicious behavior from both users and network devices. For more information about the SphynxML UEBA design details you can refer to Section 3.1.2 of D3.1 [19].

3.1.3 Implementation Details

The UEBA component is deployed as a containerized service. It accesses network data stored in one of the project's databases, specifically, an Elasticsearch instance and uses this data to build the required machine learning models. SphynxML leverages Automated Machine Learning (AutoML) to automatically evaluate multiple machine learning algorithms, producing ready-to-use predictive models once the necessary data had been collected.

All SphynxML-related metadata is stored in its dedicated PostgreSQL database. After deploying a predictive model, new events are received from EVEREST via a messaging queue, and SphynxML

deliveres its predictions through the same channel. For tools that require explanations of predictions, both messaging queues and RESTful API endpoints are available for communication.

In addition, SphynxML is integrated into the SPA platform with its own user interface, simplifying the processes of model training, monitoring, and maintenance.

An overview of the enabler's internal architecture is provided in Figure 4.

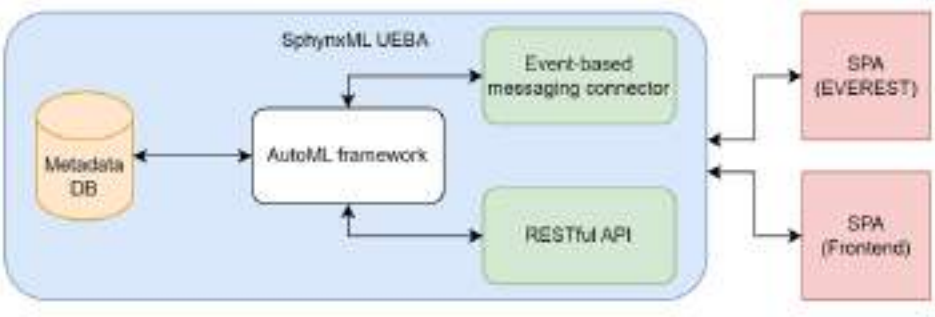

Figure 4. Internal architecture of the UEBA enabler [Source: D3.1].

3.1.3.1 Generated Data Definition

To support the development of the model, synthetic data was generated using a script. The script simulates 6 users, each executing a series of shell commands. Four of the six users are classified as normal users and perform non-malicious commands exclusively. The other two users are classified as malicious and perform a mix of non-malicious and malicious commands with 30% of the commands being malicious. The commands are selected randomly from two lists, malicious and non- malicious. The random selection of commands ensures diversity in the command sequences and minimizes bias. This data generation approach enables the creation of reproducible and configurable test datasets that can be used to evaluate detection capabilities of both malicious and non- malicious conditions.

After running the data generation script over several days, a total of 46,547 rows were produced, capturing detailed records of user activity. Each record contains 238 variables, encompassing a wide range of features including command type, user identifier, timestamp, execution context, and various derived metadata. This rich dataset provides a comprehensive foundation for training and evaluating models under diverse behavioral scenarios.

3.1.3.2 Initial Results Exploration

An initial analysis was conducted on the generated dataset using the SphynxML AutoML system. The platform was employed to automatically preprocess the data, perform feature selection, and evaluate various classification models. The best-performing pipeline identified by the system was based on the LightGBM Classifier, which demonstrated strong predictive performance in distinguishing between normal and malicious behavior. The full pipeline included multiple preprocessing steps such as label encoding, imputation of missing values, one-hot encoding, and column selection based on data type. Specifically, the final pipeline consisted of the steps illustrated in the Figure 5:

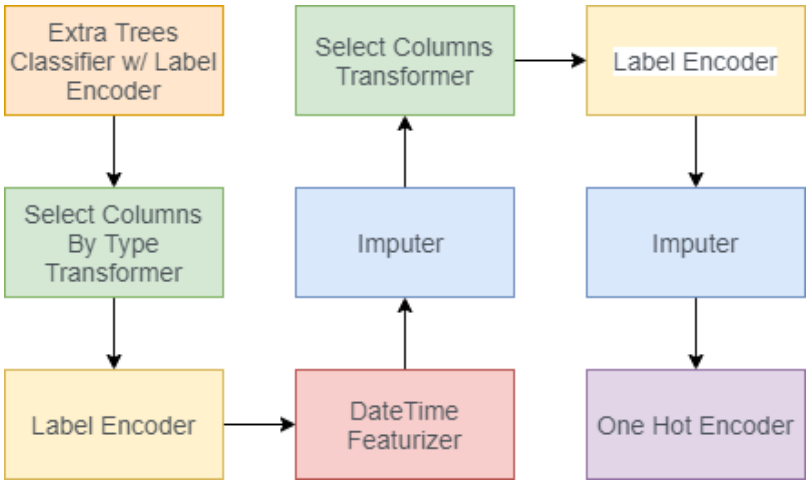

Figure 5. AutoML Pipeline

This configuration highlighted the importance of both categorical encoding and careful handling of missing data in the classification task. The results from this initial phase also guided further refinements in data generation and feature engineering for subsequent iterations. However, additional work is required to validate these results and ensure their robustness and generalizability across different user behaviors and environments.

So far, our analysis utilizes the generated synthetic data to perform an initial analysis. The methodology employed can be examined below step by step.

Initially, the local dataset is uploaded to the SPA platform by clicking in the 'Upload' button and choosing to upload a local dataset.

Figure 6. Main panel of the SphynxML AutoML

After the dataset has been uploaded, a workflow can be created in the 'Worklfow' tab. The workflow is a sequence of steps that defines how the uploaded data will be processed, analyzed, and used for modeling. It begins with selecting the dataset you want to work with, followed by configuring various parameters, namely the time budget to control the time of the analysis, the ML model categories to include in the analysis and whether to include ensembling or not.

Figure 7. Workflow creation step. The Workflow name and description can be set. Also, the use case name, workflow type, train dataset, validation dataset and target variable name are selected. Addition hyperparameters can be set

After the Workflow has been created and the training is completed, the Workflow holds information about the created pipelines. Each pipeline consists of a machine learning model and additional preprocessing steps. By clicking in the completed Workflow we can view the results as seen below.

Figure 8. Workflow results part 1. Here we can see the Workflow information and the performance of each model

Figure 9. Workflow results part 2. Below the performance we can view the variable importance, the steps of the model and the logs that were generated from the AutoML

The Workflow is completed, but no model is yet deployed and ready for predictions. The best-performing model was an **Extra Trees Classifier** with a comprehensive preprocessing pipeline including label encoding, feature selection, datetime feature extraction, imputation, and one-hot encoding. The pipeline was optimized using cross-validation across three folds.

The model demonstrated consistent performance, with mean F1 scores ranging from **0.445 to 0.458**, and high discrimination metrics including **AUC values above 0.80** and **balanced accuracy exceeding 0.80**. These results indicate moderate model performance in distinguishing between classes, indicating challenges in correctly predicting the minority class. However, **discrimination metrics** such as **AUC (~0.81)** and **Balanced Accuracy (~0.81)** suggest that the model is still effective in distinguishing between classes overall, making it a reasonable choice in contexts where class imbalance is a concern.

We continue by creating a deployment. We can create a new Deployment in the deployment tab as seen below.

Figure 10. Deployment create view. The name and description of the Deployment can be set. Also, the use case, the worklfow and type of the deployment are selected. The model to deploy can also be selected

The model is now deployed and ready to receive logs for processing and prediction. Each sample (log event) that comes for prediction follow the same preprocessing as done in the training phase. In case of a malicious sample, the deployment will trigger the SPA in the backend and inform that the specific samples is possibly malicious.

3.1.4 Final Status

The final version of the model is envisioned as a robust, production-ready pipeline capable of real-time detection of malicious and non-malicious command-line behaviour. Building on the insights gained from the initial analysis, the model will incorporate advanced feature engineering techniques, refined input data representations, and optimized preprocessing steps tailored to the specific characteristics of the domain. We anticipate the final pipeline to leverage the strengths of gradient boosting methods, such as LightGBM, due to their proven performance in handling high-dimensional, mixed-type data.

4 CTI Discovery, Analytics & Threat Hunting Enabler

This section presents the work done in Task 3.2 “CTI Discovery, Analytics & Threat Hunting”, the main outputs of which are the CTI Discovery & Analytics; the Threat Intelligence Integrator (TII); and the Threat Actor Context (TAC) Ontology.

4.1 CTI Discovery & Analytics

4.1.1 Overview

The CTI Discovery & Analytics component of PHOENI2X is designed to enhance the Cyber Threat Intelligence (CTI) process by gathering, processing, and disseminating threat information from different online sources, including dark web forums and social media. Using web scraping, machine learning, and natural language processing, it converts unstructured data into standardized formats like STIX 2.1. The processed information is then stored and sent to a TAXII server for distribution to CTI platforms such as OpenCTI, ultimately aiding in the identification of current and emerging threats. Detailed information about the purpose of the component, its functionality, data processing, information flow, integration with other systems and architectural context can be found in previous issue of this deliverable, in section 4.1 of **D3.1**.

The CTI Discovery & Analytics includes several key subcomponents: a **Crawler** that identifies new links on targeted websites; a **Scraper** for data extraction; a **PDF Decomposer** for converting PDFs to HTML; and a **Preprocessor** that organizes the extracted data. Additionally, a **Decider** module assesses the relevance of the data, while the **Internal DB Manager** structures or discards this information. The **Entity eXtraction Module (EXM)** processes and extracts STIX 2.1 entities, which are then used by a **STIX mapper** for knowledge creation. An **OpenTAXII server** facilitates sharing, supported by a custom REST API controller. Further details about the design of this component can be found in section 4.1.2 of **D3.1**, which is the previous issue of this deliverable.

Figure 11 illustrates the key elements of the Proof-of-Concept (PoC) design for the CTI Discovery & Analytics components that described above.

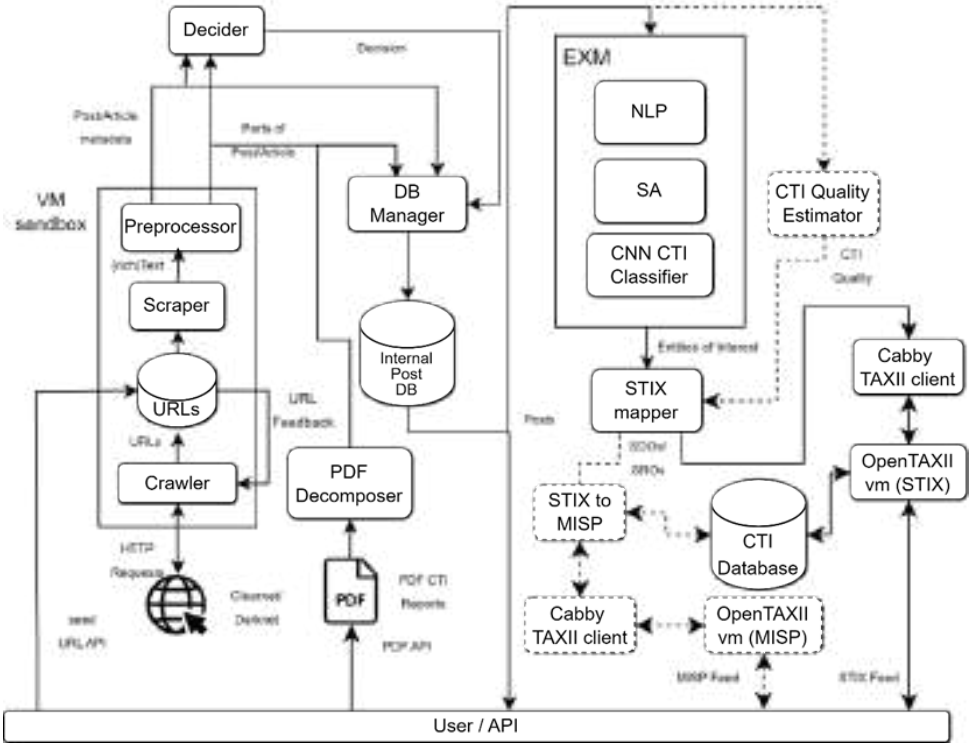

Figure 11. Internal architecture of the CTI Discovery & Analytics component’s Proof-of-Concept design (Source: D3.1)

4.1.2 Implementation Details

The Proof-of-Concept implementation of the above architecture is focused on a few modules that incorporate many of the described functions, as illustrated in Figure 12.

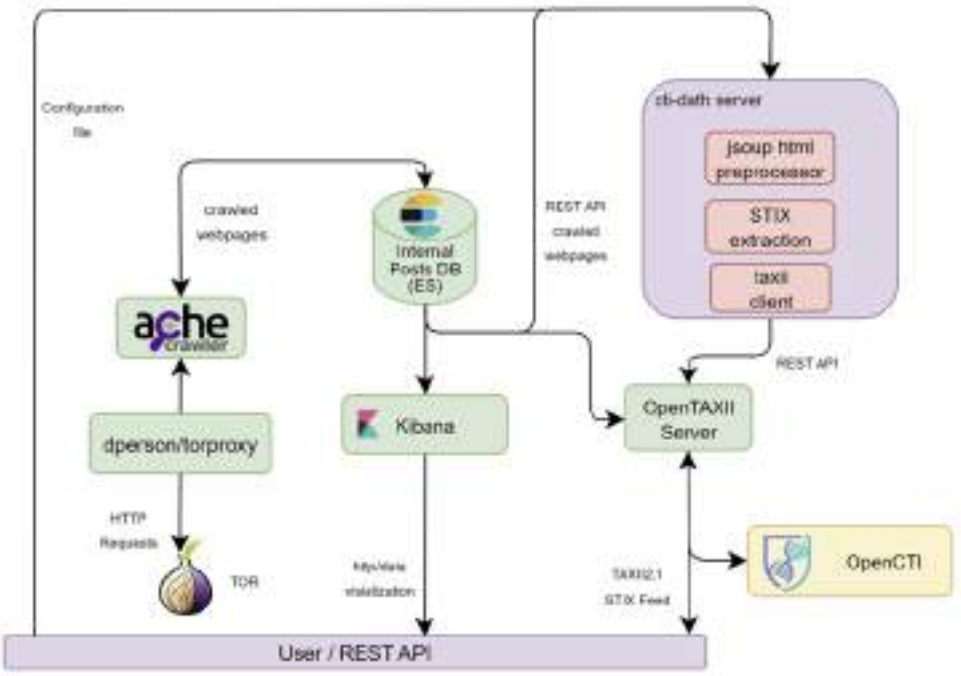

Figure 12. Internal architecture of the CTI Discovery & Analytics Proof-of-Concept implementation (Source D3.1).

A list of the functionalities covered by each module, such as the **torproxy**, **ACHE crawler**, and the **configuration file**, but also additional implementation details can be found in Deliverable **D3.1**, Section 4.1.3 “Implementation Details”.

At this stage, we are able to provide further details regarding certain configuration options that were not included in Deliverable D3.1. A more comprehensive description of these options, as defined in the **configuration file**, is presented below:

- `mainTextIdentifier` is the html tag that identifies a Forum Post text. It is used by the Jsoup HTML parser to extract it. The same goes for `authorUsernameIdentifier`,`creationDateIdentifier`,`creationDateIdentifier` and `postBoxIdentifier`.
- **`minimumCTIRelevanceThreshold`** CTI-dath uses a tf-idf statistical process to evaluate wether a forum post has CTI value or is irrelevant. A good starting point is a value of 10-15%, as below that you will get irrelevant content, and above that, almost nothing.
- **`minimumLetterPercentage`** A good way to estimate if something is human text, or some bot-generated ad, is the letter/symbol ratio: the more alphabet letters(abcABC) there are in a sentence compared to symbols (like !@$> etc.), the more possible that it is human generated.
- **`numberOfThread`**s the forum post selection process can be quite CPU-intensive, and in some cases, multithreading makes sense. This option specifies the number of threads you want to use in this process (a good starting point is a little less than your actual CPU cores).
- **`domainNameOfServer`** the domain name of the server that runs the cti-dath services: OpenTAXII,Elasticsearch and ACHE Crawler. Can also be localhost.
- **Crawler_manager.downloader.cookies:** provides the Session cookies that the crawler uses to emulate user login in some forums/websites that this is deemed necessary.

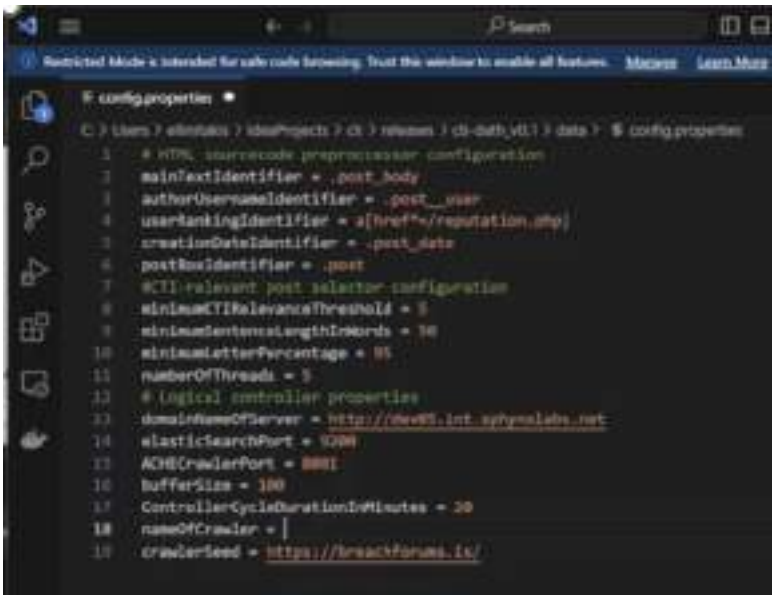

Figure 13. Configuration file of the CTI Discovery & Analytics component. (Source D3.1)

In addition, Figure 14 shows the configuration of User Session Cookies within the CTI-DATH configuration file for the blog site "BreachForums." This section specifies how user session cookies are managed, allowing the system to properly authenticate and maintain sessions when interacting with the blog. The configuration ensures that automated processes can access necessary areas of "BreachForums" by simulating valid user sessions based on predefined cookie values.

Figure 14. User Session Cookies configuration in CTI-DATH config file for the blog "BreachForums" (Source D3.1)

In summary, the implementation consolidates several key functionalities into a few core modules, as depicted in Figure 12. Additional configuration details, such as session management and content parsing parameters, have been incorporated to enhance the system's ability to automatically collect and assess CTI-relevant content from targeted sources. These implementation choices lay the foundation for the continuous development and refinement of the CTI Discovery & Analytics component.

4.1.3 Final Status

The final status of CTI Discovery & Analytics provides a comprehensive solution for exploiting unstructured threat intelligence information available in the web through the continuous collection and conversion to a standard-based format (i.e., STIX 2.1). In this way, the unstructured threat intelligence information will be ready for consumption and sharing among threat intelligence teams and security analysts. The CTI Discovery & Analytics supports the whole lifecycle of CTI, from collection through the web crawling module, conversion to STIX 2.1 format through ML and NLP, and incorporation to OpenCTI through custom scripts and GraphQL queries. After the ingestion to OpenCTI the CTI are ready to further analysed by security analysts or shared with other security teams as a CTI feed.

Moreover, as part of supporting better the activities of the CTI-DATH, namely to leverage all the information that is brought to the OpenCTI through CTI-DATH, SPHYNX have developed a tool named *CTI Assessments.*

The CTI assessment has been developed and implemented as a module of SPHYNX SPA Suite to assist security teams in the collection of specific threat intelligence information from the threat management platforms, such as OpenCTI. CTI Assessments is a tool that correlates existing SPHYNX Assurance DB assets with potentially relevant CTI information, while the latter performs arbitrary searches in the CTI database that utilize both the Assurance DB and custom keywords. The CTI Assessment and Query results, along with CTI Metrics, provide analysts or end users with meaningful insights about imminent threats and help create threat landscape awareness. More specifically, to collect threat intelligence that are directly related with an organization. For instance, it can retrieve CTI information about all malware that targets or exploits the organization's assets.

4.1.4 Lessons Learnt and Recommendations for Future Work

Due to the project's scope, direct validation with end-users across use-case scenarios was limited. Future work should include dedicated phases for usability testing and feedback collection to improve operational relevance.

Future Work:

- Conduct performance evaluations of CTI-DATH in real-world or simulated environments to assess scalability and responsiveness.
- Collaborate with domain-specific stakeholders (e.g., SOC teams, CERTs) to fine-tune usability and operational fit.
- Explore integration with additional platforms beyond OpenCTI, such as MISP or commercial TIPs, to enhance interoperability.

4.2 SPA CTI Assessment

4.2.1 Overview

CTI Assessment component is a new tool that is added to the SPA Platform and is specialized and designed to support CTI Analysts in performing dynamic threat analysis and intelligence-driven security assessments. Its primary function is to enable the creation and execution of custom queries over threat intelligence data aggregated within an OpenCTI instance. By doing so, analysts can define tailored Security Assessment procedures that align with specific organizational risk profiles, threat models, or operational contexts.

The tool is implemented using the Java Spring Framework, ensuring a robust, scalable, and maintainable backend architecture. This technology choice allows for seamless integration with existing systems, modular development, and ease of deployment in enterprise environments.

CTI Assessments acts as an intermediary between the OpenCTI data layer and downstream security assurance components. It leverages enriched and filtered CTI data to derive context-aware insights, supports decision-making through automation, and feeds structured outputs into the SPHYNX SPA platform’s and visualization layers. Ultimately, it enhances the analyst’s ability to extract actionable intelligence, define high-fidelity threat detection rules, and continuously refine security postures based on evolving threat landscapes.

4.2.2 Implementation Details

Figure 15 illustrates the implementation of a simple Cyber Threat Intelligence (CTI) search use case within the SPHYNX SPA Suite. The flow of information begins with the ingestion of external CTI feeds and concludes with the delivery of structured intelligence to the end user or analyst.

External CTI feeds from the internet are first collected by the OpenCTI platform. OpenCTI serves as the central intelligence aggregator, filtering and structuring the raw data into actionable intelligence. This filtered CTI is then passed to the CTI Assessments module, which is responsible for analyzing the data and generating assessments based on predefined logic and contextual information.

During this process, the CTI Assessments component interacts with two key modules: it reads contextual data and threat assurance metrics from the SPA Assurance Database (DB), and it performs write operations to the SPA Core. The SPA Core acts as the central logic and coordination engine for the platform.

Following the assessment, formatted results are generated and sent to the SPA Frontend. The frontend is responsible for presenting the information in a structured and user-friendly manner to the end user or analyst. This enables effective threat analysis, decision-making, and incident response.

This implementation supports a structured, modular approach to CTI management and ensures the integrity and relevance of the intelligence provided to users.

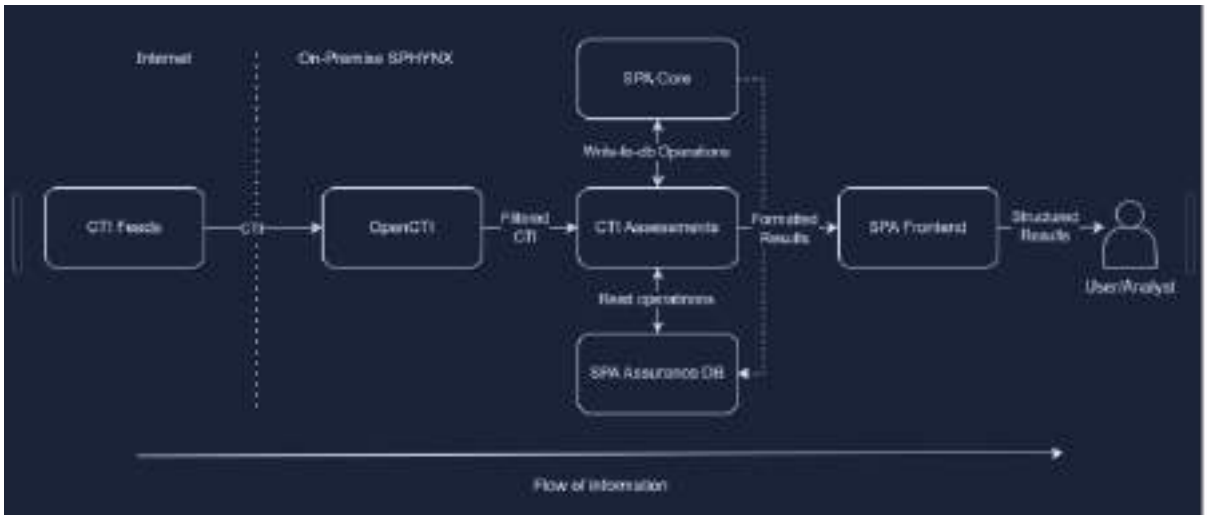

Figure 15. Simple CTI Search Usecase

The main components working together to provide the CTI Orchestrator functionality are the following:

- **SPA Core Platform (Assurance Database)**

  Contains all the information that is created in the SPHYNX Security & Privacy Assurance Core Platform [1]in a Postgres database.

- **SPHYNX OpenCTI**

[1] Deliverable D2.2 – Baseline Enablers, Section 4

It is an external tool[2] created to gather, organize, and manage dissemination of Cyber Threat Intelligence (CTI) data. After an instance[3] is set up, and feeds[4] are added, the Elasticsearch instance of the app begins storing CTI in STIX format[5]. This can later be accessed by the CTI Orchestrator to create CTI Assessments.

- **CTI Orchestrator/CTI-Assessments**

  This is the component that holds the logic of the CTI security assessments. It contains models, rules and procedures for all the information to be processed and then disseminated to the user through the web UI.

All of the components above are independent of each other, and communicating via request, queries and APIs in a standard fashion.

4.2.3 Final Status

The development and integration of the CTI Assessments tool within the SPHYNX platform have been successfully completed. The tool is fully operational and has been validated through functional and performance testing using real-world CTI data. It demonstrates reliable performance in generating context-aware assessments, handling complex query logic, and interacting with both the OpenCTI instance and SPA components, such as the SPA Assurance DB and SPA Core.

The deployment has met all the initial objectives set during the design phase, including seamless integration with OpenCTI, support for user-defined queries, and structured output generation for the SPA Frontend.

Ongoing support and minor maintenance are expected, primarily to accommodate updates from upstream CTI sources and to refine the tool based on user feedback.

4.2.4 Lessons Learnt and Recommendations for Future Work

The development of the CTI Assessments tool highlighted the importance of modular design, high-quality CTI data, and analyst-driven query customization. Allowing analysts to define custom queries significantly improved the relevance and accuracy of assessments. Regular feedback loops with users proved invaluable in refining both functionality and usability.

For future work, improvements should focus on enhancing data visualization, integrating CTI outputs with internal logs, and scaling the tool for larger datasets. Incorporating AI-driven features, such as query suggestions and threat prioritization, could further support analysts. Additionally, tighter integration with incident response workflows would increase operational efficiency and maximize the tool's impact.

4.3 Threat Intelligence Integrator (TII)

4.3.1 Overview

The Threat Intelligence Integrator (TII), as detailed in D3.1 [19], is a core component of the PHOENI2X project. TII was designed to enhance Cyber Threat Intelligence (CTI) by filtering and prioritising information from the Malware Information Sharing Platform (MISP). The architecture of TII, shown in Figure 16, includes three primary components: the event dispatcher, orchestrator, and HeuristicEngine [19].

[2] https://filigran.io/solutions/open-cti/
[3] https://docs.opencti.io/latest/deployment/installation/
[4] https://docs.opencti.io/latest/deployment/integrations/),
[5] https://oasis-open.github.io/cti-documentation/stix/intro.html

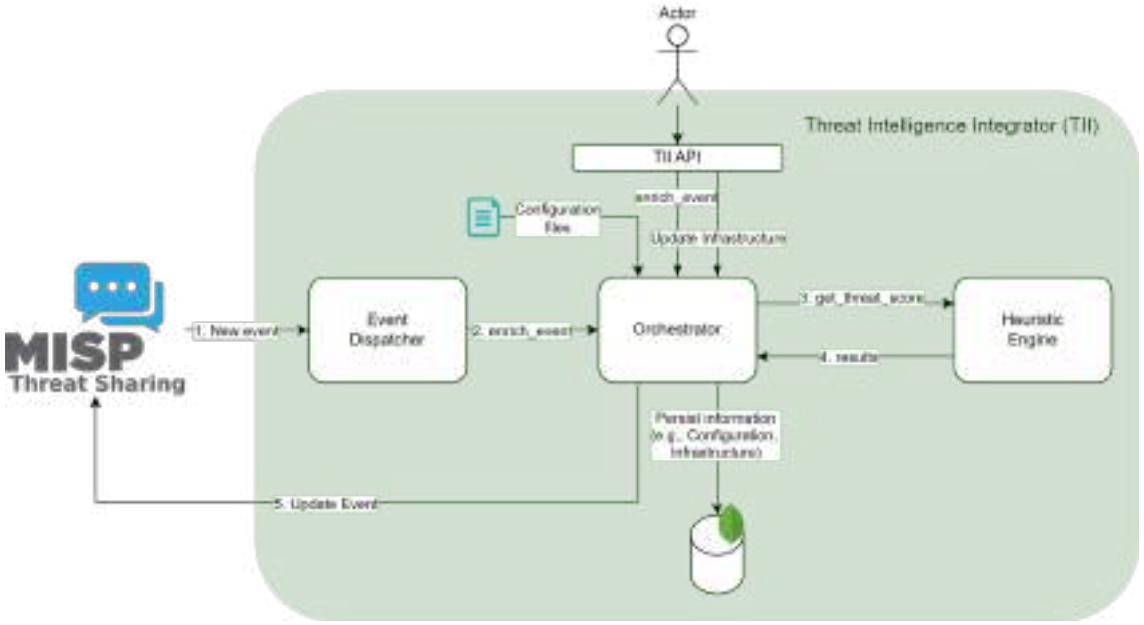

Figure 16. Overview of the architecture of the Threat Intelligence Integrator (source: D3.1 [19])

Figure 16 provides a high-level view of how these elements interact to process and evaluate threat data, ensuring efficient handling of incoming MISP events.

TII's operational workflow begins with the event dispatcher detecting new events in MISP, subsequently relaying them to the orchestrator. The orchestrator, which contains detailed infrastructure information, determines the enrichment steps required and collects necessary data. This data is then passed to the HeuristicEngine, which uses heuristics such as timeliness, trending, completeness, and relevance to assign a threat score. The resulting score is tagged to the MISP event, as Figure 17 shows [19]. This scoring mechanism is vital for prioritising threats in environments flooded with CTI data.

Figure 17. Tag added by TII into a MISP event to indicate the level of threat score (source: D3.1 [19])

During the initial phase of PHOENI2X, TII went into significant enhancements to align with project objectives. These improvements included more granular infrastructure information management, enabling precise mapping of assets and their software components. A notable advancement was the integration with Software Bill of Materials (SBOM) files, enabling improved vulnerability management through MISP vulnerability objects. Figure 18 demonstrates how TII tags events when vulnerabilities are linked to infrastructure components, offering clear visibility into potential risks [19]. Additionally, TII's input/output mechanisms were refined for flexibility, supporting seamless integration with other PHOENI2X components such as CERCA and ROAR via Kafka and API endpoints [19].

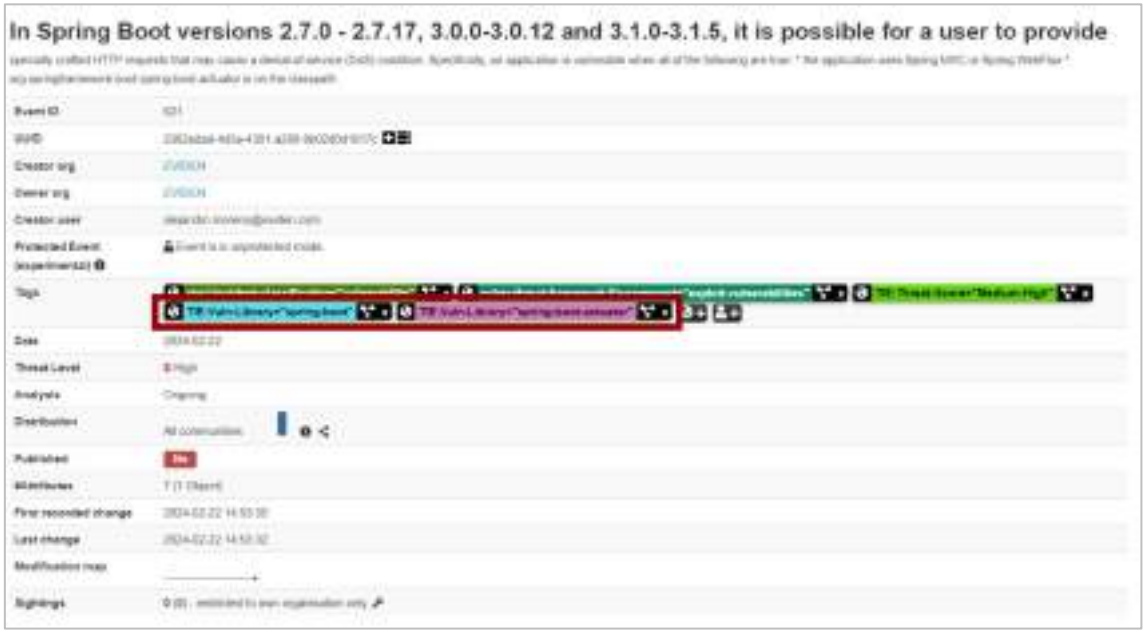

Figure 18. Tags added by TII when a vulnerability is related to libraries present in the infrastructure (source: D3.1 [19])

These enhancements have custom-made TII to address the specific needs of the use cases in PHOENI2X, ensuring it not only filters CTI effectively but also supports sector-specific operational requirements.

4.3.2 Component Final Status

Since D3.1 [19], TII has undergone important improvements as a result of its integration with the project use cases. This integration highlights TII's value and its ability to address sector-specific cybersecurity challenges, aligning with the project's main goals.

A major improvement is the completed implementation of the Software Bill of Materials (SBOM) to MISP vulnerability object events functionality, now a feature of TII. This capability allows TII to process SBOM files containing Common Platform Enumeration (CPE) information, extract software component data, and identify associated vulnerabilities using the NIST National Vulnerability Database (NVD) API. The resulting vulnerability data is then transformed into MISP events with vulnerability objects, which are automatically shared across all connected MISP instances in the PHOENI2X ecosystem. This enhancement improves vulnerability awareness and response times across the project.

This SBOM file information to MISP functionality will be fully integrated into TII's architecture by creating a dedicated Docker container. The container will have its own API to enable other systems within PHOENI2X to access the feature directly. The containerisation approach improves TII's

modularity, scalability, and its ability of deployment. The implementation process includes the following steps:

- **SBOM analysis:** TII employs tools like cve-bin-tool to analyse SBOM files (e.g., in CycloneDX format), and generate vulnerability reports for the listed software components.
- **CVE details retrieval:** for vulnerabilities classified as "CRITICAL" or "HIGH" severity, detailed CVE information, such as descriptions, severity scores, and affected CPEs, is gathered from the NIST NVD API.
- **MISP event creation:** the CVE data is converted into MISP-compatible vulnerability objects, incorporated into new MISP events, and tagged for distribution (e.g., "Vulnerability").

This process enables TII to manage software vulnerabilities, providing useful insights to the specific organisation's infrastructure. Based on D3.1's focus on reducing the volume of CTI by prioritising relevant information [19], this functionality addresses a key challenge identified in the deliverable: linking CTI to specific infrastructure components. By incorporating SBOM data, TII enhances situational awareness, fulfilling the AI-assisted objectives defined in D3.1.

Integrating TII into the project's use cases, combined with the SBOM to MISP enhancement, shows that it plays an important role in the PHOENI2X cybersecurity framework. It reduces the manual effort required for threat management, allowing cybersecurity teams to focus on critical priorities.

4.3.3 Lessons Learnt

The development of new functionalities within the TII has produced important findings that enhance its robustness and adaptability within the PHOENI2X ecosystem. These lessons build on what was already explained in D3.1 [19]:

1. **Using standard tools and APIs:** integrating with tools like cve-bin-tool and the NIST National Vulnerability Database (NVD) API has been helpful for using existing vulnerability data, reducing development overhead, and ensuring compatibility with industry standards.

- **Flexibility in SBOM parsing:** the initial focus on the CycloneDX format revealed the need for adaptable parsing mechanisms to support diverse SBOM standards. In the future, a modular parsing approach could let TII handle new formats more easily

2. **Automation efficiency:** automating the creation of MISP events from SBOM analysis has updated the CTI process, minimising manual effort and accelerating threat dissemination.
3. **Contextualisation with infrastructure information:** building on D3.1's emphasis on the importance of infrastructure information [19], the integration of SBOM data with specific assets has been useful for more accurate vulnerability handling and for setting priorities. This makes CTI more relevant to each organisation and improves understanding of the situation.

By continuing to refine its capabilities, TII will remain a robust and adaptable component of the PHOENI2X ecosystem, contributing effectively to AI-assisted cybersecurity.

4.4 Threat Actor Context Ontology

4.4.1 Overview

STIX (Structured Threat Information eXpression)[6] is a standardized, JSON-based language for representing and sharing cyber threat intelligence (CTI) in a consistent and machine-readable way. It enables better collaboration, detection, and response across cybersecurity communities. However, its

[6] https://docs.oasis-open.org/cti/stix/v2.1/stix-v2.1.html

flat JSON structure limits complex relationship modeling, makes schema evolution difficult, and hinders advanced analytics.

To address these challenges, the Threat Actor Context (TAC) ontology was developed by the University of Oslo and standardized through the OASIS TAC Technical Committee[7]. TAC offers a formal, semantic representation of the STIX 2.1 model using OWL (Web Ontology Language), enabling more expressive, flexible, and interoperable CTI. It supports reasoning, inferencing, and integration with heterogeneous data sources, making it better suited for advanced analytical and automation use cases.

TAC is implemented using open-source and Commercial-Off-The-Shelf (COTS) tools. Protégé[8] is used to build and visualize the ontology, while Stardog[9] enables knowledge graph construction and inferencing. The ontology is modular—each STIX Domain Object (SDO) has its own OWL file—and is publicly available on GitHub for customization and extension.

Organizations can use the TAC ontology to semantically enrich their CTI workflows. For example, it can represent threat actors using OWL classes and infer additional insights using external models like Intel's Threat Agent Library (TAL). By integrating internal asset and vulnerability ontologies, organizations can perform automated reasoning, stay threat-informed, and support risk-driven decision-making with contextual, machine-interpretable intelligence.

In the context of PHOENI2X, the TAC ontology, is adopted to enable improved interoperability between CTI sources (e.g., MITRE ATT&CK, MISP Galaxies), supports context-aware analysis, and facilitates automated reasoning to derive actionable intelligence.

4.4.2 AI CTI Extractor Using TAC Ontology

Large Language Models (LLMs) are a type of machine learning that are capable of processing natural language. These models excel in tasks such as content generation, text transformation and information extraction. An extension to LLMs is Agentic AI, where AI agents integrate decision making, external knowledge and tool calling functions to act more independently on a provided task.

The University of Oslo is using this Agentic AI paradigm to transform threat reports into a semantically enriched knowledge graph. This knowledge graph is the threat report content and other CTI structured using as TAC.

AI CTI Extractor uses a series of locally hosted LLMs to extract entities and relationships from the threat report, ensuring that an organization has full control over the extraction process. The advantage of this method is a semi-automated entity extraction and transformation process that is pre-trained on natural language. For each object in the STIX 2.1 specification, we assign an agent with aligning the threat report content with that specific object. This focused approach reduces the risk of AI agents having to operate on too many varying contexts.

We perform this extraction as a workflow. The workflow is a python function, ensuring a deterministic extraction process. The identified entities and relationships are first transformed into their relevant STIX 2.1 objects and then transformed into OWL for inserting into the TAC ontology.

To mitigate the risk of hallucination, we use the TAC ontology, hosted on Stardog, to act as an agent's reasoner. The reasoning rules of TAC provide the agents with deterministic knowledge and mitigate the challenges with the agent's reasoning, such as subjective attributes like "Is this threat actor hostile?".

[7] https://github.com/oasis-open/tac-ontology
[8] https://protege.stanford.edu/
[9] https://www.stardog.com/

Additional knowledge bases such as MITRE ATT&CK are indexed in TAC. Since MITRE ATT&CK provides a Python library for interacting with MITRE ATT&CK enterprise we use a pure Python approach for this indexing, instead of the agentic AI approach.

The AI CTI Extractor utilizes the Agno agentic framework for generating the workflow, instantiating the agents and providing the knowledge and tool calling functionality. The LLMs that the agents use are hosted locally and ran via the Ollama platform. It is possible to implement an alternative local platform, such as LM Studio, if required.

The knowledge graph is embedded for LLM based querying and retrieval methods. This knowledge graph retrieval augmented generation process supports context-aware analysis, more actionable intelligence, expanded threat report context and other opportunities.

By utilising the workflow, AI CTI Extractor assists organisations with generating actionable threat intelligence.

An overview of the general architecture can be found in Figure 19:

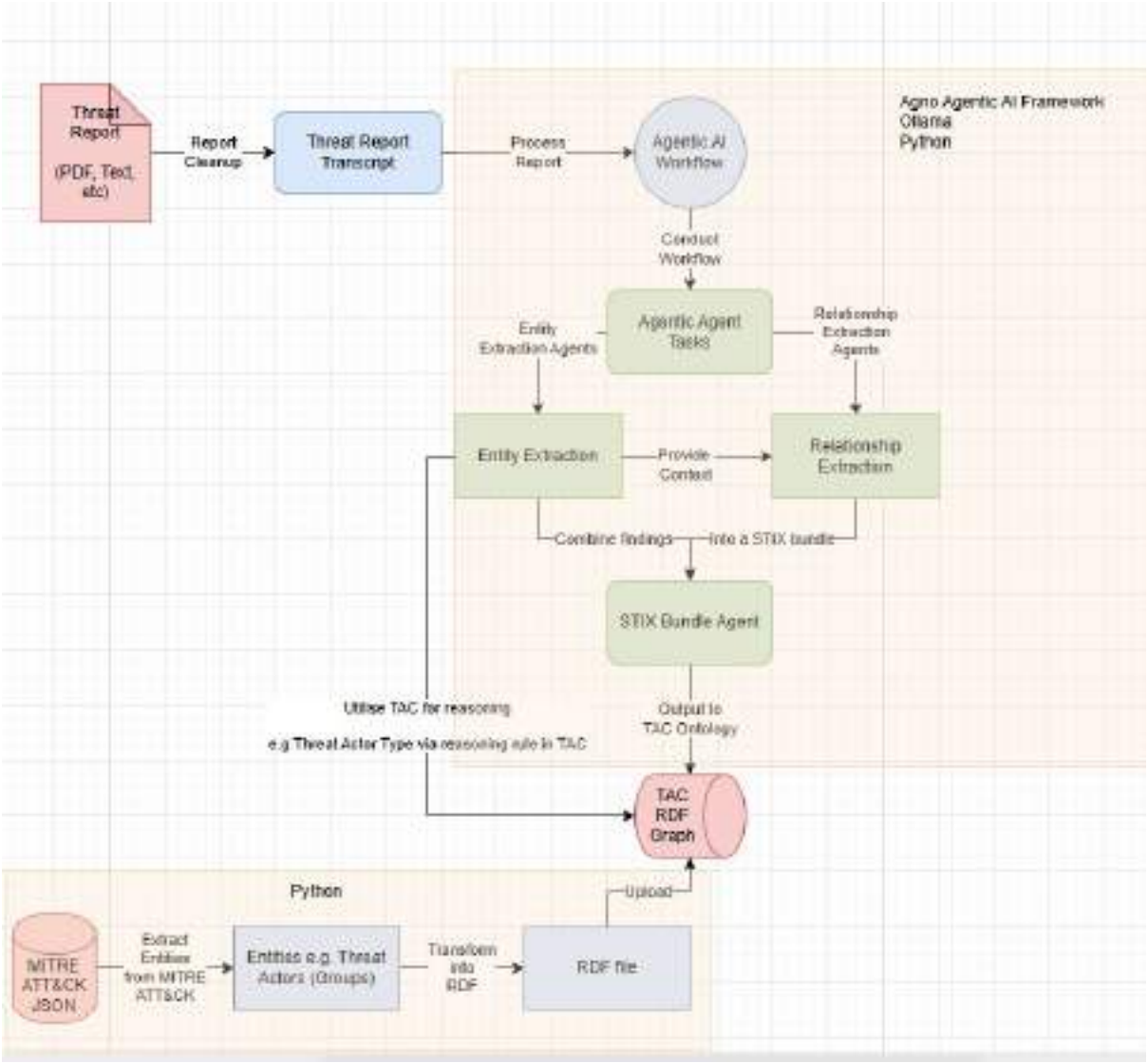

Figure 19. General architecture of the AI CTI Extractor

4.5 Pushing CACAO to the Next Level

The Collaborative Automated Course of Action Operations (CACAO) specification[10] of OASIS provides a structured framework designed to enhance the creation and sharing of cybersecurity playbooks. As cyber threats grow in scale and complexity, organizations face increasing challenges in responding effectively to security incidents. CACAO enables organizations to develop repeatable and shareable orchestration steps that facilitate the prevention, detection, investigation, mitigation, and remediation of threats. By grouping these steps into machine-readable playbooks, organizations can more effectively protect their systems, networks, data, and users in an automated manner.

CACAO enhances the interplay between cybersecurity tools, human supervisory agents, and standard operating procedures and seamlessly integrates with cyber threat intelligence (such as the STIX2 standard), enabling organizations to have a more threat-informed and automated defense. This framework promotes collaboration across organizational boundaries and also supports the testing of an organization's defensive posture through threat emulations. By implementing CACAO, organizations can enhance their preparedness against cyber threats and improve their overall cybersecurity resilience.

CACAO playbooks can be developed by the organization itself, external entities, sharing communities, or vendors, and may vary in their level of abstraction. While some focus on high-level concepts and procedures, others provide detailed, executable instructions that enable effective and more immediate implementation. Figure 20 provides a high-level architecture of CACAO playbooks.

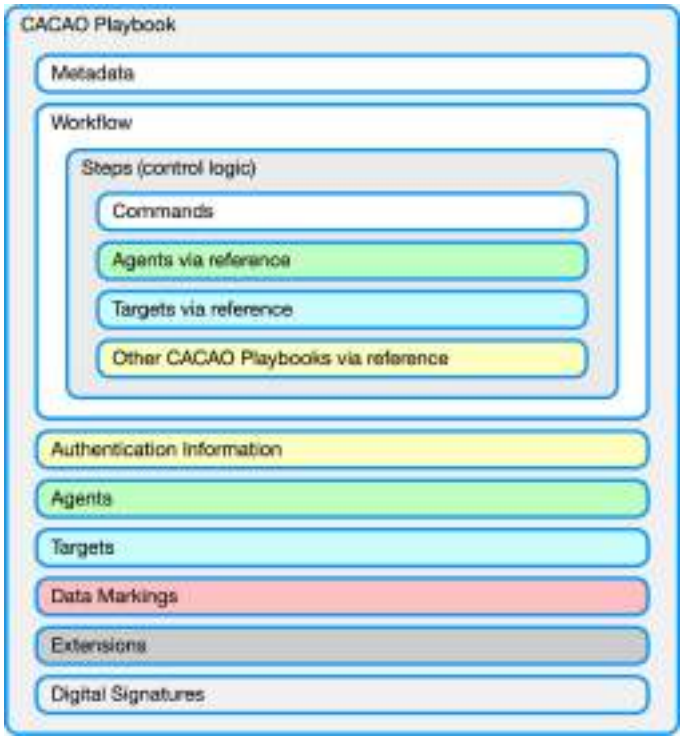

Figure 20: CACAO high-level architecture

4.5.1 CACAO Ontology

The CACAO framework faces similar challenges to STIX, including limitations in modeling complex entities and relationships that arise from the serialization formats used to encode these domains of

[10] https://docs.oasis-open.org/cacao/security-playbooks/v2.0/security-playbooks-v2.0.html

discourse. As a result, performing complex analytics and integrating and harmonizing siloed data remains a challenge. In the push to reach beyond the state-of-the-art, PHOENI2X has initiated both theoretical exploration and practical development of solving these problems by creating the CACAO ontology, aiming to enhance the framework's capabilities for reasoning and automated fact inference while allowing for seamless integration with other models and schemas. This proactive approach seeks to advance CACAO beyond current standards, fostering a more robust and interconnected ecosystem for cyber threat response and playbook orchestration.

Advancing our understanding and utilization of CACAO, we are adopting a modeling language and a serialization format that promotes interoperability and facilitates enhanced integration across various systems. By integrating ontologies and knowledge graphs into our approach, we can improve data management and retrieval. This strategy aligns with the Semantic Web Stack, which emphasizes the importance of structured data through knowledge graphs (KGs) and ontologies. By enabling seamless navigation through information silos and enhancing the ability to traverse complex data relationships, we can provide more insightful analytics and inference capabilities that support effective knowledge management.

The push towards a standards-based framework bolsters our confidence in developing a universally applicable ontology, ensuring that future integrations remain robust and versatile. Transitioning from JSON to JSON-LD and RDF triples will not disrupt existing workflows; instead, it allows for the continued sharing of playbooks in a familiar format while leveraging the ontology for enhanced system interconnections and interactions. This strategic move positions playbooks as practical representations of knowledge graphs, containing detailed action steps, commands, and integrations—essentially, the instances defined by our semantic framework. By categorizing these elements into subclasses, such as integrations, vendors, and specific tool functionalities, we reflect our commitment to a systematic approach that streamlines operations and enhances the effectiveness of our cybersecurity initiatives.

Figure 21 illustrates a hierarchical view of a CACAO ontology. Specifically, it shows the class investigation, defined as a subclass of PlaybookTypes and equivalent to CacaoPlaybooks, with the object property playbookType set to investigation. The ontology uses OWL (Web Ontology Language) to semantically structure the different types of playbooks within the cybersecurity domain. The left panel displays the overall class hierarchy, while the center pane highlights the logical axioms and instance relationships for the investigation class. This modular and semantically rich structure supports enhanced reasoning, classification, and contextualization of CACAO playbooks.

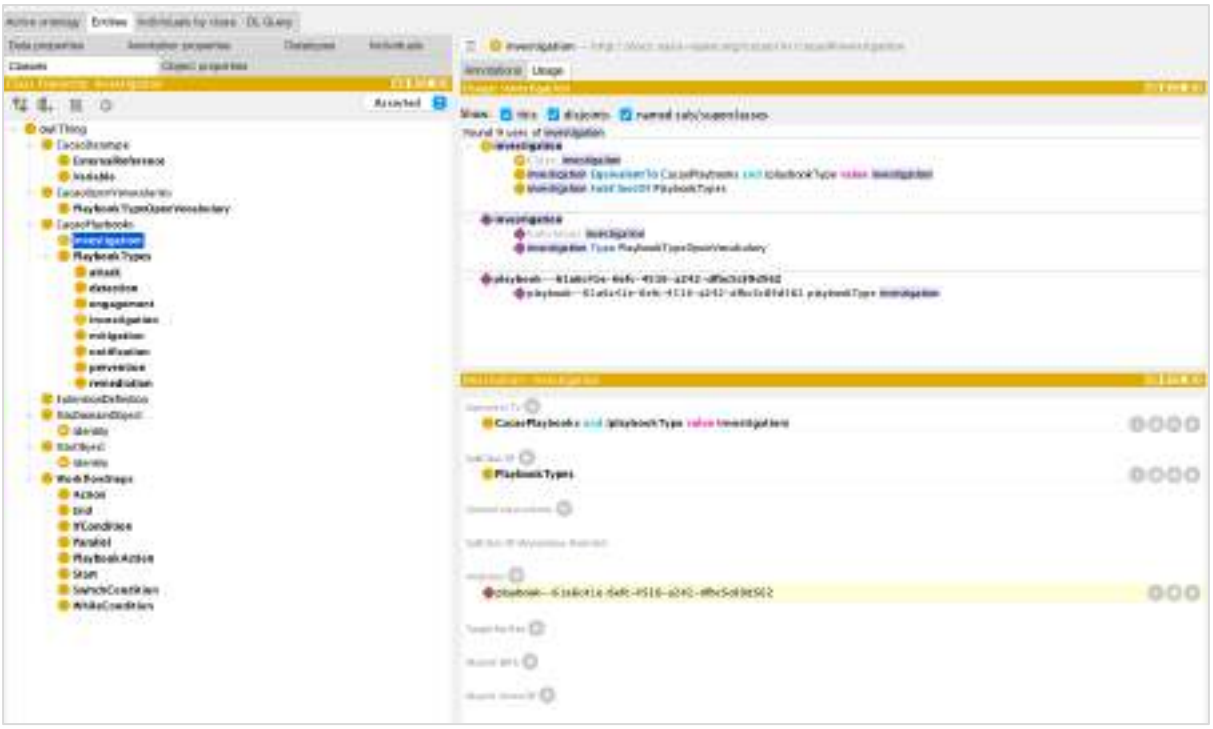

Figure 21: Ontological inference rule inferring an investigation playbook.

Figure 22 presents an individual instance of a CACAO playbook within Protégé, categorized under the class CacaoPlaybooks with a playbookType of investigation. The right panel shows an object property assertion that links this playbook to the investigation type, along with a data property assertion containing a description: "This playbook will look for FuzzyPanda on the network and in a SIEM." This reflects the use of ontologies to create machine-readable, context-aware representations of cybersecurity procedures, enabling automated reasoning and threat response planning within a structured knowledge framework. The CACAO ontology will remain an active project and will progress along with the official CACAO specification. The consortium will release this as open-source and our aim is this work to be continuously maintained by the global cybersecurity, playbooks, and ontology communities.

Figure 22: Inferring an instance to be a playbook based on its characteristics

5 Attack Prediction, Response Recommendation & Adaptation Enabler

5.1 Overview

During this reporting period, progress has been made in the development and deployment of the Attack Prediction, Response Recommendation & Adaptation Enabler, particularly in the PMEM (Predictive Maintenance) component. This enabler is designed to monitor and analyze system behaviour in order to minimize the risk of cyber-attacks by identifying anomalies, predicting potential threats, and recommending suitable mitigation actions.

The enabler consists of three interconnected sub-modules, as depicted in Figure 23:

- Attack Categorization: Identifies and characterizes potential attacks, estimating their scope and impact.
- Attack Detection and Prediction: Detects anomalous behaviors using machine learning algorithms, raising alerts in case of deviations from expected patterns. Recent developments include the capability to forecast infrastructure usage metrics, allowing the identification of Distributed Denial of Service (DDoS) conditions.
- Attack Response and Adaptation: Proposes mitigation and proactive actions aligned with the attack type and its estimated impact. These recommendations are based on the MITRE ATT&CK framework and aim to reduce the risk of cascading failures in critical infrastructures.

Since the last deliverable, the PMEM tool has been enhanced to enable advanced DDoS attack prediction. This functionality has been fully implemented in the Transportation use case. In this use case, resource usage metrics such as CPU, memory, and network bandwidth are continuously monitored and predicted using machine learning models. Deviations from expected usage patterns are automatically identified, facilitating early detection of abnormal load patterns consistent with DDoS attacks.

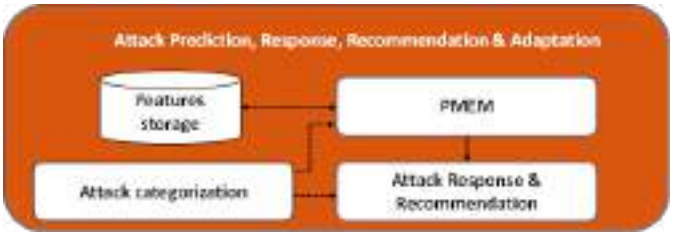

Figure 23. Attack Prediction, Response, Recommendation & Adaptation Enabler

The placement of the Attack Prediction, Response Recommendation & Adaptation Enabler within the detailed PHOENI2X architecture is shown in Figure 24.

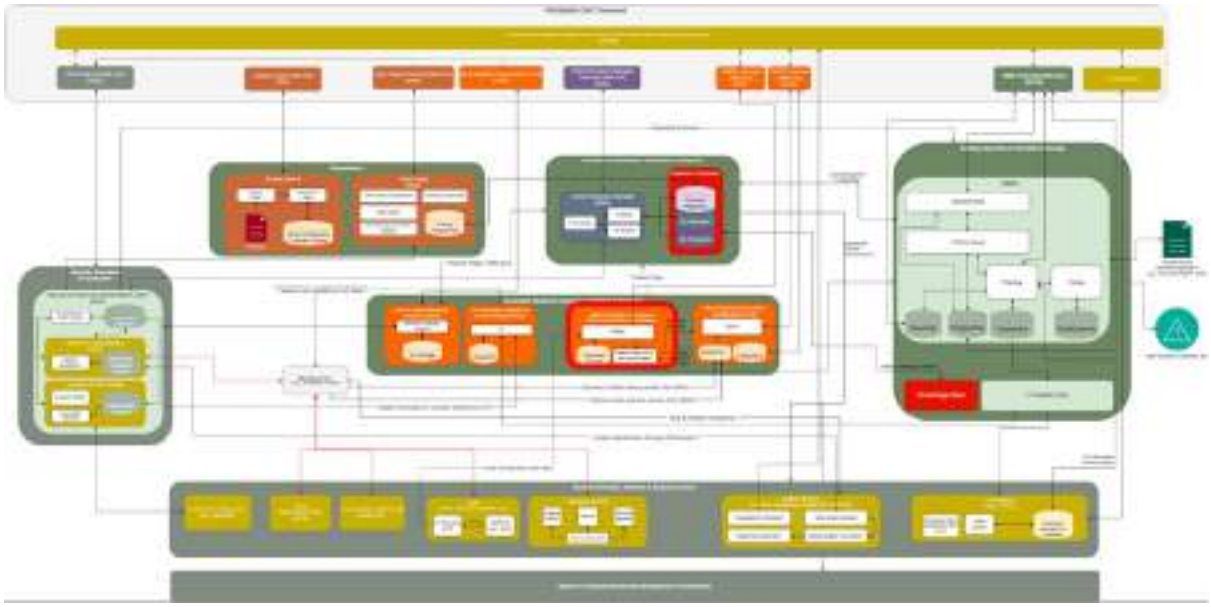

Figure 24. Attack Prediction, Response, Recommendation & Adaptation Enabler within the PHOENI2X architecture.

5.2 Design details

This section presents in detail the design done for the Attack Prediction, Response, Recommendation & Adaptation Enabler.

5.2.1 PMEM

PMEM is a machine learning-based tool developed to detect anomalies in systems. It is structured into two separate modules. The first focuses on detecting attacks in network traffic, while the second is focused toward predicting measurements and identifying abnormal deviations from expected values.

Figure 25 illustrates the PMEM architecture, with the attack detection module on the left and the forecasting module on the right.

PMEM can process different types of data as input (see Figure 25):

- Network traffic data (Input interface of the unmonitored component): Network flows extracted from the monitored infrastructure. This data can typically be captured using tools such as Tcpdump or Wireshark and then transmitted to PMEM using the Apache Kafka or MQTT protocols.
- IoT measurement data (Input interface of the forecasting component): Time series that include the measurement value captured by an IoT device and its corresponding timestamp. These series are also received via MQTT and Apache Kafka.

In addition to IoT measurements, the forecasting module has been extended to analyze system resource metrics—such as CPU usage, memory consumption, and network bandwidth—to support the detection of Distributed Denial of Service (DDoS) attacks. In this use case, deviations from predicted values in these metrics may indicate service overload or malicious activity. To this end, the forecasting module includes additional logic to compare the predicted values of resource metrics with predefined thresholds. When a value exceeds expected behavior, it triggers alerts that point to possible DDoS conditions. This expansion allows PMEM to be applied not only to IoT monitoring, but also to broader IT infrastructure protection.

PMEM stores both the input data and the corresponding output (i.e., the prediction) in a database. The results for both system components are available via a dashboard or API. Whenever an anomaly or deviation is detected, PMEM generates an alert to report potential malicious activity.

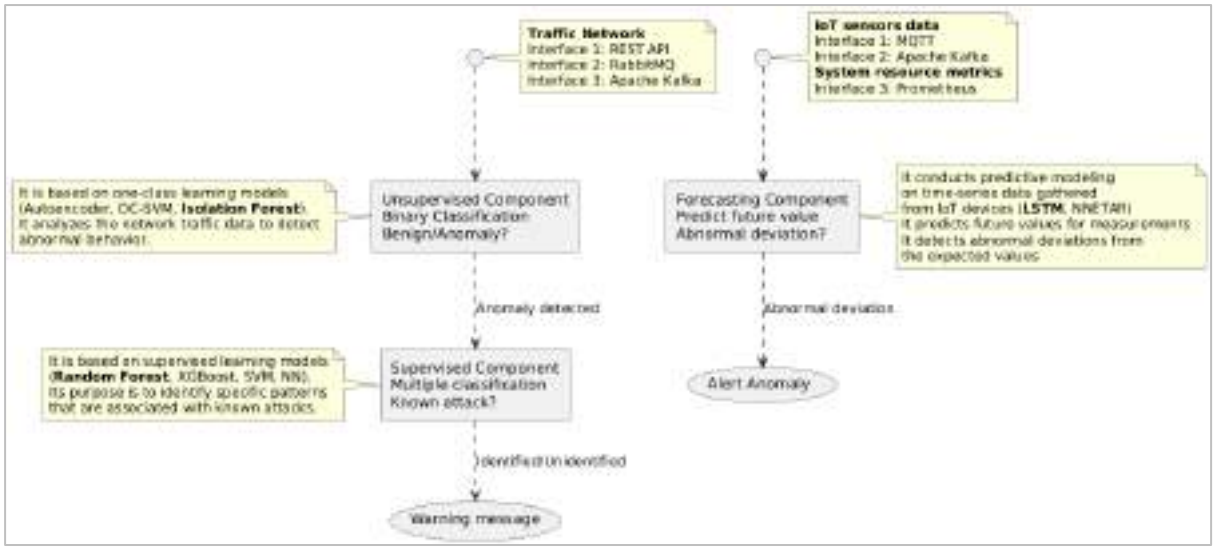

Figure 25. PMEM architecture design.

5.2.1.1 Attack detection module

The general description of this module is provided in the deliverable D3.1 [19]. In this second period of the project, the attack detection module is extended to work as per specific requirement of the attack nature in each use case. The attack detection module makes use of the network traffic which is coming from the infrastructure being monitored. The attack detection module is updated to detect the different types of the DDoS attacks generated by the adversary. The attack detection module has used the supervised machine learning approaches to classify the network flows as DDoS or Normal. The attack detection module is trained previously with the offline captured normal network flows along with the DDoS network flows simulated using different techniques. The raw network traffic which is being captured from the monitored infrastructure contains raw network packets, the bidirectional network flows are being extracted and they are being passed to the supervised component. The supervised component consists of different trained ML algorithms and each model assigns a label to each network flow. The output of each model is then merged using the softmax voting approach which assigns the final label to each network flow. The final output is feeded to the Recommendation module which based on the predicted label do further analysis and provide the list of recommended actions based on the predictions.

5.2.1.2 Forecasting module

During the current reporting period, the forecasting module has been extended to support Distributed Denial of Service (DDoS) detection. The module now ingests system resource usage metrics—including CPU usage, memory consumption, and network traffic—as time series inputs. The objective is to detect abnormal load patterns consistent with DDoS attacks by identifying sustained deviations in these metrics.

A significant design enhancement has been introduced in the model training pipeline: instead of training a single LSTM model per metric, the system now trains a set of models with different architectures and configurations. These include baseline predictors, dense networks, convolutional layers, and recurrent models. After evaluation, the model with the best performance (based on forecasting accuracy) is selected for deployment. This selection process helps to adapt the system to changing patterns in resource usage and improves detection robustness.

To detect anomalies, the predicted values are compared against dynamic thresholds derived from historical quartiles and prediction error distributions. If sustained deviations beyond the defined bounds are detected across multiple metrics, the module flags a potential DDoS attack.

5.2.2 Attack response and recommendation

The attack response and recommendation module is updated in the second iteration of the project to provide a list of suitable suggestions which can be used to totally mitigate the effect of the attack or helps in reducing the damage caused by the adversary. The module provides the list of mitigation actions as per the guidelines provided by the MITRE ATT&CK framework[11]. The attacker can use different types of the techniques to simulate the DDoS attack. The module considers different types of attack techniques such as UDP flooding, TCP flooding and Land flooding to generate an attack. In UDP flooding the attacker overwhelms a target with a high volume of UDP packets. If the larger number of packets are using the UDP protocol a recommended suggestion will be to Filter Network Traffic. This recommendation can be enforced with the help of filtering the UDP packets at the firewall. In TCP flooding, such as SYN flood attacks the attacker can use either a single source or multiple source IP address to send the traffic. In this case the recommended suggestion will be to block the src IP address. In Land flooding attack the attacker can use the same IP and destination IP address to flood the network traffic. In this case the packets which have the same IP address can be blocked. A list of the recommendation and mitigation actions are provided to the responsible module which ultimately prioritised these recommendations and enforce mitigation actions to remove the adversary impact caused by the attacker.

5.3 Implementation Details

This section describes the design activities performed during this period related to improving the PMEM component for DDoS attack detection. In the Transport use case, the forecasting module has been extended to analyze system resource usage metrics (such as CPU, memory, and network traffic) to identify abnormal patterns that may indicate Distributed Denial of Service (DDoS) attacks.

In the Energy use case, the attack detection module is extended to work on classifying the malicious and normal behavior of the network traffic intended towards the AMI-headend which handles all the communication with the smart meters. This module is now able to use the Supervised machine learning approach to classify the Normal traffic flows from the Denials of service networks flow. The Machine learning models are trained on the Normal network flows along with different attack traffic simulated for the DDoS attacks. The notification alerts are being generated when the malicious network flows are being predicted and a list of recommendations are being provided to mitigate the effect of the attack.

5.3.1 Tools

In the second period of the project, the tools used for PMEM implementation included those initially presented in D3.1 [19], as well as Paramiko[12], a Pure-python SSHv2 protocol. Paramiko has been used to securely access services without requiring complex additional configuration.

5.3.2 Enabler implementation for the Energy Use Case

The Energy use case consists of a centralised Advanced Metering Infrastructure (AMI) headend which acts as a central hub for communication for data and smart meters. The AMI-headend needs to be protected against different network Distributed Denial of service attacks that can disturb the whole communication infrastructure and can result in causing several damages if no appropriate actions are taken to mitigate the effects of the attacks. The main objective of this use case is to design and implement Machine learning models which can predict and classify the network flows as Normal or

[11] https://attack.mitre.org/
[12] https://www.paramiko.org/

Malicious. The trained ML models are able to provide early detection of the denial-of-service attack and this early detection can be used to generate some alert and provide mitigation action to the collaborating tools that can enforce some actions in the lower infrastructure to mitigate the effects of the attack to avoid further damage.

The following are the main steps used to train and validate the system against these types of threats.

- Port Mirroring the network traffic of the AMI-headend.
- Capturing the network traffic using the TCPDUMP.
- Features extraction using the CIC-FLOWMETER to extract the statistical features from the network traffic.
- Training a combination of different Machine learning models.
- The prediction results are being sent to the kafka server to be analyzed further.

5.3.2.1 Data collection

The Data collection process consists of capturing all the network traffic intended towards the AMI-headend. This network traffic contains all the network traffic being transmitted between the smart meters and AMI-headed. This network traffic is being forwarded to a separate Virtual Machine (VM) through port mirroring mechanism. All the network traffic is now intended towards the AMI-headend now can be observed on the mirrored port. The TCPDUMP is being used to capture the network traffic from this mirrored port. A bash script was developed to capture the network traffic for a specific duration of the time and dumped the network traffic in PCAP format on the local machine.

5.3.2.2 Feature Extraction

The collected PCAP file contains the raw network traffic which cannot be directly used to train the machine learning models. Different types of network statistical features need to be extracted from this network traffic to be used further to train the models. For the feature extraction task, an open-source tool CIC-FLOWMETER is being used to extract the bidirectional flows. It provides overall more than 80 statistical network features like Duration, Number of forward and backward packets, Number of bytes, Length of the packets etc. that can be used to calculate the overall number of packets in both directions. The network also contains a unique Flow ID, Source IP, Destination IP, Source Port, Destination Port to provide further insights about the users who are communicating with the AMI-headend. This info is helpful in triggering the alerts to mitigate the malicious network entities. The output of all the extracted network flows is being stored in the CSV format which is further being used. A shell script is designed to automate this process of data collection and features extraction. The script can be configured to capture the network traffic for a specific duration of time and from a specific port where the mirrored network traffic is coming. The script automatically calls the TCPDUMP for a specific duration of the time, passes the captured PCAP file to the dockerized version of the CIC-FLOWMETER and the extracted features are then transmitted to the API endpoint where the ML model is performing the Flow by Flow predictions.

5.3.2.3 Model Training

The Machine learning models are being trained using the extracted network features obtained from the PCAP file. The Model training consists of a complete workflow of preprocessing, feature importance, models training and validation on the test dataset. The training of the model is performed in the offline mode and then trained models are being used in the real time manner. For the training purpose, the Normal network traffic is being generated on the AMI-headend and the traffic is captured using the TCPDUMP. In a similar manner different types of the Distributed denial of service attacks were being simulated and the attack was performed on the AMI-headend. Overall, 4 types of the denial-of-service attacks were simulated and performed. Both Normal and attack traffic is then being used to train the system to perform the attacks classification. Initially the features are being extracted

from the PCAP files using the CIC-Flowmeter. The raw extracted features need to be pre-processed further to remove the abnormalities in the data and perform of data normalization. To ensure that the trained model is not biased towards the IP address the features containing the information about the IP address are not being used during the training stage. The system considered different ML approaches to train the models on the pre-processed data and then these models were deployed as a FAST API endpoint which works in the real time manner to perform the real-time predictions with new data.

5.3.2.4 Validation Results

The trained models were then used in the real time manner to validate their effectiveness. They were deployed as a FAST API endpoint. The bash script which is responsible for capturing the network traffic and extracting features, triggered the API endpoint to generate the prediction for the new data. The trained model performs the prediction at each flow level and the results are stored on the elastic database and different types of the visualisation can be seen on the kibana dashboard. To validate the system, the Normal activity is being performed on the AMI-headend and this traffic is being used to perform the prediction. The prediction results can be shown in Figure 26.

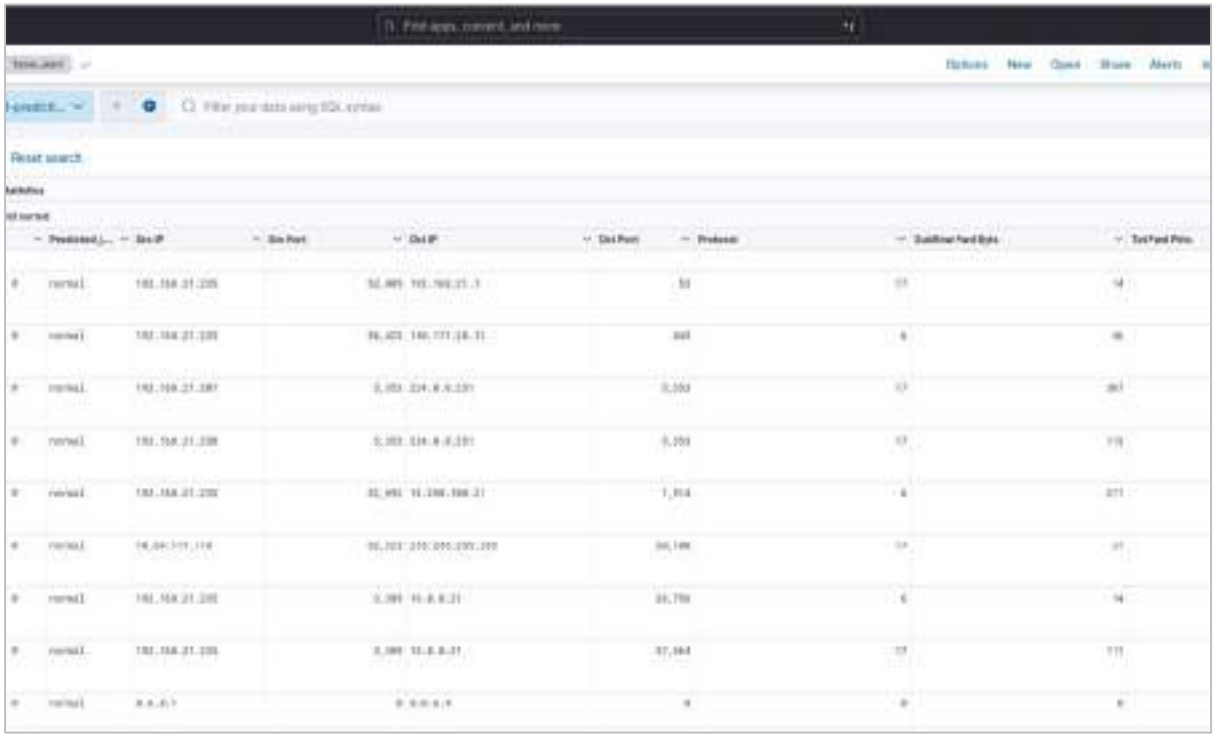

Figure 26. Kibana dashboard illustrates the Prediction Results for Normal Network Flows.

The above figure shows that under the Normal communication intended towards the AMI-headend is classified as Normal. To validate the effectiveness against prediction of the DDoS attack network flows, different techniques are being used to simulate the attack traffic which is then being used to perform the predictions. Under the influence of the attack traffic the trained model has classified these network flows as an attack. The prediction results can be shown in Figure 27.

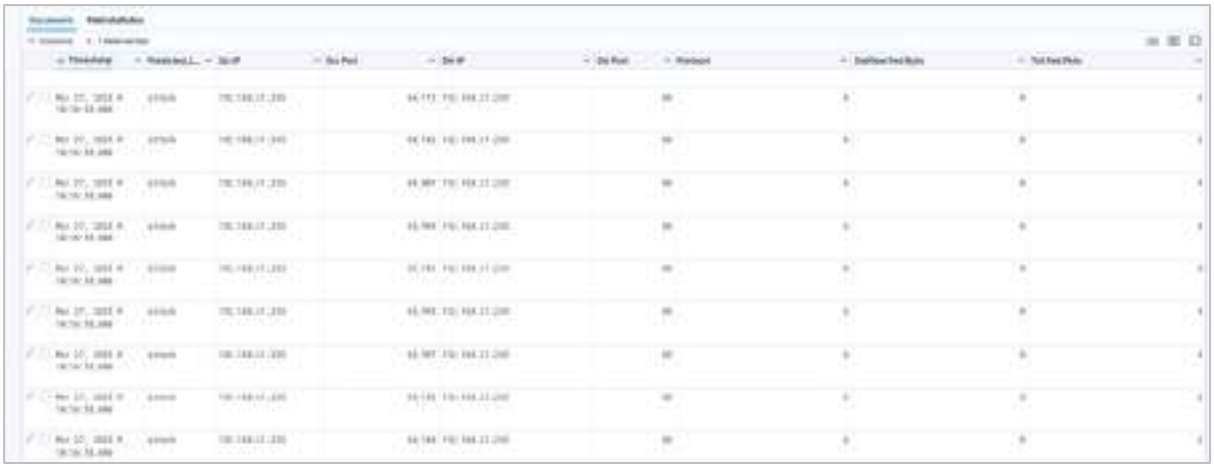

Figure 27. Kibana dashboard illustrates the Prediction Results for DDoS Network Flows.

The prediction results show that the system is able to classify the normal activities from the malicious network flows and perform the prediction in the real time manner. These prediction results are further analyzed and alerts are being generated which are transmitted to the kafka server where the other tools can see this information and can take some actions as shown in the Figure 28.

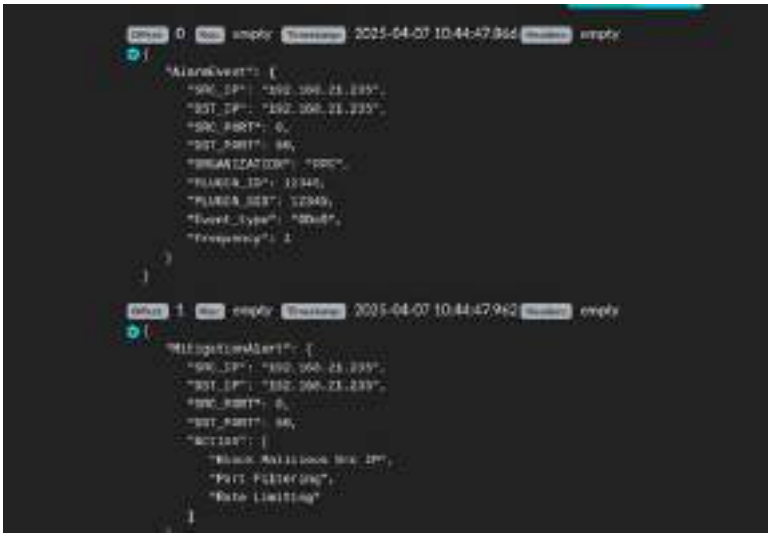

Figure 28. Notifications Alert being transmitted on the Kafka server along with Recommended Suggestion.

The alert messages which are transmitted on the kafka server also provide recommendations which can be taken to mitigate the effect of attack. In case of the DDoS attack the recommended actions are aligned with the MITRE framework; the specific action which needs to be triggered is prioritised and triggered in the infrastructure by the collaborating tools.

5.3.3 Enabler implementation for the Transport use case

This section describes the development of AI-based anomaly detection in the Transportation use case. During this period, work has focused on extending the capabilities of the PMEM module to include the detection of DDoS attacks by analyzing system telemetry data—such as CPU and memory usage and incoming and outgoing network traffic—collected from a Prometheus instance.

5.3.3.1 Forecasting module

5.3.3.1.1 Overview

The primary objective of this use case is to design and implement an anomaly detection component capable of characterizing the usage patterns of a Kubernetes cluster from its telemetry stored in a

Prometheus instance. The anomaly detection is based on training forecasting models on some of the selected variables of the available telemetry and then generating predictions to detect deviations on expected behaviour. The intention of this is to be able to detect DDoS attacks to trigger mitigations.

This component makes use of the following tools:

- Data collection: external Prometheus server and requests (python) to retrieve the data
- Data pre-/post-processing pandas, scikit-learn and NumPy
- Model generation: tensorflow
- Storage: PostgreSQL and local file system, psycopg (Python) to access the database
- Dashboard: Grafana
- SSH bridge to access data from test server within the container: paramiko (Python)

A simplified view of the module (omitting the threshold checker and supporting levels) is shown in Figure 29.

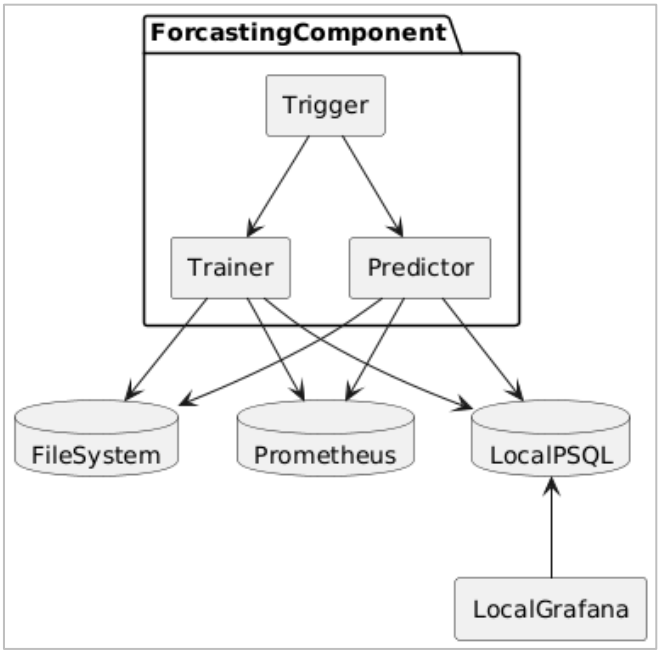

Figure 29. Forecasting components for Prometheus data diagram.

5.3.3.1.2 Data collection

The data for training and generating predictions needs to be retrieved from the Prometheus server. In case it is not accessible directly where the service is deployed, an SSH bridge with paramiko is activated. This bridge is used exclusively by the Python code, and the port is not exposed outside the Docker container. Additionally, it is bound to the localhost interface to prevent accidentally leaking the access outside.

The data is requested from the Prometheus server using a PromQL query to the respective endpoint and cached to the local PostreSQL query. If the module is attempting to train the models, it will request, for each k8s namespace, the largest reasonable block of data containing the average in the predefined increment interval of the variables “network_transmit”, “network_receive”, “cpu_usage” and “memory_usage”. When the module is running inference, it will retrieve only the relevant steps. As the first is not frequent and the second requests a reduced amount of data, the impact in the system is limited.

5.3.3.1.3 Triggering

As there is no messaging platform which the forecasts and analysis can piggyback on to start each process, it is necessary to implement a way to trigger the analysis of the data. In the implementation, loops in background threads wait for a predefined amount of time until generating a request for the relevant processes. In the current implementation, this consists on one to trigger retraining of the forecasting models after certain amount of time; and another to generate predictions for the current telemetry and check for anomalies.

5.3.3.1.4 Modelling

As the usage data is prone to change dynamically, it was decided to implement a structure where a series of model variants are checked and compared in every run to have better results.

A trainer module generates models from historical data for each namespace available in Prometheus and a configuration of different models. After this, it compares the performance of the models and selects the one that has the best performance according to certain metrics. The results are stored in the local PSQL database and the file system to be able to retrieve them afterward.

Table 1 presents the generated models.

Type	Input steps	Output steps	Neurons / Conv units	Use deltas	Add timestamp info
MultiStepLastBaseline	12	24	N/A	No	No
MultiStepLastBaseline	12	24	N/A	Yes	No
RepatLastBaseline	12	24	N/A	No	No
RepatLastBaseline	12	24	N/A	Yes	No
Multilinear	12	24	N/A	No	No
Multilinear	12	24	N/A	Yes	No
Multilinear	12	24	N/A	No	Yes
Multilinear	12	24	N/A	Yes	Yes
Dense	12	24	256	No	No
Dense	12	24	256	Yes	No
Dense	12	24	256	No	Yes
Dense	12	24	256	Yes	Yes
CNN	12	24	256 - 12	No	No
CNN	12	24	256 - 12	Yes	No
CNN	12	24	256 - 12	No	Yes
CNN	12	24	256 - 12	Yes	Yes
RNN (LTSM)	12	24	32	No	No
RNN (LTSM)	12	24	32	Yes	No
RNN (LTSM)	12	24	32	No	Yes
RNN (LTSM)	12	24	32	Yes	Yes

Table 1 Model evaluation – forecasting metrics

5.3.3.1.5 Inference, dynamic threshold generation and anomaly detection

The Predictor module gets triggered every configurable amount of time. It loads a selected model for a given device, the previous N measures and generates the M future measures. After the inference, the predictions are cached in the database for the anomaly detection process to be run on them.

The module first requests the database to generate the first and third quartiles of each variable of the last 5000 metrics and the errors of the 5000 errors between the expected and the real value. Afterward, the box-plot barriers (internal and external) are computed from the quartiles and a minimal separation of 0.5 units is enforced. The resulting values will be the thresholds considered for the anomaly detection. They are additionally stored in the database to be able to represent them in the Grafana dashboard.

Once the thresholds are generated, it is checked if two of the four measures have surpassed the external barriers six consecutive time units or the internal barriers six consecutive time units. Additionally, to consider that a barrier has been exceeded, it is necessary for the given data point to be over the threshold in both the real value and the error. The reason for requiring both is mainly to avoid excess false positives.

5.3.3.1.6 Results

As with the previous cases, a Grafana dashboard was designed to allow the illustration of the forecasting component output. The dashboard allows selecting the k8s namespace and the specific variable. In the upper panel, it shows the cached recorded telemetry (green), the predicted values (yellow) and the threshold ranges (yellow for inside the interior barriers, red in between the interior and exterior and empty for outside the exterior), as depicted in Figure 30.

Figure 30. Grafana dashboard which illustrates the output of the forecasting component for the transport use case on the Prometheus data.

Besides the time series dashboard, there is an additional dashboard (see Figure 31) to be able to check which models have been trained, which of them are currently used and their configuration and evaluation.

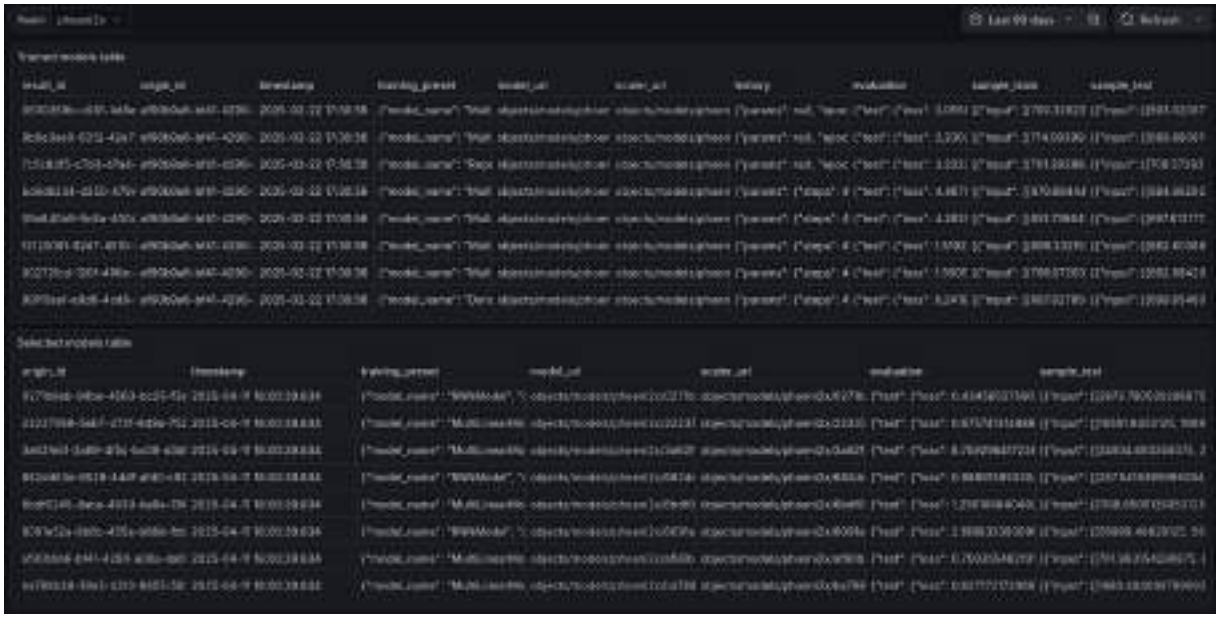

Figure 31. Grafana dashboard with the trained and selected models for the transport use case on the Prometheus data.

When running the module and connecting all the parts, the system is able to train the models continuously retrieve the data, generate predictions and look for anomalies. These are reflected in the logs, as it can be seen in Figure 32.

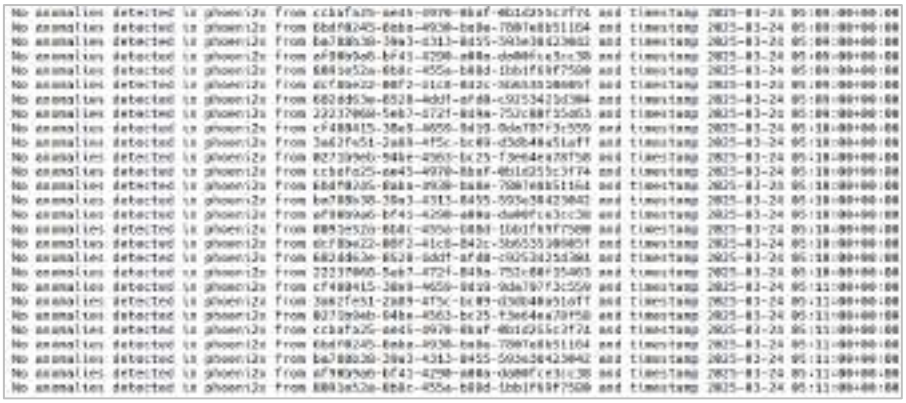

Figure 32. Logs with the results of anomaly detection process.

5.4 Final Status

PMEM was designed and implemented for the Energy and Transportation use cases, integrating anomaly detection and prediction capabilities for early attack detection.

The PMEM architecture and its components (supervised, unsupervised, and predictive) have been fully designed and implemented, encompassing the following key tasks:

- **Data collection:** A real-time data collection process has been implemented for both use cases. This data is preprocessed and stored in a database system to generate the necessary data sets.
- **Modelling**: The analysis and selection of suitable machine learning algorithms for the prediction component were completed. For both use cases, multiple models were trained and evaluated, selecting those with the best performance. In the Transportation case, a robust DDoS detection system was implemented based on system resource usage metrics (CPU, memory, network traffic), using models such as LSTM, dense networks and others, and automatically selecting the optimal model on each run. In the Energy case, an approach

focused on AMI headend network traffic was developed, with feature extraction using CIC-FLOWMETER and application of supervised models to detect anomalous patterns consistent with DDoS attacks. Further a list of recommended actions is provided which helps in mitigating the adversary effects caused by the malicious entity.

- **Storage system:** A storage structure was designed to store both the input data and the outputs generated by each component, ensuring traceability and subsequent analysis.
- **Containerization**: Both components (forecasting & anomaly detection) have been containerized using Docker containers to facilitate the importation and deployment of implemented modules.
- **Integration**: The connection of PMEM's output to the ROAR has been implemented, and the appropriate playbooks are triggered when an attack or anomaly is detected. The notification alerts and recommendations are being transmitted to the kafka server which are available to the CERCA which prioritised the list of recommendation and a suitable playbook will be triggered using ROAR.
- **Validation:** Several tests have been conducted to evaluate PMEM's performance in anomaly and attack detection.

6 Risk Impact Assessment, Alert Triage & Response Prioritisation Enabler

6.1 Overview

This chapter describes the architectural and functional developments added to CERCA since the release of D3.1 where it was initially reported. Chapter 6 of D3.1 [19] should be consulted for implementation details, description of the modules, and the initial system design. The objective of this updated chapter in current D3.2 is exclusively on the reporting of new components, integrations, and operational capabilities implemented after D3.1.

Initial Architecture of CERCA:

The initial CERCA architecture, as detailed in D3.1, sections 6.3.1, comprises five core components: i) the dashboard (Django-based GUI and REST interface), ii) data warehouse (PostgreSQL database), iii) engine (risk model evaluator in Python with R and DEXi support), iv) broker (RabbitMQ for internal communication), and v) NGINX (SSL proxy). Deployment is Docker-based, supporting scalability and modularization. Initial risk models, such as SLARA, relied on static indicators and manual reassessment triggers. Full technical documentation on these components and workflows is available in D3.1, sections 6.2 to 6.3.

6.2 Functional enhancements

The importance of automatic response and risk prioritization is essential for critical infrastructures, as they provide an essential service to society, partial or total disruption of their operations can cause major impacts in the economy and the livelihoods of citizens. To support this, CERCA now incorporates a mechanism for playbook prioritization based on simulated risk reductions:

- When an attack is detected or vulnerability reported, CERCA dynamically prioritizes available response playbooks.
- Each playbook is simulated to assess its impact on current risk levels.
- The playbook offering the highest simulated risk reduction is prioritized for execution.

This prioritization mechanism represents a shift from reactive to proactive, simulation-driven security response.

6.3 New developments

6.3.1 Top-level modules

Two new top-level modules have been added to the CERCA architecture since D3.1:

1. **A mitigation simulation module** (developed at SUNRISE project [13]) that allows to launch simulations in a sequential way (no mitigation strategy is applied), and visualize what would be the effect if a mitigation or set of mitigations is applied:
    a. A mitigation cancels or inverts the effect of (at least) an indicator
    b. The simulation sub-module runs the risk model per each target with the affected indicator values simulated
2. **A playbook prioritization module**, comprised of two sub-modules:
    a. **Simulation strategy**: this module runs simulations based on the selected strategy. For PHOENI2X project, risk reduction driven simulations are available, based on the playbooks data.

b. **Playbook prioritization**: playbooks are assigned a priority score based on the selected simulation strategy.

To allow this, the following actions are required:

- Mitigations in the playbook are mapped to CERCA mitigations, that are mapped to indicators, each mitigation cancels (at least) the effect of one indicator.
- The simulation is run for each playbook, considering all mitigations
- The playbook with the higher risk reduction is the one that is given a priority

These two modules therefore represent an addition to the overall CERCA architecture and are include in the top-level module “cerca-dashboard” or web-application, as seen in Figure 33.

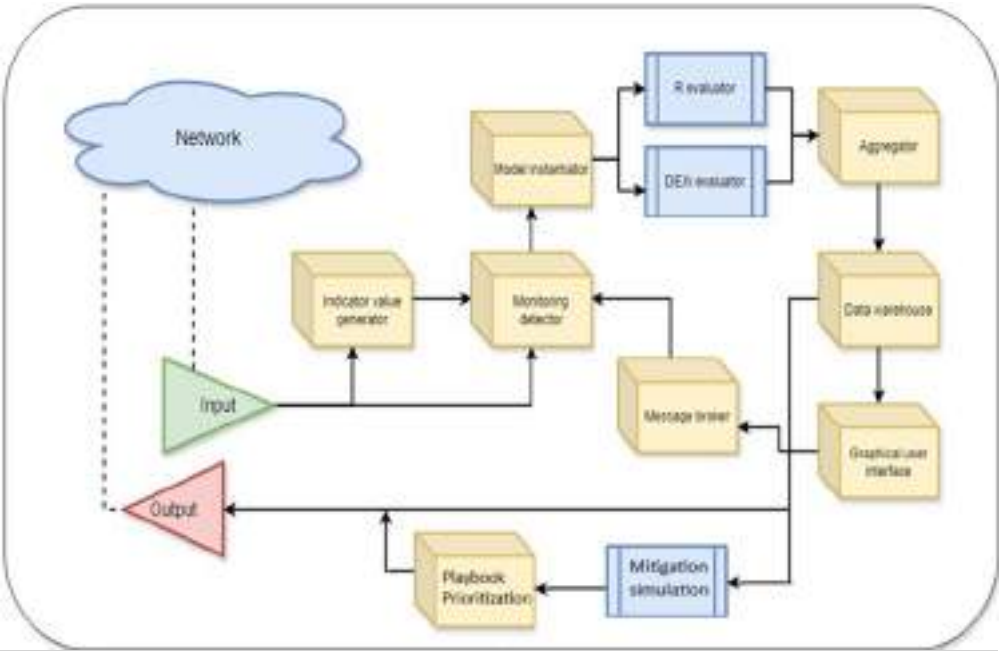

Figure 33. Final CERCA architecture.

Full details of all the components of the architecture of CERCA can be found in D3.1, section 6.2.2.

6.3.2 New integration flows and interoperability

To support automated response and updates from external sources, CERCA has been extended with the following communication and integration flows:

1) CERCA receives from PMEM:
    - A message that indicates an attack has been detected. This message contains basic information for the risk assessment such as the affected target and IP address (based on the available SBOMs) and the type of attack.
    - Suggested mitigations. PMEM suggest mitigations to apply in the simulation

    The overall process is illustrated in the following figures.

    CERCA performs an initial risk assessment with a baseline risk level:

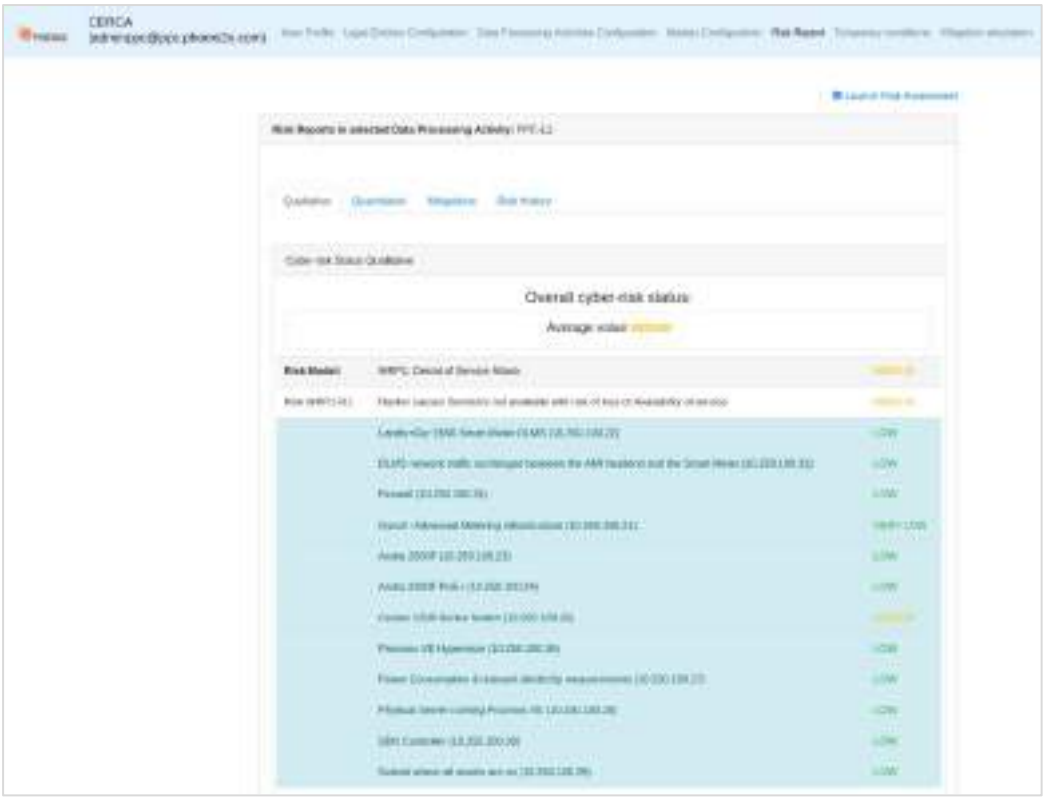

Figure 34. Initial qualitative risk assessment for CERCA.

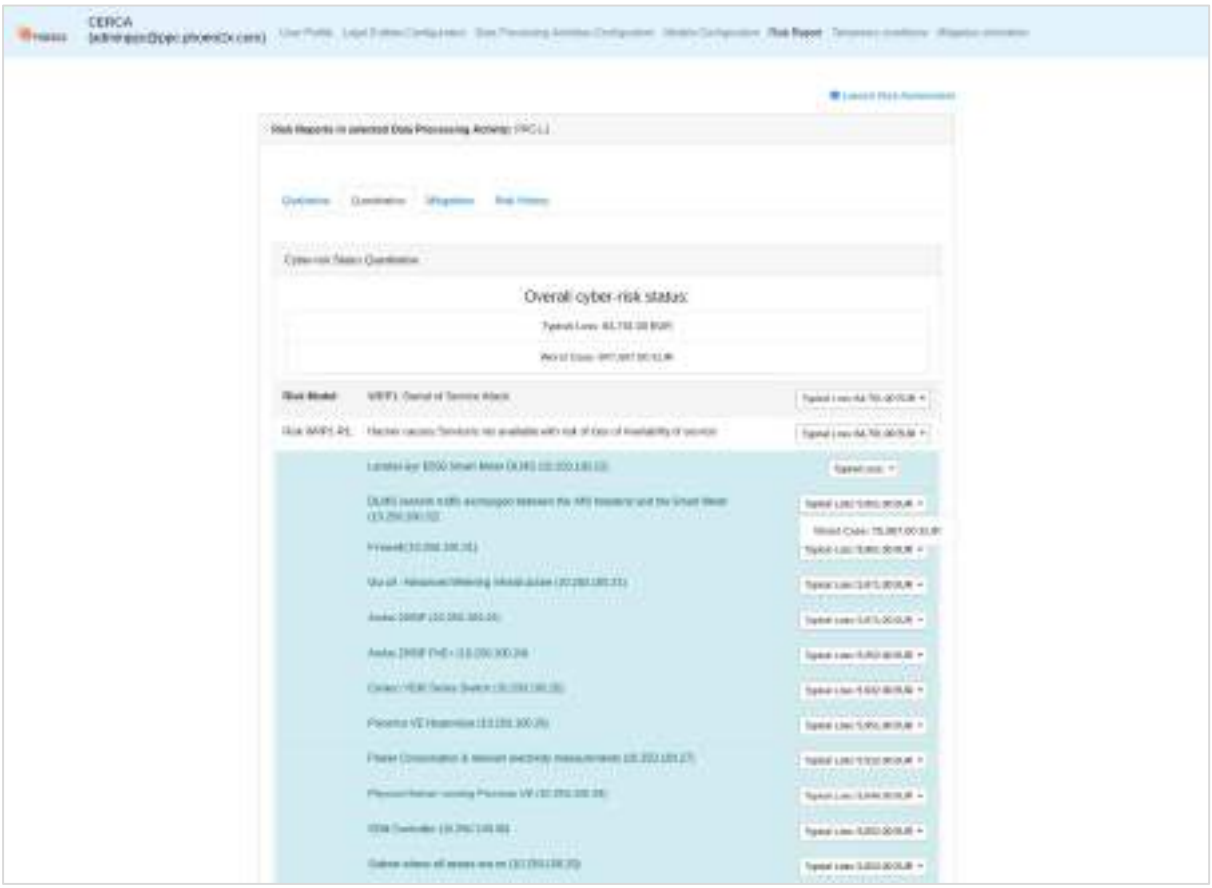

Figure 35. Initial quantitative risk assessment for CERCA.

After the PMEM message for “attack detected” is processed:

Figure 36. CERCA log showcasing receiving PMEM alarm.

The associated indicator for the relevant target is updated and a new risk assessment is computed:

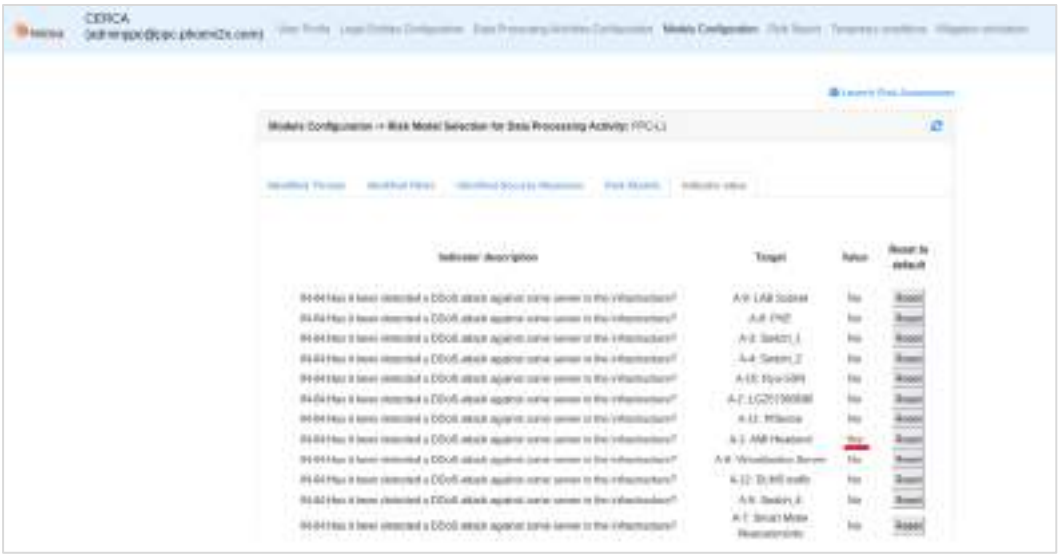

Figure 37. CERCA indicator for AMI Headend and Ddos attack updated.

An updated risk report is generated, that showcases the new risk :

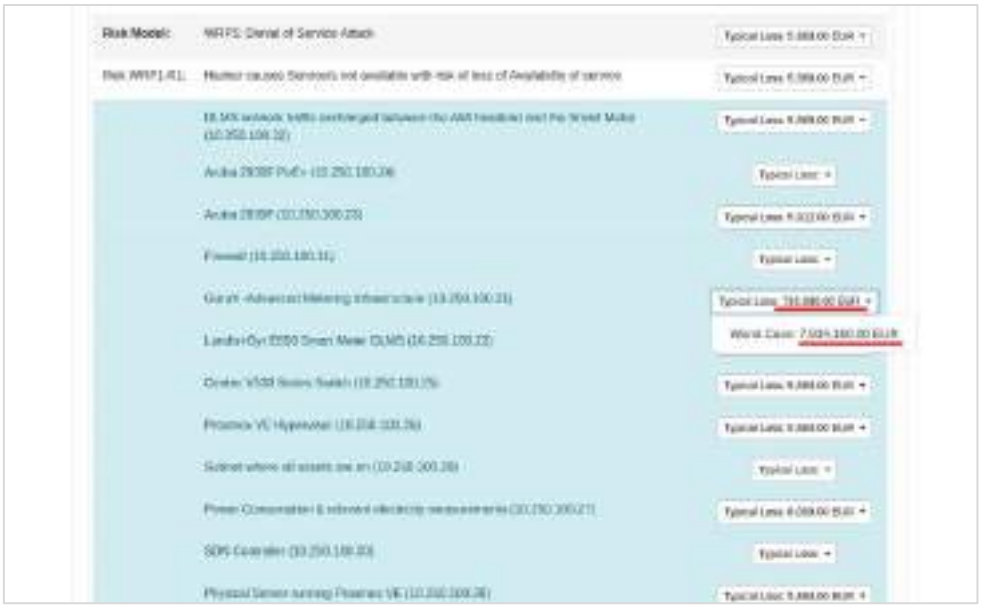

Figure 38. Updated quantitative risk assessment for CERCA.

2) SPA suite integration, CERCA receives from SPA suite:

   - Vulnerability report. These vulnerabilities are mapped to indicators and assets based on the available SBOMs. When the vulnerability is discovered, the risk assessment is launched with the relevant indicator activated:

Figure 39. SPA vulnerability message processed by CERCA.

- Detection of attack, including different phases of the attack.

3) ROAR integration, CERCA communicates with ROAR to:

- Launch the prioritized playbooks based on the priority strategy, in this case the playbook that most reduces the overall risk.
- Receive the "end of playbook" notification: CERCA is informed by SPA of the finalization of the playbook, meaning the mitigations contained in the playbook have been executed, and a new risk assessment is executed with a new risk report

A flowchart depicting the most relevant blocks of the new module architecture can be found in Figure 40.

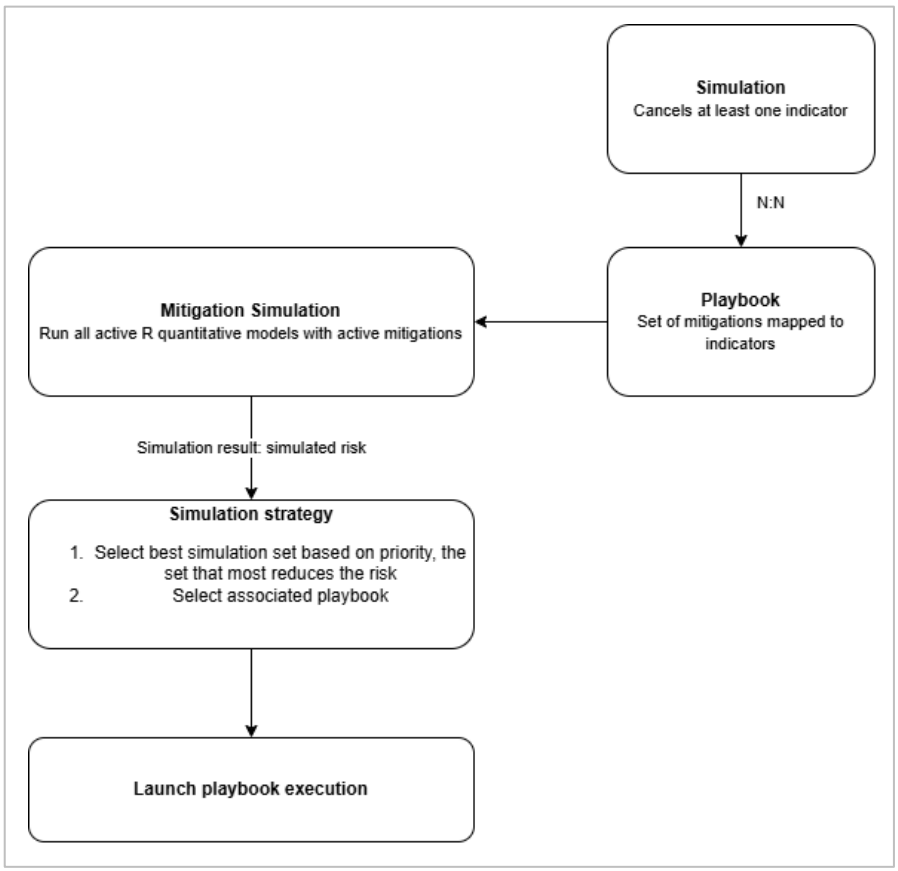

Figure 40. Playbook prioritization module.

These extensions establish CERCA as a key orchestrator within a broader cybersecurity ecosystem, enabling continuous assessment, dynamic prioritization, and rapid response.

6.4 Lessons Learnt and Recommendations for Future Work

Future improvements:

- Playbook prioritization: introduction of different simulation strategies based on:
    - Cost-driven strategy: by assigning a cost to the mitigations contained in a playbook, cost reduction strategies can be applied. Cost could mean monetary cost, energy cost, personnel cost, etc.
    - Time-driven strategy: assigning an execution time to the mitigations, prioritizing the quickest playbook.
- Enhanced simulation strategies: new simulation strategies can be introduced in the playbook prioritization module.

7 Conclusions

This deliverable presents the final status of the AI-assisted Situational Awareness, Prediction & Response enablers. It details the technical progress, functional enhancements, and integration efforts made since the previous version of the deliverable D3.1 on the four enablers.

The enablers have achieved their goals, achieving a significant level of maturity. The main enhancements of each enabler are the following:

- **User & Entity Behaviour Analytics (UEBA)** has improved its anomaly detection based on behaviour modelling.
- **The CTI Discovery & Analytics** component has expanded its asset visibility and supports intelligent asset classification.
- **The Threat Intelligence Integrator (TII)** now filters and prioritizes threat intelligence through heuristic scoring and SBOM-based enrichment.
- **The Threat Actor Context Ontology** has improved how cyber threat intelligence is modeled. Also an AI-based extraction system has been developed that allows to transform threat reports into a semantically enriched knowledge graph.
- **The Attack Prediction, Response Recommendation & Adaptation (PMEM)** component has improved in predictive maintenance, notably improving DDoS attack forecasting using system telemetry and machine learning.
- **CERCA** has evolved to become a proactive orchestrator of risk response, adding simulation-based playbook prioritization for dynamic and efficient mitigation.

Also, the enablers now are better integrated through the use of common communication protocols, APIs and orchestrators such as Kafka and ROAR. Overall, the toolset improves the automation of cyber defence methods, not only in the domains involved in the project use cases but also in the wider critical infrastructure market.

References

[1] K. Parsons, A. McCormac, M. Butavicius, M. Pattinson und C. Jerram, "Determining employeeawareness using the Human Aspects of Information Security Questionnaire (HAIS-Q)," Computer & Security, Bd. 42, pp. 165-176, 2014.

[2] M. Karjalainen, "Improving employees' information systems (IS)security behaviour: toward a metatheory of is security training and a new framework for understanding employees' is security behaviour," University of Oulu, Oulu, Finland, 2011.

[3] S. Egelman und E. Peer, "Scaling the security wall: Developing a security behaviour intention scale (SeBIS)," in CHI, Seoul, Republic of Korea, 18-23 April 2015.

[4] R. G. Netemeyer, W. O. Bearden und S. Sharma, Scaling Procedures: Issues and Applications, SAGE Publications Inc., 2003.

[5] C. Faklaris, L. D. und J. I. Hong, "A self-report measure of end-user security attitudes (SA-6)," in USENIX Symposium on Usable Privacy and Security (SOUPS), Santa Clara, CA, USA, August 11 - 13, 2019.

[6] P. Rajivan, P. Moriano, T. Kelley und L. J. Camp, "Factors in an end user security expertise instrument," Information & Computer Security, Bd. 25, Nr. 2, pp. 190-205, 2017.

[7] L. Hadlington, "Human factors in cybersecurity; examining the link between Internet addiction, impulsivity, attitudes towards cybersecurity, and risky cybersecurity behaviours," Heliyon, Bd. 3, Nr. 7, Jlu 2017.

[8] C. G. Coutlee, C. S. Politzer, R. H. Hoyle und S. A. Huettel, "An Abbreviated Impulsiveness Scale (ABIS) Constructed through Confirmatory Factor Analysis of the BIS-11," Arch Sci Psychol, Bd. 2, Nr. 1, p. 1–12., April 2014.

[9] R. A. Davis, G. L. Flett und A. Besser, "Validation of a new scale for measuring problematic internet use: implications for pre-employment screening," Cyberpsychol Behav, Bd. 4, Nr. 5, pp. 331-345, August 2002.

[10] G. Öğ˘ütçü, Ö. M. Testik und O. Chouseinoglou, "Analysis of personal information security behavior and awareness," Computer & Security, Bd. 56, pp. 83-93, 2016.

[11] H.-Y. Huang, S. Demetriou, R. Banerjee, G. S. Tuncay, C. A. Gunter und M. Bashir, "Smartphone Security Behavioral Scale: A New Psychometric Measurement for Smartphone Security," in https://arxiv.org/abs/2007.01721, 2020.

[12] AEuropean Commission, "Cyber security cOmpeteNCe fOr Research anD InnovAtion," 1 January 2019. [Online]. Available: https://cordis.europa.eu/project/id/830927.

[13] European Commission, "Strategies and Technologies for United and Resilient Critical Infrastructures and Vital Services in Pandemic-Stricken Europe," 1 October 2022. [Online]. Available: https://cordis.europa.eu/project/id/101073821.

[14] European Commission, "rEsilient and seLf-healed EleCTRical pOwer Nanogrid," 1 October 2023. [Online]. Available: https://cordis.europa.eu/project/id/101021936.

[15] Casey, T. (2007). Threat agent library helps identify information security risks. Intel White Paper, 2.

[16] Ngcobo, T. J., & Ghayoor, F. (2022). An overview of DLMS/COSEM and g3-plc for smart metering applications. *International Journal on Smart Sensing and Intelligent Systems*, *15*(1).

[17] Burenok, D. S., & Voevodin, V. A. (2023). On the information security controls database developed in accordance with ISO/IEC 27002: 2022. International Journal of Open Information Technologies, 11(9), 118-124.

[18] ISO/IEC 27002:2022, Information security, cybersecurity and privacy protection. Information security controls.

[19] PHOENI2X, 2024, Deliverable D3.1 AI-assisted Situational Awareness, Prediction & Response Enablers v1

[20] PHOENI2X, 2022, Part A of the Description of the Action (DoA), project number 101070586, A European cyber resilience framework with artificial intelligence -assisted orchestration & automation for business continuity, incident response & information exchange

[21] PHOENI2X, 2023, Deliverable D2.1 PHOENI2X Requirements & Architecture