HORIZON EUROPE PROGRAMME

HORIZON-CL3-2021-CS-01-01

PHOENIIX

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

D2.2: Baseline Enablers

Abstract: This deliverable shows the progress made in tasks T2.3 (Baseline Prevention, Detection & Response Toolset), T2.4 (Network & Compute Infrastructure Management & Orchestration), and T2.5 (PHOENI2X Security Assurance & Certification). The document describes the open-source tools that have been used as PHOENI2X baseline, how to deploy these tools, and where they are deployed. In addition, the document details the infrastructure where PHOENI2X's tools are deployed, and how monitoring, vulnerability testing, Cyber Threat Intelligence (CTI), and impact assessments are combined to assure and certify the security, privacy, and trust posture of the PHOENI2X platform.

Contractual Date of Delivery	30/09/2023
Actual Date of Delivery	30/09/2023
Deliverable Security Class	PU - Public
Editor	Alejandro A. Moreno Sancho (ATOS)
Contributors	UPAT, UPC, WS, AEGIS, NPS, COSM, FGC, WS, SANL, EUNL, NCSA
Quality Assurance	SEA, COSM

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: #9 SOCIAL ENGINEERING ACADEMY (SEA)

Reviewer #2: #4 COSMOTE KINITES TILEPIKOINONIES AE (COSM)

Revisions

Version	Date	By	Overview
1	12/06/2023	Editor	ToC released
2	27/06/2023	UPC	Input to Section 2.5
3	17/07/2023	SANL	Input to "Considered Open-Source Tools" for all categories (i.e., sections 2.X.1)
4	19/07/2023	AEGIS	Input to Section 2.7
5	19/07/2023	Editor	Input to Section 1 and overall improvements
6	21/07/2023	Editor	Input to sections 2.2, 2.3, and 2.8
7	27/07/2023	Editor	Input to sections 2.2.2, 2.3.2, and 2.8.2
8	27/07/2023	NCSA	Input to section 3.2.1
9	3/08/2023	WSE	Input to sections 2.4 and 2.4.2
10	18/08/2023	SANL	Input to section 4
11	22/08/2023	Editor	Input to section 2.2.3
12	23/08/2023	Editor	Input to section 2.3.3 and 2.8.3
13	25/08/2023	UPAT	Input to section 3.1
14	29/08/2023	AEGIS	Input to section 2.7.3
15	29/08/2023	COSM	Initial review comments
16	30/08/2023	Editor	Input to section 5
17	31/08/2023	Editor	Overall Improvements
18	31/08/2023	UPAT	Input to section 2.6
19	31/08/2023	PPC	Input to section 3.2
20	31/08/2023	NPS	Input to section 3.4
21	31/08/2023	UPC	Input to section 2.5.3
22	06/09/2023	WSE	Input to section 2.4.3

24	08/09/2023	NCSA	Input to section 2.3.3
25	12/09/2023	Editor	Modifications to section 3.1
26	14/09/2023	WSE	Input to section 3.3
27	18/09/2023	COSM	Final review of D2.2
28	26/09/2023	SEA	Final review of D2.2
29	27/09/2023	Editor	Overall Modifications considering the reviews

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CTI: Cyber Threat Intelligence

WP: Work Package

OES: Operators of Essential Services

RPs: Resiliency Playbooks

UPAT: University of Patras

CRCs: Critical Response Capabilities

UCs: Use Cases

ZAP: Zed Attack Proxy

WPS: Wifi Protected Setup

SET: Social Engineering Toolkit

CPE: Common Platform Enumeration

SIEM: Security Information & Event Management

USM: Unified Security Management

IDMEF: Intrusion Detection Message Exchange Format

VM: Virtual Machine

TTPs: Tactics, Techniques, and Procedures

MISP: Malware Information Sharing Platform

IOCs: Indicators of Compromise

CIRCL: Computer Incident Response Center of Luxemburg

TII: Threat Intelligence Integrator

EDR: Endpoint Detection and Response

EPP: Endpoint Protection Platform

HIDS: Host-based Intrusion Detection System

SIM: Security Incident Management

NDR: Network Detection and Response

NIDS: Network Intrusion Detection System

IP: Internet Protocol

NIPS: Network Intrusion Prevention System

GSoC: Google Summer of Code

SMB: Server Message Block

FTP: File Transfer Protocol

ICS: Industrial control Systems

IoT: Internet of Things

FVT: Forensics Visualization Toolkit

AMQP: Advanced Message Queuing Protocol

JMS: Java Message Service

MQTT: Message Queuing Telemetry Transport

NAT: Network Address Translation

VPN: Virtual Private Network

CMT: Connectivity Management Tool

KVM: Kernel-based Virtual Machine

UEBA: User and Entity Behaviour Analytics

ROAR: Resiliency Orchestration, Automation, and Response

CVE: Common Vulnerabilities and Exposures

NVD: National Vulnerability Database

CVSS: Common Vulnerability Scoring System

SDOs: STIX Domain Objects

ALTAI: Assessment List for Trustworthy AI

1 Introduction

1.1 Purpose of the Document

The purpose of the document is to describe the outcomes of the three WP2 tasks: Task 2.3 – Baseline Prevention, Detection & Response Toolset; Task 2.4 – Network & Compute Infrastructure Management & Orchestration; and Task 2.5 – PHOENI^{2}X Security Assurance & Certification.

In D2.2, the tools that serve as crucial enablers for the PHOENI2X project will be described in Section 2. These tools form the foundation upon which the rest of the framework will be built, playing a vital role in enabling intelligent assessments and detections based on the monitoring capabilities they provide. The deliverable will thoroughly analyse the different categories of tools, presenting some state of the art within each category, with a focus on open-source solutions. This analysis will provide valuable insights into established open-source solutions for each tool category that may be useful for organisations to monitor and maintain their security posture. Moreover, the deliverable will delve into the selection of the appropriate baseline tools from among the considered options. This selection process will involve an evaluation to identify the tools that are representative of their category, are widely adopted & and well-maintained, as well, but also ones that best align with the project's requirements and objectives. Furthermore, the deliverable will outline how each selected tool fits within the specific Use Case environments of the PHOENI2X project. In fact, any pre-existing tools already available in the use case environment will also influence this selection. This comprehensive analysis will highlight the relevance and applicability of each tool within the context of different scenarios and use cases.

Within this deliverable, the Network & Compute Infrastructure aspect of the PHOENI2X project will also be addressed in Section 3. As established in WP2, Task 2.4, the purpose of this infrastructure management and orchestration is twofold: facilitating the deployment and management of the PHOENI2X framework and supporting the adaptation actions of the OES (Operators of Essential Services) infrastructure encoded within RPs (Resilience Playbooks). To achieve this objective, the UPAT (University of Patras) will provide a comprehensive cloud platform that will serve as the foundation for deploying and managing the various components of the PHOENI2X architecture described in D2.1. In addition, the use case owners will also offer resources for full or partial deployment of PHOENI2X, so that the flexibility that the framework offers can be demonstrated. The number of resources needed to support the PHOENI2X framework and ensure its smooth operation will be discussed.

Finally, in Section 4 a detailed description of the Security Assurance & Certification solution shall be provided. Given the increase in the attack surface within the OES domain upon the introduction of new technologies and tools inherent in PHOENI2X, and the criticality of the application and its access to sensitive organizational processes, instilling trust in PHOENI2X Critical Response Capabilities (CRCs) assumes paramount importance. The Security Assurance & Certification solution is intricately linked to Task 2.3 within the project, as it heavily relies on the tools provided in that task. These tools serve as the foundation for the comprehensive analysis conducted by the Security Assurance & Certification solution, which primarily focuses on static testing, dynamic testing, CTI, and continuous runtime monitoring.

1.2 Overall Methodology

This section provides an overview of the methodology followed in the preparation of this deliverable, building upon the activities described in the previous deliverable, D2.1. The methodology employed in D2.1 places special emphasis on the separation between public content and RESTREINT UE/EU RESTRICTED content. D2.2 falls within the public part of the methodology, particularly related to the technical activities outlined in D2.1 “revolving around updating the proposal phase material”.

Furthermore, certain sections of this deliverable also fall under the second pillar, “Sector specific”, as they provide descriptions of the infrastructure that constitutes the testbeds for each of the use cases. Task 2.3 activities begin by assigning partners who have technical knowledge in the categories in which they have expertise. Subsequently, a sequence of steps was pursued, which can be summarized as follows:

1. The first step involved gathering information on each of the categories, understanding the essence of each category and defining the functionalities that each of the categories would bring to PHOENI2X.
2. Based on the gathered information, a list of open-source tools covering the expected functionalities in each category was compiled. Brief descriptions of the functionalities provided by each tool were also included. This step enabled a subsequent evaluation of the tools to select the most suitable ones.
3. The selection of tools involved an evaluation process to identify an indicative tool from each category that: (i) is an established (well-known, widely used) open-source tool that is well maintained (i.e., kept up-to-date by its developers and/or the community); (ii) is best aligned with the project's requirements and objectives in terms of functionality & features; (iii) is compatible with the requirements and overall setup of the use case environments (e.g., also considering any pre-existing tools already adopted by the use case owners). Overall, it should be clarified that the goal of this selection is not to identify the “best” tool in each category, but rather identify a characteristic, representative open-source tool for each category, while also showcasing the heterogeneity of tools that can be integrated with PHOENI2X, as well as the framework‘s adaptability: integrating this framework into an environment does not require an additional investment to install new baseline tools and discard the ones already deployed
4. In parallel with the selection of tools, the backend infrastructure of PHOENI2X was prepared. This process is described in section 3, “Network & Compute Infrastructure”, where the infrastructure for deploying and managing the various components of the PHOENI2X architecture is described. The selected tools from the baseline were deployed and configured, ready to be integrated with the rest of the PHOENI2X components.
5. Further, in parallel to the above, work was carried out to integrate and customise, as needed, a security assurance & certification solution within the overall framework that would facilitate the protection of the PHOENI2X assets themselves (as, inevitably, they expand the attack surface of the OES where they are deployed), as well as the OES’ assets, as needed.

1.3 Relationship with other PHOENI2X deliverables

Deliverable 2.1 provides a brief overview of the components that form the architecture of PHOENI2X, including the different categories that will constitute the baseline tools. In this deliverable, the description of the baseline tool is extended, detailing the tools used for each category and the selection process for each one. Additionally, the deliverable 5.1 will detail the management and orchestration aspects of the infrastructure, which will be used to represent the use cases described in D2.1, in D5.1 additional information will be provided regarding the testbed of every use case.

2 Baseline Prevention, Detection & Response Toolset

In this section, we present the open-source tool options for each of the categories considered in task 2.3. These categories of tools form the foundation for the development and proper functioning of the rest of the components within the PHOENI2X framework. Each tool or set of tools selected in this section represents the respective categories to which they belong. By doing so, we aim to emphasize that PHOENI2X is a framework that does not rely on a set of specific tools but can adapt to environments with a pre-installed toolset. This is why some of the tool choices depend directly on the availability of these tools in the use cases. We contemplate the following categories:

- Dynamic & Static Testing: Tools that perform tests on applications that are either actively running or on the code of the application.
- Security Information & Event Management (SIEM): Platforms that allow the collection and correlation of real-time information from various sources, including other categories mentioned later.
- Cyber Threat Intelligence (CTI): Providing information that can be correlated, aggregated, and offers a comprehensive view of the threat landscape.
- Endpoint Detection and Response (EDR): Detecting and responding to threats on endpoints. The information can be sent to the SIEM and other components of PHOENI2X for further actions.
- Network Detection and Response (NDR): Detecting and responding to threats on the network, similar to NDR, where information can be sent to the SIEM or other components.
- Deception Tools: Enabling the emulation of production systems to gather advanced information about attacks.
- Digital Investigation and Forensics: Tools that facilitate investigations into data obtained from the previous categories, allowing the identification of system compromises and digital artifacts.
- Message Bus: Enables the intercommunication of the baseline tools, and from the baseline tools to the rest of the components of PHOENI2X.

In the following subsections, we delve deeper into each category, outlining the functionalities performed, potential tools that fall within that category, and finally, the selection of tools that represent each respective category.

2.1 Dynamic & Static Testing

This category includes tools that can proactively identify and assess vulnerabilities, including static approaches (such as static vulnerability analysis) and dynamic ones (e.g., through penetration testing tools that actively probe assets).

The subsections below provide a set of popular, off-the-shelf tools considered of each of the two categories, which could be potential candidates for integration with PHOENI2X.

2.1.1 Considered Open-Source Tools

2.1.1.1 Dynamic Testing tools

Dynamic testing tools are popular and diverse, covering a number of use cases (e.g., local vs. remote assessments), involvement (automated, manual, etc.) and aggressiveness levels (e.g., vulnerability analysis vs. actually testing identified vulnerabilities for exploitation), as well as typically being tailored to specific types of assets (e.g., tools dedicated to assessing web applications or tools dedicated to

assessing SQL databases). Key categories and some indicative, popular tools within each category are presented in the subsections below.

2.1.1.1.1 Vulnerability analysis

Vulnerability analysis applications interact with assets (typically via network scanning to identify information about available systems, etc.) and, based on databases of known vulnerabilities, are able to produce reports highlighting important findings in terms of vulnerable assets, misconfigurations, etc. Examples include:

- **Open Vulnerability Assessment System – OpenVAS** [1]; a fully featured vulnerability scanner (& vulnerability management tool).
- **Burp Suite** [2]; an integrated platform and graphical tool for performing security testing of web applications, it supports the entire testing process, from initial mapping and analysis of an application's attack surface, through to finding and exploiting security vulnerabilities.
- **Grabber Vulnerability scanner** [3]; a black box web application vulnerability scanner that looks for SQL Injection, Blind SQL injection, XSS vulnerability and File include injection.
- **Wapiti WebApp vulnerability scanner** [4]; a tool that performs “black-box” scans of the web application by crawling the webpages of the deployed webapp, looking for scripts and forms where it can inject data.
- **Nikto Web server scanner** [5]; a vulnerability scanner that scans webservers for dangerous files/CGIs, outdated server software and other problems. It performs generic and server type specific checks.
- **Arachnni** [6]; a web application security scanner framework that can detect changes caused while travelling through the paths of a web application’s cyclomatic complexity and is able to adjust itself accordingly.

2.1.1.1.2 Penetration Testing

Penetration testing tools are typically a bit more active (or “aggressive”) in their scanning, compared to vulnerability analyser tools by, for instance, offering the option to try to exploit identified vulnerabilities. Since exploits are very application-specific, the tools themselves tend to be more specialised as well to specific types of assets. Some key categories & relevant tools are presented in the subsections below.

2.1.1.1.2.1 Web Application testing

Typical tools to assess the security of web applications include:

- **Web Application Attack and Audit Framework - W3af** [7]; a web application security scanner which helps developers and penetration testers identify and exploit vulnerabilities in their web applications.
- **SQLMap** [8]; a penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers.
- **Wfuzz** [9]; a web application security fuzzer tool and library for Python that provides a framework to automate web applications security assessments and can help secure web applications by finding and exploiting web application vulnerabilities.
- **OWASP Zed Attack Proxy (ZAP)** [10]; an integrated penetration testing tool for finding vulnerabilities in web applications which, when used as a proxy server, allows the user to manipulate all of the traffic that passes through it (including traffic using HTTPS).

2.1.1.1.2.2 Network Security Testing

For network-based security testing, tools that could be highlighted include:

- **Reaver Brute Force** [11] targeting WiFi Protected Setup (WPS) to recover WPA/WPA2 passphrases.
- **T50** [12]; a mixed packet injector for stress testing.
- **Aircrack-ng** [13]; a suite of tools to assess WiFi network security.
- **Btljuice** [14]; implements Man-in-the-Middle attacks on Bluetooth Low Energy devices.
- **Spooftooph** [15]; provides automated means for spoofing or cloning Bluetooth device information.
- **Nogotofail** [16], aimed to test TLS/SSL connections.
- **Metasploit Framework** [17]; accommodates a collection of tools and exploits to assess a large range of applications, operating systems, and platforms. It also provides the means to enable the automatization of the assessment.

2.1.1.1.2.3 Security Control Testing

Tools that are dedicated to testing specific security controls (e.g., antivirus) are also valuable in the cyber defenders'arsenal. Examples include:

- **Veil** [18]; a tool designed to generate Metasploit payloads that bypass common anti-virus solutions.
- **Hyperion** [19]; a runtime encrypter for 32-bit portable executables.
- **Phantom Evasion** [20]; provides polymorphic code and antivirus sandbox detection techniques to evade Antivirus software.

2.1.1.1.2.4 Social Engineering

Finally, a set of penetration testing tools has also emerged that focus on – what is often referred to as the weakest link in cybersecurity – the human factor. Examples of social engineering tools include:

- **Social Engineering Toolkit (SET)** [21]; a social engineering penetration testing framework.
- **KingPhisher** [22]; a Phishing campaign toolkit that can be used to run campaigns ranging from simple awareness training to more complicated scenarios in which user aware content is served for harvesting credentials.

2.1.1.2 Static Testing tools

Static testing tools are typically used "offline" and offer a more passive assessment of the inner workings of target applications & services. Such tools include:

- **SonarQube** [23] offering code Inspection; exposing vulnerabilities, used to measure the source code quality of a web app.
- **Cppcheck** [24] is a static analysis tool for C/C++ code. It provides unique code analysis to detect bugs and focuses on detecting undefined behaviour and dangerous coding constructs.
- **OWASP Dependency Check** [25], detects publicly disclosed vulnerabilities contained within a project's dependencies.
- **National Vulnerability Database dependency checker library (nvd-clojure)** [26], a utility that attempts to detect publicly disclosed vulnerabilities contained within project dependencies. It does this by determining if there is a Common Platform Enumeration (CPE) identifier for a given dependency. If found, it will generate a report linking to the associated CVE entries.

2.1.2 Final Selected Tools

To cover the baseline security testing (assessment) capabilities that PHOENI2X can build upon, two main capabilities need to be provided: dynamic & static testing. Selection of additional tools is, of course, welcome in an actual operational environment (e.g., a tool assesses web applications, another to assess databases), but beyond the scope of the work here.

Therefore, for the dynamic testing functionality required to showcase the PHOENI2X framework, the OpenVAS vulnerability scanner & tester, as even its "Community" edition (which features a friendly licensing scheme; namely GNU GPL-2) is a fully featured scanning & testing software that includes features such as (unauthenticated & authenticated) testing, various internet & industrial protocols, performance tuning for large-scale scans, etc. Further, OpenVAS is well maintained, with regular updates. OpenVAS integration, to automate its execution and the ingestion of its results, will happen through the SPA suite (see Sect. 4 below). This will allow for seamless integration of OpenVAS testing within the PHOENI2X workflows, as well as a useful contextualisation of the scanning results through their mapping to the assets modelled within SPA. Furthermore, while basic integration of OpenVAS within the PHOENI2X workflows will be achieved through the SPA Suite (which, out of the box support scheduled execution of OpenVAS assessments), if more complex workflows are needed, these can be automated through the ROAR component of PHOENI2X.

For static testing, the baseline functionality will be provided through the vulnerability scanning provided by the relevant component within the SPA Suite. This is similar to tools such as the nvd-clojure mentioned above, providing a mapping of assets to their CPEs and then identification of relevant vulnerabilities of said assets, always referring to an up-to-date view of the NVD vulnerability database. More details on this feature are provided in Section 4.

2.2 Security Information & Event Management

A Security Information and Event Management (SIEM) system is a comprehensive cybersecurity tool that plays a crucial role in monitoring and analysing an organization's security-related events and activities. Some of its capabilities are collection, aggregation, normalization, correlation and analysis of real-time and historical security events from heterogeneous data sources. SIEM platforms are designed to centralize and process vast amounts of data generated by various network devices, applications and systems, providing security professionals with valuable insights into potential security threats and vulnerabilities.

In the context of PHOENI2X, the adoption of a SIEM system holds significant importance. It provides a holistic view of the organization's security posture, allowing security teams to identify, prioritize, and respond to security incidents effectively. With SIEM's ability to gather data from various sources, it enriches the analysis process, aiding in the detection of complex and advanced threats that may otherwise remain unnoticed. The real-time capabilities of SIEM platforms allow for timely detection and response to security incidents, reducing the mean time to detect and respond to potential breaches, which is crucial in minimizing the impact of cyberattacks. The SIEM acts as a link between the events occurring in the infrastructure and the rest of the PHOENI2X components, providing those components with the correlated information to be further analysed.

2.2.1 Considered Open-Source Tools

This section proposes some of the existing open-source tools that could be used as SIEM in PHOENI²X.

- ***OSSIM*** [27] is the open-source version of AlienVault's Unified Security Management (USM). It is one of the more popular open-source SIEM platforms, combining event collection, processing, normalization, and correlation. It includes and relies upon an assortment of open-

source projects, such as the OSSEC, Snort, Suricata and OpenVAS tools mentioned in previous subsections.

- ***ELK stack*** [28] is the acronym for three open-source projects: Elasticsearch (search and analytics engine), Logstash (server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and "ships" it) and Kibana (visualisation layer). When combined, and through appropriate customisation, these three solutions can become a very scalable and versatile SIEM platform. In fact, many open-source and commercial SIEM offerings build upon or partly rely on the ELK stack, due to the powerful log processing, storage, and visualisation capabilities.
- ***Prelude OSS*** [29] is a SIEM framework unifying various established open-source tools. It allows for collecting, normalising, sorting, aggregating, correlating, visualising, and reporting on all security-related events, as aggregated from various types of logs (system logs, syslog, flat files, etc.) and enriched from additional sources (e.g., Snort, OSSEC). Prelude aggregates and stores all events in a unified format, the Intrusion Detection Message Exchange Format (IDMEF), as defined in IETF's RFC4765 [30], created for sharing information of interest to intrusion detection and response systems and to the management systems that may need to interact with them.
- ***SIEMonster*** [31], another open-source SIEM framework that relies on an assortment of individual open-source tools (e.g., ELK, Wazuh, as well as threat intelligence and ticketing solutions). Of particular interest is SIEMonster's support for cloud deployment, through Docker containers, virtual machines (VMs) or on bare metal servers (a feature usually only found on commercial SIEMs).
- ***Wazuh*** [32]***,*** is an open-source platform that can protect across on-premises, virtualized, containerized, and cloud-based environments. Wazuh has its own multi-platform agent that runs on the endpoints to be monitored. It provides prevention, detection, and response capabilities.
- ***OSSEC*** [33]***,*** *it* can be considered a Host-based Intrusion detection system as it performs integrity checking, windows registry monitoring, rootkit detection, time-based alerting, and active response. But it has a centralized architecture that allows analysis and correlation of logs from multiple devices and formats.

2.2.2 Final Selected Tools

In order to select a tool that represents the SIEM category, we first identify the characteristics sought in a SIEM. After distinguishing these characteristics, it is necessary to assign specific importance to each of them, as some will hold greater significance than others. Subsequently, comments are provided regarding these characteristics for each of the proposed tools. The criteria or characteristics considered for this category of baseline tools are as follows:

- **Real-time analysis**. In order to prevent threats, real-time analysis is crucial. This criterion relates with scalability or with the number of events per second that the SIEM can handle. Considering that we have infrastructures with a large number of devices, the importance of this criteria is high.
- **Integration with sources:** For the SIEM to detect as many threats as possible, the information must come from different sources, such as NIDS (Snort, Suricata), HIDS (OSSEC), logs (Syslog)... The number of sources or the ease of integrating those sources is really valuable, that is why this criterion has high importance.
- **Log file Management:** Log management is the ability to deal with large volumes of generated logs, although this is crucial, the weight of this criterion is not so high as many of the tools provide log management with external programs, for example, Syslog is commonly used with

OSSIM, or ELK which is used by Wazuh. As we can complement the SIEM with external tools if there are no log file management capabilities, the weight of this criteria is medium.

- **Deployment possibilities:** Refers to the possible ways the SIEM can be deployed (Virtual machines, on-premises, Dockerized, Kubernetes, etc). The architecture is not considered to limit the way we deploy the SIEM at present. Therefore, this criterion could be considered but the effect in the decision is going to be low.
- **Documentation and community:** Lack of documentation and community can make deployment and development slow and tedious. In addition, the project being in use with an active community indicates a degree of maturity of the project, which is why the importance of this criteria is high.
- **Ease of installation:** Indicates how easy it is to install the SIEM. The faster we can install the SIEM, the faster we can keep working on other tasks such as configuration and creation of the necessary rules. However, the importance of the criteria is low as it may be costly to install the tool but later have more documentation and be easier to configure.

Considering these criteria, Wazuh has been identified to be a representative tool of the SIEM category. Moreover, Wazuh is among the tools available in use case 3, which is compelling reason to choose it, as it allows us to demonstrate the adaptability of PHOENI2X to already existing tools in the infrastructure. Although this is not necessary for the choice, we have also identified some capabilities that make Wazuh a good selection for use cases other than those considered in PHOENI2X. Wazuh boasts multi-platform agents that facilitate data collection from endpoints; it serves as a fork of OSSEC, expanding upon its rule set. Moreover, Wazuh can be deployed in a multi-mode cluster, ensuring significant scalability; a capability not offered by other SIEMs such as OSSIM, where even others limit the number of endpoints and require a paid version like SIEMonster. Wazuh's real-time analysis capabilities surpass those of other tools like PreludeOSS. The documentation of PreludeOSS acknowledges these limitations and recommends using it primarily for experimental purposes rather than production scenarios. Wazuh boasts an extensive array of integrations with various sources. Its deployment is straightforward and can be achieved through various methods, ranging from ready-to-use machines to Docker and Kubernetes containers. The documentation provided is comprehensive and intuitive, and the community backing Wazuh is substantial compared to other tools like OSSIM and SIEMonster. Lastly, Wazuh is seamlessly integrated with the ELK stack, granting it excellent data search and visualization capabilities. In contrast, other tools like OSSEC lack this integration or log management capabilities, requiring users to undertake the task themselves.

2.3 Cyber Threat Intelligence

Cyber Threat Intelligence (CTI) refers to the knowledge and information gathered from various sources about potential and existing cybersecurity threats, their actors, tactics, techniques and procedures (TTPs). It involves the collection, analysis and dissemination of intelligence to enhance an organization's ability to detect, prevent and respond to cyberattacks effectively. CTI serves as a critical resource, providing with a comprehensive understanding of the threat landscape, enabling preventive measures. CTI facilitates proactive threat hunting, allowing security analysts to actively search for and identify hidden threats within their networks.

A CTI platform is a critical component in PHOENI²X. It serves as a centralized system that collects, processes, and analyses a wide range of cyber threat data from various sources. The platform aggregates this diverse data and performs advanced analytics, enabling security professionals to gain comprehensive insights into the evolving threat landscape. CTI platforms offer several essential functionalities to strengthen an organization's cybersecurity posture. Through data analysis and correlation, they can identify patterns and trends indicative of potential cyber threats. Moreover, CTI

platforms facilitate information sharing and collaboration among organizations, helping them stay updated on the latest threat intelligence and improving collective defence against cyber threats.

2.3.1 Considered Open-Source Tools

There is currently a large number of open-source CTI platforms, the most popular ones are mentioned below:

- ***Malware Information Sharing Platform - MISP*** [34]**,** is an open-source platform for sharing, storing and correlating Indicators of Compromise (IoCs) of targeted attacks. MISP is used today in multiple organisations. MISP allows you to collaborate on malware and use the IOCs to support the detection and prevention of attacks. The project is developed by a team of developers from CIRCL, Belgian Defence, NATO and NCIRC and funded by the European Union (through the Connecting Europe Facility) and the Computer Incident Response Center Luxembourg.
- **OpenCTI** [35], an open-source platform meant for processing and sharing knowledge for cyber threat intelligence purposes. Initially developed by ANSSI and CERT-EU, but now made available to the CTI community. It allows CTI analysts to structure, store, organize, visualize and share their knowledge.
- ***MineMeld*** [36], an open source, community-supported framework that can support the consumption and sharing of threat intelligence for SOC operations. Of particular utility is MineMeld's capability to automate daily SOC tasks and to manipulate lists of indicators (transform/aggregate) for consumption by third party enforcement infrastructures, allowing the automated enforcement certain actions based on the aggregated threat intelligence (e.g., create specific URL filtering profiles, populate URL allow and block lists, create block policies based on indicators of compromise detected in the wild). These features are tailored to SOCs allowing, for example, aggregation of CERT and ISAC Threat Intelligence feeds, preparing these inputs as needed (e.g., removing duplicates, expiring entries, and consolidating attack directions and confidence levels), before forwarding them to other tools (e.g., for enforcement).
- ***SOCRadar Community Edition*** [37], a unified threat intelligence platform that offers asset-focused risk management features alongside with proactive protection through monitoring threat landscape elements (External-facing Asset Discovery, SSL Certificates Security Grading, Phishing Domain Detection, IP Reputation / Blacklist Monitoring, Vulnerability Data Feeds).
- ***IntelOwl*** [38], an open-source OSINT solution providing a scalable way to retrieve threat intelligence data about a specific file or observable (e.g., IP, URL, domain, or hash) from a single API at scale. It also features a dashboard allowing visualizations of queries and their analysis data.
- ***Cortex*** [39], another open-source observables (e.g., IP and email addresses, URLs, domain names, files, or hashes) analyser, offering a web UI, powerful automation mechanisms, an assortment of integrations with external tools and services, as well as tight integration with MISP (Cortex can invoke MISP modules and MISP can invoke Cortex analysers).
- ***Machinae Security Intelligence Collector*** [40], a tool for collecting intelligence from public sites and feeds about various security-related observables, including IP addresses, domain names, URLs, email addresses, file hashes and SSL fingerprints.
- ***Yeti*** [41], allows you to organize observables, indicators of compromise, TTPs, and knowledge on threats in a single repository. It can resolve domains, geolocate IPs and much more in order to enrich observables. Yeti obtains information from a wide variety of sources (e.g., MISP instances, JSON feeds, XML feeds…). The platform provides a graphic interface and a web API that can be used to automate queries.

2.3.2 Final Selected Tools

Among the plethora of open-source CTI platforms available, MISP stands out as the selected tool for the PHOENI2X. MISP is a powerful platform designed for sharing, storing, and correlating IOCs related to targeted attacks. Its ability to support multiple organizations in collaborating on malware data and IOCs is a testament to its effectiveness in enhancing incident detection and prevention.

One of the primary reasons for selecting MISP over other platforms, particularly OpenCTI, is the presence of existing components within PHOENI2X that are built on top of MISP. For instance, the Threat Intelligence Integrator (TII) leverages MISP's capabilities. This existing integration significantly reduces development efforts and speeds up the project's overall implementation. Furthermore, MISP's feature set aligns closely with the project's needs, enabling it to efficiently structure, store, organize, visualize, and share CTI knowledge. Its focus on processing and sharing knowledge for cyber threat intelligence purposes is well-suited to PHOENI2X's goals of enhancing the understanding and detection of cybersecurity threats. The active community support surrounding MISP ensures continuous development, bug fixes, and feature enhancements, contributing to the platform's reliability and relevance. The ongoing development by a team of experts ensures that MISP stays up to date with the evolving threat landscape, providing timely and effective threat intelligence. Finally, its dashboard and visualization features offer valuable insights into queries and analysis data, allowing security analysts to gain a comprehensive understanding of potential threats.

2.4 Endpoint Detection and Response

An Endpoint Detection and Response (EDR) system is an identification and protection system for the organization's IT equipment and infrastructure.

An EDR is constantly collecting information from endpoint devices connected to an organization’s network such as desktops, servers, laptops or tablets. This information is normally contextual activity happening between the endpoint and network such as logins, session types, IP address location, accessibility data, etc.

An EDR processes the collected information and searches for any irregularities in the events taking place, such as an abnormal session type or a login from a non-recognized IP address. In case of incident detection, it will automatically initiate response functions to contain the incident in that endpoint, investigate the event further, notify the relevant personnel, and possibly even provide insights on how to remediate the incident.

Although it shares tasks with traditional antivirus, also known as EPP (Endpoint Protection Platform), such as the detection, identification and prevention of the effects of malware and exploits, EDR can also detect advanced threats. These may include polymorphic malware, 0-day vulnerabilities, social engineering attacks, persistent threats or APTs, compromised accounts, etc.

As per Anton Chuvakin, in his 2013 article for the company Gartner titled “On Endpoint sensing” [42], an EDR should primarily detect security incidents at endpoints by:

- Collecting endpoint data such as running processes (including those with no file on disk), handles, DLLs, network connections, open ports, loaded drivers, Windows services, users, select files, registry entries, routing and ARP table entries, browser history, etc.
- Centralizing the data (by either periodic polling or near-real-time collection or distributed data access) and make it available and accessible for fast, indexed searching.
- Post-processing (analysing) the data to identify anomalies such as rare processes, unusual connections, and other higher-level patterns by using baselining or other means.

- Provide an interactive data exploration interface that allows the Security Manager to explore the data and hunt for indicators.

Furthermore, an EDR is also tasked to contain the incident at the endpoint. Typically, the agent has the ability to block network connections, stop running processes, and delete malicious files, among other actions. As an additional control, EDR can prevent further encryption over the network by isolating the endpoint.

Subsequently, an EDR should help investigate the incident that took place. This process is supported by Threat Intelligence databases, which compare the available incident data with Malware databases, IP reputation databases, etc. Furthermore, investigating the reasons on why the network could not stop the incident is very valuable, in order to reveal issues and prevent future exploitation via the same threat vector.

In the end, an EDR should provide remediation guidance on how to eliminate the threat, if the conclusion of the investigation is that the incident was real and the endpoint is compromised. An EDR providing visibility over the entire timeline of a file is crucial, as the remediation of the endpoint may also imply remediation of multiple parts of the network.

2.4.1 Considered Open-Source Tools

The most relevant EDR solutions available as open-source are listed below:

- **Osquery** [43] is an open-source endpoint security tool that allows real-time visibility and monitoring of operating systems in a cross-platform environment. It works by exposing the OS as a SQL database, enabling users to execute SQL queries to gather information about the system, such as processes, logged-in users, hardware details, file integrity, and more in real time. Osquery doesn't provide logging capabilities and has reduced detection capabilities. Regarding integrations with other tools, this product can be integrated with various security tools and SIEM systems, but it may be more suitable for smaller environments due to its focus on real time monitoring and querying.
- **OSSEC** [33] is a widely-used open-source intrusion detection and prevention system. It serves as a host-based intrusion detection system (HIDS) and provides real-time log analysis, real-time windows registry monitoring, file integrity monitoring, and active response capabilities. It performs log analysis, integrity checking, Windows registry monitoring, rootkit detection, real-time alerting, and active response. It runs on most operating systems. It mixes together all the aspects of HIDS, log monitoring, and Security Incident Management (SIM)/SIEM together in a unique platform for monitoring and controlling your systems.
- **Wazuh** [32] is an open-source security monitoring platform that integrates OSSEC HIDS with other capabilities, such as centralized management, analysis, and threat hunting. It offers a more comprehensive approach to security by combining endpoint detection with centralized monitoring and analysis. Additionally, Wazuh provides real-time alerts and dashboards to help security teams identify and respond to security threats across the infrastructure.
- **OpenEDR** [44], powered by Xcitium, stands as an open-source endpoint detection and response platform, offering analytic detection combined with Mitre ATT&CK visibility. Its primary purpose lies in correlating events and conducting real-time root cause analysis of adversarial cyber threat activities and behaviours. OpenEDR gathers endpoint device activity data and employs sophisticated analysis to identify malicious activities. This powerful platform facilitates comprehensive monitoring, visualizes essential endpoint security information, and conducts thorough malware analysis. Open EDR has recently been released to the market, and despite being an open-source tool, it requires registration on the Xcitium platform.

2.4.2 Final Selected Tools

After considering the previously mentioned tools and based on the specific needs of the project, available infrastructure, and the organization's framework, it has been decided to use Wazuh for PHOENI2X platform.

- Real-time Threat Detection: Wazuh provides real-time monitoring and analysis of security events and logs, enabling swift detection of potential threats and suspicious activities.
- Highly Customizable: Wazuh provides an open API that allows easy integration with other security tools, SIEM solutions, and third-party applications.
- Agent-based: Wazuh can scale to accommodate the monitoring needs of the infrastructure.
- User-Friendly Dashboard: Wazuh offers a user-friendly web-based interface that provides a comprehensive view of the security events, alerts and reports for efficient monitoring and management.

In summary, Wazuh provides organizations a customizable, scalable, and cost-effective platform that delivers real-time threat detection, comprehensive security monitoring, and the flexibility to integrate with other tools for a robust defence against cyber threats.

2.5 Network Detection and Response

A network detection and response (NDR) mechanism is a cybersecurity solution designed to monitor and detect suspicious activity and threats within a computer network through machine learning and data analysis. It combines several complementary techniques and technologies to automatically gain real-time visibility of the network, detect, analyse and enable rapid response/act and mitigate sophisticated cyber threats. In addition, NDR tools allow to generate attack alerts, advanced threat detection, forensic security analysis, encrypted traffic analysis.

PHOENI2X framework will provide baseline prevention, detection, and response capabilities that through a baseline toolset that will use open-source tools covering NDR. NDR tools will focus on monitoring network traffic for malicious actors and suspicious behaviour, while also supporting reaction and response to detected cyber threats which are present in the network.

2.5.1 Considered Open-Source Tools

This section reviews the existent open-source tools for network detection and response, which could prove valuable for developed in the PHOENI2X project. We have classified them in the following four categories: response, detection, testing and packages.

2.5.1.1 Response

- ***pfSense*** [45]***:*** a firewall or router software distribution based on FreeBSD. It is an open-source software which can be installed into a computer or any kind of virtual machine to make a dedicated network for a firewall or router. pfSense is a stateful firewall software, then it can automatically allow replying to the traffic which is stored in a state table. Though its primary use is a software of router or firewall, sometimes it can be used to set up a Wi-Fi access point, DHCP server, Wi-Fi access point, DNS server and VPN server. It also allows connecting multiple internal networks. pfSense is a free, open-source, stateful firewall, highly available software. However, it is sometimes complex to configure and it requires powerful hardware.
- ***Untangle*** [46]***:*** an open-source network gateway which provides some basic firewall services based on Debian 6.0 (squeeze) and 2.6.32 kernels. It allows detecting malware and spam blocking, phishing and intrusion protection, as well as web filtering. The untangle firewall can

be deployed in different formats such as: (i) run in the Intel-based server at the gateway to the network; (ii) run in a virtual machine, and; (iii) run in the public cloud to ensure the performance network connectivity, reliability, safety and protection for the applications, data and users. It has zero touch provisioning, some simplified dashboard, bandwidth management rights, and more deep perception in real time.

2.5.1.2 Detection

- ***Snort*** [47]***:*** an open-source Network Intrusion Detection System (NIDS) and intrusion prevention system (IPS). Nowadays, it is one of the most used IPS. It uses a series of rules which will be helpful to define some malicious network activity as well as to apply some of those rules to find some packets that totally match opposed to them, and then it will show some alert for users. It can be used to perform traffic analysis in real time, packet logging on Internet Protocol (IP) networks, protocol analysis, content searching and matching. Snort sensor can be deployed on the router as a virtual container service. It has some advantages since it is an open-source free network, so it is highly customizable and can be used for recording some unwanted presence of traffic as a passive trap, to show the records of data packets in a human-readable form, for both home DSL connection and a corporate website to supervise and monitor.

- ***Suricata*** [48]***:*** an open-source based Intrusion Detection System (IDS) and intrusion prevention system (IPS). It allows setting rules and signatures to detect and prevent the threads, and it takes some actions on the events and attempts for blocking the traffic. It enables deep packet inspection and security monitoring. On a gateway host network, Suricata can be deployed for scanning any kind of incoming and outgoing network traffic from any other systems. Alternatively, it can also be run locally. It is an open-source, flexible, a lightweight and powerful IPS. It has low cost and high performance. Furthermore, it allows the users to create a horizontal scale on one device by attaching more processing threads packets.

2.5.1.3 Evidence collection

- ***Zeek*** [49]***:*** a passive network traffic analyzer, mostly used as a network traffic monitor to support the collection of the evidence for the identification of normal and suspicious activity. Zeek can be installed on a single machine or can be configured to work with multiple clusters. At very high level, its architecture contains two components, event engine and policy Script Interpreter. The main function of the event engine is to get the input from the network interface and extract high level events. The second module, the script interpreter, supports different set of security policies written in Zeek language to get notification and logs. These scripts can be used to create different real time alert messages. Zeek provides transaction data and extracted content data, summarizing all the logs of protocols. Zeek plays an important role in matching and hunting task during analysis of security investigations.

2.5.1.4 Packets

- ***Arkime*** [50]***:*** an open-source tool which supports searching, analysis and packets capture for further analysis by different tools. It is designed to work with multiple clustered systems. It has three main components, capture, viewer and elastic search. The capture is used to monitor a specific network port and write the output in PCAP format and send the metadata information to the elastic module. Viewer is a node.js application which handles the web interface and transfers the PCAP files. The elastic search module supports multiple views of the information of the collected data. All information is stored locally and can be accessed using the HTTPS web server proxy of the Arkime. This tool is not a replacement of the IDS rather supports more visibility of the data along with exposed API which allows data to be downloaded in the Json formatted and PCAP format.

- ***Tstat*** [51]***:*** a passive tool which is able to reconstruct the internet traffic at the IP, UPD and network level. Tstat can work in a total passive way without altering the network packets.

Tstat generates three different types of measurement collections, Log files for storing the flow level measurements, histograms storing the distribution during a time interval, and RRD databases for storing the histograms. Tstat can be used to work in the real time analysis of the traffic. For live mode, it should be configured with the edge router to get the Newtork flows in the passive modes. More than 80 histograms can be viewed in the previous version of Tstat gui. One of the biggest advantages of the Tstat is that it supports lots of input file formats collected by different tools for the analysis purpose. Tstat is tested with linux systems, their primary use is to support the command line tool.

- ***Netflow*** [52]***:*** a professionally developed tool by Cisco to provide different set of key services IP application, network traffic accounting, network usage and other capabilities supported by the tool. It was developed for cisco routers to collect the network traffic entering and leaving the network. NetFlow is basically a feature in Cisco routers to collect different information about the network. Usually flow monitoring using the NetFlow consists of three main components, Flow exporter, flow collector and analysis applications. These logs and flow data can be used for the traffic profiling and intrusion detection. Netflow supports a wide range of network protocols and thus is quite useful for a wide series of commercial routers.
- ***TCPDUMP*** [53]***:*** an open-source tool to capture the Raw traffic patterns from the networks. It is a packet analyser which can allow users to capture both the incoming and outgoing traffic from a specific network interface. These log files contain different traffic patterns of HTTPS, HTTP, SMTP, POP3, IMAP, SSH, and FTP protocol. It's quite simple and easy to configure with the network. TCPDUMP is easy to install and supports a wide range of network traffic protocols. It is usually used to capture the raw network traces and store them for future analysis for intrusion detection systems. The traffic captured is stored in the standard format of PCAP files for future analysis. It can Decrypt IPSEC traffic by providing an encryption key.
- ***CICFlowmeter*** [54]***:*** a network traffic flow generator and analyser. It's an open-source software compatibile with Linux and Windows systems. The most relevant feature of CICFLowMeter is that it can simulate different types of network traffic to aid in various testing scenarios and performance evaluations. This feature allows users to assess the impact of different network conditions on their systems, applications, or infrastructure.
- ***Wireshark*** [55]***:*** a widely utilized network protocol analyser renowned for its effectiveness in network troubleshooting, analysis, and security auditing. Notably, it is an open-source tool that is compatible with multiple operating systems. Wireshark exhibits the capability to monitor and analyse diverse forms of network traffic, encompassing Ethernet, Wi-Fi, TCP/IP, and UDP. The software boasts several features, including an intuitive graphical user interface, robust filtering and search capabilities, the ability to reconstruct network streams, support for an extensive range of protocols, and the option to save captured data for subsequent analysis. These features contribute to Wireshark's effectiveness and usability in various network-related tasks.

2.5.2 Final Selected Tools

From the list of tools presented in previous section, and considering the requirements of the PHOENI2X project, as well as the protocols used in the use cases, especially in the Energy and Transport scenarios, we have chosen the following tools for the NDR: pfSense, Zeek, CICflowMEter and Wireshark.

These tools complement each other very effectively and support the LoRaWAN and the DLMS/COSEM protocols. The Wireshark will be responsible for analysing the network in a highly precise manner, combined with Zeek, capable of detecting anomalies in network traffic. Once the anomalies or attacks are detected, pfSense will take care of halting them. Finally, to test its operation in different scenarios, we have the CICFlowMeter traffic analyser that will extract the features which will be used by the Attack Prediction Response Recommendation and Adaptation for training the AI models.

2.6 Deception Tools

Deception tools is a category of cybersecurity solutions that aims at detecting threats as early as possible. This technology deploys realistic decoys (e.g., domains, databases, directories, servers, apps, files, etc.) in a network alongside real assets to act as lures for potential attackers. Their main goal lies in 1) hiding the actual critical assets and 2) allowing security personnel to obtain information about cyber-attack Tactics, Techniques and Procedures (TTPs) etc., in order to protect the actual production system more effectively. In other words, deception tools do not directly protect a critical system, but instead they assist its security team to identify system vulnerabilities and weaknesses, devise countermeasures, as well as remediate exploitable breaches.

The most vital characteristic of the deception tools is the resemblance to the actual system, so that cyber-attackers are led in the belief that they are targeting the actual system. To achieve such resemblance, all the data of the actual system must be initially collected, analysed and programmed, for a deception tool to behave similarly (e.g., send and respond with the same messages). Apart from that, these solutions also collect all the logs and traces of a cyber-attack as evidence for analysis from the security team. The most widely used deception tools are honeypots. These are solutions that are mostly based on (are programmed to resemble) traditional IT systems. Within PHOENI2X, more sophisticated honeypots, which focus on industrial protocols and systems, will be used and enhanced. These honeypots will be deployed in OES infrastructures and will be able interpret the data and commands from industrial protocols, as well as transmit commands as they constituted the actual system. Another scenario that shall be tested within PHOENI2X is the exposure of a public interface of the honeypot, to attract cyber-threat actors for launching their attacks. This scenario has as a prerequisite that the honeypot has no interaction with the actual system nor the OES infrastructure, to avoid a possible lateral movement from the cyber-threat actors.

2.6.1 Considered Open-Source Tools

2.6.1.1 Stand-Alone tools

- ***Cowrie Honeypot*** [56], *"Cowrie is a medium to high interaction SSH and Telnet honeypot designed to log brute force attacks and the shell interaction performed by the attacker. In medium interaction mode (shell) it emulates a UNIX system in Python, in high interaction mode (proxy) it functions as an SSH and telnet proxy to observe attacker behaviour to another system."* It can be a valuable detector of attack attempts and can analyse the behaviour of the attackers targeting a specific organisation. It is also dockerised and can be easily deployed to a large number of hosts.
- ***Kippo*** [57] is a medium interaction SSH honeypot that was designed to log brute-force attacks with the capability of logging the entire shell interaction performed by the attacker.
- ***Dionaea*** [58] is a low interaction malware-capturing honeypot initially developed under The Honeynet Project's 2009 Google Summer of Code (GSoC). Dionaea's main goal is to trap malware exploiting vulnerabilities exposed by services offered over a network, and ultimately obtain a copy of the malware. It supports IPv6 and TLS and uses Python as scripting language to simulate many popular services such as Server Message Block **(**SMB), the File Transfer Protocol (FTP), MySQL and others, and libemu to detect shellcodes.
- ***SNARE*** [59] and ***TANNER*** [60]: SNARE is a web application honeypot and is the successor of Glastopf [61], inheriting most of its features. It also has the ability to convert existing Web pages into attack surfaces using TANNER. Every detected event is sent from SNARE to TANNER and TANNER decides dynamically how SNARE should respond to the client, improving its camouflage abilities. SNARE when fingerprinted by attackers shows that it is a Nginx Web application server.

- ***Conpot*** [62] is a low interaction Industrial Control Systems (ICS) honeypot. It can emulate various vulnerable services which are common in an Internet of Things (Iot) setting, such as Modbus, SNMP, HTTP, and others. It offers logging capabilities and can report events made by threat actors.
- ***Thug*** [63] is a honeyclient tool written in Python working as a low-interaction client. It aims at mimicking the behaviour of a web browser in order to detect and emulate malicious contents. Client-side honeyclients allow the study of client-side attacks.
- ***ElasticPot*** [64] is an Elasticsearch honeypot simulating a vulnerable Elasticsearch server that is publicly available.
- ***CanaryTokens*** [65] allow the implantation of tracking tokens, similar to the transparent images which track when someone opens an email, but for different resources in a production system. These tokens can be embedded in HTTP, DNS, Web image, PDF, Word, Excel, Windows Directory, in executable files, QR codes, SVN, AWS API Keys, Fast Redirect, Slow Redirect, Slack API and SQL server.

2.6.1.2 Honeypots collection

- ***T-Pot 20.06*** [66] is a highly dockerised solution that includes more than fifteen honeypots, mostly low-interaction and specific honeytokens ready to be deployed. Moreover, it includes specific tools for monitoring, visualising, and managing the deployed honeypot detectors.
- ***Modern Honey Network*** [67] is a centralised server for data collection and management of different honeypots. It allows the deployment of sensors and the collection of data to a centralised point viewable by a visual interface.

2.6.2 Final Selected Tools

With respect to the chosen deception tools (UPAT), we have opted for Conpot, as our honeypot solution due to its specialized capacity in emulating Industrial Control Systems (ICS). Its emulation of vulnerable services for ICS protocols aligns with the PHOENI2X strategy for safeguarding OES infrastructures. Conpot's low-interaction approach captures malicious activities while avoiding operational risks. Its robust logging capabilities offer insights into attack patterns and vulnerabilities for proactive mitigation. Integration into our infrastructure is seamless, as it works well with common ICS protocols. Overall, Conpot focuses on low-interaction design, extensive logging, and easy integration making it an optimal choice for enhancing the cyber-resilience of OES infrastructures. Specifically, this is accomplished by analyzing sophisticated attacks against the deployed honeypots and developing detection mechanisms for them.

2.7 Digital Investigations and Forensics

This category includes tools that offer investigative capabilities to find signs of malicious activity (e.g., through memory and file analysis) and to allow the extraction of digital artefacts (e.g., binary extraction, dumps of configuration data).

There is a number of available tools and services in the market that could be used as stand-alone or in a combination, in the platform:

- **Traditional digital forensics tools**: Many stand-alone software solutions and specialized hardware devices are available for specific forensic tasks, such as data recovery, analysis, and reporting. However, these tools often require manual operation and might not be well-integrated, leading to longer investigation times and inefficiencies.
- **Open-source tools and frameworks**: A variety of open-source tools and frameworks to conduct digital forensics investigations could be used. While these solutions can be cost-

effective, they often lack the comprehensive features, support, and regular updates provided by commercial solutions.

- **Managed security services**: Outsourcing the cybersecurity incident response and forensics tasks to external providers is another option. While this can alleviate some of the burden on the internal team, it might not provide the same level of control and customization as an in-house solution.
- **Manual investigation and analysis**: A manual processes for digital forensics and incident response can be time-consuming, error-prone, and may not scale well with increasing volumes of data and threats.

2.7.1 Considered Open-Source Tools

More specifically, the list below provides a set of popular, off-the-shelf tools which could be potential candidates for integration with PHOENI2X:

- ***Redline*** [68] is a free tool provided by FireEye that offers host-investigative capabilities to users to find signs of malicious activity through memory and file analysis and the development of a threat assessment profile. With Redline, analysts can audit all running processes and drivers (from memory, file-system metadata, registry data, event logs, network information, services, tasks, and web history), and streamline memory and IOC analyses.
- ***GRR Rapid Response*** [69] is an incident response framework by Google, focusing on remote live forensics. It relies on agents installed on target systems and a server to manage and interact with the agents. GRR supports forensics and investigations in a fast, scalable manner, allowing analysts to quickly triage attacks and perform analysis remotely.
- ***Rekall*** [70] is an advanced forensics framework by Google, with incident response elements. It initially focused on the extraction of digital artefacts from volatile memory samples, but it has evolved to a full end-to-end solution to incident responders and forensic analysts.
- ***Volatility*** [71] is an advanced memory forensics framework, which comes with a plug in / add-on tool ecosystem, allowing it to offer additional features (e.g., malware analysis). For example, ***VolatilityBot*** [72] offers automations pertaining to the binary extraction phase and can help analysts in the first steps of performing a memory analysis investigation. ***MalConfScan*** [73] is a Volatility plugin that searches for malware in memory images and dumps configuration data.
- ***Forensics Visualization Toolkit (FVT)*** [74] is a commercial tool offered by AEGIS. It is a versatile solution for digital forensics investigations, analysis of situational awareness digital evidence and advanced visualizations that assist non-experienced in cyber-security end-users in handling security related incidents. It also aids forensic investigators analyse digital information collected from monitored components (i.e., computers, network devices, switches, as well as API-based Cloud services) in order to better understand abnormal system operation and draw data-backed informed conclusions.

2.7.2 Final Selected Tools

In the PHOENI2X platform, the tool offered by the PHOENI2X partner AEGIS, i.e., FVT, will be used. The FVT offers several competitive advantages and innovative aspects that set it apart from the alternative solutions:

- ***Integrated approach***: The toolkit seamlessly combines physical and cyber security data, enhancing risk awareness and enabling more effective threat hunting. This holistic approach provides a comprehensive view of the security landscape, allowing users to identify and address vulnerabilities that might be overlooked with traditional tools.

- ***Advanced analytics***: The FVT leverages advanced analytics techniques to discover patterns, behaviours, and correlations of data items. This powerful feature helps cybersecurity experts uncover hidden connections and trends, improving the accuracy and speed of investigations and decision-making.
- ***Tailored for cybersecurity***: Unlike generic digital forensics tools, the FVT is specifically designed for the cybersecurity sector. This focus ensures that the toolkit is optimized for the unique challenges and requirements of this domain, providing users with a more targeted and effective solution.
- ***User-friendly interface***: The toolkit is designed with an intuitive user interface, making it accessible to both novice and experienced users. This ease of use will allow PHOENI2X to quickly onboard members and maximize the value of the toolkit in its cybersecurity operations.
- ***Scalability and adaptability***: The FVT is built to scale with the evolving cybersecurity landscape, making it a future-proof investment. As new threats and vulnerabilities emerge, the toolkit can be easily updated and adapted to address these challenges, ensuring that users are always equipped with the most cutting-edge tools and techniques.

In summary, the FVT provides a tailored, integrated, and user-friendly solution for cybersecurity incident investigation and response. Its advanced analytics and focus on both physical and cyber security data enable users to uncover hidden patterns and trends, leading to more effective threat hunting and ultimately, better protection against cyber threats.

2.8 Message Bus

During the development of this deliverable, the different connections between the various components of PHOENI2X were not fully consolidated. Therefore, a centralized communication mechanism is proposed (i.e., the message bus). The message bus acts as an intermediary, allowing different security tools to communicate in a standardized manner. It plays a critical role in coordination among the components as it allows the information coming from the baseline tools to be used by the different components easily. It promotes interoperability, allowing diverse components to communicate regardless of their underlying technologies or protocols.

The idea is to use a message bus operating on a publish-subscribe model. This allows us to abstract ourselves from the tools used in the baseline and focus on the type of information that each category provides. By creating different channels or topics for every category, any baseline tool used will write into that topic, then the components from PHOENI2X that require that kind of information will subscribe to the topics.

2.8.1 Considered Open-Source Tools

Below is a list of some of the open-source tools that could be used as publisher-subscriber message bus in PHOENI²X.

- **Apache Kafka** [75] is a distributed event streaming platform known for its high throughput, fault-tolerance, and scalability. It is designed to handle real-time data streams. Its architecture allows it to store and process large volumes of data from multiple sources, making it suitable for big data and real-time analytics use cases. Its unique design ensures reliable message delivery and efficient data replication, making it popular for building data pipelines and streaming applications.
- **ZeroMQ** [76] is a lightweight, high-performance messaging library that enables asynchronous communication between applications and processes. It provides various messaging patterns, including publish-subscribe, request-reply, and push-pull. It is designed to be language-

agnostic, making it easy to integrate with applications written in different programming languages. Its simplicity and low latency make it ideal for building distributed systems, microservices, and inter-process communication in resource-constrained environments.

- **RabbitMQ** [77] is a widely used message broker implementing the Advanced Message Queuing Protocol (AMQP). It facilitates reliable message delivery between applications and systems, providing features like message queuing, routing, and persistence. RabbitMQ supports various messaging patterns, including publish-subscribe, point-to-point, and topic-based messaging. Its robustness, flexibility, and cross-platform compatibility make it suitable for enterprise-level applications, ensuring message delivery and enabling smooth communication between distributed components.
- **Apache ActiveMQ** [78] is an open-source message broker and messaging system based on Java Messaging Service (JMS) and Message Queuing Telemetry Transport (MQTT) protocols. It offers various messaging patterns, including publish-subscribe, point-to-point, and request-reply. ActiveMQ supports message persistence, high availability, and message filtering, making it suitable for diverse use cases like communication between microservices, IoT, and enterprise messaging.

2.8.2 Final Selected Tools

The chosen tool for the message bus in the PHOENI2X project is Apache Kafka. Kafka's key strengths lie in its high throughput and fault-tolerance capabilities, making it a reliable choice for handling large volumes of real-time data streams. As the project aims to interconnect multiple security tools, Kafka's ability to efficiently manage data flow and communication between components becomes crucial.

Scalability is another important aspect, and Apache Kafka's architecture is designed to scale seamlessly, accommodating growing data demands without compromising performance. This scalability ensures the message bus can handle diverse data volumes, which can be very high, for example with the reception of logs from different use cases.

In addition, the reliability of message delivery provided by Kafka is of paramount importance in a security-focused project like PHOENI2X. Messages are delivered in a robust and fault-tolerant manner, ensuring data integrity and consistency across all interconnected components. Kafka's publish-subscribe model further contributes to the project's success by facilitating loose coupling between components. This abstraction from underlying technologies used in the baseline tools allows PHOENI2X to focus on the information each category provides, rather than the specific tools themselves. Creating different channels or topics for each category simplifies the integration of new tools and enhances flexibility. Moreover, Kafka's support for real-time analytics and data pipelines aligns perfectly with the project's requirements. Real-time data processing empowers PHOENI2X to conduct timely and informed security analyses. Additionally, Apache Kafka benefits from a strong open-source community and a vast ecosystem. Continuous development, bug fixes, and feature enhancements are driven by the active community, ensuring the tool remains up-to-date and reliable.

Last but not least, Apache Kafka has been chosen due to some components of the PHOENI2X platform having the ability to produce output directly to Kafka topics. Obviously, by using a message bus for which the tools of the various partners already have as a communication mechanism, the team can focus on more important tasks, accelerating the project's progress.

3 Network & Compute Infrastructure

This section details the work done in task 2.4. It describes the design of the infrastructure and network over which PHOENI2X will operate. It is intended that with each use case different deployment models of the platform are performed. In this way we can demonstrate the flexibility offered by the PHOENI2X architecture. For use case 1, PHOENI2X will be deployed in premises, this means that all the infrastructure used for use case 1 demonstrations will be inside PPC. In use case 2, a hybrid approach will be used, part of the tools will be deployed in WSE, especially the baseline tools, while the rest of the components will be deployed in the UPAT infrastructure. Finally, PHOENI2X is deployed with cloud model for use case 3, where all the tools including part of the baseline tools are located in UPAT, while NPS has its tools pre-installed and redirects the logs of those tools to PHOENI2X. Where the tools selected in the previous section are deployed will depend on the use case, the Table 1, Table 2, and Table 3 display where and which tools are deployed. It is notable that MISP, belonging to the Cyber Threat Intelligence category, is deployed in ATOS premises, this is because this tool works in conjunction with TII, which due to intellectual property cannot be deployed in non-ATOS environments.

Table 1. Information about the infrastructure used for the baseline tools for Pilot 1: Energy.

Pilot 1: Energy		
Category	**Tools**	**Deployment Info**
Dynamic & Static Testing	• OpenVAS • SPA Vulnerability Scanning	Deployed on PPC testbed
Security Information & Event Management	• Wazuh	Deployed on PPC testbed
Cyber Threat Intelligence	• MISP	Deployed in ATOS premises
Endpoint Detection and Response	• Windows Logging via Wazuh Agent	Deployed on AMI HeadEnd within PPC testbed
Network Detection and Response	• Zeek	Deployed on PPC testbed
Deception Tools	• Conpot	Deployed on PPC testbed
Digital Investigations and Forensics	• FVT	Deployed on PPC testbed
Message Bus	• Kafka	Deployed on PPC testbed

Table 2. Information about the infrastructure used for the baseline tools for Pilot 2: Transport.

Pilot 2: Transport		
Category	**Tools**	**Deployment Info**
Dynamic & Static Testing	• OpenVAS • SPA Vulnerability Scanning	Static Testing through SPA in UPAT. Dynamic testing, OpenVAS deployed in WSE, SPA in UPAT.
Security Information & Event Management	• Wazuh	Deployed on WSE testbed
Cyber Threat Intelligence	• MISP	Deployed in ATOS premises

Endpoint Detection and Response	• Wazuh Agent	Deployed at the same environment of CMT Cloud at WSE
Network Detection and Response	• ad-hoc solution that collects metadata on the LongRange radio network parameters	Deployed in WSE testbed
Deception Tools	• Conpot	Deployed in WSE testbed
Digital Investigations and Forensics	• FVT	Deployed in UPAT premises
Message Bus	• Kafka	Deployed in UPAT premises

Table 3. Information about the infrastructure used for the baseline tools for Pilot 3: Health.

Pilot 3: Health		
Category	**Tools**	**Deployment Info**
Dynamic & Static Testing	• OpenVAS • SPA Vulnerability Scanning	Static Testing through SPA in UPAT. Dynamic testing, OpenVAS deployed in WSE, SPA in UPAT.
Security Information & Event Management	• Wazuh	Deployed in NPS testbed
Cyber Threat Intelligence	• MISP	Deployed in ATOS premises
Endpoint Detection and Response	• Using Beats to collect data from existing testbed hosts	Deployed in NPS testbed
Network Detection and Response	• Suricata running as opnSense module	Deployed in NPS testbed
Deception Tools	NA	NA
Digital Investigations and Forensics	• FVT	Deployed in UPAT premises
Message Bus	• Kafka	Deployed in UPAT premises

Therefore, in this section we will describe the UPAT infrastructure, used as cloud infrastructure, and the infrastructure of the use cases. However, it will be in deliverable 5.2 where more detail on the testbed of each of the use cases will be provided. In deliverable 5.2 we will also provide more details on how the management and orchestration of the infrastructure is performed.

3.1 PHOENI2X Infrastructure Description

UPAT cloud platform offers a total computing power of 450 CPUs and 1,5 TB of RAM and 50 TB of storage. All servers are interconnected on TOR 10GbE/40GbE NVIDIA Cumulus switches with dual 10GbE NICs DPDK enabled. Kubernetes clusters are available and can be created on demand for the users. The cloud platform hosts a large number of services like smart cities applications (https://sense.city/, https://safeamea.gr/), UPAT research, EU and national funded projects etc. Apart from the typical cloud resources, the cloud also provides specialized 5G standard-conformant components, Core Network infrastructure and Integration of 5G Core and 5G RAN with its Opensource

based NFV platform. It supports various flavors and installations of the 5G System, that are both NSA and SA depending on the scenarios that the customer wants to support. GEANT connectivity is also available.

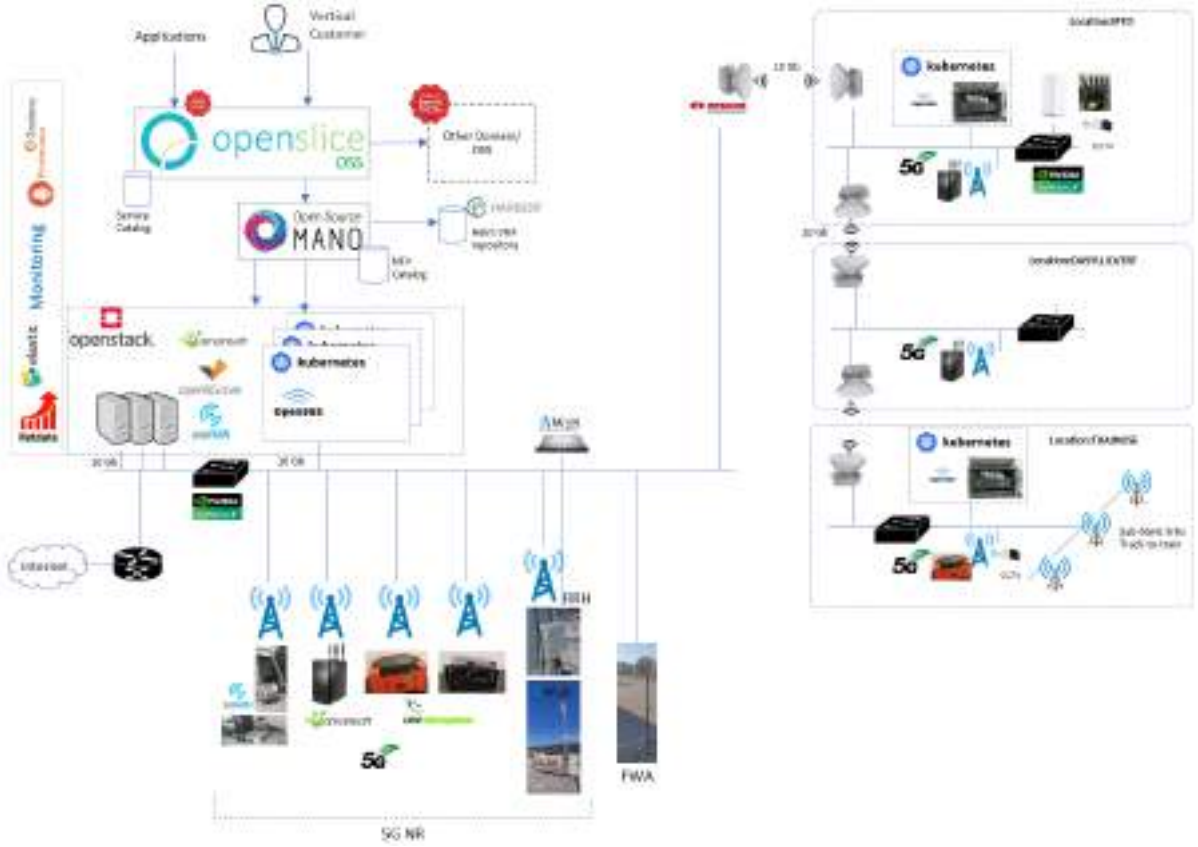

Figure 1. UPAT cloud and 5G specialized components

For the cloud deployment for PHOENI2X, a set of VMs have been created and connected inside a private network. Only one VM (Gateway) has direct access to the internet and runs a Network Address Translation (NAT) service to provide internet access to the rest of the private network. A Virtual Private Network (VPN) service has also been installed and configured inside the Gateway VM, to allow the various technical partners connect to the private network's VMs and work on their components.

For validating PHOENI2X framework, three different pilots have been defined (energy, transport, health). To ensure that this phase will be able to evaluate multiple different deployments of the overall framework, we have designed three diverse implementations. For the first one, the complete PHOENI2X framework will be installed inside the pilot premises. In the second one, some PHOENI2X components will be installed in the pilot premises and the rest inside the UPAT cloud (hybrid). Finally, third implementation, the pilot will not host any of PHOENI2X's components and all of them will reside inside the UPAT cloud.

Based on this approach, the cloud resources provided by the University of Patras are 6 VMs, 46 CPU, 74 RAM, 460 Disk and are analyzed as follows:

Table 4. Provided resources by UPAT for Pilot 1

Pilot 1: Energy
No resources

Table 5. Provided resources by UPAT for Pilot 2

Pilot 2: Transport	
Instance 1	
Name:	Phoeni2x_Sphynx
Flavor:	VCPU 8, RAM 16GB, DISK 120GB
Owner	SANL
PHOENI2X Components:	• Dynamic/Static Testing • Security Assurance Platform • Resilience CR • UEBA • ROAR - Engine • ROAR - Playbooks
Instance 2	
Name:	aegisfvt
Flavor:	VCPU 2, RAM 12GB, DISK 40GB
Owner	AEGIS
PHOENI2X Components:	• Digital Forensics • Visualization • ELK (Elasticsearch, Kibana, Beats, and Logstash)
Instance 3	
Name:	atossiem
Flavor:	VCPU 16, RAM 24GB, DISK 80GB
Owner	ATOS
PHOENI2X Components:	• SIEM (Wazuh)
Instance 4	
Name:	atosmessagebus
Flavor:	VCPU 8, RAM 6GB, DISK 80GB
Owner	ATOS
PHOENI2X Components:	• Apache Kafka

Table 6. Provided resources by UPAT for Pilot 3

Pilot 3: Health	
Instance 1	
Name:	Phoeni2x_Sphynx
Flavor:	VCPU 8, RAM 16GB, DISK 120GB

Owner	SANL
PHOENI2X Components:	• Dynamic/Static Testing • Security Assurance Platform • Resilience CR • UEBA • ROAR - Engine • ROAR - Playbooks
Instance 2	
Name:	aegisfvt
Flavor:	VCPU 2, RAM 12GB, DISK 40GB
Owner	AEGIS
PHOENI2X Components:	• Digital Forensics • Visualization • ELK (Elasticsearch, Kibana, Beats, and Logstash)
Instance 3	
Name:	atossiem
Flavor:	VCPU 16, RAM 24GB, DISK 80GB
Owner	ATOS
PHOENI2X Components:	• SIEM (Wazuh)
Instance 4	
Name:	atosmessagebus
Flavor:	VCPU 8, RAM 6GB, DISK 80GB
Owner	ATOS
PHOENI2X Components:	• Apache Kafka

Table 7. Provided resources by UPAT for General Tools (i.e., all pilots)

General Tools	
Instance 5	
Name:	Phoeni2x_GW
Flavor:	VCPU 8, RAM 16GB, DISK 80GB
Owner	UPAT
Purpose	This VM acts as the gateway of the PHOENI2X cloud infrastructure. It hosts the VPN server as well as the NAT service.
Instance 6	
Name:	upatrasgit

Flavor:	VCPU 4, RAM 8GB, DISK 80GB
Owner	UPAT
Purpose	This VM acts hosts the project's Gitlab to act as the code repository and enable collaborative software development.

Figure 2 depicts the VMs that have been created so far to host the PHOENI2X components.

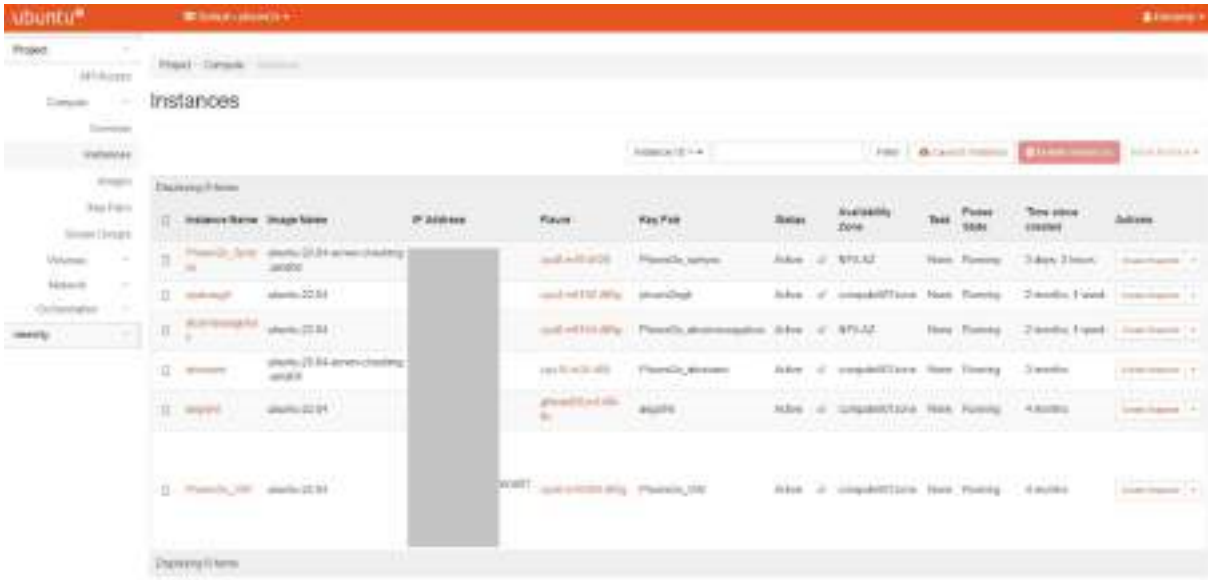

Figure 2. PHOENI2X cloud resources

Besides the resources provided by UPAT, ATOS will supply resources for deploying its assets in order to preserve Intellectual Property Rights (IPR). ATOS provides a virtual machine with a public IP; the assets within this machine are used for the three pilots. The virtual machine has the following resources:

Table 8. Provided resources by ATOS for all pilots.

ATOS VM for Pilot 1, 2 & 3	
Instance 7	
Name:	MISP-TII-CERCA-MISP
Flavor:	VCPU 14, RAM 25GB, DISK 140GB
Owner	ATOS
PHOENI2X Components:	• MISP • TII • CERCA • SMIR

3.2 Use Case 1 – Energy Infrastructure Description

The energy infrastructure is based on the R&D testbed of PPC's Innovation Hub. The testbed is split into different network segments to enforce an air-gap between the critical devices. Moreover, the testbed includes a dedicated Infrastructure server, where virtualization Proxmox Virtual Environment is running that hosts virtual machines. The server characteristics are HPE ProLiant ML350 Gen10 Intel

Xeon Golf with CPU 2.30 GHz 32 threads, 128 GB RAM memory and an SSD slot for 5.44 TB additional RAM.

Furthermore, the overall network topology also includes SDN switches with controller software and dashboard as well as all the processes and applications are containerized through docker images. In terms of security, access to the infrastructure is restricted as it includes VPN connection as well as firewall mechanisms using pfsense to block malicious activity. The firewall logs are collected and analyzed as well as regular updates on the rules are also happening based on the monitored network traffic. The overview of the R&D testbed infrastructure is illustrated in Figure 3Figure 3 - Energy use-case infrastructure overview.

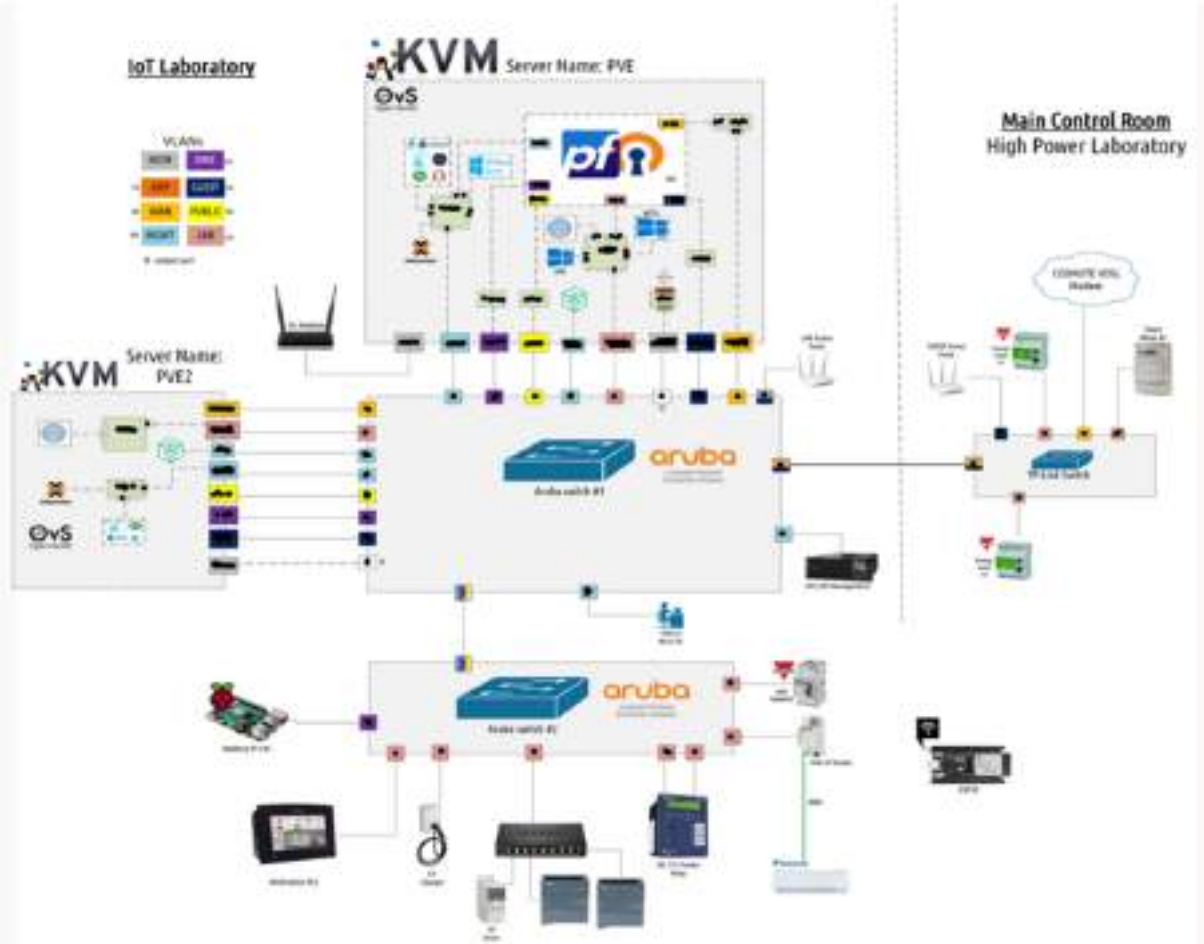

Figure 3 - Energy use-case infrastructure overview

3.3 Use Case 2 – Transport Infrastructure Description

The controlled environment set-up for the Use Case 2 is based on a replica of the Connectivity Management Tool (CMT) production software, under a specific environment where all the Phoeni2X baseline tools can be deployed and connectivity with the UPAT testbed.

CMT is a complex software solution using several software components, and therefore an architecture based on using Kubernetes is used. Such architecture isolates several components from the scalability requirements that a Cloud based software being offered as SaaS.

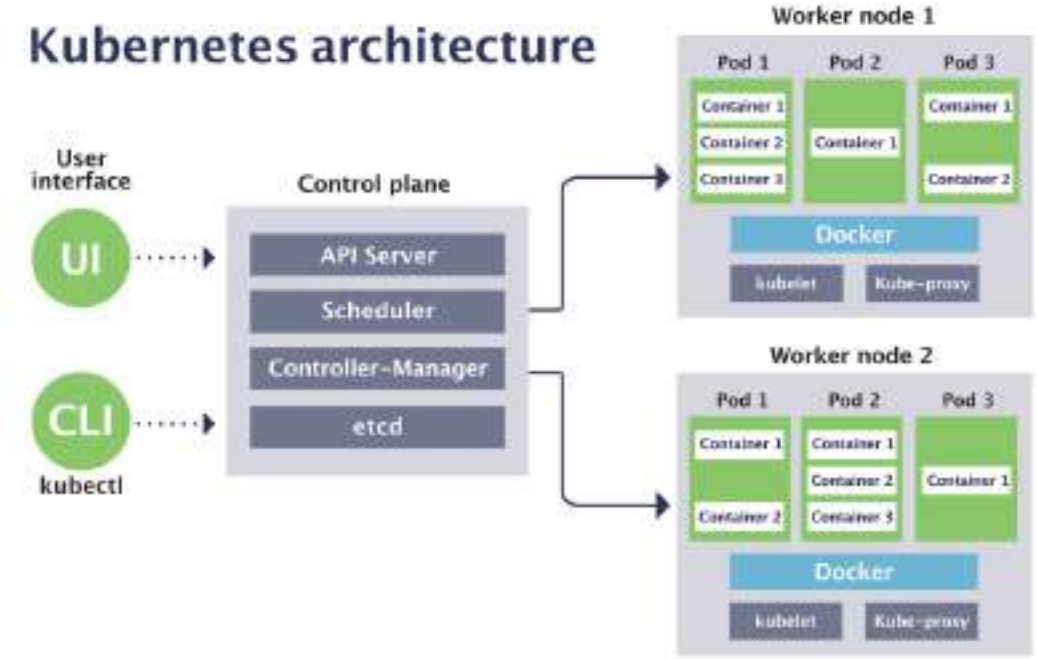

Figure 4. Kubernetes general architecture with two worker nodes

CMT general architecture uses 5 VM where three of them are Kubernetes worker nodes, one is the load balancer, and one is used for permanent storage purposes. The VM are provided by Google Cloud Platform for the cloud-based deployments but can be provided and managed by any VM management solution like OpenStack or VMWare. The requirements for the VM for the K8 worker nodes are 4 vCPU and 16GB RAM, the load balancer needs 2vCPU and 4GB RAM, and the permanent storage VM has a requirement for 4vCPU, 4GB RAM and 20GB SSD.

Some baseline tools and one PHOENI2X tool will be deployed at the same controlled environment in separate VMs. The baseline tools deployed will be EDR and SIEM, deployed at a single VM with 4 vCPU and 16GB RAM, and the NDR will also be deployed at the same environment, as it is an ad-hoc solution, using a specific VM with 2 vCPU and 4GB RAM. While the PHOENI2X tool will be PMEM, deployed at another VM with 4 vCPU and 16GB RAM. A secure channel will establish connectivity with UPAT servers where the other baseline and PHOENI2X tools will be deployed.

3.4 Use Case 3 – Health Infrastructure Description

Use Case 3 uses a variety set of technologies hosted on an oVirt Cluster. This is a server virtualization management platform able to configure, monitor and manage a Linux Kernel-based Virtual Machine (KVM) environment with enterprise-grade performance. oVirt is an Open-source project adopted by all major Linux Distributors, in this case the Oracle version is used (OLVM 4.4.10). oVirt is installed on top of three (3) Dell Servers each one with 28 cores and 256GBs of RAM running Oracle Linux 8.8 Hypervisors. These servers are connected via redundant 10Gbps Ethernet switches. The IBM 7000 is used as storage appliance, connected to the hosts using two Brocade redundant Fiber Channel Fabrics in multizone setup, supporting the Data domain of the oVirt cluster. The use case traffic is isolated in a network island using multiple VLANs for compartmentation. An OPNsense firewall is responsible for every network aspect of the use case and because of its modular/plugable capabilities it is also used for the supporting services, such as DNS and NTP. OPNsense is responsible for the OpenVPN tunnel that connects the use case 3 with the central PHOENI2X servers at UPAT. All guests VMs are able to

connect to the central ELK and there are rules, routes and NAT to enable clients from Phoeni2x servers to connect to the Control Plane SSL endpoint. All use case 3 services are running on Ubuntu 22.04 VMs. Most of them are containerized using Docker/container runtimes and docker-compose scripts.

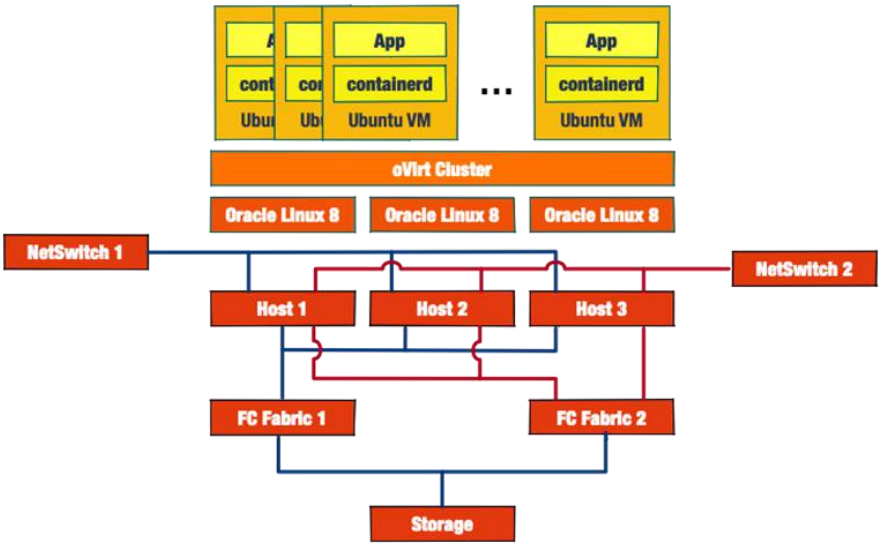

Figure 5. The UC3 infrastructure.

4 Security Assurance & Certification

This section introduces the Security Assurance & Certification module, describing the work done on task 2.5 and detailing the design and operation of the module.

4.1 Overview

As already highlighted since the proposal preparation phase, the introduction of PHOENI2X in the OES environment is expectedly going to increase the attack surface with all these new technologies and tools. Thus, also considering the criticality of the application and access to sensitive processes within the organization, building trust into PHOENI2X CRCs is of fundamental importance.

Towards that, the proposed architecture of PHOENI2X CRCs includes a continuous, evidence-based Security Assurance and Certification solution (developed within Task 2.5 of the project). This will provide comprehensive assessment coverage of every component of the PHOENI2X CRCs and their operation, covering all assets (e.g., network, compute, data, processes) comprising each CRC.

The overarching aim of this component is to provide an auditable, evidence-based security assessment and certification of the assurance posture of PHOENI2X CRCs to the OES that deploys it. The solution will combine: (i) static testing (e.g., vulnerability assessment); (ii) dynamic testing (e.g., penetration testing); (iii) CTI assessments, leveraging the CTI capabilities of the baseline toolset; (iv) continuous runtime monitoring, driven by temporal event patterns and rules implementing signature and anomaly-based security incident detection, and (v) hybrid assessments, combining evidence from assessments of types (i)-(iv).

To obtain the monitoring and test evidence required for the assurance assessments, the platform will be hooked to different systems programmatically (e.g., through event captors, test tools). Operation will be driven by assessment models developed to enable assessments & certifications according to specific schemes (e.g., ISO security standards, GDPR, NIS Directive), but also the requirements specified within EC's "Ethics Guidelines for Trustworthy Artificial Intelligence" [79], producing a novel Trustworthy AI Assessment Model against which the AI enablers of the platform will be evaluated.

More details on this aspect of PHOENI2X and the enablers that will allow for this capability are provided in the subsections that follow.

4.2 Design Details

The Security Assurance & Certification capabilities of PHOENI2X CRCs will be covered through integration of the SPHYNX Security and Privacy Assurance Suite (SPHYNX SPA Suite); an integrated suite of tools that provides comprehensive cyber security risk detection and management for enterprise systems.

The operation of the platform is model-driven, based on underlying comprehensive Security & Privacy assurance models, to provide a common (and uniform) basis for all sorts of reasoning required and provide extensibility to the operation of the platform (e.g., to cover new types of assets or assessment types, as needed). Overall, the SPA Suite:

- Combines runtime monitoring and dynamic runtime testing to ensure correct and effective operation of security controls.
- Can be hooked to different systems programmatically through appropriate probes (e.g., event captors, test tools) to obtain the monitoring and/or test evidence required for assurance and/or certification assessments.
- Operates based on models that determine the operational evidence that should be captured from systems and how it should be assessed (e.g., what conditions it should satisfy) to assess the correctness and effectiveness of implemented system security controls.

- Enables the runtime assessment of temporal event patterns and rules that can express signature or anomaly-based patterns.

Key features include [80]:

- Comprehensive client asset modelling and automated client asset discovery.
- Automated threats and vulnerabilities detection.
- Sophisticated event processing capabilities and continuous runtime monitoring (SIEM-like features).
- Automated cyber threat intelligence ingestion and hunting.
- Penetration testing and support for ingestion of penetration testing reports using third-party tools.
- Hybrid risk assessments for comprehensive technical and economic cyber risk estimates.

In addition, the SPA suite can offer seamless integration with the User and Entity Behaviour Analytics (UEBA), incident response (i.e., ROAR), and CTI components of PHOENI2X.

4.2.1 Internal Architecture

A high-level architecture of the SPA Suite is shown in Figure 6; the following main modules can be identified:

- **Core Platform components**: These include the **Asset Loader**, i.e., the component that provides handles the core asset model and the relations between the assets, and the **Assessment Loader**, i.e., the component that handles the assessments, including the execution of an assessment, the retrieval of assessment results etc.

- **Asset-based Vulnerability Assessment**: Based on the Vulnerability Analyzer, this module is responsible to identify known vulnerabilities of assets defined within an organisations' asset model. This component automatically constructs the CPE [81] (a structured naming scheme for information technology systems, software, and packages) per asset and then retrieves the relevant Common Vulnerabilities and Exposures (CVE [82], a reference-method for defining unique, common identifiers for publicly known information-security vulnerabilities and exposures) entries, by searching in a local copy of the National Vulnerability Database (NVD [83], a U.S. government repository of standards-based vulnerability management data, maintained by the National Institute of Standards and Technology, NIST [84]). This copy is continuously updated by utilising an in-house component that fetches the latest known CVEs from NVD's JSON files.

- **CRISES:** The hybrid assessment module, responsible for combing the different types of assessments / evidence (e.g., monitoring, testing), whereby users can create standalone or hybrid risk detection models, execute them using the CRISES engine, and leverage the results to evaluate the security status of the cyber system.

- **EVEREST** [85] [86] [87]: A runtime monitoring engine, built-in Java, that offers an API for establishing the monitoring rules to be checked. This module is composed of two submodules: (a) the monitoring database and (b) the monitor. The role of the module is to forward the runtime events from the application's monitored properties and finally obtain the monitoring results. **EVEREST** basic implementation is based on Drools logical language, while it is modelled in Event Calculus, a first order temporal logic that can both represent and reason actions and their effects over time. By abstracting the above concepts, Event Calculus basic elements are comprised of events and fluents. **EVEREST** performs continuous assessments and is based upon these core logic factors of Event Calculus in terms of comprising the rules continuously checked in a system.

- **Dynamic Testing**: The component responsible for initiating the (penetration) testing assessments. The module consists of two components: (a) the dynamic tester and (b) the dynamic (penetration) testing tool itself; typically, an external tool (more on that below).
- **Event Captors**: The Event Captor is a tool, implemented In Java, that based on collected data and triggering events, formulates a rule or a set of rules and pushes the latter towards the monitoring module for evaluation. Data and events are mostly collected through Elastisearch[1] based on lightweight shippers (namely Beats), such as Filebeat[2], Metricbeat[3], Packetbeat[4], etc., that forwards and centralizes log data. Data can also be collected through Logstash[5], an open server-side data processing pipeline that ingests data from a multitude of sources transforms it, and then sends it to Elasticsearch. The Event Captor is initiated through the respective REST calls from the monitoring module. In addition, Elasticsearch provides build-in mechanisms to support the secure communication with SSL communication and hence the interaction is performed via the HTTPS protocol.

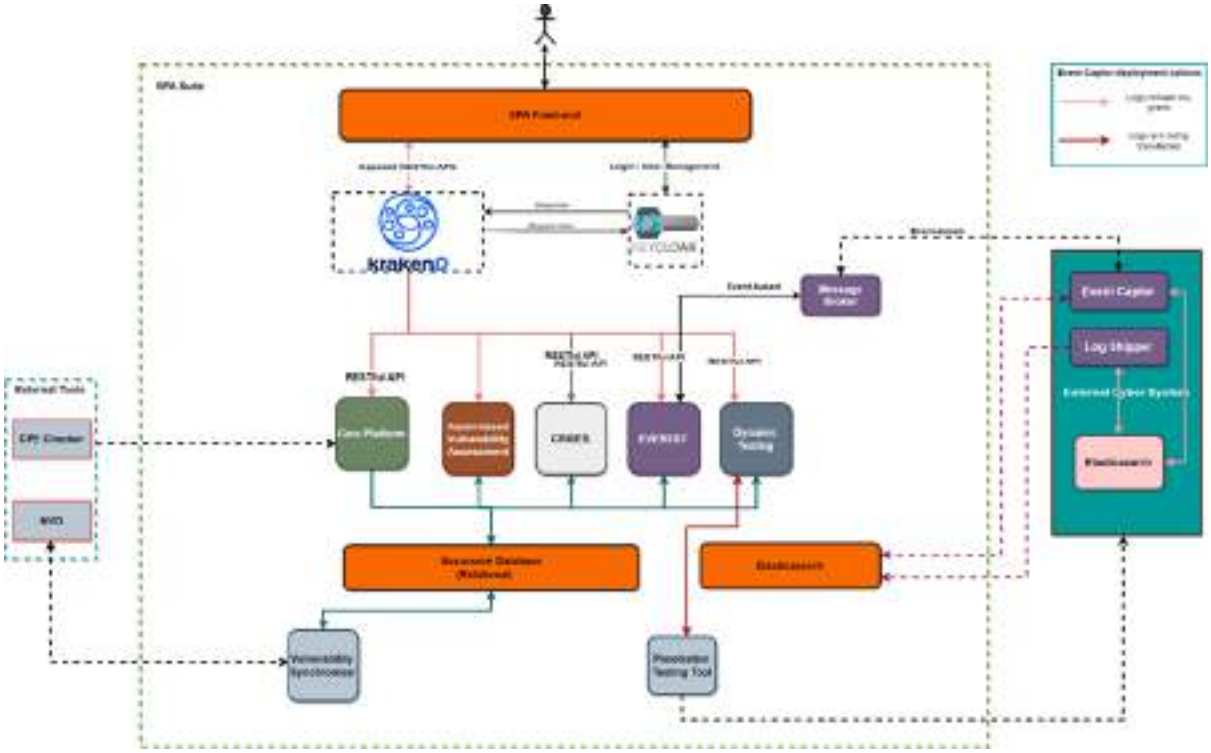

Figure 6. SPA Suite core modules (internal architecture)

Moreover, the operation of the platform relies on the following databases:

- **EVEREST Database**: Holds the monitoring rules and the overall process done by EVEREST if the templates and other values important for monitor procedure to conclude.
- **Assurance Database**: Holds the cyber system asset model, its components, and the results of the assessments.

[1] https://www.elastic.co/
[2] https://www.elastic.co/beats/filebeat
[3] https://www.elastic.co/beats/metricbeat
[4] https://www.elastic.co/beats/packetbeat
[5] https://www.elastic.co/logstash

- **Vulnerabilities Database**: Holds the known vulnerabilities, as retrieved from the NVD database.

Finally, the Assurance tool interfaces with a set of external components, namely:

- **Message Broker (RabbitMQ):** The message broker is a messaging bus that allows communication between the external components and the assurance platform using AMPQ protocol. This will also act as the Trust Broker in IntellIoT; see Sect. 6.
- **OpenVAS[6]:** This is a software framework of several services and tools offering vulnerability scanning and vulnerability management. The tool is used as part of the dynamic testing offered by the Assurance Platform

4.2.2 The PHOENI2X Assurance Model

Being model-driven, at the core of the operation of the Security Assurance Platform is the specification of the associated Assurance Model. Thus, the PHOENI2X Assurance Model must be defined, and should holistically cover PHOENI2X CRC assets and, optionally, other assets of the use case deployment itself that need to be assessed & protected.

In more detail, the assurance model should specify all assets of the target cyber system to be protected, known threats that may affect the physical or software components of the system; assumptions regarding the external environment of the cyber system and the behaviour of agents (human- or system-agents) related to it that may affect it (i.e., prevent or introduce threats); and security controls used to mitigate the risks arising from the threats.

The assurance model also specifies assessment procedures, to identify vulnerabilities, assess the proper function of security controls, and to determine how to detect and respond to cyber-attacks. To support this feature, it can also specify any assessment tools that should be employed for the aforementioned assessments prior the deployment of the system (i.e., static analysis and testing tools) or during its operation (i.e., monitoring and dynamic testing tools). Moreover, it specifies parameters determining how the attacks may manifest, how the security controls may respond to them (e.g., the attack manifestation events captured and detection time, the undertaken response actions) and the outputs that the deployed assessment tools will generate for the situation.

[6] https://www.openvas.org/

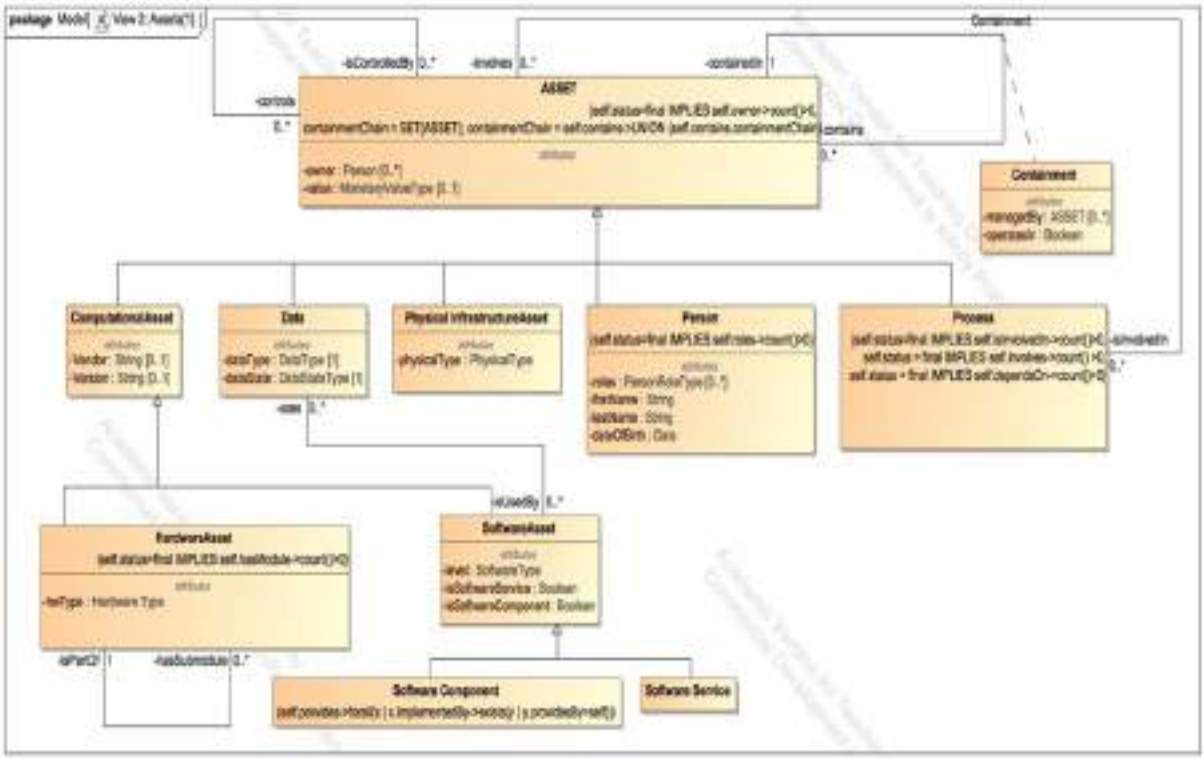

Figure 7. Part of the Assurance Model, showing a view of the Asset sub-model

Since it encompasses all the above, the final model is quite complex - for the sake of brevity, only a high-level description is provided herein. More specifically, the Assurance model consists of the following:

- **Security Assurance Model Element:** The core element of the model, inherited by every other class within the model. Its specification includes the name of the element, the timestamp of the insertion of the element in the assurance platform, the timestamp of the deletion of the element from the assurance platform, a brief description of the element that states its purpose, and the user that inserted this element in the platform.
- **Asset sub-model:** Describes the different assets of the cyber system (i.e., all IntellIoT assets). In addition to the attributes inherited from the Security Assurance Model Element, it also includes the owner (person that owns the asset), its value (in monetary terms) and a number of associations, such as if it is protected by one or more security control(s), if it is contained within another asset and/or if it contains and/or controls other asset(s), and its exposed interfaces. Moreover, the Asset sub-model includes a number of sub-models for different types of assets; namely: *Physical Infrastructure Asset sub-model, Hardware Asset sub-model, Software Asset sub-model, Data Asset sub-model, Person Asset sub-model, Process Asset sub-model and Security Control Asset sub-model*.

A partial view of the Assurance Model, covering the different types of assets in the Asset sub-model and their relationships, is provided in Figure 7.

To create an accurate instance of the PHOENI2X assurance model – one for each use case deployment - a specific process is established whereby all partners (i.e., asset owners) are involved. In more details, all Asset sub-model fields needed for assets' specification have been mapped onto tabs and tables of an Excel-based form (see Figure 8 for a partial view of said form). Said file was created to allow all partners to define the needed fields for their assets (PHOENI2X components, as involved in each of the three use cases) without having to interact with the Assurance Tool itself.

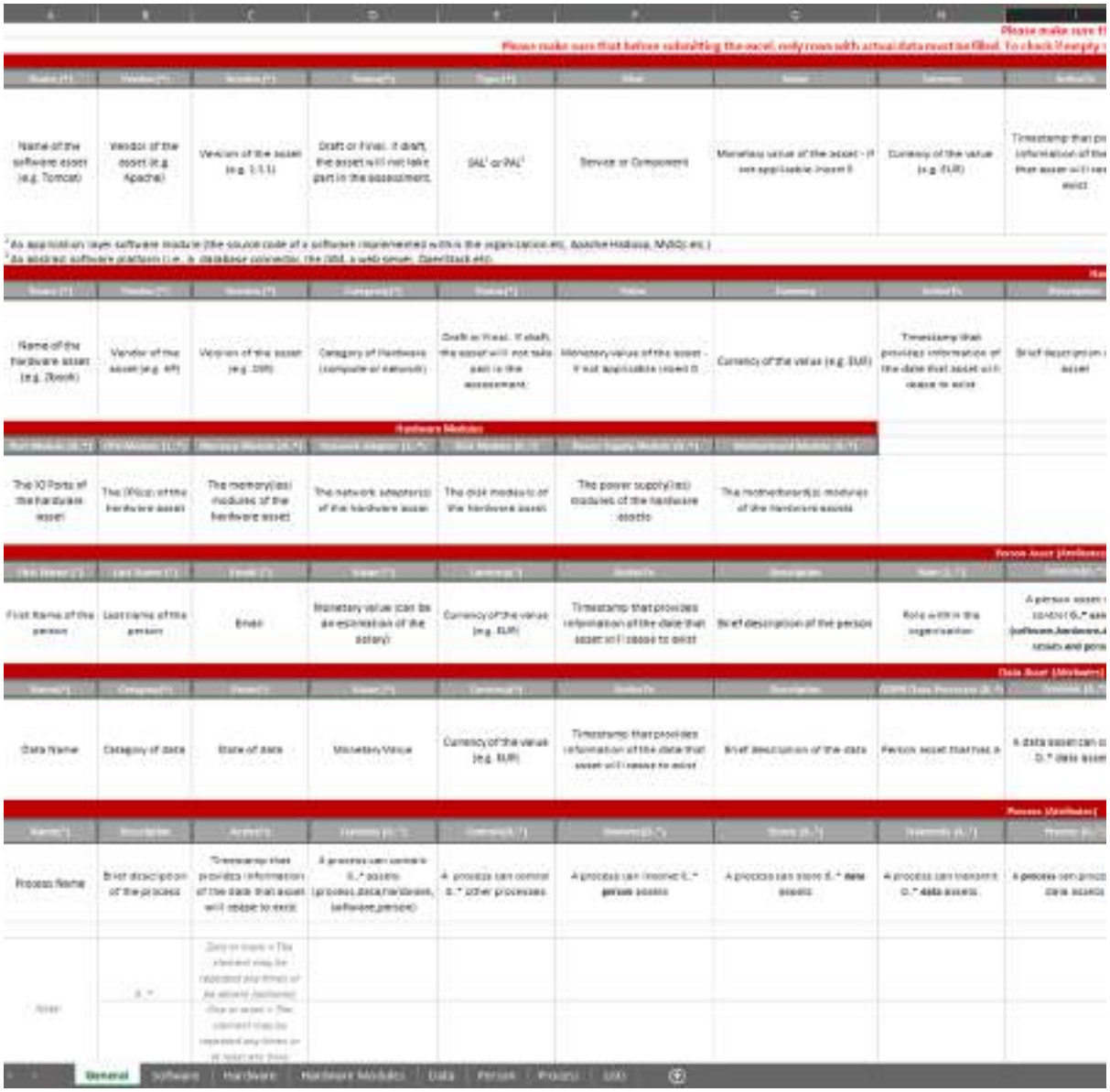

Figure 8. Partial view of the PHOENI2X Asset Model specification form (cover page with instructions)

In addition to the Excel-based form mentioned above, the asset model can be populated through the SPA front-end (a "wizard" guides the user through the steps), though this is a more time-consuming process, not intended for bulk specification of assets.

A third approach is using the Asset Model Grammar that is aligned with the structure of the underlying asset model. This method enables the bulk specification of assets within the SPA, where using the Excel form is considered inefficient (e.g., large number of assets). An example of an asset definition (in this case an instance of the Ubuntu Operating System; i.e., a software asset) using the Grammar is provided below:

```
SoftwareAsset(vendor("Ubuntu"),version("18.04
LTS"),name("Ubuntu"),kind(Service),type(PAL),project("PHOENI2X"),organisation("SPHY
NX"),description("Server OS"))
```

A more complex example of a hardware asset defined in the SPA's Asset Model Grammar is provided below:

```
HardwareAsset(vendor("DELL"),version("6800"),name("AssuranceVM"),
value(3000.0),currency(EUR),hwType(compute),project("PHOENI2X"),organisation("SPHYN
```

```
X"),CpuModule(name("Intel i7-
8700"),manufacturer("GenuinIntel"),numberOfCores(4),numberOfThreads(8),baseClock(2.
4),cache(12),MemoryModule(name("VENGEANCE
LPX"),size(32),type("DDR4"),speed(3200.0),manufacturer("Corsair")),NetworkAdapterMo
dule(connectionType(Integrated),MAC("00:0a:95:9d:68:16"),NetworkConfiguration(ipv4(
"195.201.62.1"),Subnet Mask("255.255.0.0"))))
```

Overall, once the asset definition process is completed, three instances of the PHOENI2X Assurance Model will be derived, one for each use case deployment, covering the specific assets involved in said deployment. These will, of course, initially reflect the first release of PHOENI2X and its associated demonstrators, while a second (and final) version of the models will be specified before the final release & demonstration. The latter will cover both the new/updated components involved in the final release of PHOENI2X, but also the extended Use Case environments that will be used for the final validation.

4.3 Model-driven Assessments

SPA, based on the PHOENI2X Assurance Model described in the previous section, performs various types of security assessments, based on the corresponding assessment models. More specifically, as mentioned above, SPA will support:

(i) Monitoring Assessments, based on EVEREST described above;
(ii) Vulnerability Analysis Assessments, leveraging the Vulnerability Analyser capabilities described above;
(iii) Dynamic Testing Assessments, using the Dynamic Tester component and, in this case, the OpenVAS vulnerability assessment solution described above;
(iv) CTI Assessments, allowing the contextualisation of CTI information aggregated by OpenCTI, through their correlation with properties stored within the asset model & existing assessment results (e.g., vulnerability assessments);
(v) Hybrid Assessments, combing (i)-(iv) above, as enabled through CRISES, a pipelined, query-type language.

An overview of the dashboard with different types of executed assessments & relevant KPIs on said dashboard are shown in Figure 9.

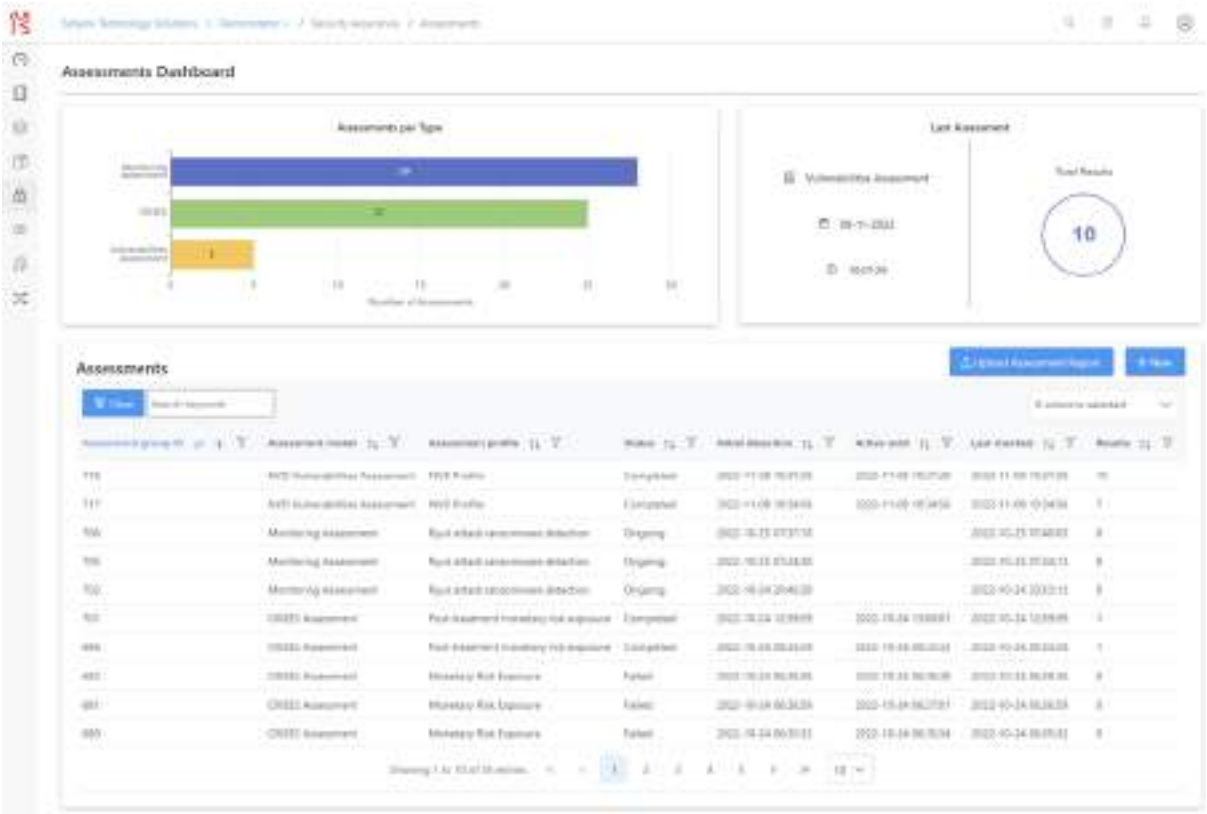

Figure 9. Listing of executed types of Security Assessments on SPA front-end.

Through this page, the PHOENI2X user operating the SPA can: (i) initiate a new assessment, (ii) upload an existing assessment report and (iii) view the existing assessments. Each existing assessment holds a number of general information such as: the assessment model that was utilised (e.g., NVD Vulnerability Assessment); the assessment profile that was used (e.g., focusing on Confidentiality), (c) the status of the assessment (e.g., ongoing); the assessment's initial detection date, (e) how long the assessment is valid (this is mostly used by the monitoring assessment); when the assessment was last checked, and; the number of current assessment findings.

A user can enter any assessment page and view the results by clicking anywhere on the corresponding line. As shown in Figure 10 (left side), the user is being presented with the basic information of the assessment such as a brief description, the project it belongs to, its creator, the date it was last updated, the tool/service that is being used, the security properties it checks, its status, the involved assets, and a brief description of the assessment profile. Some key statistics and other information are provided to the user in the form of charts and graphs (e.g., distribution of findings by severity, most vulnerable assets). Further below (right side of Figure 10), the user can see individual assessment results (findings), with some general parameters such as the asset it involves, the security property, the normalized likelihood, an initial detection timestamp (if applicable), the time it was last checked and the time it ceased to exist (if applicable).

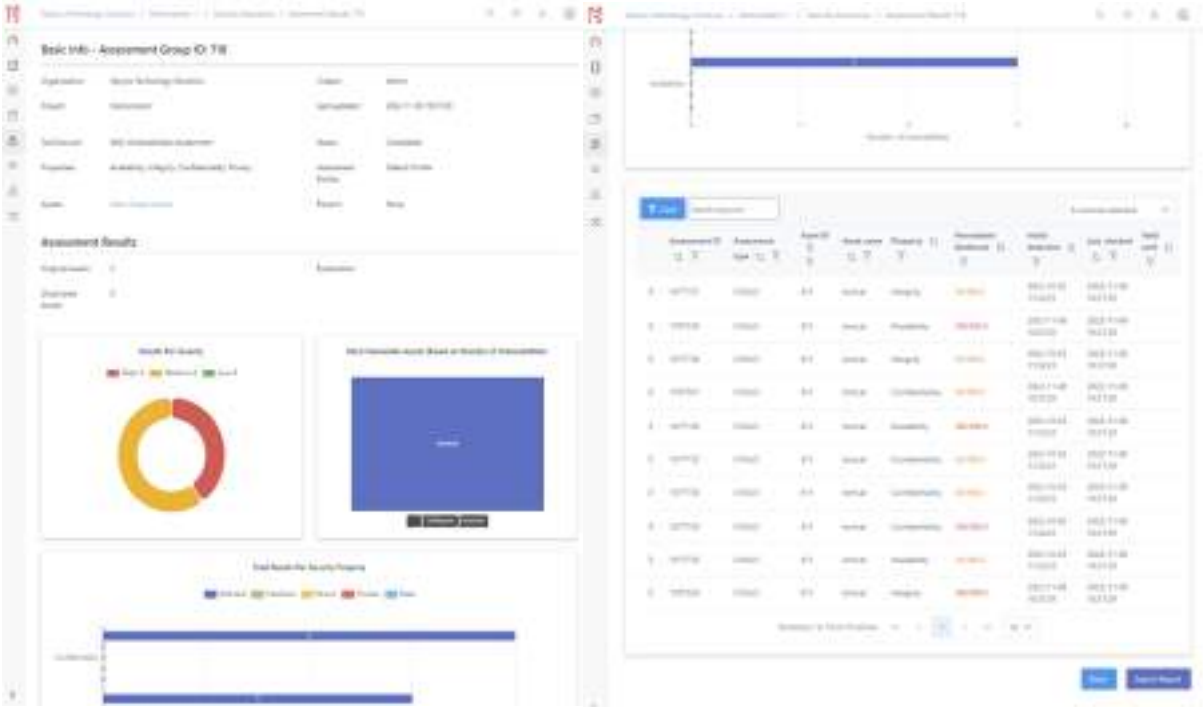

Figure 10. Sample of an assessment results screen (in this case NVD assessment) within the SPA front end (overview left, detailed findings right).

More specific information about each result can be seen by clicking on it. For instance, a specific result from a **Vulnerability Assessment** is shown in Figure 11, whereby CVE-2021-30640, pertaining to certain versions of the Apache Tomcat server is found in the modelled assets. In this case, the presented information follows the latest version (V3.1) of the Common Vulnerability Scoring System (CVSS [88]), that provides a standardised way to capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity.

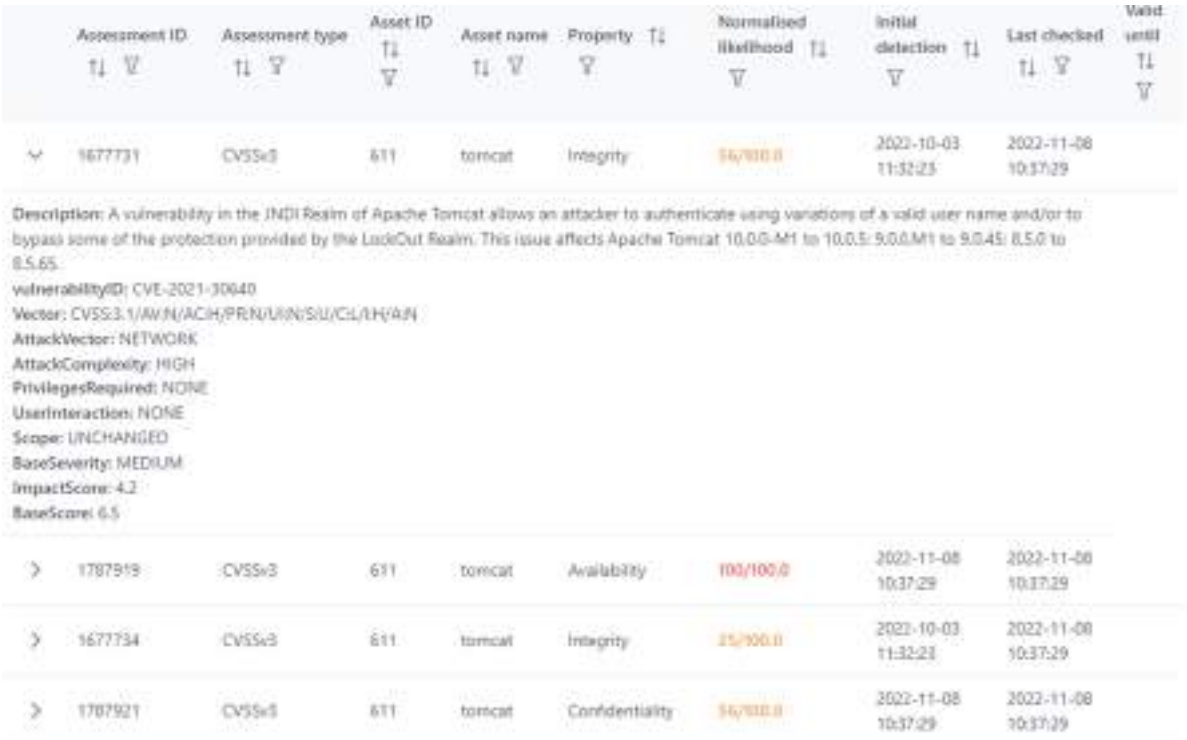

Figure 11. Viewing details of a specific finding of a Vulnerability Assessment.

The user is presented with similar screens for other assessment types as well, but each tailored to the intricacies of the assessment type (covering all four assessment types noted above) & associated assessment results. Another example, this time expanding on a specific finding of a **Dynamic Testing Assessment** carried out using the OpenVAS tool is shown in Figure 12. In this case, a finding is presented that pertains to the SSH server found during the testing, and the tool identifies a specific vulnerability that stems from a misconfiguration of said server (revealed by directly interacting with the server). Here additional fields are provided, such as how the tool discovered the vulnerability, the confidence in the result (from 0-100, with 100 being that the vulnerability is certainly exploitable), and recommendations to mitigate the finding.

Figure 12. Viewing details of a specific finding of a Dynamic Testing Assessment.

Further, Figure 13 presents details on a finding from a **Monitoring Assessment** (i.e., using the EVEREST Monitoring Module). Here the finding is created through the "Violation" of a monitoring rule that was defined to observe for a specific pattern that may indicate ransomware activity. The result provides details on which Event Captor triggered the rule (i.e., where the violation was observed), the exact source and receiver, the time stamp, and other details that allow to uniquely identify the event (and, consequently, the suspicious behaviour), allowing triggering of appropriate mitigation actions, if needed (e.g., through the IRM tool), and providing the necessary evidence required for audit purposes.

Figure 13. Viewing details of a specific finding of a Monitoring Assessment.

As mentioned, **CTI Assessments** can also be executed through SPA, in tight integration with a CTI platform, such as OpenCTI which was used for the proof-of-concept implementation presented here. A CTI assessment follows the same conventions as all other Assessments executed through the Sphynx SPA Suite. Firstly, a CTI Assessment Criterion must be created. The specification of the criterion is populated dynamically using dropdown menus, following the established STIX ontology (an example seen in Figure 14, whereby the possible relationships of the objects are enforced via the available dropdown menu options.

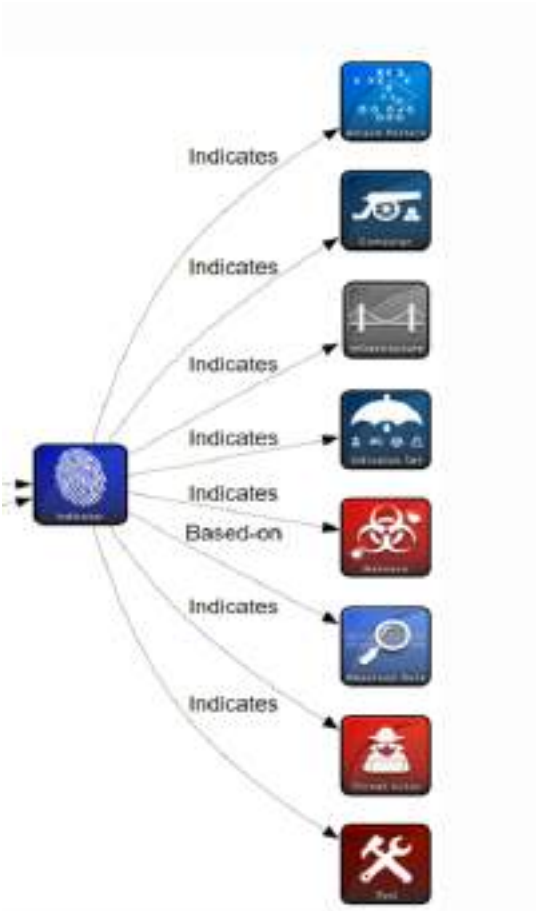

Figure 14. The relationships of the Indicator STIX object.

Multiple conditions can be added in a CTI criterion. The source Object of each condition is forced to be the same across all conditions.
The input of each condition can originate from our platform (information that is specific to our own organisation), or custom input can be used. More specifically, the following are supported:

- **Input from Assurance Model**: Could be the location of our organisation, the sector our organisation works on, or even more specific information like Assets used and vulnerabilities found in our systems.
- **Custom input**: Anything that STIX supports, that is any of the 18 SDOs (Stix Domain Objects). E.g., Information about a specific campaign, intrusions set, malware etc.

After the specification creation has been completed, the criterion is ready to be used. One sample criterion could be:

*Find all **Malware** that **Target** our **Identity** (which translates to our sector)*
AND
***Use** our **Tools** (which translate to our Assets)*

A screenshot of the front end used to define this criterion appears in Figure 15.

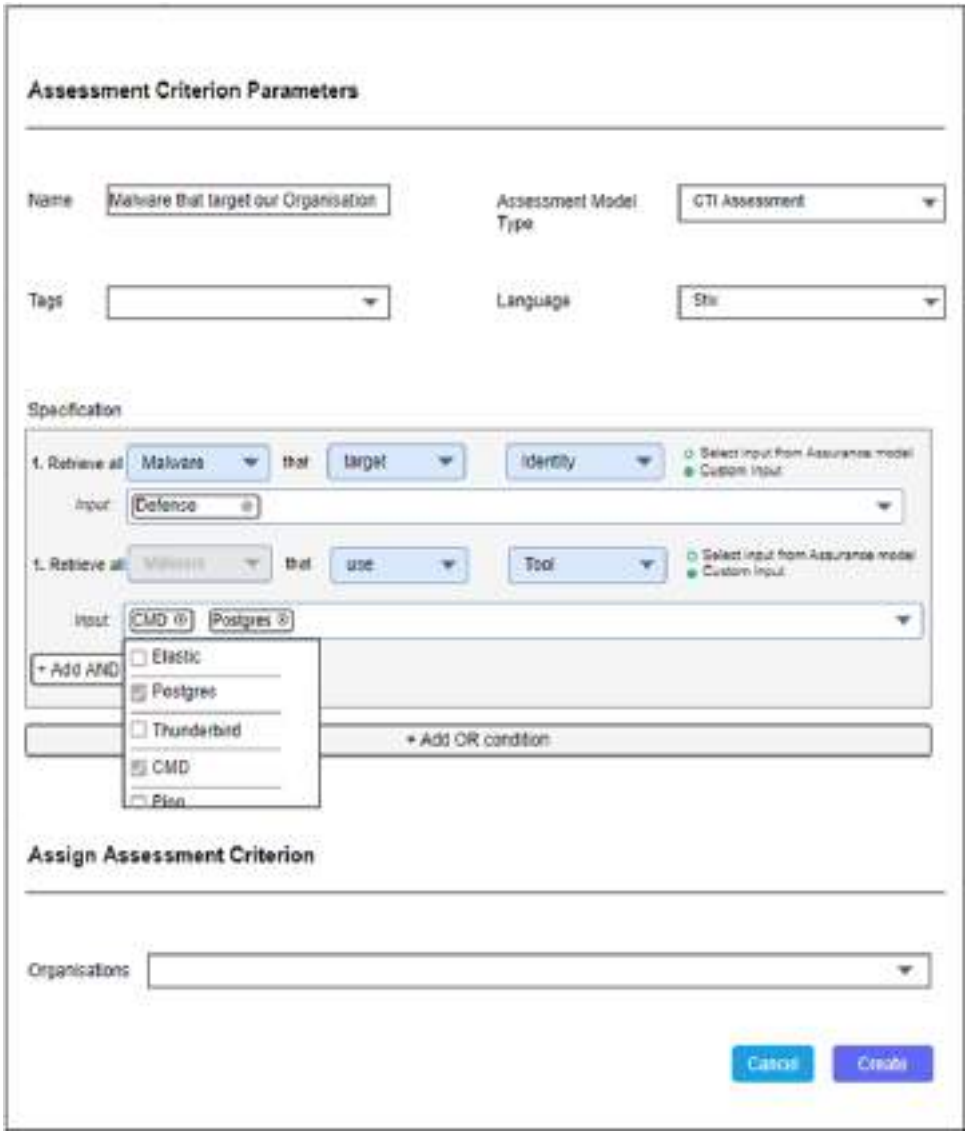

Figure 15. Creation of a CTI Assessment Criterion.

Multiple criterions can be used in a CTI profile. The total results will consist of the Union of every CTI criterion that took part in the assessment.

In addition to the notions of AND, OR, UNION, a specification can contain concatenated queries, e.g.: *Find all Indicators that indicate malware that target our Assets.* In that case, the output of the first query (Indicators that Indicate Malware) is used as input to the second query (Malware that Target Assets).

The specification is created using dropdowns for better user experience and to reduce mistakes. However, in the background, the dropdowns are translated into plain text. Therefore, it is possible to create criteria by using text format directly, e.g., the specification above would be translated into Figure 16. As can be seen, it follows the same logic as the dropdown specification.

Figure 16. Plaintext-based specification of a CTI Assessment Criterion option.

The Assessment results will contain the most important information of each SDO, and will be viewed in the Assessments Page, along with every other assessment result of the other tools. As it can be seen in Figure 17, information like the name, entity type and confidence are displayed, as they are crucial. There are more properties, like OpenCTI URL, which redirects to the OpenCTI local instance, where the data is stored.

Figure 17. CTI Assessment Results & their properties, along with a link to corresponding OpenCTI page.

In terms of **Hybrid Assessments** (i.e., the most complex assessment type), a purpose-defined language is developed by SPHYNX – CRISES - to support such Hybrid Assessments that will combine the results from other Assessments (as the ones shown above) based on criteria defined by the user.

In more detail, CRISES is a pipelined, query-type language that allows for the definition of arbitrary risk models for the propagation of assessments produced by SPA assessments. CRISES also allows the execution of a hybrid analysis, i.e., an analysis that combines two or more of SPA's assessments.

CRISES can be as simple as:

```
FETCH y1:Assessment_Model_Execution, y2:Assessment_Model_Execution,
y3:Assessment_Model_Execution, x:Asset, z:Asset, ar1:Monitoring_Assessment_Result,
ar2: CTI_Assessment_Result, ar3:Monitoring_Assessment_Result

SUCH THAT (y1.INVOLVES=x AND y2.INVOLVES=x AND y1.PRODUCES=ar1 AND y2.PRODUCES=ar2
AND y3.INVOLVES=z AND y3.PRODUCES=ar3)

FILTER (x.ASSET_TYPE=DATA AND z.ASSET_TYPE=PROCESS, ar1.timestamp IN
(CURRENT_TIMESTAMP - 365,CURRENT_TIMESTAMP) AND
x.CATEGORY=authentication/authorization AND z.DEPENDS=x)

GENERATE

ASSESSMENT_RESULT nar.assessment_Result

ASSET nar.asset IN (z)

SECURITY_PROPERTY (nar.Security_Property = ar3.Security_Property)

LIKELIHOOD (nar.likelihood = MAX((ARO(ar2)),(COUNT(ar1.likelihood=1)/COUNT(ar1))))
```

```
VALUE (nar.value = FORMULA(ar3.variable(downtime)*z.value))
```

As you may observe, in CRISES each line of the query is a transformation of the previous line's result. Further, CRISES is designed considering the following principles:

- *Pipelined*: A CRISES query is a linean pipeline of transformations.
- *Extensible*: CRISES is based on SPHYNX's Assurance Model (see the PHOENI2X Asssurance Model noted above). As the model evolves, CRISES will be extended to support the definition/utilisation of the newly introduced components (e.g., new assets or assessments).
- *Analytical*: CRISES queries are analytical, as they emphasize data transformations, and speed.

Therefore, through the SPA front-end, CRISES (hybrid) assessments will be able to be defined and executed.

Finally, a novel aspect in the assessment models, in addition to models developed to enable assessments & certifications according to specific schemes (e.g., ISO security standards, GDPR, NIS Directive), is the specification of an assessment model covering the requirements specified within EC's "Ethics Guidelines for Trustworthy Artificial Intelligence" [79], producing a novel Trustworthy AI Assessment Model against which the AI enablers of the platform can be evaluated.
Citing the above reference, the following requirements have been defined (presented below in verbatim):

- ***Human agency and oversight:*** *AI systems should empower human beings, allowing them to make informed decisions and fostering their fundamental rights. At the same time, proper oversight mechanisms need to be ensured, which can be achieved through human-in-the-loop, human-on-the-loop, and human-in-command approaches.*
- ***Technical Robustness and safety:*** *AI systems need to be resilient and secure. They need to be safe, ensuring a fallback plan in case something goes wrong, as well as being accurate, reliable and reproducible. That is the only way to ensure that also unintentional harm can be minimized and prevented.*
- ***Privacy and data governance:*** *besides ensuring full respect for privacy and data protection, adequate data governance mechanisms must also be ensured, taking into account the quality and integrity of the data, and ensuring legitimised access to data.*
- ***Transparency:*** *the data, system and AI business models should be transparent. Traceability mechanisms can help achieving this. Moreover, AI systems and their decisions should be explained in a manner adapted to the stakeholder concerned. Humans need to be aware that they are interacting with an AI system and must be informed of the system's capabilities and limitations.*
- ***Diversity, non-discrimination and fairness:*** *Unfair bias must be avoided, as it could have multiple negative implications, from the marginalization of vulnerable groups to the exacerbation of prejudice and discrimination. Fostering diversity, AI systems should be accessible to all, regardless of any disability, and involve relevant stakeholders throughout their entire life circle.*
- ***Societal and environmental well-being:*** *AI systems should benefit all human beings, including future generations. It must hence be ensured that they are sustainable and environmentally friendly. Moreover, they should take into account the environment, including other living beings, and their social and societal impact should be carefully considered.*
- ***Accountability:*** *Mechanisms should be put in place to ensure responsibility and accountability for AI systems and their outcomes. Auditability, which enables the assessment of algorithms, data and design processes plays a key role therein, especially in critical applications. Moreover, adequate an accessible redress should be ensured.*

Considering the need to derive an assessment model from the above, of particular importance is the Assessment List for Trustworthy AI (ALTAI), which was developed by the "High-Level Expert Group on Artificial Intelligence" [89] set up by the European Commission.

In fact, ALTAI's objective is to specifically help assess whether any AI system that is being developed, deployed, procured or used, complies with the 7 principles noted above. To achieve this, the AI High-Level Expert Group on Artificial Intelligence translated these requirements into a detailed Assessment List, taking into account feedback from a six-month long piloting process within the European AI community.

The ALTAI prototype is available as a website [90] (see screenshot in Figure 18), and the parameters (questionnaires, check lists, etc.) encoded within that prototype will guide the definition of the Trustworthy AI assessment model defined within PHOENI2X.

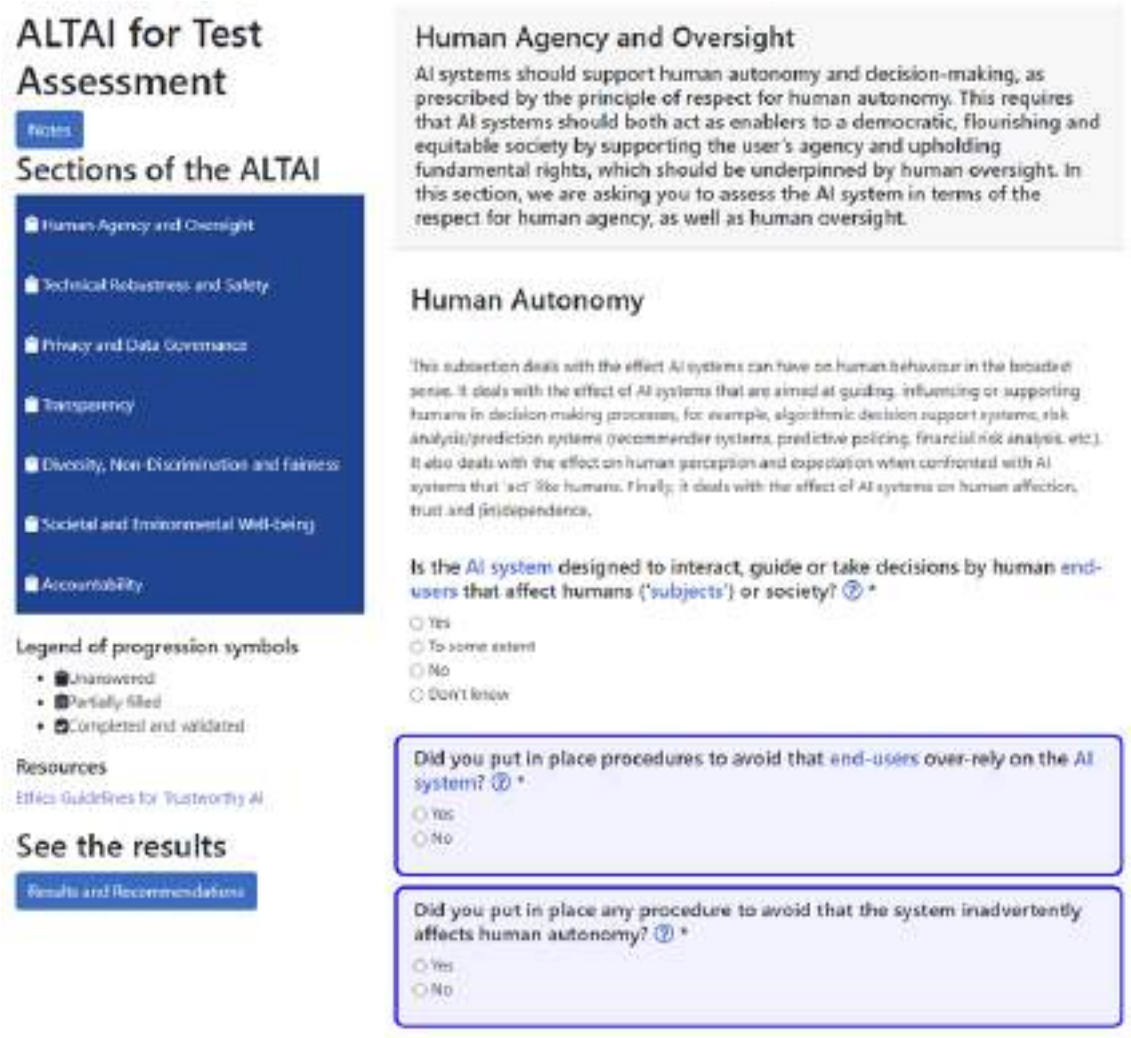

Figure 18. Part of an ALTAI Trustworthy AI Assessment.

5 Conclusion and Next Steps

1 introduces the purpose of this document and provides with an overview of the overall methodology employed in the preparation of this deliverable. It has highlighted the steps involved in selecting the appropriate baseline tools, considering the use cases, and preparing the infrastructure for integrating the selected tools into the PHOENI2X project. The described methodology ensures that the document aligns with the objectives and requirements of the project while leveraging existing tools and infrastructure to facilitate the development and evaluation of the PHOENI2X framework.

In 2, the baseline tools for the PHOENI2X platform have been established. These tools enable the rest of the PHOENI2X components to obtain information regarding the infrastructure. First, a brief description of each of the categories is given, indicating the functionalities of the tools that belong to it. Subsequently, multiple open-source tools within each category are provided, stating the capabilities of each one. Then, one or multiple tools are chosen for each category, the selection is made so the tools are representative of the category to which they belong, also taking into account the existing tools in each of the use cases. More details on the deployment and interconnection of these tools will be provided in deliverable 5.2.

Furthermore, this document addresses in 3 the technical aspects of the infrastructure for the deployment of the tools in section “Network & Compute Infrastructure”. Depending on the use case, the deployment of PHOENI2X varies slightly. The goal is to demonstrate the platform’s flexibility by deploying it as a service, on-premises, or in a hybrid setup. In addition, each use case encompasses distinct tools; the intention behind this is to underscore that PHOENI2X does not necessitate the installation of a predefined set of specific tools for use as a baseline, although the ones mentioned in this deliverable are representative of each category. This holds significant relevance for infrastructures aiming to leverage the functionalities of PHOENI2X while employing alternate tools for monitoring their infrastructure.

Lastly, the need for a “Security Assurance & Certification” platform is elaborated and the detailed design of this functionality is presented in 4.

This deliverable primarily sets the groundwork for integrating the different components of PHOENI2X and paves the way for the development of specific functionalities within each of these components.

6 References

[1] G. N. GmbH, "OpenVAS Scanner," [Online]. Available: https://github.com/greenbone/openvas-scanner. [Accessed 17 07 2023].

[2] PortSwigger Ltd., "What do you want to do with Burp Suite?," [Online]. Available: https://portswigger.net/burp. [Accessed 17 07 2023].

[3] amoldp, "Grabber-Security-and-Vulnerability-Analysis," [Online]. Available: https://github.com/amoldp/Grabber-Security-and-Vulnerability-Analysis-. [Accessed 17 07 2023].

[4] N. SURRIBAS, "Wapiti; The web-application vulnerability scanner," [Online]. Available: https://wapiti-scanner.github.io/. [Accessed 17 07 2023].

[5] sullo, "Nikto," [Online]. Available: https://github.com/sullo/nikto. [Accessed 17 07 2023].

[6] Sarosys LLC , "Arachni - Web Application Security Scanner Framework," [Online]. Available: https://github.com/Arachni/arachni. [Accessed 17 07 2023].

[7] A. Riancho, "w3af - Web Application Attack and Audit Framework," [Online]. Available: https://github.com/andresriancho/w3af. [Accessed 17 07 2023].

[8] B. Damele A. G. and M. Stampar, "sqlmap," [Online]. Available: https://sqlmap.org/. [Accessed 17 07 2023].

[9] X. Mendez, "Wfuzz - The Web Fuzzer," [Online]. Available: https://github.com/xmendez/wfuzz. [Accessed 17 07 2023].

[10] The ZAP Dev Team, "Zed Attack Proxy (ZAP)," [Online]. Available: https://www.zaproxy.org/. [Accessed 17 07 2023].

[11] Google, "reaver-wps," [Online]. Available: https://code.google.com/archive/p/reaver-wps/. [Accessed 17 07 2023].

[12] nbrito, "T50: an Experimental Packet Injector Tool," [Online]. Available: https://github.com/nbrito/source/tree/master/c/t50. [Accessed 17 07 2023].

[13] aircrak-ng, "Aircrack-ng," [Online]. Available: https://github.com/aircrack-ng/aircrack-ng. [Accessed 17 07 2023].

[14] Digital Security, "BtleJuice Framework," [Online]. Available: https://github.com/DigitalSecurity/btlejuice. [Accessed 17 07 2023].

[15] shadowcave, "SpoofTooph," [Online]. Available: https://sourceforge.net/projects/spooftooph/. [Accessed 17 07 2023].

[16] Google, "nogotofail," [Online]. Available: https://github.com/google/nogotofail. [Accessed 17 07 2023].

[17] rapid7, "Metasploit," [Online]. Available: https://github.com/rapid7/metasploit-framework. [Accessed 17 07 2023].

[18] Veil-Framework, "Veil," [Online]. Available: https://github.com/Veil-Framework/Veil. [Accessed 17 07 2023].

[19] nullsecurity.net, "Hyperion," [Online]. Available: https://github.com/nullsecuritynet/tools/tree/master/binary/hyperion. [Accessed 17 07 2023].

[20] oddcod3, "Phantom Evasion 3.0," [Online]. Available: https://github.com/oddcod3/Phantom-Evasion. [Accessed 17 07 2023].

[21] TrustedSec, "The Social-Engineer Toolkit (SET)," [Online]. Available: https://github.com/trustedsec/social-engineer-toolkit. [Accessed 17 07 2023].

[22] RSM US LLP, "King Phisher," [Online]. Available: https://github.com/rsmusllp/king-phisher. [Accessed 17 07 2023].

[23] Sonar, "SonarQube," [Online]. Available: https://github.com/SonarSource/sonarqube. [Accessed 17 07 2023].

[24] Source Forge, "Cpp check - A tool for static C/C++ code analysis," [Online]. Available: https://cppcheck.sourceforge.io/. [Accessed 17 07 2023].

[25] jeremylong, "Dependency-Check," [Online]. Available: https://github.com/jeremylong/DependencyCheck. [Accessed 17 07 2023].

[26] R. Hull, "nvd-clojure," [Online]. Available: https://github.com/rm-hull/nvd-clojure. [Accessed 17 07 2023].

[27] AT&T Business, "AlienVault OSSIM," [Online]. Available: https://cybersecurity.att.com/products/ossim. [Accessed 21 07 2023].

[28] Elasticsearch B.V., "Elasticsearch, Logstash, and Kibana (ELK)," [Online]. Available: https://www.elastic.co/. [Accessed 21 07 2023].

[29] J.-P. Lang, "Prelude-SIEM," [Online]. Available: https://www.prelude-siem.org/. [Accessed 21 07 2023].

[30] H. Debar, D. Curry and B. Feinstein, March 2007. [Online]. Available: https://www.ietf.org/rfc/rfc4765.txt. [Accessed 21 07 2023].

[31] SIEMonster, "SIEMonster," [Online]. Available: https://siemonster.com/. [Accessed 21 07 2023].

[32] Wazuh Inc, "Wazuh - The Open Source Security Platform," [Online]. Available: https://wazuh.com/. [Accessed 21 07 2023].

[33] OSSEC PROJECT TEAM, "OSSEC," [Online]. Available: https://www.ossec.net/. [Accessed 21 07 2023].

[34] MISP project, "MISP - Malware Sharing Information Platform," [Online]. Available: https://www.misp-project.org/. [Accessed 21 07 2023].

[35] Filigran, "OpenCTI - Open Threat Intelligence Platform," [Online]. Available: https://filigran.io/solutions/products/opencti-threat-intelligence/. [Accessed 21 07 2023].

[36] Palo Alto Networks, "What is MineMeld?," [Online]. Available: https://github.com/PaloAltoNetworks/minemeld/wiki. [Accessed 21 07 2023].

[37] SOCRadar, "SOCRadar - Experience the edges of Extended Threat Intelligence," [Online]. Available: https://socradar.io/free-edition/. [Accessed 21 07 2023].

[38] IntelOwl Project Org, "Intel Owl," [Online]. Available: https://github.com/intelowlproject/IntelOwl/. [Accessed 21 07 2023].

[39] TheHive Project, "Cortex," [Online]. Available: https://github.com/TheHive-Project/Cortex. [Accessed 21 07 2023].

[40] Hurricane Labs LCC, "Machinae Security Intelligence Collector," [Online]. Available: https://machinae.hurricanelabs.com/. [Accessed 21 07 2023].

[41] Thomas Chopitea et al., "YETI - Your Every Threat Intelligence," [Online]. Available: https://yeti-platform.github.io/. [Accessed 21 07 2023].

[42] A. Chuvakin, "On Endpoint Sensing," 2023. [Online]. Available: https://www.gartner.com/en/insights. [Accessed 3 08 2023].

[43] Project License; The Linux Foundation Projects, "Osquery - Performant endpoint visibility," [Online]. Available: https://www.osquery.io/. [Accessed 3 08 2023].

[44] OPENEDR - PART OF THE © XCITIUM, LLC CYBERSECURITY COMPANY, "OPENEDR - What is Open Source Endpoint Detection and Response (EDR)?," [Online]. Available: https://www.openedr.com/. [Accessed 3 08 2023].

[45] Electric Sheep Fencing, LCC, "pdfsense," [Online]. Available: https://www.pfsense.org/. [Accessed 27 06 2023].

[46] Arista Networks, Inc., "NG Firewall," [Online]. Available: https://edge.arista.com/ng-firewall/. [Accessed 27 06 2023].

[47] Cisco and/or its affiliates, "Snort - What is Snort?," [Online]. Available: https://www.snort.org/. [Accessed 27 06 2023].

[48] OISF, "Suricata - Observer. Protect. Adapt.," [Online]. Available: https://suricata.io/. [Accessed 27 06 2023].

[49] The Zeek Project, "Zeek - An Open Source network Security Monitoring Tool," [Online]. Available: https://zeek.org/. [Accessed 27 06 2023].

[50] Arkime, "Arkime - Full Packet Capture," [Online]. Available: https://arkime.com/. [Accessed 27 06 2023].

[51] Telecommunication Networks Group - Politecnico di Torino, "Tstat - TCP STatistic and Analysis Tool," [Online]. Available: http://tstat.polito.it/. [Accessed 27 06 2023].

[52] Cisco, "Netflow," [Online]. Available: https://www.cisco.com/c/en/us/tech/quality-of-service-qos/netflow/index.html. [Accessed 27 06 2023].

[53] The Tcpdump Group, "TCPDUMP," [Online]. Available: https://www.tcpdump.org/. [Accessed 27 06 2023].

[54] A. Habibi Lashkari, "CICFlowmeter," [Online]. Available: https://github.com/ahlashkari/CICFlowMeter. [Accessed 27 06 2023].

[55] Sysdig, "Wireshark," [Online]. Available: https://www.wireshark.org/. [Accessed 27 06 2023].

[56] Cowrie Project, "Cowrie," [Online]. Available: https://github.com/cowrie/cowrie. [Accessed 31 08 2023].

[57] T. Upi, "Kippo," [Online]. Available: https://github.com/desaster/kippo. [Accessed 31 08 2023].

[58] DinoTools, "dionaea - catches bugs," [Online]. Available: https://github.com/DinoTools/dionaea. [Accessed 31 08 2023].

[59] MushMush, "SNARE," [Online]. Available: https://github.com/mushorg/snare. [Accessed 31 08 2023].

[60] MushMush, "TANNER," [Online]. Available: https://github.com/mushorg/tanner. [Accessed 31 08 2023].

[61] MushMush, "Glastopf," [Online]. Available: https://github.com/mushorg/glastopf. [Accessed 31 08 2023].

[62] MushMush, "Conpot," [Online]. Available: https://github.com/mushorg/conpot. [Accessed 31 08 2023].

[63] A. Dell'Aera, "Thug," [Online]. Available: https://github.com/mushorg/conpot. [Accessed 31 08 2023].

[64] V. Bontchev, "ElasticPot: an Elasticsearch Honeypot," [Online]. Available: https://gitlab.com/bontchev/elasticpot. [Accessed 31 08 2023].

[65] Thinkst Applied Research, "CanaryTokens," [Online]. Available: https://github.com/thinkst/canarytokens. [Accessed 31 08 2023].

[66] "T-Pot - The All In One Multi Honeypot Platform," [Online]. Available: https://github.com/telekom-security/tpotce. [Accessed 31 08 2023].

[67] Pwnlandia, "Modern Honey Network," [Online]. Available: https://github.com/pwnlandia/mhn. [Accessed 31 08 2023].

[68] Fireeye, "Redline," [Online]. Available: https://fireeye.market/apps/211364. [Accessed 19 07 2023].

[69] Google, "GRR Rapid Response," [Online]. Available: https://github.com/google/grr. [Accessed 19 07 2023].

[70] Rekall Forensics, "Rekall," [Online]. Available: http://www.rekall-forensic.com/. [Accessed 19 07 2023].

[71] Volatility Foundation, "Volatility Framework - Volatile memory extraction utility framework," [Online]. Available: https://github.com/volatilityfoundation/volatility. [Accessed 19 07 2023].

[72] M. Korman, "Volatility Bot," [Online]. Available: https://github.com/mkorman90/VolatilityBot. [Accessed 19 07 2023].

[73] JPCERT Coordination Center, "MalConfScan," [Online]. Available: https://github.com/JPCERTCC/MalConfScan. [Accessed 19 07 2023].

[74] AEGIS IT Research, "Forensics Visualization Toolkit (FVT)," [Online]. Available: https://aegisresearch.eu/solutions/forensics-visualization-toolkit-fvt/. [Accessed 19 07 2023].

[75] Apache Software Foundation, "Apache Kafka," [Online]. Available: https://kafka.apache.org/. [Accessed 21 07 2023].

[76] The ZeroMQ authors, "ZeroMQ - An open-source universal messaging library," [Online]. Available: https://zeromq.org/. [Accessed 21 07 2023].

[77] VMWare Inc, "RabbitMQ," [Online]. Available: https://www.rabbitmq.com/. [Accessed 21 07 2023].

[78] The Apache Software Foundation, "ActiveMQ - Flexible & Powerful Open Source Multi-Protocol Messaging," [Online]. Available: https://activemq.apache.org/. [Accessed 21 07 2023].

[79] European Commission, "Ethics guidelines for trustworthy AI," [Online]. Available: https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai. [Accessed 18 08 2023].

[80] Sphynx Technology Solutions, "The SPHYNX SPA Suite," [Online]. Available: https://www.sphynx.ch/products/. [Accessed 18 08 2023].

[81] NIST, "CPE," [Online]. Available: https://nvd.nist.gov/products/cpe. [Accessed 18 08 2023].

[82] The Mitre Corporation, "CVE," [Online]. Available: https://cve.mitre.org/. [Accessed 18 08 2023].

[83] NIST, "National Vulnerability Database," [Online]. Available: https://nvd.nist.gov/. [Accessed 18 08 2023].

[84] NIST, "National Institute of Standards and Technology," [Online]. Available: https://www.nist.gov/. [Accessed 18 08 2023].

[85] T. Tsigkritis, G. Spanoudakis, C. Kloukinas and D. Lorenzoli, "Diagnosis and Threat Detection Capabilities of the SERENITY Monitoring Framework. Security and dependability for ambient intelligence," pp. 239-271, 2009.

[86] D. Lorenzoli and G. Spanoudakis, "EVEREST + run-time SLA violations prediction," *In Proceedings of the 5th International Workshop on Middleware for Service Oriented Computing,* pp. 13-18, November 2010.

[87] K. Mahbub, G. Spanoudakis and T. Tsigkritis, "Translation of SLAs into Monitoring Specifications," *Wieder, P., Butler, J., Theilmann, W., Yahyapour, R. (eds) Service Level Agreements for Cloud Computing,* 2011.

[88] FIRST, "Common Vulnerability Scoring System v3.1: Specification Document," [Online]. Available: https://www.first.org/cvss/v3.1/specification-document. [Accessed 18 08 2023].

[89] European Commission, "High-level expert group on artificial intelligence," [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/expert-group-ai. [Accessed 18 08 2023].

[90] European Commission, "Welcome to the ALTAI portal!," [Online]. Available: https://futurium.ec.europa.eu/en/european-ai-alliance/pages/welcome-altai-portal. [Accessed 18 08 2023].

[91] J. Ermerins, N. van Noort, J. de Novais Marques and L. Velasco, Scoring model for IoCs by combining, Amsterdam, 2020.

[92] GmbH, Greenbone Networks, "OpenVAS Scanner," [Online]. Available: https://github.com/greenbone/openvas-scanner. [Accessed 17 07 2023].