D2.3

The HEIR facilitators package: Final version

Project number	883275
Project acronym	HEIR
Project title	A secure Healthcare Environment for Informatics Resilience
Start date of the project	September 1st, 2020
Duration	36 months
Programme	H2020-SU-DS-2019

Deliverable type	Demonstrator
Deliverable reference no.	D2.3
Workpackage	WP2
Due date	**31/10/2022 [M26]**
Actual submission date	28/02/2023

Deliverable lead	IBM
Editors	Eliot Salant (IBM)
Contributors	Eliot Salant (IBM), Bogdan Prelipcean (BD), Andreas Alexopoulos, Miltiadis Kokkonidis, Aris Sotiropoulos, Chronis Ballas (AEGIS), Marwan Darwish Khabbaz (TUD), Apostolis Zarras, Eftychia Lakka (FORTH)
Reviewers	Hervé DEBAR (IMT), Iulia Ilie (SIE)
Dissemination level	PU
Revision	Final / 1.2
Keywords	Threat Hunting, Vulnerability Assessment, Forensics Analysis, Privacy Aware, Prototype

Abstract

Deliverable D2.3 serves as the report which describes the final version of the HEIR facilitators. The content is illustrating the development of the demonstrator based on the activities carried out within the context of WP2 and the relations with the other technical WPs, namely WP3 and WP4.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 883275

Executive Summary

This deliverable presents the work that has been carried out towards the final delivery of the HEIR facilitators package, the intelligent threat monitoring and hunting module providing treat detection as a service, and the blockchain-based collaborative privacy aware framework providing sensitive data trustworthiness sharing. The development of the 1st complete prototype was demonstrated in the first project review.

The HEIR facilitators packaged for the final version includes (i) the Vulnerability Assessment module; (ii) the HEIR Interactive Forensics Module; (iii) the Anomaly Detection and Threats Classification module and (iv) the advancements of the Privacy Aware Framework and the Auditing Mechanisms.

The current development status of these facilitators is presented in this document.

Table of Contents

EXECUTIVE SUMMARY **2**

1. INTRODUCTION **5**

1.1 SCOPE AND OBJECTIVES 5
1.2 RELATION TO OTHER TASKS AND WORK PACKAGES 5

2. THE HEIR FACILITATORS PACKAGES **6**

2.1 ARCHITECTURE OVERVIEW 6
2.2 COMPONENT DESCRIPTION 6

2.2.1 Vulnerability assessment Module *6*

2.2.2 SIEM - Threat Detection *9*

2.2.3 HEIR Interactive Forensics Module *10*

2.2.4 ML anomaly detection and threat classification *16*

2.2.5 HEIR Collaborative privacy-aware framework (PAF) – task 2.4 ..... *22*

2.3 HEIR FACILITATORS IN THE USE CASES 27

3. CONCLUSION **29**

List of Figures

FIGURE 1: FACILITATOR ARCHITECTURE 6
FIGURE 2: VULNERABILITY ASSESSMENT MODULE - DATA FLOW 7
FIGURE 3: HEIR AGENT OVERVIEW 9
FIGURE 4: FVT - OVERVIEW & DEVICES 11
FIGURE 5: CONNECTED DEVICES 11
FIGURE 6: ANALYSIS DASHBOARD 12
FIGURE 7: FVT TIMELINE WIDGET 13
FIGURE 8: FVT - DETAILS WIDGET 13
FIGURE 9: FVT - PERFORMANCE METRICS CHART WIDGETS 14
FIGURE 10: FVT - EVENTS ANALYSIS 15
FIGURE 11: FVT - INSPECT DEVICE 16
FIGURE 12: LSTM TEST ACCURACY / LOSS 20
FIGURE 13: HIGH-LEVEL IMPLEMENTATION 22
FIGURE 14: REGO RULE FOR JOIN ON CONSENT RESOURCES 24
FIGURE 15: SEQUENCE DIAGRAM OF JOIN OPERATION 25
FIGURE 16: THE ASSET YAML FOR A FHIR OBSERVATION 26
FIGURE 17: GUI TO TOGGLE PII STATUS OF FHIR OBSERVATION FIELDS 27

List of Abbreviations

FVT	Forensics Visualization Toolkit (AEGIS)
HCC	HEIR Cryptographic Checker
HCG	HEIR Client GUI
HET	HEIR Exploit Tester
HNM	HEIR Network Module
IOCs	Indicators of Compromise
Kafka	Apache Kafka
ML	Machine Learning
MVP	Minimum Viable Product
RAMA	Risk Assessment for Medical Applications
SIEM	Security Information and Event Management

1. Introduction

1.1 Scope and objectives

This document reports the work carried out during the developmental cycle between M18-M30.

For each component, the involved partners provide the research challenges and advancements achieved for the final version of the Facilitators package, (M12-M18).

1.2 Relation to other Tasks and Work Packages

This deliverable is the outcome of the works performed by all Tasks of WP2, and builds upon the deliverable D2.2, the MVP version of the HEIR facilitators package.

Moreover, there is a close interrelationship between this deliverable and the WP3 deliverables. More specifically, this deliverable is tightly connected to the "D3.2 - The HEIR 1st layer of services package: 1st complete version" and D3.3 deliverables, as the 1st layer of services (Client services) use the results computed by the facilitators' package.

2. The HEIR Facilitators Packages

2.1 Architecture Overview

Erreur ! Source du renvoi introuvable. illustrates a high-level overview of the architecture o f Facilitators components. In the following subsections, the components will be described in more details.

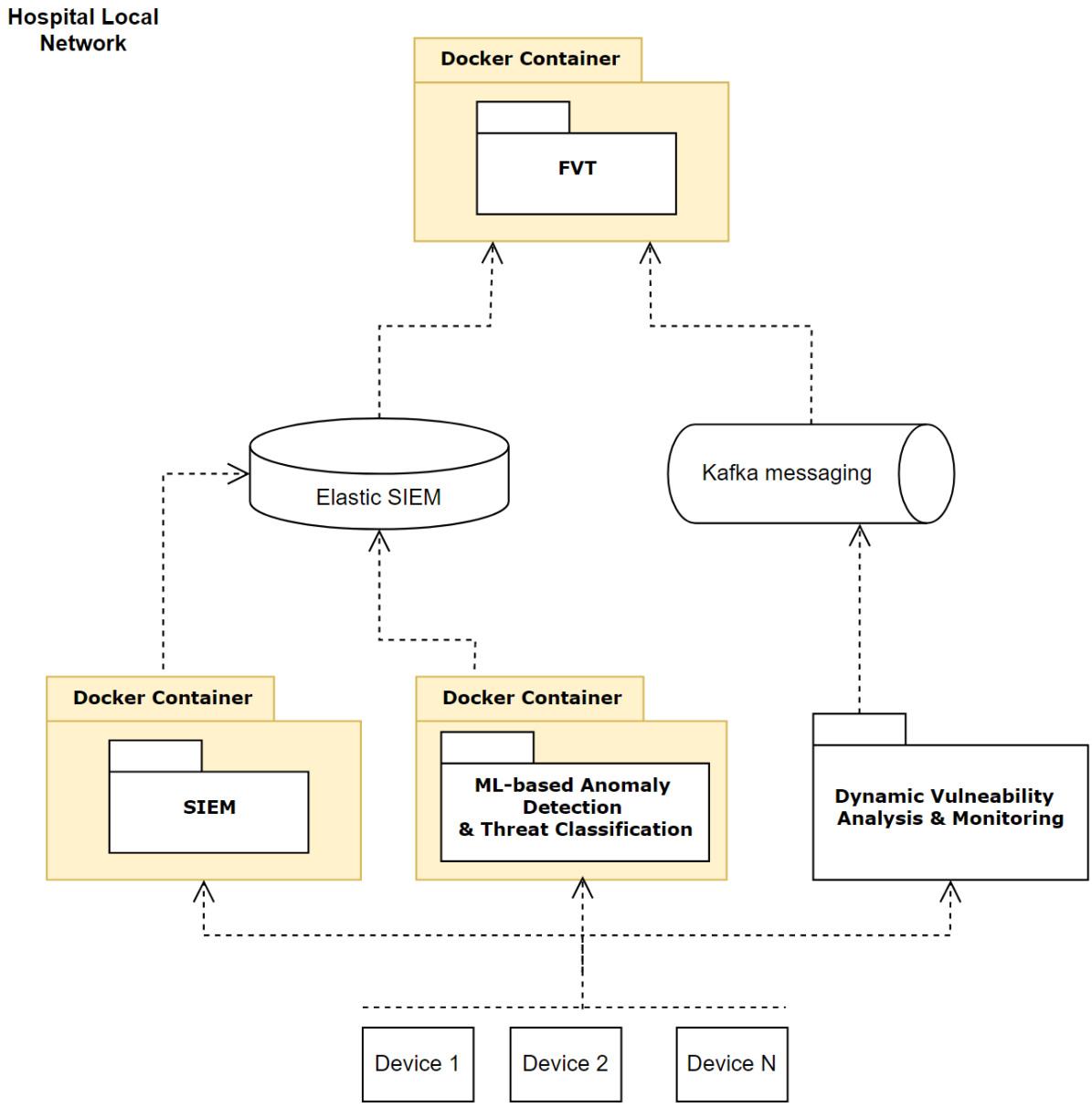

Figure 1: Facilitator architecture

2.2 Component description

2.2.1 Vulnerability assessment Module

The Dynamic Vulnerability Assessment and Monitoring module (also denoted as the Vulnerability Assessment module) exists within the context of the HEIR Agent. The HEIR

Agent is the software tool deployed at an endpoint level that manages and collects the data necessary for further analysis.

For the final version of the facilitator package, the HEIR Agent was updated making room for further integration for any other modules at the endpoint level. In this case we consider any scanning scenario (files, installed programs, running processes). Also, an update mechanism was put in place such that the integrated modules get the necessary updates.

We recall the functionality of the Vulnerability Assessment module. It extracts information about the operating system configurations and application information. In case that these points of interest are not properly configured, or the applications are outdated they can pose a security risk for the endpoint and after that to the entire medical environment. In Figure 2, we have an overview of the data flow. For the final version the flow remained the same; no interventions over the mechanism were necessary.

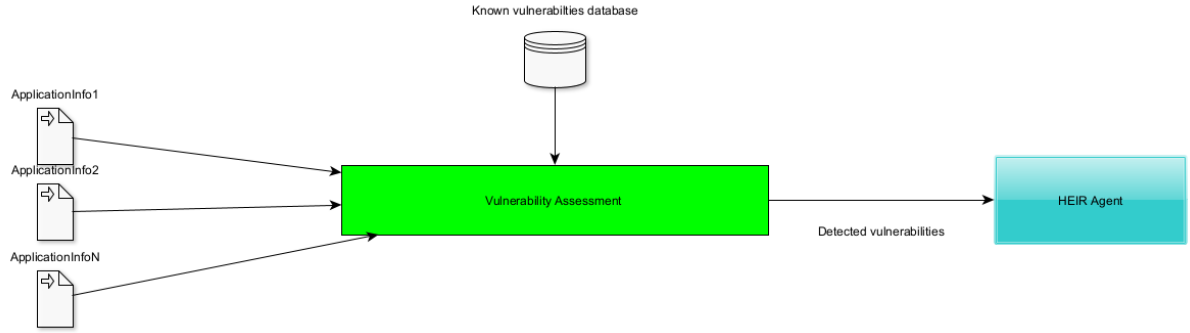

Figure 2: Vulnerability Assessment Module - Data flow

The collected information is:

- Application Name
- Version

The full results from this module are collected by the Local Correlation Component of the HEIR Agent and then forwarded to the HEIR Client by the message broker (Kafka in this case). The component has a database with known CVE[1] that are matched against the existing information on the installed applications (name and version). The output of the module will contain:

- the CVE number
- the vulnerability description
- the publishing date of the CVE
- severity score

[1] Common Vulnerabilities and Exposures

An example of such output is as follows:


```
            "application_name": "Docker Desktop",
            "cves": [
                {
                    "cve": "CVE-2018-10892",
                    "description": "The default OCI linux spec in oci/defaults{_linux}.go in Docker/Moby from 1.11 to current does not block /proc/acpi pathnames. The flaw allows an attacker to modify host's hardware like enabling/disabling bluetooth or turning up/down keyboard brightness.",
                    "publish_date": "2018-07-06T16:29Z",
                    "score": 50
                },
                {
                    "cve": "CVE-2020-11492",
                    "description": "An issue was discovered in Docker Desktop through 2.2.0.5 on Windows. If a local attacker sets up their own named pipe prior to starting Docker with the same name, this attacker can intercept a connection attempt from Docker Service (which runs as SYSTEM), and then impersonate their privileges.",
                    "publish_date": "2020-06-05T14:15Z",
                    "score": 72
                }
            ],
            "version": "2.2.0.4"
        }
```


For the final version, a final iteration over the rules was made. The updated set provides a complete analysis regarding the existing vulnerabilities based on the information that is available. We recall that the database with known vulnerabilities is updated in a real-time manner.

Another important aspect is the usage of HEIR Agent as the backbone for HEIR Client endpoint components as Heir Exploit Tester, Threat Detection Module and the local version of HEIR Network Module.

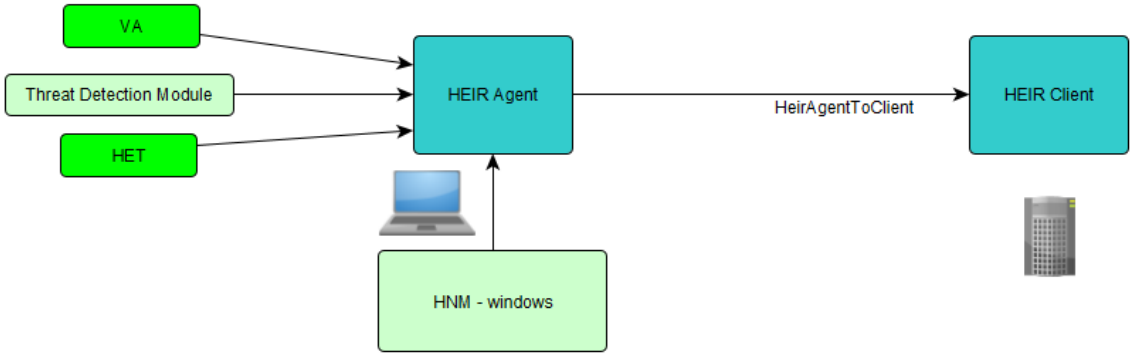

Figure 3: HEIR Agent Overview

2.2.2 SIEM - Threat Detection

The HEIR SIEM component supplies various security related data from all endpoints to the HEIR Interactive Forensics Module, AEGIS' FVT, as well as BD's HEIR client.

The SIEM solution used in HEIR is built around ITML's SAAS product, named Security Infusion[2], which incorporates a plethora of open-source software tools in a unified framework. Regarding the SIEM functionality in the HEIR solution, event data is collected from lightweight agents installed on the endpoints (i.e., workstations, servers etc.)

The solution is based on the Wazuh[3] open-source software which provides a multitude of security related services that continuously monitor an IT infrastructure. The events reported by the Wazuh agents are the outcome of a wide range of tasks such as

- Inventory of running processes and installed applications.
- Log and events data collection
- File and registry keys integrity monitoring
- Monitoring of open ports and network configuration
- Configuration assessment and policy monitoring

These events are received by the Wazuh server and processed through a toolset of decoders and rules, using threat intelligence to look for well-known Indicators of Compromises (IOCs). As a result of this analysis, all events are appointed a severity level enabling the administrators to focus on the crucial issues that need to be addressed.

Subsequently, the SIEM can identify a specific combination or sequence of events as a culprit of a specific threat indication. For example, although individual failed user login events do not necessarily bear a high attack risk, a sequence of multiple failed logins followed by a successful login holds a high probability that i.e., a brute force attack was deemed successful. As a result, in such a case a high severity level event would be produced.

[2] https://security-infusion.com/
[3] https://wazuh.com/

In the 1st iteration of the SIEM component, all the security related events were solely used in feeding the FVT dashboard, enabling a security administrator to recognize potential security incidents. To this end, by collecting data originating from the Security Infusion agent such as the endpoint's resources utilization, it provides the ability of performing a thorough forensic analysis by correlating the reported events with additional computer metrics (i.e., Disk & Memory utilization etc.). However, the real-time automatic reporting of high severity detected event was not yet implemented.

In the 2nd iteration of the SIEM component, an extra Elastic Connector component was deployed which extracts all high severity events from the Wazuh data stored in the Elastic Search and sends them to a Kafka topic which is monitored by the HEIR client, adding one more source of security related events that are used to calculate the RAMA score.

2.2.3 HEIR Interactive Forensics Module

The Forensics Visualization Toolkit (FVT) demonstrates security information for a selected department of the organization and provides users with a timeline-based representation of the security events captured by the SIEM sub-module and processed by ML Anomaly Detection Module (see section **Events Analysis)**. It is accessed through the 1st layer of visualizations and is meant to represent the captured events in a more detailed way. **Authorized** users who belong to the hospital / Healthcare staff groups / domains and have access to the HEIR Client GUI (HCG) can further investigate any of the connected HEIR Clients of the hospital through the FVT.

In the landing page of FVT (Figure 4), users can observe department specific insights derived from the HEIR Agent. (I.e., the outcome of the Exploit Tester, Cryptographic Checker, Vulnerability Assessment and Network module.)

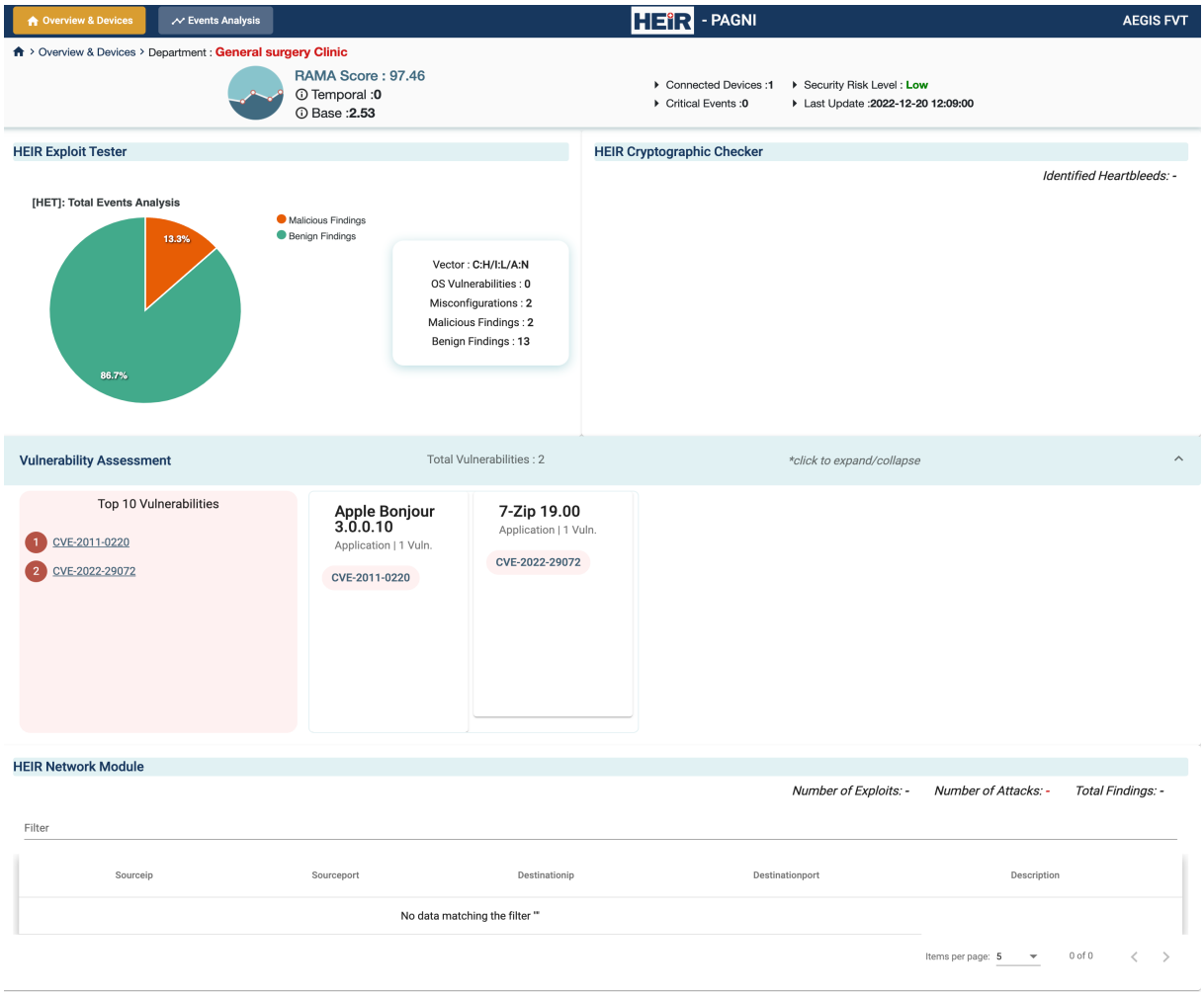

Figure 4: FVT - Overview & Devices

At the bottom of the screen, monitored devices of the selected department are listed. By clicking the "Inspect" button on any device, users will access the main analysis dashboard of FVT (Figure 6).

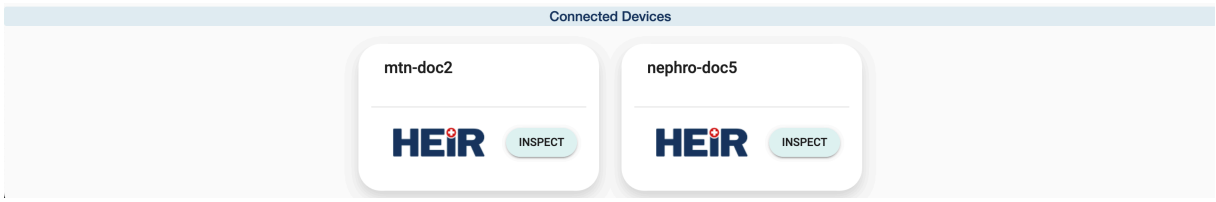

Figure 5: Connected devices

The main analysis dashboard (Figure 6) allows users to choose from a set of widgets containing different types of visualizations, that refer to different system metrics and network related information. The widgets could be standalone and support discrete input sources, but their combination offers a comprehensive depiction and meaningful visualization of the data. The user is also able to request historical data.

For the final version of the facilitators, FVT's main dashboard was enriched with advanced clustering capabilities and data aggregated custom views per widget, based on the volume of the data. A severity accurate filtering functionality is also now available to the users.

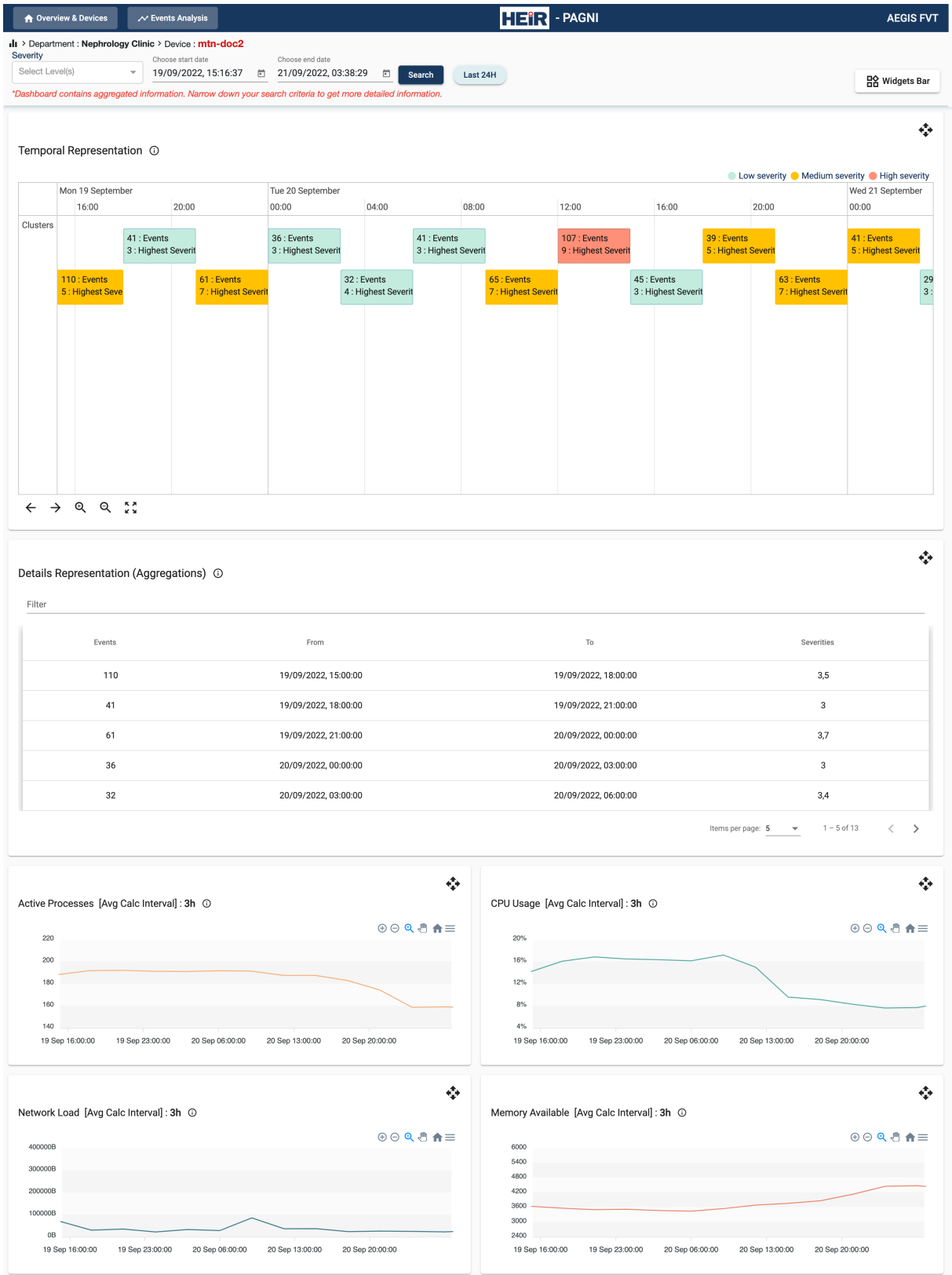

Figure 6: Analysis Dashboard

Temporal representation of incoming logged events captured by the SIEM is available through the Timeline widget (Figure 7). By changing the range period (e.g., zoom interactions) in the timeline all the available widgets will be timewise synchronized and updated accordingly. For the final version, the timeline widget was enhanced with resizing options, ability to add

timewise annotations to the rest of the chart widgets (Figure 8Figure 9) (double click on any detected event in time) and adaptive visualization of the events based on the amount of data.

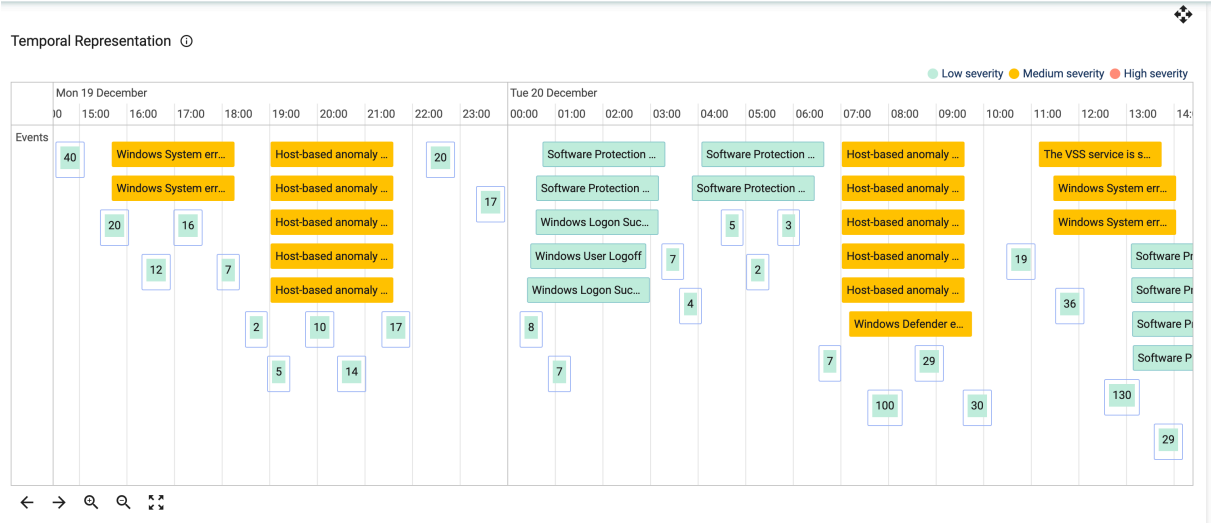

Figure 7: FVT Timeline widget

Detailed information about the incoming events will be presented in the Details widget (Figure 8). This widget also supports a standalone text filtering capability to search through the available information.

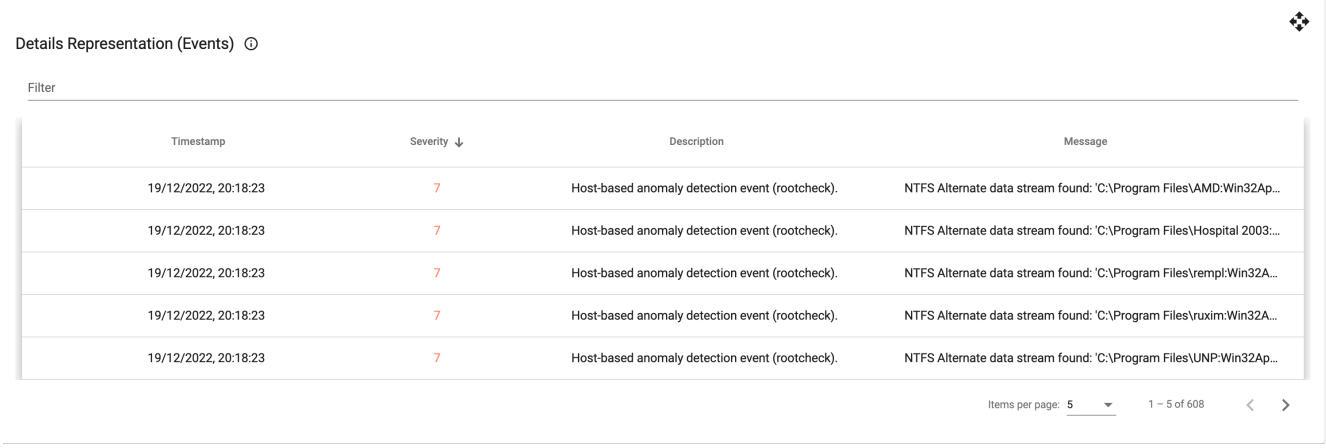

Figure 8: FVT - Details Widget

A variety of different device related metrics can also be analyzed through the available Line Chart widgets (Figure 9). The corresponding widgets support zooming, panning, downloading options, and are interconnected as they share the same time series (x axis).

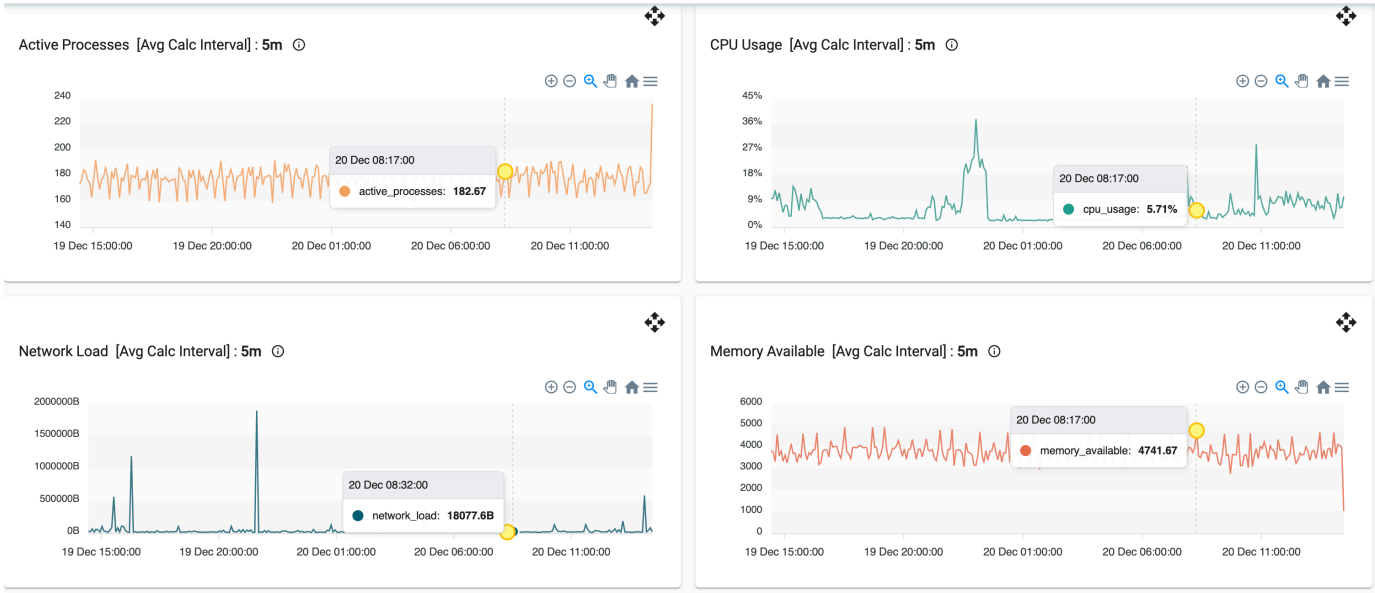

Figure 9: FVT - Performance Metrics Chart Widgets

Users are allowed to discard from the view any of the available widgets or change the order and the topology of the current ones, based on the investigation needs.

For the final version of the platform, a new ‘Events Analysis’ screen was developed in FVT (Figure 10). The ‘Events Analysis’ screen integrates and displays the results of the Anomaly Detection component, thus allows authorized users to investigate, side by side, the output of the ML anomaly detection module and the SIEM across all the monitored devices of the selected department.

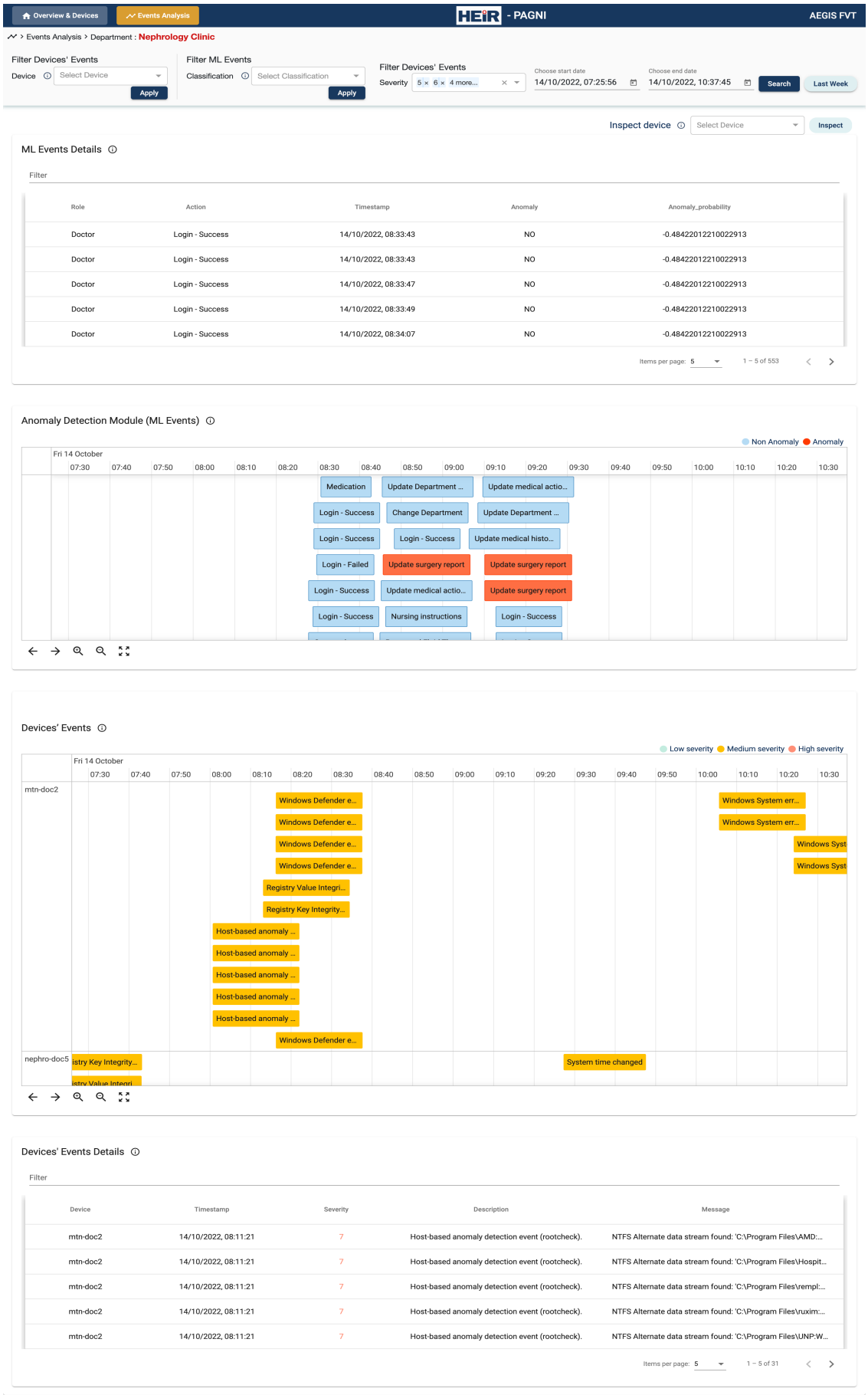

Figure 10: FVT - Events Analysis

Therefore, a security operator can investigate in parallel the captured events for all the connected devices in the department and the reported events from the anomaly detection component. Historical data requests, filtering capabilities, color coded events, extended tooltip information, additional tabular data representations, resizable timelines and zoom interactions synchronization are the key features that have been developed for the facilitation of the investigation.

Finally, via the 'Inspect device" mechanism on the top right of the 'Events Analysis' screen (Figure 10), the end-user has the ability to navigate to the main analysis dashboard of FVT for a selected device in the same time window that is currently applied. This mechanism aims to support a security operator to easily drill down to detailed information regarding a specific device that raised the need of further investigation. (e.g., Anomalies have been detected in a time window where high severity events from a specific device have also been reported)

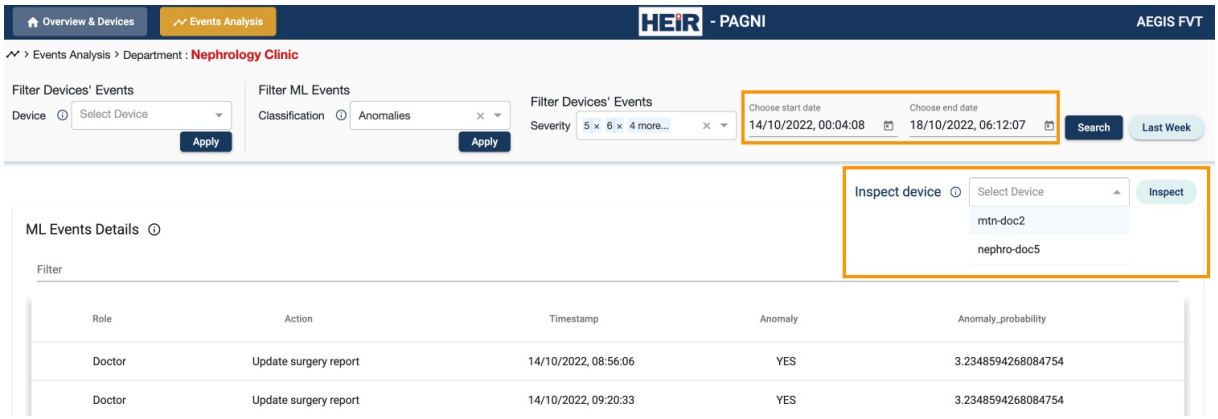

Figure 11: FVT - Inspect Device

FVT consists of two main inner modules, a) an Angular Framework component that serves as the front-end application and b) a middleware in Node.js that is responsible for the dependency communication inside the HEIR environment. FVT has been successfully deployed across all Pilots of HEIR, in a dockerised version. Tailor made adapters and secure pipelines are enabled so to guarantee the end-to-end transmission of data between FVT and the rest HEIR environment's components.

2.2.4 ML anomaly detection and threat classification

This module provides efficient event and threat data classification based on specific rules related to cyber security requirements and cyber-threats level of criticality and novel machine-learning (ML) models. In particular, adaptations of existing ML models utilized in anomaly detection and/or threat classification are incorporated, which match the requirements of the healthcare cybersecurity systems. The ML module takes the input from HEIR IoT (Logs) and processes the records to differentiate between anomalies and non-anomalies. After that, the ML component processes the results in a detailed report. The result is visualized in the FVT toolkit.

The choice of model/algorithm and whether it performs supervised/ unsupervised machine learning depends on the use case (PAGNI, CROYDON). Each use case has its own model which was built to suit the data structure and classify the anomalies in a good order. Both use cases are explained below in details.

2.2.4.1 PAGNI Use Case

Our approach for the PAGNI use case is based on a voting mechanism; we use six cutting-edge algorithms, collect the results, and then aggregate them. We determine the severity of the event or probable anomaly based on the vote results. The following will include descriptions of each of the algorithms taken into consideration for this approach: KNN, SUOD, PCA, LOF, COPOD, and HBOS are described below.

A. KNN (K-Nearest Neighbors): This classification algorithm works based on finding the K nearest neighbors to a given sample point and classifying the sample point based on the majority class of its nearest neighbors.

B. SUOD (Single-class Unsupervised Outlier Detection): This is an unsupervised machine learning technique that detects outliers in data. It is based on computing a similarity score between each sample point and all other samples and flagging the sample points with low similarity scores as outliers.

C. PCA (Principal Component Analysis): This is a dimensionality reduction technique used to reduce the complexity of data by projecting it onto a lower dimensional space while retaining most of the information.

D. LOF (Local Outlier Factor): This is an unsupervised machine-learning technique for detecting data outliers. It is based on computing the local density of each sample point and flagging sample points with a low local density as outliers.

E. COPOD (Cluster-based Outlier Probability Density): This is an unsupervised machine-learning technique for detecting data outliers. It is based on the idea of clustering the data and computing the outlier score for each sample point based on the probabilities of its assigned cluster.

F. HBOS (Histogram-based Outlier Score): This is an unsupervised machine-learning technique for detecting data outliers. It is based on the idea of computing the outlier score for each sample point based on the distribution of feature values in the data.

In order to proceed with the voting engine of the machine-learning component, several steps must be taken beforehand to clean, format, and initialize the data correctly.

2.2.4.1.1 Data Preprocessing

The provided dataset from 01.01.2016 to 08.31.2017 contains **2,497,457 records**; we have different features for actions related to the hospital. The features are shown in the below table.

Column Name	Description
Hospital	indicates the hospital where the action took place.

Role	indicates the person who performed the action
Action	indicates the action that was performed
VPN	indicates whether the action was performed using a VPN
Department	Indicates which department the specific role locating
Date Time	indicates the date and time the action was performed

The values from role and action had to be preprocessed and transferred to integer values without any logic because they were encoded as strings. Furthermore, the date time column was divided into three shifts to cover 24 hours in the hospital **(First shift at 07:00–17:00; Second shift at 17:00–23:59, and Third shift at 00:00–06:59)**.

2.2.4.1.2 Model Building

All the unlabeled data from the hospital, role, action, vpn, and datetime columns were input to all algorithms (KNN, SUOD, PCA, LOF, COPOD and HBOS).

2.2.4.1.3 Data Outcome

After training all six algorithms with the archived data, the voting system can capture and classify the anomalies according to a probability counter-set for each record in the provided dataset. The maximum number of votes was six, whereas the minimum one was four votes. The table below explains the outcome details of the PAGNI archived dataset.

No of Votes	No of Anomalies
1	40
2	10040
3	25010
4	3231
5	14
6	6

The total percentage of anomalies captured out of total events, including all votes, was around **1,6%.**

2.2.4.2 CROYDON Use Case

Our approach for the CUH use case was to use supervised learning. The training data used to train the model covers the regular/irregular patterns (**50 regular patterns / 90 irregular patterns from different series of time**) from the other samples of CTG data of Team3 fetal monitoring devices. The algorithm used is LSTM (Long Short-Term Memory), a Recurrent Neural Network (RNN) commonly used for analyzing time series data and solving tasks related to sequential information, such as speech recognition, language translation, and stock price prediction. In this case, it can classify signals captured from Team3 devices and detect anomalies. LSTM networks are trained on the data sequence and use memory cells to capture long-term dependencies in the input data, making them well-suited for time series analysis. Furthermore, various evaluation techniques were implemented to obtain the highest resolution. Below is the context of the essential features that were used to train the model:

- **FHR (Fetal heart rate):** A regular fetal heart rate typically ranges from 110 to 160 beats per minute (bpm).
- **TOCO (tocodynamometry):** is a measure of uterine contractions. A regular TOCO reading would be consistent with the normal uterine contractions described above.
- **MHR (maternal heart rate):** the mother's heart rate. A normal MHR typically ranges from 60 to 100 bpm
- **Last Modified:** This column indicates the date and time the action was performed.

2.2.4.2.1 Model Objective

Multivariate Time Series Classification: An anomaly might be due to sudden changes in various signals. For example, a sudden change in fetal heart rate (e.g., a rapid increase or decrease)

could be an anomaly. The screenshot below illustrates the test accuracy (**around 99%**) for the labeled data from Team3 devices.

```
Epoch 9/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0045 - accuracy: 0.9987
Epoch 10/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0041 - accuracy: 0.9987
Epoch 11/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0037 - accuracy: 0.9988
Epoch 12/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0035 - accuracy: 0.9988
Epoch 13/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0032 - accuracy: 0.9989
Epoch 14/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0029 - accuracy: 0.9989
Epoch 15/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0027 - accuracy: 0.9989
Epoch 16/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0025 - accuracy: 0.9989
Epoch 17/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0023 - accuracy: 0.9989
Epoch 18/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0021 - accuracy: 0.9990
Epoch 19/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0019 - accuracy: 0.9992
Epoch 20/20
1116/1116 [==============================] - 2s 2ms/step - loss: 0.0018 - accuracy: 0.9994
2232/2232 [==============================] - 3s 1ms/step
558/558 [==============================] - 1s 1ms/step - loss: 0.0559 - accuracy: 0.9835
Test Loss:  0.055933889001607895
Test Accuracy:  0.9834752678871155
```

Figure 12: LSTM Test Accuracy / Loss

The table below represents both the input used for PAGNI/CUH and the machine learning output to post the results in a tangible way using the FVT toolkit.

Module interfaces		
Input		
Name	Type	Short Description
HEIR LOG	SQL, CSV (provided by the partners)	SQL log was provided, and it was changed into CSV format in order to be processed within the ML component. The PAGNI file contains (id, user, hospital, department, role, action, caseID, vpn, datetime) The CUH file contains (id, CTGId, FHR1, FHR2, FHR3, TOCO, MHR, Status, LastModified)
Output		
Name	Type	Description
ML output for PAGNI	JSON (provided by machine learning component)	JSON file was generated as an outcome from ML component to be used in AEGIS side by creating UI tile like SIEM. The file contains (id, user, hospital, department, role, action, caseID, VPN, datetime and Anomaly) + ML score in the

		bottom of the JSON format. For Instance (from one use Case): **{"id": "100000",** **"user": "1188",** **"hospital": "2",** **"department": "102",** **"role": "Doctor",** **"action": "Nursing instructions",** **"caseID": "2051641978",** **"vpn": "0",** **"datetime": "2016-01-27 22:12:25",** **"Anomaly": "NO"}**
ML output for CROYDON	JSON (provided by machine learning component)	JSON file was generated as an outcome from ML component to be used in AEGIS side by creating UI tile like SIEM. The file contains (id, CTGId, FHR1, FHR2, FHR3, TOCO, MHR, Status, LastModified and Anomaly) + ML score in the bottom of the JSON format. For Instance (from one use Case): {"Anomaly": "NO", “Signal”:[{ "id": "100000", "CTGId": "1188", "FHR1": "698", "FHR2": "699", "FHR3": "699", "TOCO": "21", "MHR": "388", "Status": "0", "LastModified": "2016-01-27 22:12:25"}, { "id": "100001", "CTGId": "1189", "FHR1": "698", "FHR2": "698", "FHR3": "698", "TOCO": "20", "MHR": "387", "Status": "0", "LastModified": "2016-01-27 22:12:25"}]}

The figure below illustrates the high-level implementation starting from training the model till the bulk ingestion of the results to depict the results.

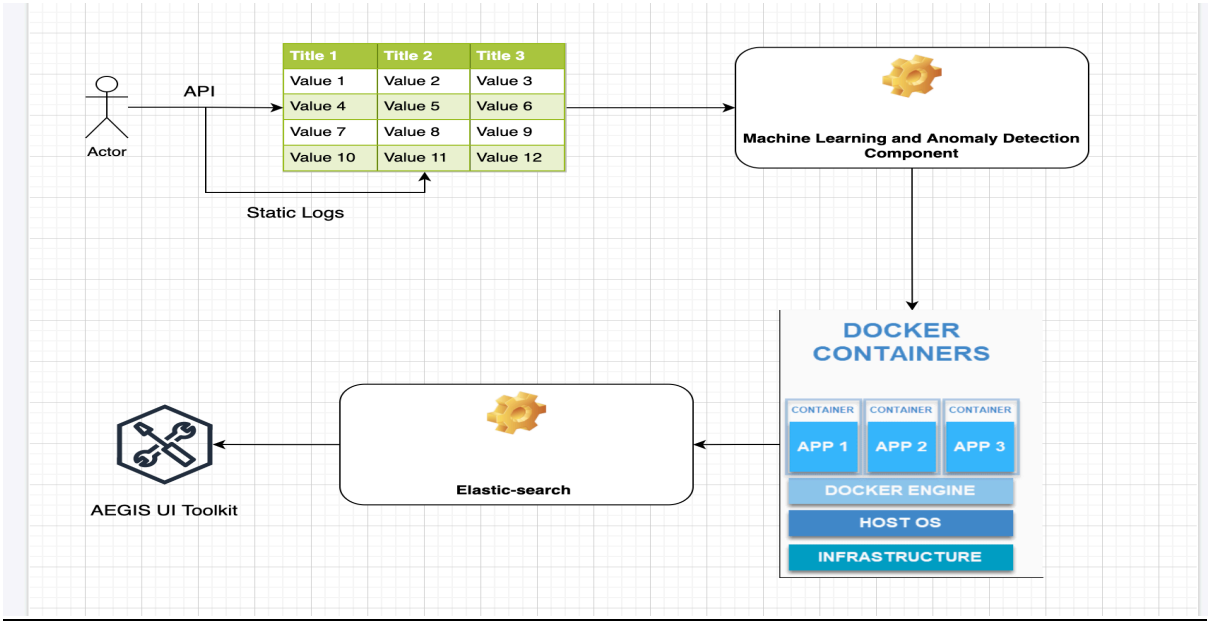

Figure 13: High-Level Implementation

2.2.5 HEIR Collaborative privacy-aware framework (PAF) – task 2.4

2.2.5.1 Privacy-aware framework

The goal of the privacy-aware framework is to provide a secure path to a data source, where data access is controlled by a set of policies typically provided by an organization's Governance Officer. The privacy-aware framework is built on top of the Open Source Fybrik framework[4], which in turn is built on top of leading Open Source technologies such as Kubernetes and Istio for service mesh implementation, and Open Policy Agent.[5]

Using Fybrik, the PAF essentially creates a locked-down path between a data requester and the backend data store – which for our HEIR use cases, is a FHIR server holding simulated medical data. In order to provide policy-compliant data access, the FHIR server and Fybrik components run in a Kubernetes cluster, and PAF configures an *ingress gateway* through which all FHIR requests for data must pass. The ingress gateway then proxies the incoming FHIR request to the *Fybrik Module*, which obtains the policy evaluation decision from the Fybrik Policy Manager, which forwards the original FHIR request to the FHIR server, and then intercepts the returned data, subsequently applying any policy-driving redaction require. An illustration of this is shown in Figure .

[4] https://fybrik.io/
[5] https://www.openpolicyagent.org

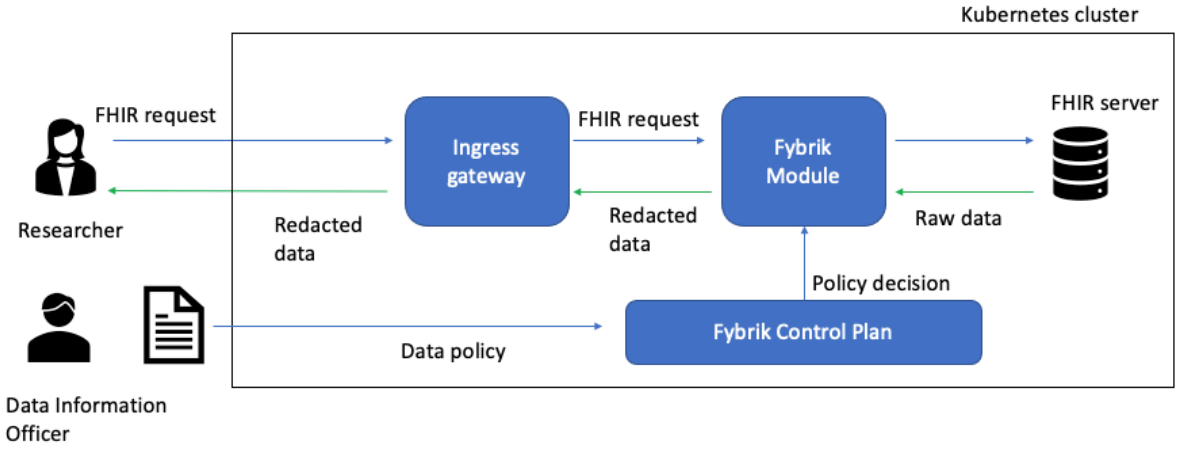

Figure 12: Privacy-aware Framework conceptual architecture

In the first half of the project, we used this architecture to demonstrate two use cases, exporting data to a third-party and policy-driven redactions of FHIR Observation resources. In both of these use cases, the starting assumption was that patient consent for third party data access was dealt with outside of the scope of the use case, and hence all the data in the FHIR server could be made accessible to a third party under the general data policy rules.

In the second half of the project, we expanded our focus to develop a general method to factor in patient-level consent to data, in addition to the general data policy rules.

Task 2.4 together with the Norwegian use case partners examined different ways for allowing patient-level consent.

One method, in the spirit of FHIR, uses the FHIR Consent resource to specify consent parameters for the sharing of data on a per patient basis. In particular, we focused on patient-specification of a time period in the Consent resource to allow his/her data to be shared. An example of this from a Consent resource is shown below:

```
"provision": {
"type": "permit",
"provision": [{
"period": {
"start": "2021-08-02T12:00:00-05:00",
"end": "2023-05-02T12:00:00-04:00"
}
}]
```

Enforcing Consent

The goal behind the use case was to show that patients can specify consent conditions and these conditions would be automatically applied by the Privacy-aware Framework whenever a researcher attempted a FHIR query. An overarching design goal of our solution is to maintain the separation of policy definition from module code, which is done by using Open Policy Agent (OPA) as a policy engine (Policy Decision Point) and defining policies as text files written in the Rego language.

We wanted to be able powerful queries conditions, such as could be done on an SQL database, performing an INNER JOIN between Observation and Consent table with a

sophisticated WHERE clause. While FHIR defines a sophisticated query language, FHIR query mechanisms such as composite search parameters and chained searches are limited both in their power and to the degree that FHIR server manufactures have implemented the specification. Additionally, while the use cases leverage the Consent resource as the data source for the Inner Join, a more general solution includes supporting legacy consent systems, which may not be FHIR-based, such as a SQL database or even a CSV-based data store.

The developed solution

The solution developed to handle our design goals allows for a seamless, behind the scenes merge of the original FHIR query with the power of an SQL JOIN to handle powerful Consent conditions. The original FHIR query is passed by the Fybrik module as-is to the backend FHIR server, and the returned information is stored as an SQL temporary table. The JOIN and WHERE clause to enforce Consent are encoded by the Data Information Officer in the Rego policy definition, as shown below.

```
rule[{"action": {"name": "JoinResource", "joinTable" : "Consent",
"whereclause" : " WHERE consent.provision_provision_0_period_end >
CURRENT_TIMESTAMP", "joinStatement" : " JOIN consent ON
observation.subject_reference = consent.patient_reference "}, "policy":
description}]  {
    description := "Executes a JOIN on the Consent table"
    1 == 1
}
```

Figure 14: Rego rule for JOIN on Consent resources

The Fybrik module then executes a FHIR query against the FHIR server to select the Consent resource and stores in the results in a temporary SQL query. Using the JOIN and WHERE conditions specified in the policy file, an SQL query is composed, and executed against the temporary table held in memory. After executing the query and obtaining the result of the JOIN, the Fybrik module then applies any redaction criteria which are in the policy definition and evaluated by the Fybrik Policy Manager.

A sequence diagram of this process is shown in Figure 15.

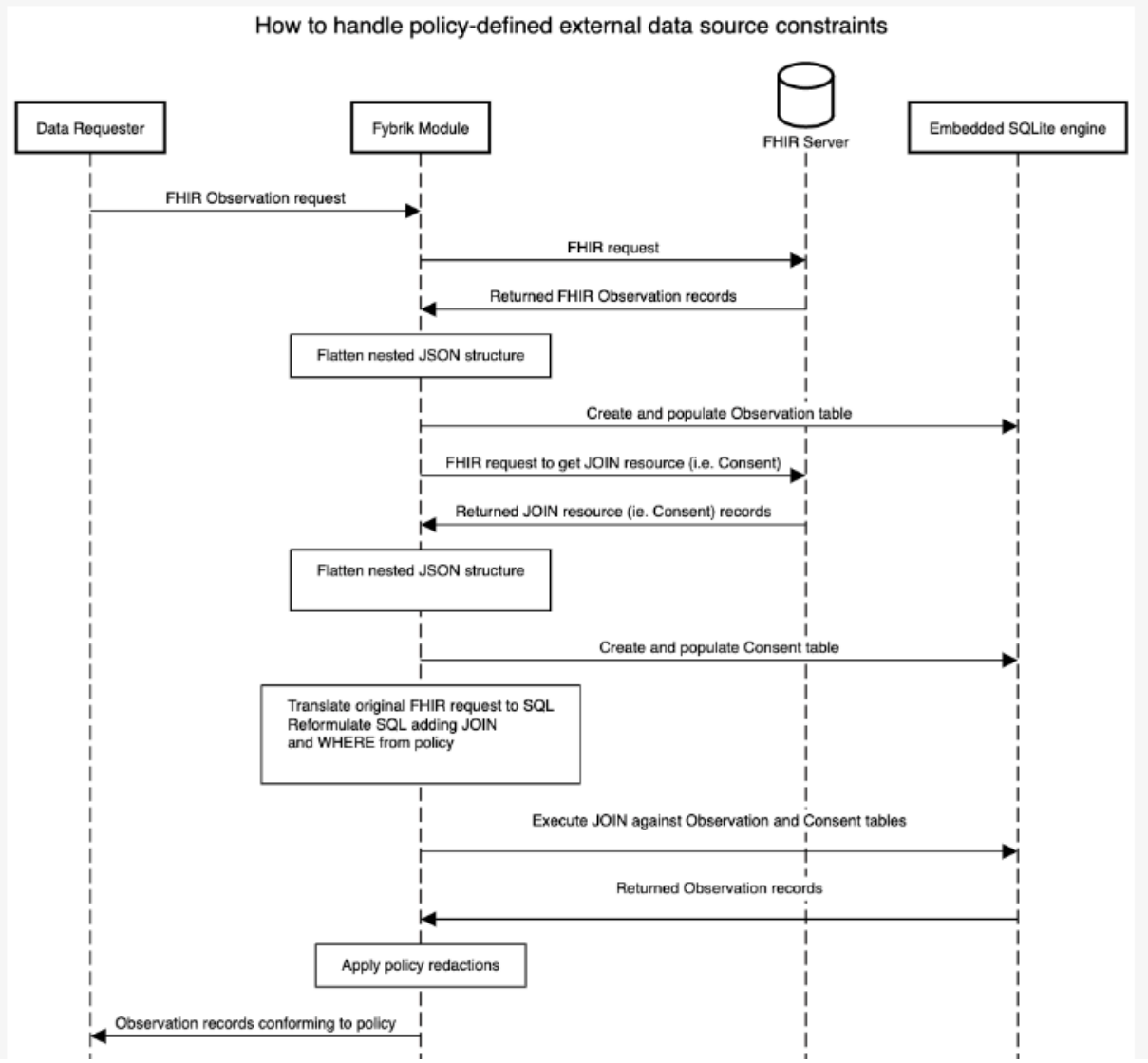

Figure 15: Sequence diagram of JOIN operation

Usability

In order to allow for redaction of sensitive data fields, the schema of each data source is categorized and stored in a catalog in the Fybrik Control Plane. This catalog may be an industrial-strength catalog, such as IBM's Watson Knowledge Catalog, or for non-proprietary installations (such as in HEIR), an internal catalog is used. With the internal catalog, data sources are represented as a Kubernetes Custom Resource called an Asset and represented by a YAML file. An example of an Asset representing a FHIR Observation is shown in Figure 16. In this file, one can see the name normalized name of the tagged FHIR attributes, (e.g. "subject.reference"), the corresponding tag associated with this attribute (e.g. "PII"), and whether the tag is enabled ("True") or not. This YAML file is submitted to the Fybrik Control Plane before the data path to the FHIR server is established.

For the purpose of training potential users in HEIR InfoDays, a simple GUI was written which reads the Kubernete Asset resource and displays the names of all of its attributes and the status (True/False) of their PII field. (See Figure 17). Clicking on an attribute will toggle the PII value. When the "OK" button is pressed, the updated PII statuses are saved. Subsequently, the results of the next data request will be filtered based on the updated PII statuses.

```
apiVersion: katalog.fybrik.io/v1alpha1
kind: Asset
metadata:
  name: observation-json
  namespace: rest-fhir
spec:
  details:
    connection:
      fhir:
        fhirServer: https://fhir-sandbox.heirproject.eu/fhir-
server/api/v4/
        namespace: rest-fhir
        port: 9443
        secretRef: fhir-credentials
      name: fhir
    dataFormat: json
  metadata:
    columns:
    - name: id
      tags:
        PII: false
    - name: subject.reference
      tags:
        PII: true
    - name: subject.display
      tags:
        PII: true
    - name: performer.reference
      tags:
        PII: true
    - name: performer.display
      tags:
        PII: false
    - name: encounter.reference
      tags:
        PII: true
    geography: UK
    name: observation
    tags:
      observation: true
  secretRef:
    name: fhir-credentials
```

Figure 16: The Asset YAML for a FHIR Observation

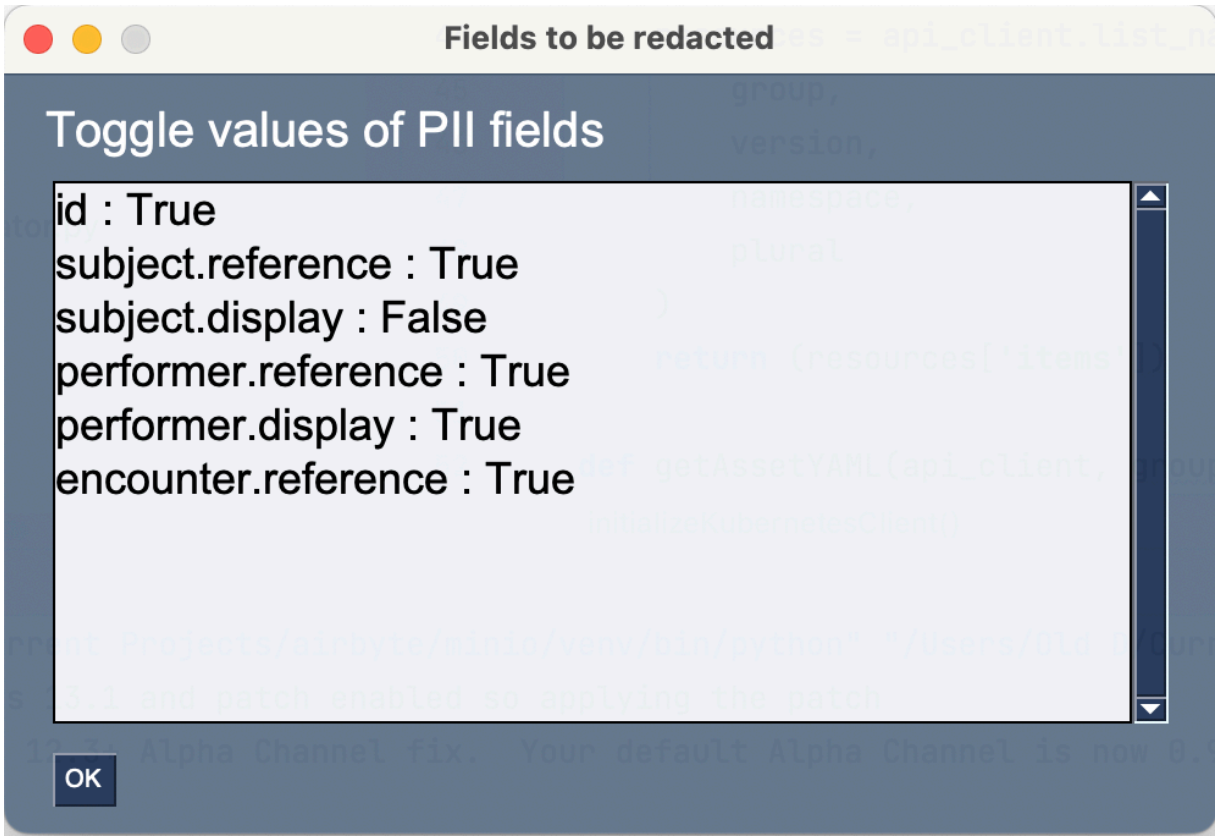

Figure 17: GUI to toggle PII status of FHIR Observation fields

2.2.5.2 HEIR Blockchain-based Auditing mechanism

Being closely related to the Privacy Aware Framework (PAF), the HEIR Auditing mechanism was developed to support logging data access attempts with a view for auditing in the NSE-NOKLUS use case. This was demonstrated in the first project review.

2.3 HEIR Facilitators in the Use Cases

The **HEIR Vulnerability Assessment Module** is a tool that efficiently operates on the complex health infrastructures and analyses them for attack surfaces and possibilities of vulnerability exploitation. It reports information regarding security and privacy by providing warning regarding existing vulnerabilities through the HEIR Agent and utilizing the HEIR Interactive Forensics Module. Currently, it is deployed in all pilots, **(i) PAGNI, (ii) HYGEIA, (iii) CROYDON and (iv) NSE**. The deployments to the pilots are continuously updated in order to keep the technology ready for detecting vulnerabilities that can be exploited by threats.

The HEIR Anomaly Detection and Threat Classification Modules are realized as a component to detect and classify unusual or malicious activity in healthcare infrastructures. This module used a combination of machine learning, statistical analysis, and rule-based algorithms to identify the patterns of behavior that deviate from the expected activity. Currently, it is deployed **in** two pilots, **PAGNI** and **CROYDON**. The component is up and running with all the necessary requirements to fulfill both use cases and ensure scalability in the long run.

The **HEIR Interactive Forensics module** consists of two main inner modules, a) an Angular Framework component that serves as the front-end application and b) middleware in Node.js that is responsible for the dependency communication inside the HEIR environment. FVT has been successfully deployed across all pilots of HEIR in a dockerised version. Tailor-made

adapters and secure pipelines are enabled to guarantee the end-to-end transmission of data between FVT and the rest of the HEIR environment components. This module provides forensics visualization services to enhance the cyber security awareness across the Healthcare systems.

The **Privacy-aware Framework** is deployed in the **NSE** use case as described in Section 2.2.5.1. Additionally, PAF has been used to supply a REST interface to policy-driven access/redaction of blockchain metadata. This interface will be called from the First-level GUI in the NSE use case.

3. Conclusion

As shown in the NSE use case, the Privacy Aware Framework was able to handle a much broader range of data transformation/redaction actions than was envisioned in the original Fybrik design – for example the handling of consent management. Additionally, the ability to handle the very different data access requirements for both FHIR healthcare data and blockchain metadata through one HEIR Fybrik Module was a validation of power of the underlaying PAF infrastructure.

On the other hand, the Privacy-aware Framework was not a good match for some of the other use case environments which run tightly locked down environments that did not allow for the interjection of PAF components into the existing, regulated workflow.

The contribution of the SIEM component in the overall solution, which participates in all four HEIR pilots, is deemed of high value. It enables multiple security related data and metrics to be correlated and exploited in a seamless monitoring of a multi end-point environment via the FVT dashboard. Additionally, adding an extra layer of threat detection logic that is applied on the available data, it solely identifies potential unknown security incidents and helps diminish the effort needed for the security personnel to track down high severity security events.

The HEIR Agent is the main facilitator as integrates the endpoint components as Vulnerability Assessment module but also has the role as backbone for the HEIR Client modules that requires information from the endpoint machines. It provides a continuous extraction of security related data that enables further the up-stream components to assess the risks and to offer enough information to consider proper mitigations.

Anomaly Detection and Threat Classification Modules identify suspicious activity that falls outside of established normal behavior patterns. The implementation was only implemented for two out of four use cases due to the incompatible data logs to be processed with machine learning. In both PAGNI & CROYDON, machine learning adopts different algorithms (i.e., supervised/unsupervised) to find suspicious records captured from either monitoring or team3 devices. The ML component was trained using different archived datasets and scaled (i.e., using different algorithms) to reach the highest accuracy. A complete lifecycle was implemented, starting from training, benchmarking, integrating, and deploying the use cases in a way to be available and scalable in the long run.

FVT serves as the main visualization tool across the facilitators of HEIR, integrating all the outputs into diverse visualization screens based on the needs and installed modules of each pilot. Tracing the differences between our pilots, flexible data adapters and tailor-made visualization widgets were developed in order to facilitate the forensics analysis based on the available capabilities of each environment. FVT was able to handle several connected devices, HEIR clients and a huge amount of events in different formats. However, this module couldn't provide its maximum capabilities, such us additional advanced widgets, preconfigured visual views and multi configurable filters, in cases where a) limited data were available or b) the data formats were not time-wise serialised.