D2.2

The HEIR facilitators package: 1st complete 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.	D2.2
Workpackage	WP2
Due date	**28/02/2022 [M18]**
Actual submission date	11/03/2022

Deliverable lead	AEGIS
Editors	Michalis Vakalellis, Leonidas Kallipolitis, Andreas Alexopoulos
Contributors	Eliot Salant (IBM), M-Marwan Darwish Khabbaz (TUD), Dragoș Gavriluț, Bogdan Prelipcean (BD), Andreas Alexopoulos (AEGIS), George Tsakirakis (ITML), Dimitrios Karamitros (WEL)
Reviewers	Eftychia Lakka (FORTH), Chan Wilson (CUH), Chang John (CUH)
Dissemination level	PU
Revision	Final / 1.0
Keywords	Threat Hunting, Vulnerability Assessment, Forensics Analysis, Privacy Aware, Prototype

Abstract

Deliverable D2.2 serves as the report which describes the 1st complete 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 delivery of the HEIR 1st complete version of the HEIR facilitators package, the intelligent threat monitoring and hunting module providing treat detection as a service, and theblockchain-based collaborative privacy aware framework providing sensitive data trustworthiness sharing. The development of the 1st complete prototype demonstrates the effective integration of the HEIR Facilitators' components into a simple, yet substantial integrated prototype with advanced functionality that showcases the potentials of the HEIR platform.

The 1st complete version advances the development efforts of the Minimal Viable Product (MVP) delivered on M12 and will serve as the basis for further developments and will drive the implementation towards the release of the final complete version (M26). The 1st complete version enhances the functionality of the proof-of-concept demonstrator and additionally will be used to approach HEIR stakeholders and validate fundamental business hypotheses (or leap-of-faith assumptions).

The HEIR facilitators packaged for the 1st complete 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 are briefly presented in this document and their contributions to the thorough HEIR Platform will be demonstrated on D5.3, in regards to the intermediate version of the HEIR integrated product.

Table of Contents

List of Figures

List of Abbreviations

FVT	Forensics Visualition 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 and the advancements performed after the delivery of the initial release of the MVP package of the HEIR facilitators towards the 1st complete version of the HEIR facilitators package. This work is accomplished in the context of the tasks T2.1 - T2.4 and it is the foundation for the future developments envisioned for the next developmental cycle of M18-M26.

For each component, the involved partners provide the research challenges and advancements achieved for the 1st complete version of the Facilitators package, (M12-M18). This deliverable does not consider how facilitators' components will be integrated with the overall HEIR platform, nor how each component communicates with the other components of the HEIR clients and the Observatory, which are going to be reported in D5.3: HEIR integrated framework intermediate Version.

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 D2.1, 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 strongly connected to (a) "D3.2 - The HEIR 1st layer of services package: 1st complete version" as the 1st layer of services (Client services) use the results computed by the facilitators package.

1.3 Structure of the document

The deliverable is organized into five sections whose purpose is briefly described next.

Section 1 introduces the deliverable and highlights relationship to other HEIR deliverables and tasks.

Section 2 presents the architecture of the Facilitators package that supports the rationale behind the 1st complete version development progress and the subsequent steps towards the deployment of the fully functional prototypes.

Section 3 describes the current status of the Facilitators components highlighting the advancements and the works performed, for all the involved components, in the context of the 1st complete version of the HEIR Facilitators package.

Section 4 briefly presents the involvement of the facilitators in each use case for the realization of the 1st complete version of services.

Finally, section 5 highlights the overall conclusions and future plans.

2 1st complete version of Facilitators packages: Architecture

2.1 Architecture Overview

In this subsection, we provide an overview of the 1st complete prototype architecture, regarding the facilitators of the HEIR Platform. Figure 1 illustrates a high level overview of the 1st complete prototype architecture of Facilitators components. In the upcoming subsections, the components will be described in more details.

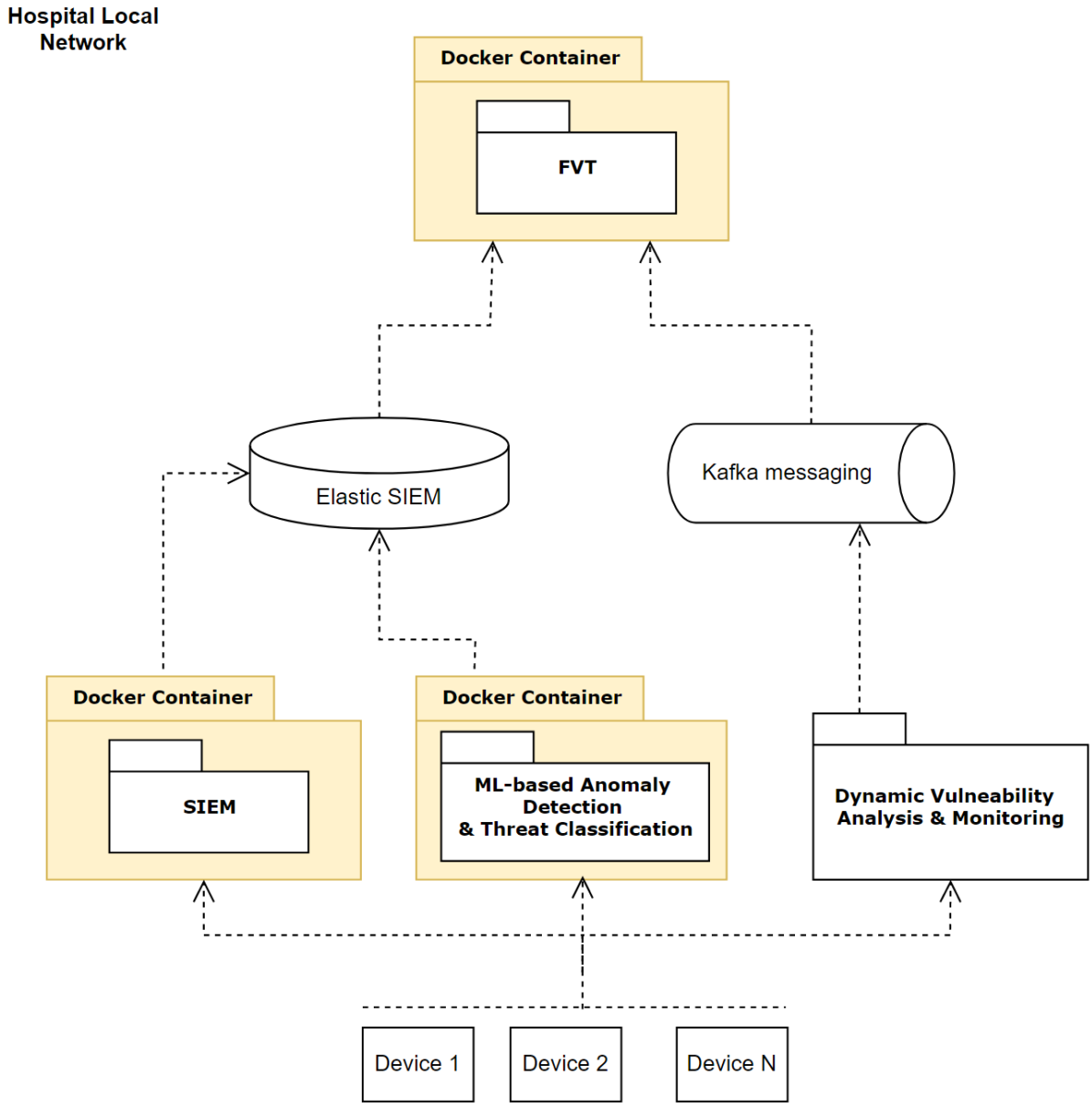

Figure 1: Facilitators Architecture

3 HEIR facilitators package – Component's description

3.1 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 1st complete version of the facilitator package, the HEIR Agent was separated from the HEIR Client (in the MVP it was the same tool), built and deployed as a standalone tool.

The Vulnerability Assessment module 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

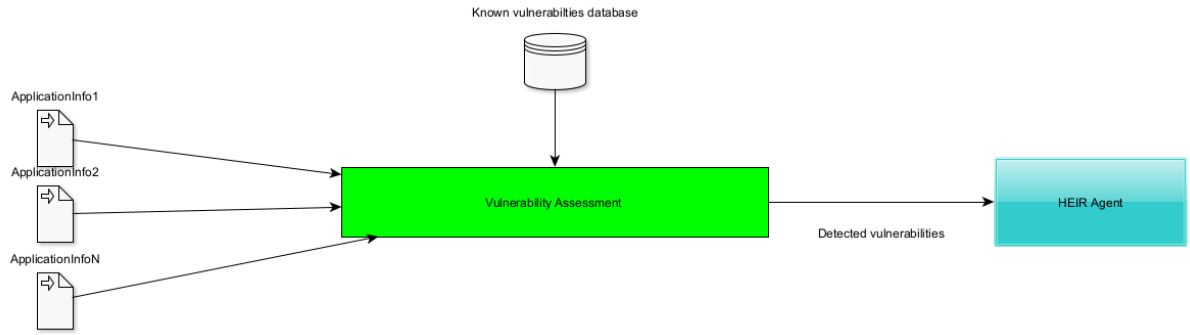

Figure 2: Vulnerability Assessment Module - Data flow

The collected information is:

- Application Name
- Applicant 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, http://cve.mitre.org/

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 1st complete version the rules were updated providing a complete analysis regarding the existing vulnerabilities based on the information that is available. Also the database with known vulnerabilities is updated in a real-time manner.

3.2 SIEM - Threat Detection

The HEIR SIEM component supplies various security related data from all endpoints to the HEIR Interactive Forensics Module, AEGIS' FVT, and it is planned to support the Vulnerability Assessment Module providing another source of security information that can be evaluated and exploited together with the rest of the available components and data sources.

It is based on the Wazuh[2] open source solution which provides a multitude of security related services that continuously monitor an IT infrastructure. All data is collected by lightweight agents which run on the monitored systems, collecting events and forwarding them to the Wazuh Manager, where data is aggregated, analyzed, indexed and stored. This ensures that the resources needed at the client level is kept to a minimum since the security intelligence and data analysis is solely performed at the server level. Wazuh clients run on many different platforms, including Windows, Linux, Mac OS X, AIX, Solaris and HP-UX.

[2] https://wazuh.com/

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 Compromise (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. This is further delivered via customized alerts that are sent to an Elastic Stack[3] which also provides a powerful interface for data visualization and analysis via its integration with Kibana. [4]

In addition to logs and events deriving from the operating system, Wazuh is able to collect and integrate logs derived from network devices such as routers, firewalls etc. either by monitoring the log files themselves or via forwarding log messages through Rsyslog[5]. This can potentially facilitate the collection of logs from medical devices that would need to be monitored within the Healthcare Use Case environments.

The Wazuh event flow management is depicted in Figure 3.**Erreur ! Source du renvoi introuvable.**

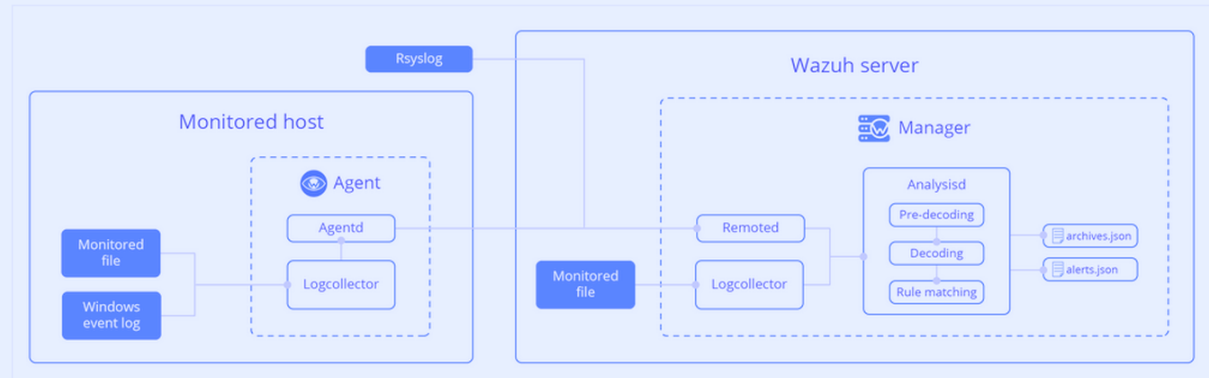

Figure 3: Wazuh event flow management

Furthermore, Wazuh offers a powerful RESTful API that allows the interaction of the Wazuh manager with web browsers, command line tools like cURL [6], or any scripts or programs that can make web requests. This, combined with the RESTful APIs provided by ElasticSearch, will greatly aid to the seamless accumulation of the HEIR SIEM security metrics in the integrated HEIR Client and its role in the Local RAMA score calculation.

[3] https://www.elastic.co/elastic-stack/
[4] https://www.elastic.co/kibana/
[5] https://www.rsyslog.com/
[6] https://curl.se/

For the initial stage of the MVP, the HEIR SIEM main role is providing all necessary information via Elastic to be depicted in the Forensics Visualization Toolkit (FVT)

3.3 HEIR Interactive Forensics Module

The Forensics Visualization Toolkit (FVT) 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.

Overview and Devices

Users accessing the FVT will firstly see the 'Overview and Devices' screen of the selected client (Department). Generic information about the selected client (e.g., connected devices, total critical events detected, security status etc.) and the calculated Local RAMA score are presented in the top panel. (Figure 4)

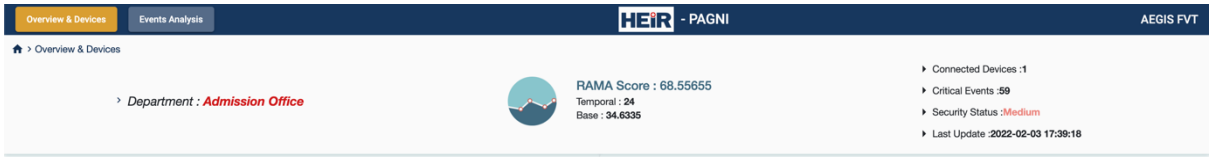

Figure 4: RAMA Score

The results of the HEIR Exploit Tester and Cryptographic Checker are displayed via graphical representations and lists (Figure 5).

Figure 5: HEIR Exploit Tester and Cryptographic Checker

The Vulnerability Assessment's metadata are demonstrated in an expandable section, where the users can quickly identify the top 10 vulnerabilities and the full list of vulnerabilities per application. Each vulnerability is clickable and linked to MITRE's CVE knowledgebase[7]. (Figure 6)

Figure 6: Vulnerability Assessment

HEIR Network Module's analysis results are currently presented in a tabular view (Figure 7). Critical information, such us the number of exploits, attacks etc., are presented in the top of the section. In addition to sorting capabilities, there is also a standalone text filter, so the users can search through the available information and quickly identify meaningful details.

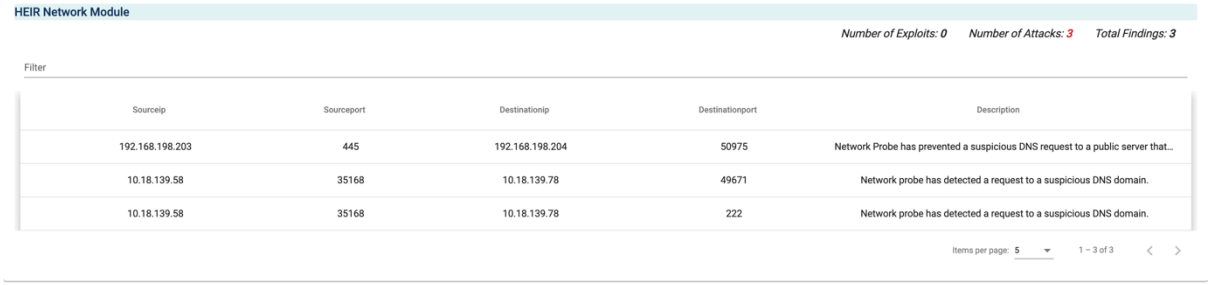

Figure 7: HEIR Network Module

At the bottom of the screen, users can choose from the devices (that are connected to the HEIR Client) the one they want to investigate. (Figure 8)

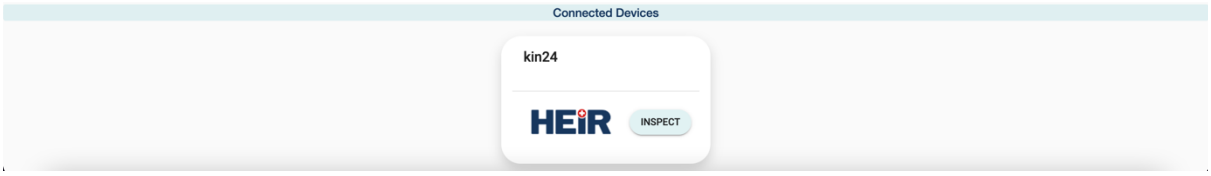

Figure 8: Connected Devices

[7] https://cve.mitre.org/

The complete page of HEIR Client's Overview and Devices is available in Figure 9 below.

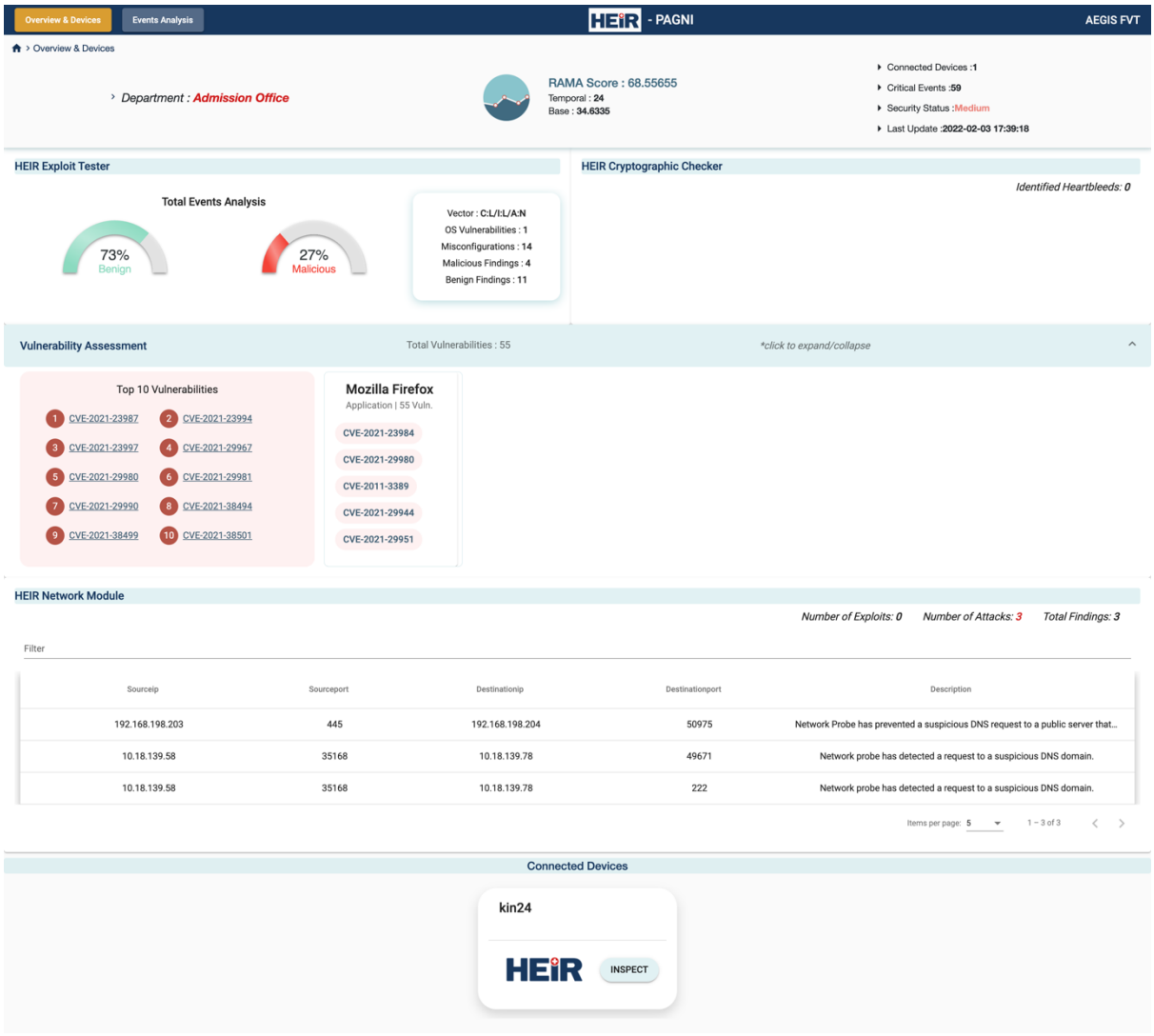

Figure 9: HEIR Client's Overview and Devices

Inspect device

The FVT's device inspection main dashboard (Figure 10) allows users to choose from a set of widgets containing different types of visualizations, that refer to different System metrics and Security event 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 will also be able to filter the incoming data by different meaningful parameters and to search historical data.

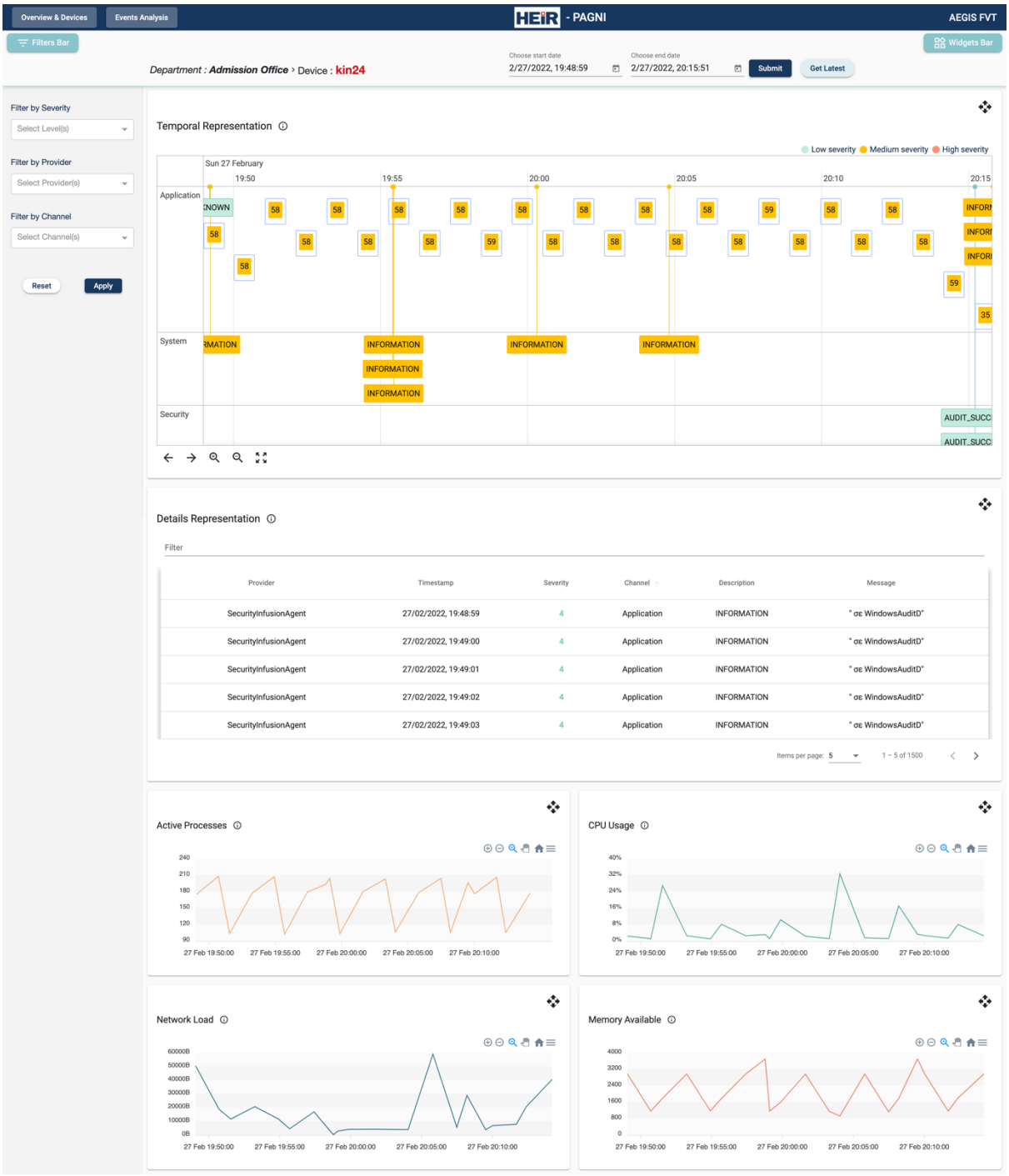

Figure 10: Inspect Device page

Temporal (point in Time) representation of incoming logged events captured by the SIEM will be available through the Timeline widget (Figure 11). By changing the date period ('Start date' to 'End date') in the timeline all the available widgets will be timewise synchronized and updated accordingly. In the filter bar (widget menu in top left side), the user can further filter the results by Severity, by data provider (e.g. Security Infusion Agent, control manager etc) and by channel (e.g. application, system etc).

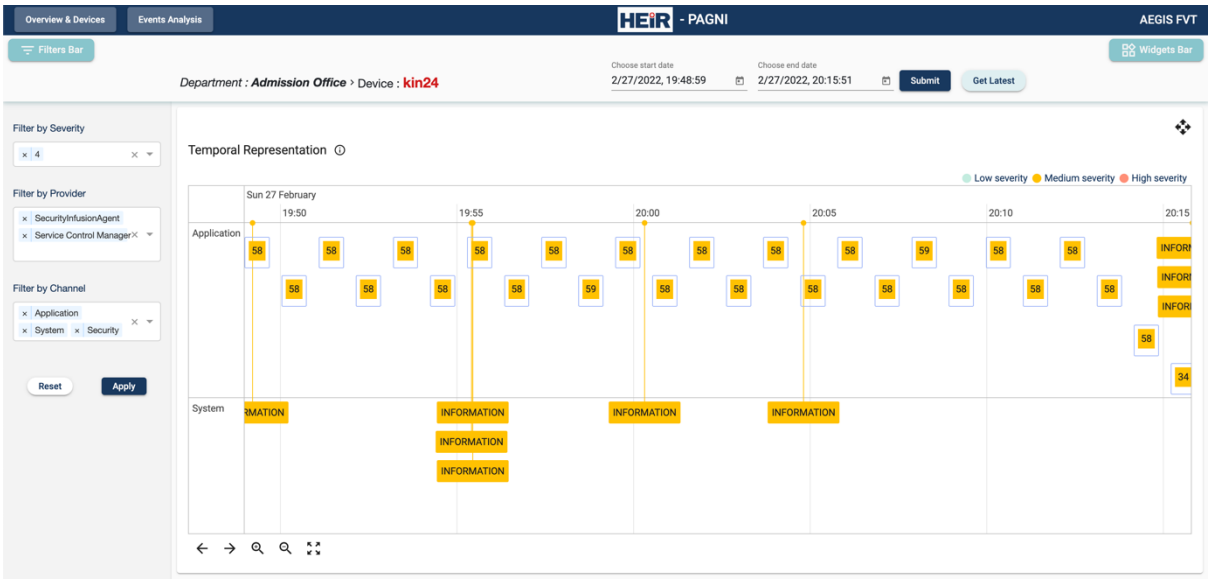

Figure 11: Temporal Representation

Detailed information about the incoming events will be presented in the Details widget (Figure 12). This widget also supports a standalone text filtering capability in order to search through the available information.

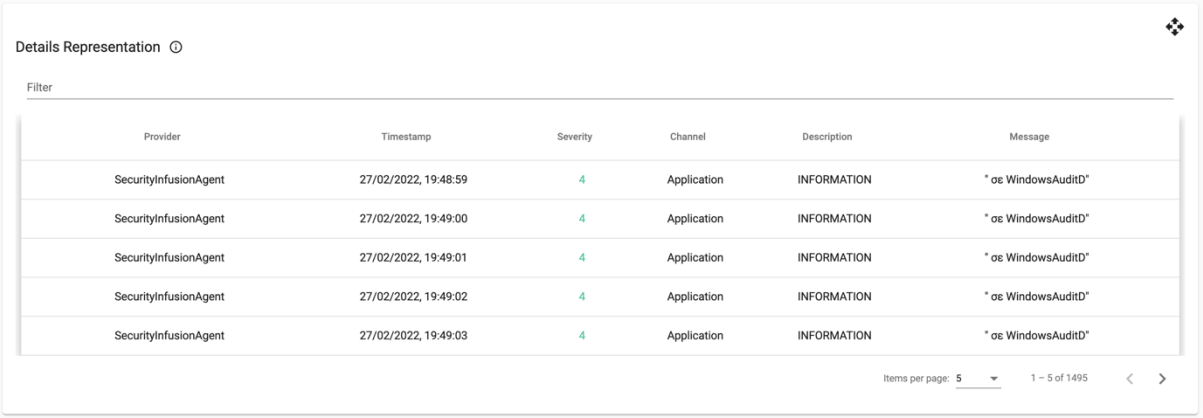

Figure 12: Details Representation

A variety of different device-related metrics can also be analyzed through the available Line Chart widgets (Figure 13), including the active processes, the CPU usage, the network load and the available memory of this device. The corresponding widgets support zooming, panning, downloading options and are interconnected as they use the same time series (x axis).

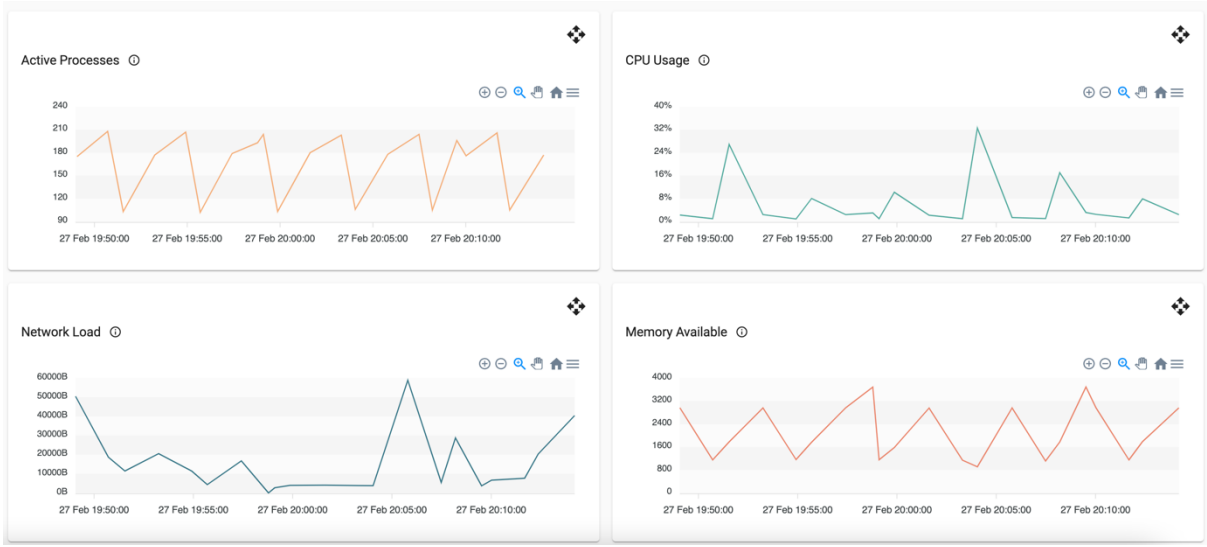

Figure 13: Device's Metrics

Events Analysis

FVT also provides an Events Analysis screen, in which the logged events from the department's (client) connected devices and the output of the Anomaly Detection's module (ML) are presented. The top layer of the screen contains the current available filters and the datetime picker for historical data requests (Figure 14).

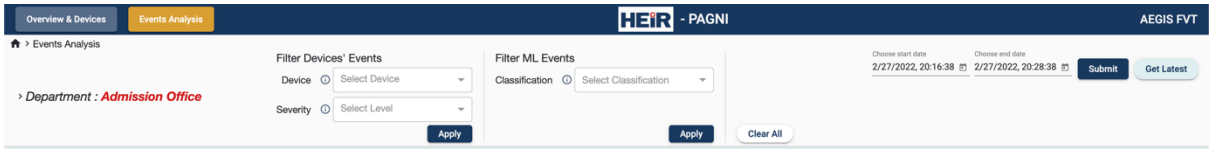

Figure 14: Events Analysis filter section

Temporal (point in Time) representation of incoming logged events captured both by the Anomaly Detection Module (ML) and the selected devices will be available through the Timeline widget (Figure 15). By changing the range period in the timeline all the available widgets will be timewise synchronized and updated accordingly. At any time, the current zoomed period is displayed above the Timeline (upper right corner). A functionality to select a connected device and navigate to FVT's device inspection main dashboard (Figure 15) in the current zoomed period, is available. This functionality aims to enhance the forensics analysis, by offering timewise parallel comparison of the detected anomalies in the department and the devices' logged events.

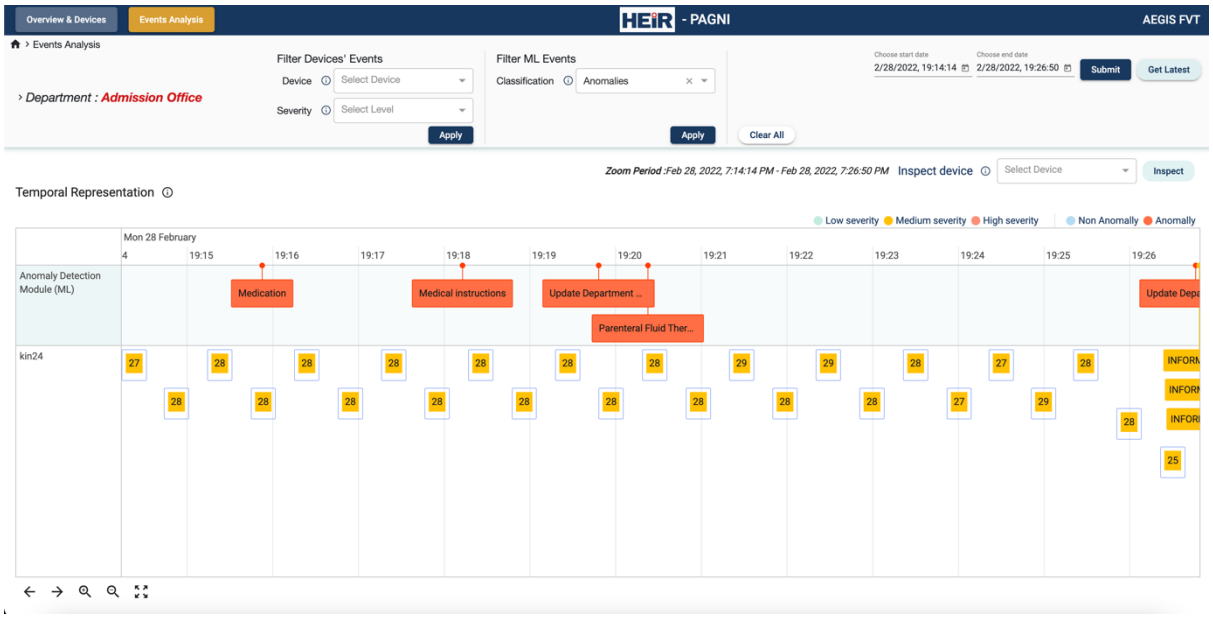

Figure 15: Temporal Representation

Detail representation widgets, both for the Anomaly Detection Module (ML) output and the logged events (SIEM), are available (Figure 15, Figure 16). Independent filtering and sorting capabilities are also available for these widgets (e.g. sort by a specific column, or focus on specific table line).

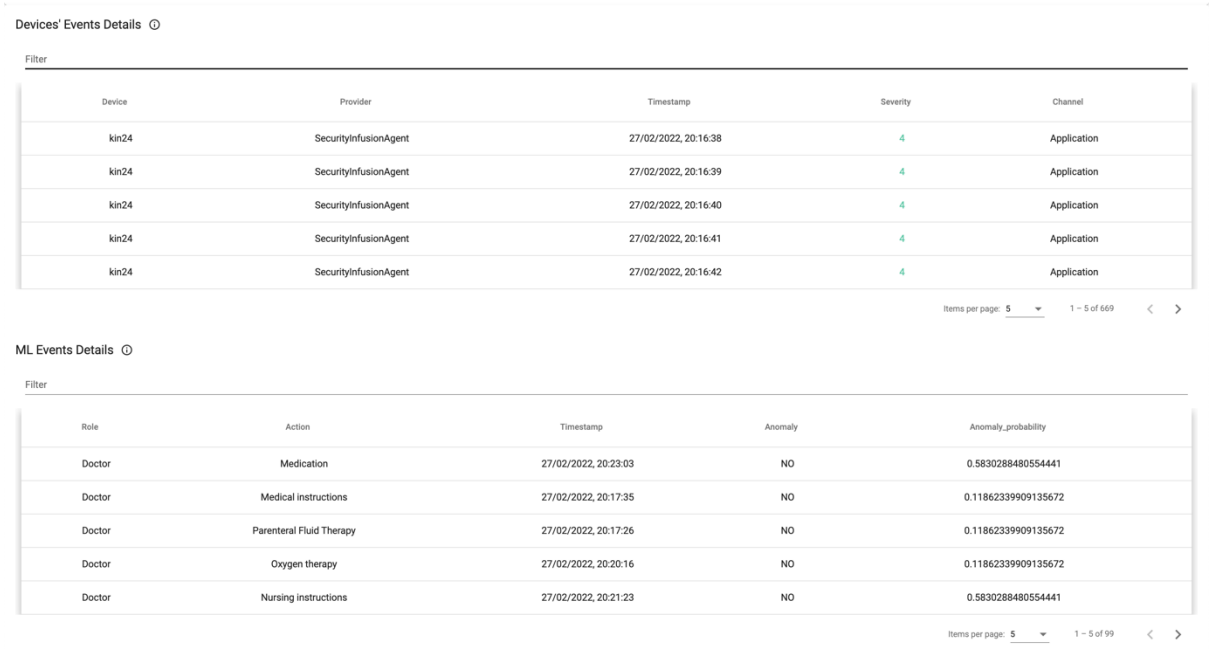

Figure 16: ML Events and Devices' Events Details Representations

The full page of Events Analysis is available below, in Figure 17.

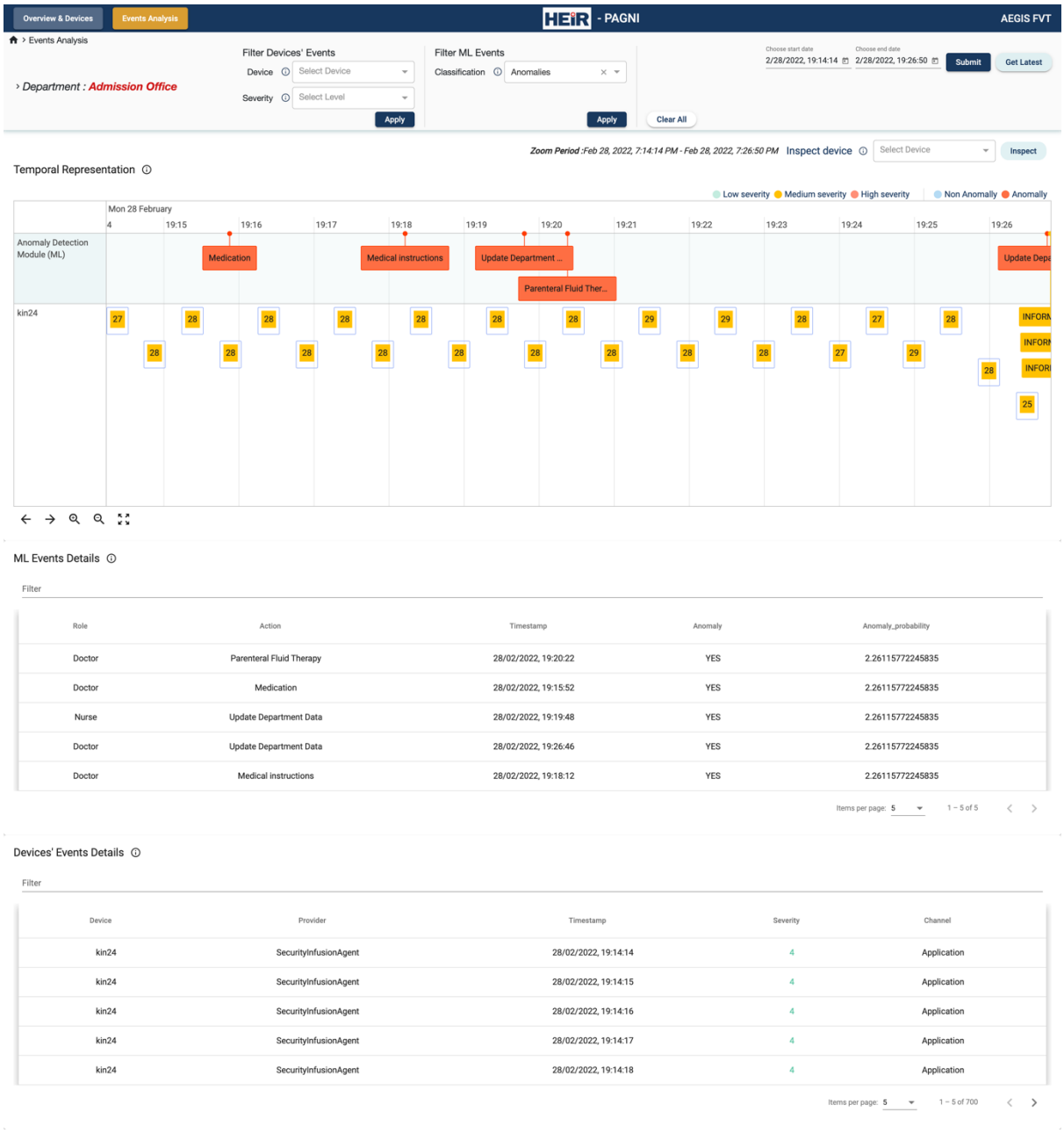

Figure 17: Events Analysis Full Page

3.4 ML anomaly detection and threat classification

ML anomaly detection and threats classification provide an efficient event and threat data classification based on specific rules related to cyber security requirements and cyber-threats level of criticality, novel machine-learning (ML) models will be developed in this task. In particular, it is expected that adaptations of existing ML models utilized in anomaly detection and/or threat classification will be incorporated, which will match the requirements of the health systems.

The machine learning module will take the input from HEIR IoT (Logs) and process the records in a way to differentiate the anomalies and non-anomalies. Furthermore, the ML component will process the results in a detailed report. The result will be visualized in AEGIS toolkit to

represent the results in a tangible way. The machine-learning component is a part of the HEIR facilitators service, as shown in Figure 18, below.

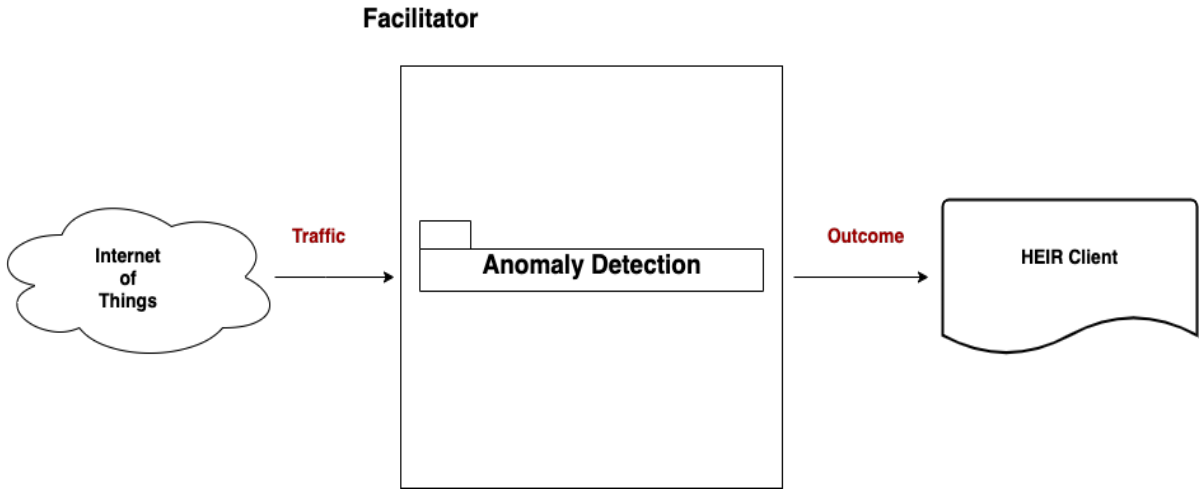

Figure 18: ML Overview

Algorithm Definition

Random forest algorithm is a supervised learning algorithm that uses Ensemble Learning method[8] for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. Figure 19 will illustrate the structure of Random Forest algorithm.

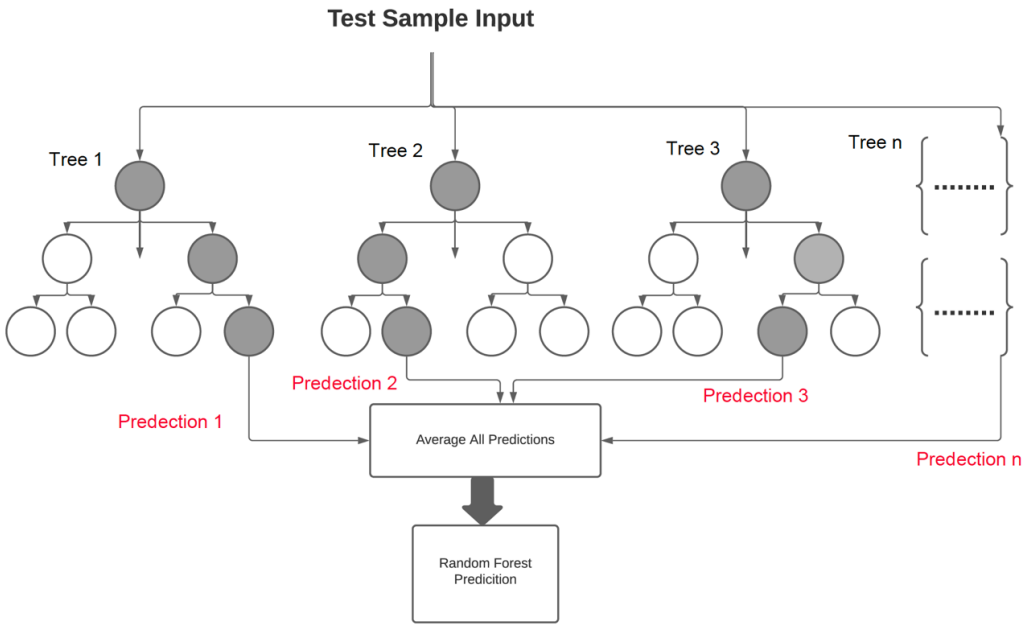

Figure 19 Algorithm Structure

[8] https://en.wikipedia.org/wiki/Ensemble_learning

Module Interface

- Input: SQL log was provided, and it was changed into CSV format in order to be processed within the ML component. The file contains (id, user, hospital, department, role, action, caseID, VPN, DateTime)
- Output: JSON file was generated as an outcome from ML component to be used by the AEGIS technical partner in creating a UI tile like SIEM. The file contains (id, user, hospital, department, role, action , caseID, VPN, DateTime, Anomaly, Z-score) + ML score in the bottom of the JSON format.

Current Status of Machine Learning Component

The Machine Learning component has followed different processes to achieve the desired results by extracting the anomalies from HEIR logs. The processes are explained below.

1. Implementing the Random Forest Algorithm
2. Training the model using dummy data
3. Evaluating the achieved model with other supervised algorithms
4. Initialized the ML component to adapt the Healthcare partner site logs (eg. PAGNI) :
    - Padding: value to use to fill holes
    - Label Encoding: refers to converting the labels into a numeric form
    - Dependent/Independent values: What the changes/ What is being studied
    - Variables Importance: quantify the usefulness of all the variables
5. Completed Output: JSON file was generated as an outcome from ML component to be used by the Technical partner (AEGIS) to create a UI tile like SIEM. The file contains (id, user, hospital, department, role, action, caseID, VPN, DateTime, Anomaly, Z-score).
6. Machine Learning Score: A total number for both anomalies and non-anomalies was added to the generated output to follow the same pattern as SIEM
7. Containerizing machine learning component
    - Select the base image we want to use
    - Select the files we want to copy inside the Docker image
    - Install application's dependencies
    - Select Docker volume to generate the output
8. Integrating the desirable output after executing the docker application into the Elasticsearch platform
9. Utilizing the output from AEGIS's FVT to visualize the results accordingly.

3.5 HEIR Collaborative privacy-aware framework (PAF)

3.5.1 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[9], 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.[10]

Advancing beyond the MVP, we have expanded the capabilities of the privacy-aware framework, including the development of a HEIR Fybrik module, to provide policy-driven control of data accessed from a FHIR server[11], token authentication for data requests, as well as logging data transaction requests to Kafka.

The creation of a data pipeline through Fybrik invocation is initiated by a specified requester. All FHIR requests for data must include a JSON Web Token (JWT) in the REST header which encodes the requester. The privacy-aware framework will reject any received FHIR request which encodes a different requester than the one that invoked the data pipeline.

The developed HEIR Fybrik module utilizes decisions from the Fybrik Policy Manager to provide fine-grained access both to individual FHIR resources, as well as redact individual fields within specified FHIR resources.

All requests to access data are logged as JSON records to a dedicated topic on Kafka for consumption by the HEIR blockchain framework. An example of a logged access request can be seen in:


```
{
"Timestamp" : "2022-02-06 15:41:43", "Requester": "EliotSalant", "Query": "Observation","ClientIP": "127.0.0.1","assetID": "sql-fhir/observation-json","intent": "research","Outcome": "UNAUTHORIZED"
}
```


Note that the "Outcome" field will show whether or not the requester is allowed access to the requested FHIR resource.

3.5.2 HEIR Blockchain-based Auditing mechanism

Being closely related to the Privacy Aware Framework (PAF), the HEIR Auditing mechanism is developed to originally support the notion of logging data access attempts with a view for auditing in the NSE-NOKLUS use case. Details on the use case are available in deliverable D6.1

[9] https://fybrik.io/
[10] https://www.openpolicyagent.org
[11] https://www.hl7.org/fhir

The Use Case scenario for the usage of the auditing mechanism currently in the NSE-NOKLUS Use Case is described in the following section.

In its simplest form, a user needs to access patient medical data. The user can on one hand belong to an organization, requiring accessing patient medical data that are stored internally in the organization. On the other hand, a 3rd party user, such as a researcher, would like to access anonymized or redacted data (for a variety of different reasons such as research, further statistical processing, visualization etc.). In the latter case, this data may or may not reside inside that Healthcare organization. A user-initiated access to data is authorized based on data access policies set by the Healthcare organization itself. The Privacy Aware Framework checks against these access control policies and allows or blocks these requests.

The auditing mechanism stores all these data access requests performed by users, regardless of whether the requests were eventually authorized or not by the Privacy Aware Framework. Storing these access requests, along with their metadata, the goal of the Auditing mechanism is to provide:

- An immutable record of all data access attempts
- A filtering mechanism to identify malicious unauthorized access attempts on a regular (daily, weekly, monthly etc) basis
- A timeline of events in the form of abnormal data access requests, facilitating the trace of malicious user behaviors back in time. This timeline could be used in conjunction with the HEIR threat detection modules, as complementary contextual data.

The Auditing mechanism developed is based on Hyperledger Fabric[12], a well-known and enterprise mature framework used for the development of permissioned blockchain applications. The Fabric executes distributed applications written in general purpose programming languages across a few peer nodes, allowing for both isolated, single-organization isolated ledgers, or cross-organization auditing. While the Auditing mechanism is equipped with server-side client applications that provide a REST API, integration with the Privacy Aware Framework is currently achieved via a dedicated Kafka topic, where all access logs are transmitted (Figure 20 illustrates this point).

[12] https://www.hyperledger.org/use/fabric

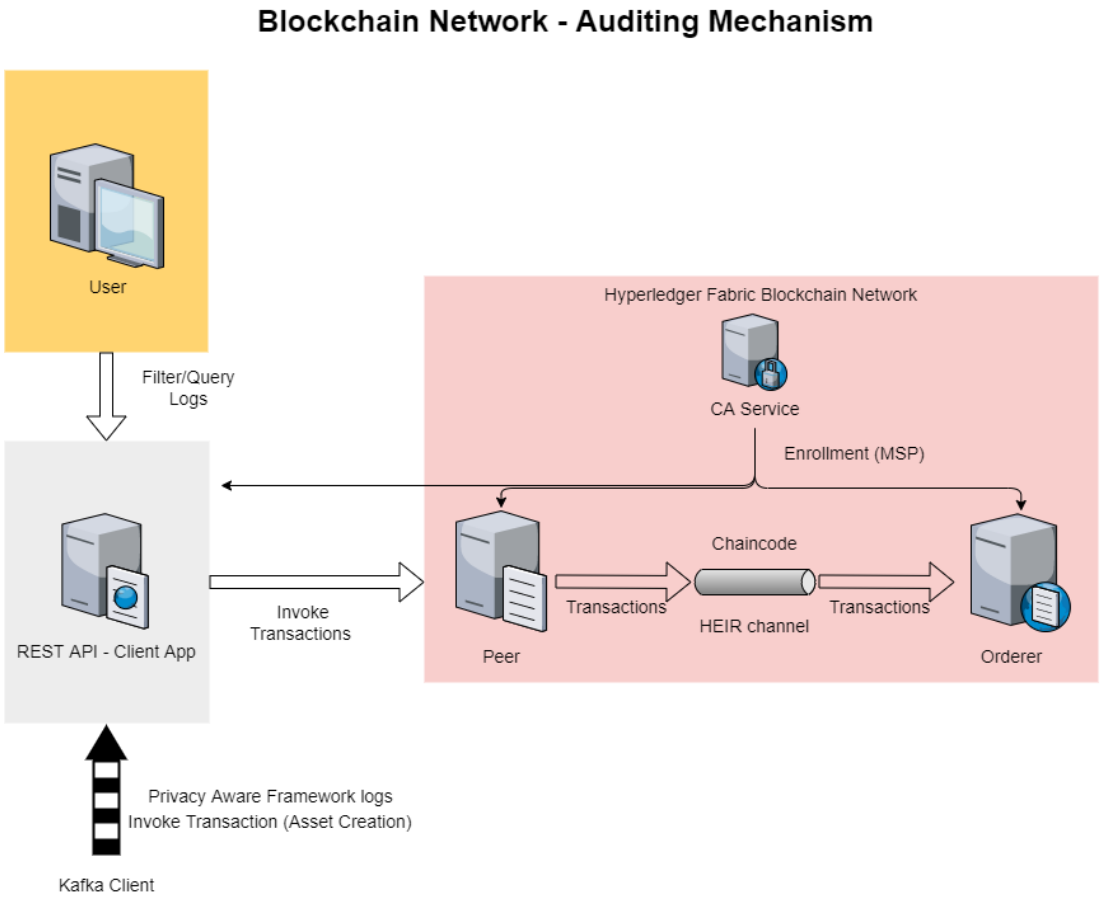

Figure 20: Auditing Mechanism

The developed auditing mechanism is presented in Figure 20. The components of this system are briefly described below:

Fabric User: a digital identity that has permitted access to the stored audit logs and can perform queries/filtrations on them. This digital identity is required by the app in order to invoke transaction with the smart contract implemented on the blockchain. Currently, the user is internally certified for testing purposes. next steps for the evolution of the auditing framework and its complete integration in the HEIR platform, an end-to-end authentication/authorization service and interface is going to be developed and integrated in the system for the purposes of user registration and management.

Client App: String based application acting as a client to the blockchain network, with Kafka based integration with Fybrik.

REST API: provides the capability of querying/filtering the stored audit logs.

Organization: the NSE organization where the blockchain network is implemented.

Peer: a blockchain node that stores all transactions on a joining channel (HEIR channel).

HEIR Channel: the communication channel where the chaincode is implemented. In general, the Fabric channel facilitates logical and physical separation of stored data on a ledger, therefore providing the equivalent of multitenancy in the blockchain.

Chaincode: refers to the smart contract, a program code that implements the application logic. It constitutes the central part of a distributed application in Fabric as it determines the blockchain's transactions functionalities. The implemented smart contract supports the creation of assets and various queries/filtrations on the stored assets.

Orderer: a service responsible for ordering transactions, creating a new block of ordered transactions and distributing a newly created block to all peers on a specific channel.

CA service: it is responsible for managing user management aspects, such as user certificates, user registration, user enrollment, user revocation. Hyperledger Fabric uses an X.509 standard certificate-based mechanism, fully extensible, to represent permissions, roles and attributes for each user. A Fabric user can query or invoke any transaction on any channel based on the possessed permissions, roles and attributes.

Kafka Client: used for the integration with Fybrik and the storage of Privacy Aware Framework logs in the blockchain network.

Currently our blockchain network consists of one organization with one peer, one orderer and one CA. In terms of the NSE Use Case scenario, the Client App, integrated with Fybrik via Kafka, is consuming the Access Log messages that represent secure data access logs.

When the Fabric network first starts up, initialization is required. For this reason, a Docker container is launched that performs the registration and configuration of the channel, as well as the deployment of the chaincode (aka. smart contract). The console output of this container (Figure 21) demonstrates the configuration steps taken.

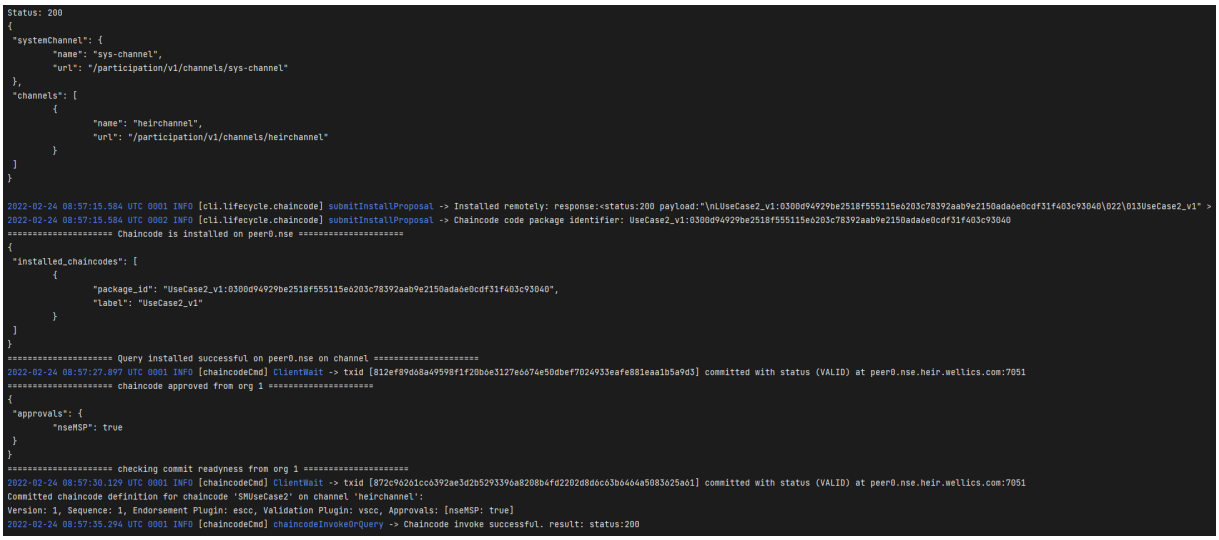

Figure 21: Fabric network intialisation

The fields of the incoming logs are:

- Timestamp (When the access request took place)
- Requester (who made the access request)
- Query (the query performed)
- ClientIP
- assetID
- intent (the purpose of access request)
- Outcome (the outcome of the access request, authorized or not)

Example of the produced Kafka message is the following (in JSON format):

```
{
    "Timestamp": "2022-02-06 15:41:43",
    "Requester": "userID",
    "Query": "Observation",
    "ClientIP": "127.0.0.1",
    "assetID": "sql-fhir/observation-json",
    "intent": "research",
    "Outcome": "UNAUTHORIZED"
}
```

Upon reception of a log via the Kafka client, the Client App encrypts it with the AES algorithm and invokes the asset creation function supported by the deployed smart contract. The fields of the created asset are:

- Timestamp
- UserID (requester)
- query
- ClientIP
- intent
- outcome
- encryptedLog

The created asset is then stored on ledger dictated by the specific channel used in this scenario, the NSE channel. Figure 22 show the console of the client app, from the reception of a log to the invocation of the smart contract and its storage in the blockchain.

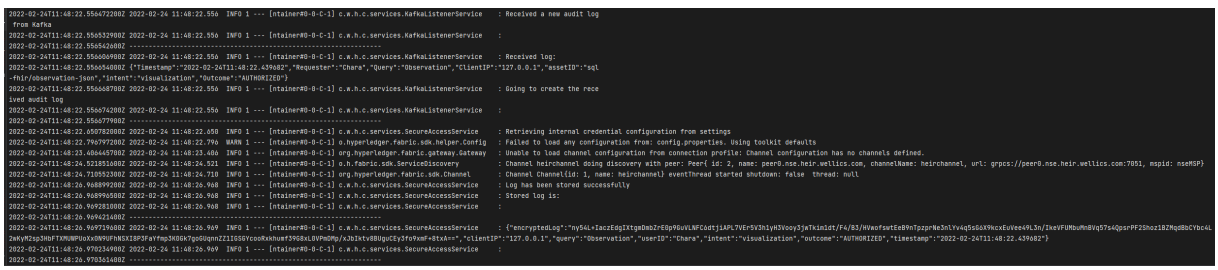

Figure 22: Processing and storage of incoming access logs

Various queries/filtering operations are exposed by the smart contract itself and can be accessed via the REST API. Examples include queries based on userID, outcome, intent, executed and the time range.

A description of the services exposed by the API is presented in Table 3-1 below.

Table 3-1: Auditing mechanism Client application REST API

API reference			
Method	**HTTP request**	**Params**	**Description**
GetAllLogs	GET <domain>/queryAllLogs	-	Returns the list of the stored logs
queryExists	GET <domain>/queryExists	Timestamp	Returns a boolean value indicating the existence of specific log
queryLog	GET <domain>/queryLog	Timestamp	Returns specific log
getLogsByRange	GET <domain>/getLogsByRange	startDate, endDate (yyyy-mm-dd, yyyy-mm-dd)	Returns the list of the stored logs for the desired time range
QueryLogsByID	GET <domain>/queryLogsById	userID	Returns the list of the stored logs for specific userID
QueryLogsByIntent	GET <domain>/queryLogsByIntent	Intent (analysis, research, visualization etc)	Returns the list of the stored logs for specific intent
QueryLogsByOutcome	GET <domain>/queryLogsByQuery	Query (observation etc)	Returns the list of the stored logs for specific query
QueryLogsByQuery	GET <domain>/queryLogsByOutcome	Outcome (AUTHORIZED or UNAUTHORIZED)	Returns the list of the stored logs for specific outcome
getLogsByRangeAndID	GET <domain>/getLogsByRangeAndID	startDate, endDate, userID	Returns the list of the stored logs for the desired time range and for specific userID
getLogsByRangeAndOutcome	GET <domain>/queryLogsByRangeAndOutcome	startDate, endDate, Outcome	Returns the list of the stored logs for the desired time range and for specific outcome
getLogsByRangeAndIntent	GET <domain>/queryLogsByRangeAndIntent	startDate, endDate, Intent	Returns the list of the stored logs for the desired time range and for specific intent

API reference			
Method	**HTTP request**	**Params**	**Description**
QueryLogsByIDAndOutcome	GET <domain>/queryLogsByIdAndOutcome	userID, Outcome	Returns the list of the stored logs for specific userID and outcome
QueryLogsByIDAndIntent	GET <domain>/queryLogsByIdAndIntent	userID, Intent	Returns the list of the stored logs for specific userID and intent

Hyperledger Fabric, thus the Auditing mechanism, utilizes a complex identity management scheme which is internal to the framework. Despite the Auditing mechanism not yet being integrated with the user management scheme, preparatory work has been performed with respect to the internal user management. The current implemented version already contains extensive logic to handle the creation of Fabric users, along with the definition of access rights on the stored logs on a per channel basis. Upon integration, external to the Auditing mechanism users will be mapped to internal users, creating a seamless end to end authorization channel. To this end, additional REST endpoints are implemented (but not currently exposed) that would allow user administration from the HEIR platform or the clinical organization side.

The identity management capabilities that Fabric offers, along with the fact that different Fabric channels may be instantiated, over which different peer nodes communicate and sync their ledgers, facilitates the establishment of a complex blockchain network, where multiple organizations' internal audit logs may reside. While the auditing mechanism applies for now to the NSE/NOKLUS Use Case only, our current implementation allows to easily plug the Auditing mechanism wherever the Privacy Aware Framework resides and functions.

On top of that, a distinct channel may additionally be established and configured, facilitating audit logs of access to data which are not specific to an organization. Our plans for the immediate future include the log collection of access attempts to the HEIR Observatory. With respect to the identity management, either for each organization or centrally for the Observatory, different and independent groups of users can be registered with appropriate access rights to the audit logs.

4 HEIR Facilitators in Use Cases

At the time of writing most of the HEIR facilitators have been ~~relialized~~ released and integrated into the HEIR integrated framework (intermediate version) within the relevant HEIR Use Case infrastructure. In the latter phase of this Project, the plan is to have all of them deployed to all HEIR Use Case infrastructure environments. More details on how each module is released and deployed in each Use Case is described below.

The **HEIR Vulnerability assessment module** (see sect **Erreur ! Source du renvoi introuvable.**) is released as a novel tool that monitors efficiently the complex health infrastructures and analyses them for different threats. It reports intelligent real-time security, privacy and data protection warnings, utilizing the HEIR Interactive Forensics Module. Currently, it is deployed at the (i) PAGNI, (ii) HYGEIA and (iii) CROYDON Use Case HEIR infrastructure environment and it is foreseen to be deployed at (iv) NSE pilot HEIR infrastructure as well.

The **HEIR Interactive Forensics module** (see sect. **Erreur ! Source du renvoi introuvable.**) is realized as two sub-module components that utilize Machine Learning models in order to provide forensics visualisation services. Currently, it is deployed at (i) PAGNI, (ii) HYGEIA and (iii) CROYDON Use Case HEIR infrastructure environments and it is foreseen to be deployed at the (iv) NSE Use Case when the environment can support this.

The **HEIR ML anomaly detection and threat classification module** (see sect.**Erreur ! Source du renvoi introuvable.**) is released as a tool that will adapt existing ML models utilized in anomaly detection and/or threat classification that match the requirements of the Healthcare systems. Currently, it is used at the (i) PAGNI Use Case, and foreseen to be deployed at (ii) HYGEIA, (iii) CROYDON and (iv) NSE Use Case sites.

The **HEIR Collaborative privacy-aware framework** (see sect **Erreur ! Source du renvoi introuvable.**) is released as a tool that provides 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, and provides policy-driven control of data accessed from a server, token authentication for data requests, as well as logging data transaction requests to Kafka. It is foreseen to be used at the (i) NSE Use Case and also demonstrated as the (ii) Observatory Use Case.

5 Conclusion

In this document, the current developmental status of the HEIR facilitators' components is presented. The Facilitators follow a modular approach for the realization of the Use Cases and the multi-layered hierarchical architecture that the HEIR product suggests. The final HEIR Client services are supported by the facilitators, including the threat hunting module and the collaborative privacy-aware framework for sharing and processing healthcare sensitive data.

The development efforts are organized in three cumulative stages, the delivery of the MVP, the upgrade to the 1st complete version and the final version of the facilitators packages, and the full realization of the HEIR services for a complete cybersecurity environment of the healthcare domain.

The next steps will be concentrated on the completion of HEIR's final complete version of the services package. This will be based on the progress of the technical work packages, as well as the feedback to be obtained from this development cycle and the evaluation process completed in D6.1: HEIR Demonstration - initial execution and evaluation. The next completed version of services will be reported in D2.3 - "The HEIR facilitators package: Final complete version".