D5.5

Best practices for maintaining/operating the framework in the long-run– TRL7

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.	D5.5
Work package	WP5
Due date	08-2023 – M36
Actual submission date	30/08/2023

Deliverable lead	WCS
Editors	Dimitrios Karamitros, Ioannis Ladakis
Contributors	P.Rodosthenous, N.Dimakopoulos, G.Tsakirakis (ITML), S.Athanassopoulos (HYGEIA), D.Deyannis, I.Xanthopoulos, M.Smyrlis, A.Zacharakis (STS), E.Salant (IBM), D.Tsolovos (STELAR), M.Darwish Khabbaz (TUD), I.Ilie (SIE), B.Prelipcean (BD), C.Ballas, A.Alexopoulos, A.Sotiropoulos (AEGIS)
Reviewers	M.Muzny (NSE), G.Tsakirakis (ITML)
Dissemination level	PU
Revision	FINAL 1.0
Keywords	Maintenance, Deployment manuals, User manuals, Legal, ethical and privacy concerns, Usefulness verification

Abstract

This deliverable outlines the objectives and key efforts aimed at supporting the commercialization of HEIR, a solution tailored for the healthcare industry and similar end-to-end environments. The focus is on establishing best practices for the long-term maintenance and operation of the framework, initially addressing bugs and gradually implementing a structured maintenance program. The validation of solutions' effectiveness and scalability is critical, achieved by integrating HEIR into a real-life environment while ensuring legal, ethical, privacy, and accessibility considerations. The development of deployment and user guides is highlighted as a crucial resource ensuring the applicability and usability of HEIR solution. Overall, the document includes all the activities taken in order to promote and communicate a stable and reliable solution for the healthcare industry and beyond, revolutionizing the approach to address complex challenges responsibly.

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

Executive Summary

This document's objective is to support the commercialization efforts of HEIR by delivering a robust and dependable solution tailored for the healthcare industry and other comparable end-to-end environments facing similar challenges. Based on task 5.3, the quality tests of T6.3[1] and the demonstrator's execution, our efforts focus on designing and detailing the best practices for the long-term maintenance and operation of the framework, with a focus on handling software updates. Initially, maintenance will be performed on an ad-hoc basis to address any bugs discovered during internal testing, while meticulously logging and analysing all modifications to establish an effective ongoing maintenance program.

The validation of the effectiveness of the proposed solutions and their efficiency in large-scale utilization is crucial to ensure the scalability of the existing work. To achieve this, we will integrate the HEIR framework into a real-life operating environment, allowing us to assess the scalability of its building blocks and demonstrate a full stack showcase of the solution's capabilities. In doing so, we will meticulously address legal, ethical, privacy, operational, and accessibility concerns to ensure the solution's suitability for real-world scenarios.

An integral part of our project is the development of an end-user guide, providing comprehensive instructions for installing, deploying, and utilizing the HEIR framework and its components. This guide will serve as a crucial resource to facilitate smooth implementation and utilization by end-users.

Through these efforts, the ultimate goal of this document is to establish a stable and reliable solution that meets the unique requirements of the healthcare industry and other end-to-end environments. The HEIR framework is poised to revolutionize the way such industries tackle challenges, offering an innovative and comprehensive approach to address complex issues effectively and responsibly.

[1] Task 6.3 (WP6): Evaluation and impact analysis

Table of Contents

List of Figures

1. Introduction

1.1 Scopes and objectives

The primary aim of this document is to support the commercialization endeavours of HEIR by introducing a resilient and dependable solution tailored for the healthcare sector and analogous end-to-end environments grappling with similar challenges. Our primary focus revolves around shaping and articulating optimal protocols for the enduring maintenance and operation of the framework, with specific attention to managing software updates. In the initial phases, an overall maintenance plan will be designed to address any bugs discovered during internal testing as well as to ensure the continuous monitoring of any updates and upgrades of the solution.

Central to the sustainability of our efforts is the validation of the proposed solutions' efficacy. To realize this objective, the integration of the HEIR framework within a real-world operational context is vital. Throughout this process, we are committed to thoroughly addressing concerns spanning legality, ethics, privacy, operational logistics, and accessibility, affirming the adaptability of the solution to real-world settings.

An indispensable component of our project involves the creation of deployment and user guides/manuals for each one of the HEIR's components, furnishing comprehensive instructions for the installation, deployment, and effective utilization of the HEIR framework and its constituent elements. This guide assumes a central role in facilitating the implementation process and enhancing the accessibility of the solution for end-users.

1.2 Document structure

The structure of the document is the following:

- Section 2: provision of an overview of best practices for maintenance and operational planning and presentation of a maintenance plan for HEIR framework
- Section 3: presentation of results regarding the verification of framework's usefulness based on the quality tests of T6.3
- Section 4: presentation of legal, ethical, privacy and operational concerns
- Section 5: provision of installation and user manuals per HEIR component
- Deliverable conclusion

1.3 Relation to other tasks and work packages

This deliverable is based on the technical work that was part of all Tasks of WP2 and WP5 for developing deployment/installation manuals and user manuals for all components of HEIR framework. Moreover, it is closely related to T6.3 for usefulness verification and T7.5, T7.6 for the discussion around legal, ethical and privacy concerns.

2. Maintenance planning

The initial production release of a software system is always a very important step. However, it would be naïve to assume that this is the end for the product development teams. This very first release of software always requires a significant amount of attention to make sure that it not only remains functional but also evolves by means of bug fixing, functionality adaptations, and new feature incorporation.

The HEIR platform developed in the scope of the project is a quite complex and distributed software system that has been thoroughly tested and evaluated during the project's pilot studies by different healthcare organizations. Its complexity is only further increased because its subcomponents have been developed by different stakeholders, while their configuration, coordination and integration into a well-orchestrated system has been challenging.

A maintenance plan would outline the comprehensive strategies, processes, and resources required to ensure the effective and efficient maintenance of the HEIR framework throughout its lifecycle. It is a crucial part for maximizing system performance, addressing software issues, and meeting the evolving needs of users and stakeholders. The maintenance plan's primary objective is to maintain and enhance the software's functionality, reliability, and usability in the long term.

2.1 Software maintenance plan goals

Overall, a well-designed and carefully thought of maintenance plan aims to achieve the following goals:

- Minimize system downtime by promptly addressing software issues and applying necessary fixes.
- Enhance system performance through performance optimization and tuning.
- Ensure the software remains compatible with evolving technology platforms and environments.
- Implement security updates and patches to protect against vulnerabilities and cyber threats.
- Continuously improve the software by incorporating user feedback and implementing new features.
- Adhere to regulatory and compliance requirements specific to the software domain.

A solid maintenance plan would need to not only take into consideration all the software components and modules of the platform software, both on an individual level but, but additionally all the backend infrastructure, databases, user interfaces, and integrations required to bring the HEIR platform to life. The skeleton for HEIR's framework maintenance plan will be discussed over the following subsections, initially presenting the concerns it must address, then its design procedure as well as the best practices it must encompass.

2.2 Types of software maintenance

Software maintenance can be performed for a variety of purposes and reasons. All systems, regardless of their current lifespan, undergo maintenance actions on a regular basis. During the lifetime of software, the type of maintenance required can vary depending on its nature. Maintenance tasks can range from routine bug fixes reported by users to significant events that involve large-scale maintenance activities. Some of the types of maintenance that would be applicable to the long-term operation of the HEIR framework. are[2]:

- Corrective Maintenance: Addressing reported bugs, errors, or malfunctions in the software. This involves thorough debugging, troubleshooting, and applying patches or fixes.
- Adaptive Maintenance: Modifying the software to accommodate changes in the technology landscape, such as updates to operating systems, libraries, or hardware platforms.

[2] https://www.tutorialspoint.com/software_engineering/software_maintenance_overview.htm

- Perfective Maintenance: Enhancing existing features, optimizing performance, and improving user experience based on user feedback and emerging needs.
- Preventive Maintenance: Taking proactive measures to prevent potential issues, such as conducting routine system checks, optimizing resource utilization, and applying security patches.

Figure 1 presents in a nutshell how a request for a change in a production system is classified and always ends up being an action contained in a corrective, preventive, adaptive, or perfective maintenance plan.

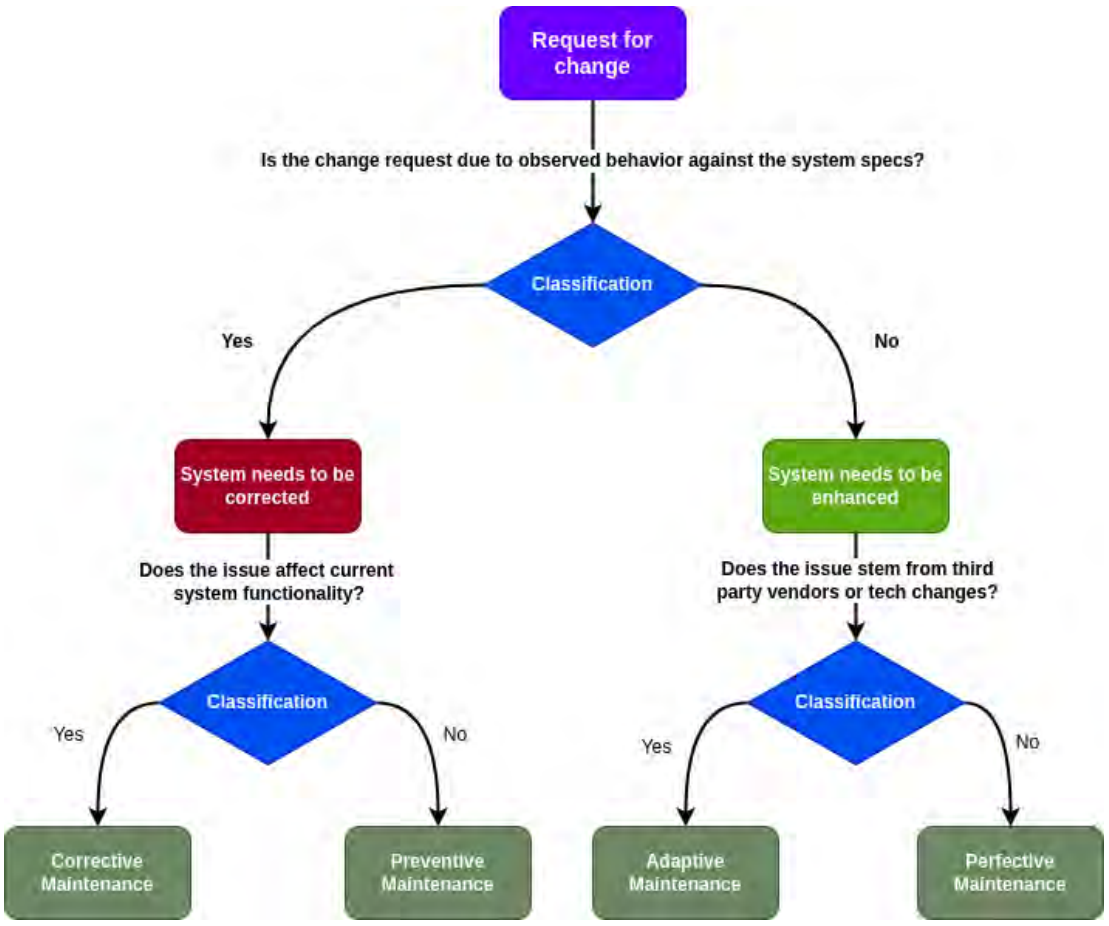

Request for change
Is the change request due to observed behavior against the system specs?
Classification
Yes
No
System needs to be corrected
System needs to be enhanced
Does the issue affect current system functionality?
Does the issue stem from third party vendors or tech changes?
Classification
Classification
Yes
No
Yes
No
Corrective Maintenance
Preventive Maintenance
Adaptive Maintenance
Perfective Maintenance

Figure 1: Types of software maintenance

2.3 The concerns of maintaining a software framework in the long run

An extensive maintenance plan is required to address a range of concerns that are critical for the successful management and sustainability of a software application or framework in the long run. These concerns encompass various aspects, including technical, operational, and strategic considerations. The key concerns of such a maintenance plan are:

Software Stability and Reliability

Maintaining software stability and reliability is a primary concern. No system is designed, developed, and rolled out to production with a short life in mind. The maintenance plan should outline procedures for identifying, addressing, and resolving software bugs, errors, and malfunctions. It should include strategies for conducting comprehensive testing, debugging, and applying necessary fixes to ensure that the said system operates reliably and consistently.

Performance Optimization

The optimization of any software system in terms of performance is essential to deliver a seamless user experience. The maintenance plan should define performance metrics, set targets, and establish monitoring mechanisms to identify performance bottlenecks, optimize resource utilization, and enhance system responsiveness. It should also address scalability concerns to ensure that the software can handle increasing user loads or growing datasets efficiently.

Security and Vulnerability Management

Addressing security concerns is crucial to safeguard the software and protect user data. The maintenance plan should encompass strategies for identifying and addressing security vulnerabilities as they occur, applying patches and updates, and ensuring compliance with relevant security standards and regulations. It should include routine security audits, vulnerability assessments, and the implementation of appropriate security measures to protect against potential threats.

Technology and Compatibility

Based on today's extremely fast pace with which software technologies advance, keeping the software components of a system up-to-date and compatible with evolving technology platforms is vital. The maintenance plan should outline procedures for assessing the impact of new technological versions, operating system updates, or hardware changes on the software. It should include strategies for adapting the software to ensure compatibility and interoperability, such as updating libraries, APIs, or dependencies.

Change Management

Managing software changes is essential to maintain stability and prevent unintended consequences. A maintenance plan should incorporate change management processes, including version control, configuration management, and documentation updates. These processes should take into consideration the fact that different software components which are part of the system are offered by different vendors, as is the case in HEIR. Therefore, it is imperative that the different component vendors (or project partners in the HEIR example), keep an always up-to-date record of changes performed, including changelogs, configuration, or deployment/installation changes.

In a highly complex system like HEIR, where different components interact to provide a unique service within the HEIR premise, isolated changes can potentially have a negative impact on the system performance and optimal functional behavior. Therefore, procedures for assessing the impact of changes need to be in place and respected, conducting thorough testing, and ensuring proper deployment and rollback mechanisms.

Continuous Improvement and Innovation

Promoting continuous improvement and innovation is essential for the software's long-term success. The maintenance plan should include mechanisms for gathering user feedback, tracking emerging needs, and incorporating enhancements and new features. It should establish processes for soliciting ideas, conducting research and development, and aligning the software with evolving industry trends and best practices.

User Support and Issue Resolution

Addressing user issues and providing effective support is a critical concern. On the client facing front, the maintenance plan should define channels for users to report problems, specify response times, and establish procedures for issue triage, categorization, and resolution. It should include a knowledge base, FAQs, and self-help resources to empower users to resolve common issues independently. The plan should also outline escalation procedures for critical or complex issues, ensuring timely and effective support.

From the administrative point of view, support and administrator roles need to be assigned to dedicated, well-known stakeholders in the organizations that adopt HEIR in their daily routines and processes (operations and support personnel).

Business Continuity

Ensuring business continuity is a strategic concern. The maintenance plan should address disaster recovery, backup strategies, and contingency planning. It should include procedures for scheduled and recurring data backups, replication, and restoration processes to mitigate the risk of data loss or system downtime. Furthermore, the maintenance should also consider infrastructure made redundant or obsolete over time and failover mechanisms. A significant aid towards this direction may be the definition and execution of business continuity exercises to ensure that the HEIR system, both on a component level but mainly and most importantly on a global level, can recover rapidly from disruptions, no matter how catastrophic they might be.

Cost effectiveness

Any software platform needs to meet its business requirements and serve the purpose it was built for but to what financial cost? Although it does not seem or feel like it, the cost-effectiveness of a platform is crucial to its long-term success and sustainability. A final responsibility of the maintenance plan is to always make sure that a platform's operational costs are kept at a minimum without compromising the platform's effectiveness.

2.4 Design of the maintenance plan

A maintenance plan exists to address all required activities that will keep a software system or platform fully operational and fulfilling its purpose in the long run.

For the design of the HEIR project's maintenance plan, we adopted IEEE's framework for sequential maintenance activities[3]. This framework includes a series of steps (Figure 2):

[3] Mamone, S. (1994). The IEEE standard for software maintenance. *ACM SIGSOFT Software Engineering Notes*, *19*(1), 75-76.

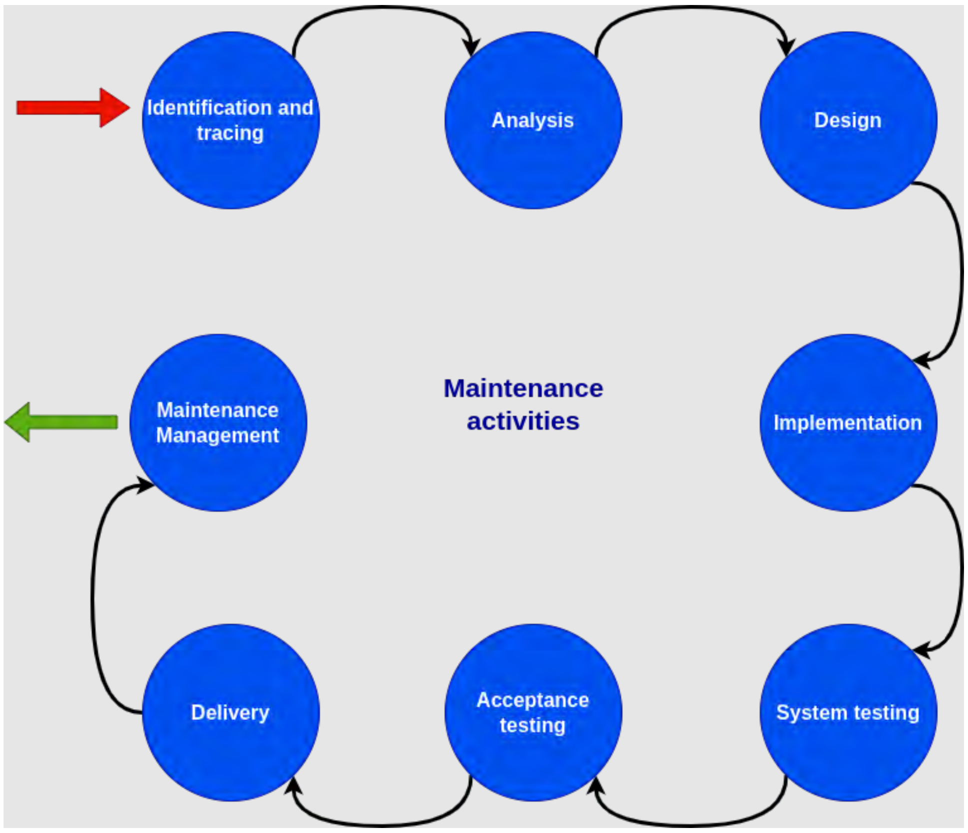


Figure 2: Maintenance process activities

In the remainder of this section, the phases of maintenance activities are described, along with best practices for each phase. Additionally, best practices that concern the complete maintenance cycle will be reported for reference.

- **Identification & Tracing Phase** – This phase encompasses tasks related to identifying the need for modifications, be it updates or fixes to the operational system. This phase can be triggered either by a user experiencing issues while operating the system or an issue detected automatically through logs or error messages. It is in this phase that the type of maintenance to be applied is classified in one of the types described in Section 2.2.

- **Analysis Phase** - The analysis of system modifications to be applied considers their impact on various aspects, such as system safety and security, interoperability, and integration. If the potential impact is deemed insignificant, then the whole maintenance cycle is limited in duration. If the potential impact is deemed significant, alternative solutions may need to be explored. The identified set of required modifications is then transformed into detailed requirement specifications. Additionally, the cost of the modifications or maintenance is carefully analyzed, and a thorough estimation is conducted.

- **Design** - The design of new modules, intended for replacement or modification, is based on the requirement specifications established in the previous stage. In the case of production bug hot fixes, the design phase may be skipped. Test cases (unit, integration, regression) are developed to ensure proper validation and verification of the redesigned modules or the bug fixes at hand.

- **Implementation** - The coding of the new modules is carried out based on the structured design developed in the previous step. Each software engineer assigned to the task is responsible for conducting unit testing simultaneously with the coding process.

- **System Testing** - Integration testing is performed to ensure seamless interaction among the newly developed, modified, or patched modules. Additionally, integration testing is conducted between the target modules and the overall system. Subsequently, the system undergoes comprehensive testing as a unified entity, following regression testing procedures. A staging environment where all these tests will be performed, resembling the exact same conditions of the production system, is crucial for the success of this phase.

- **Acceptance Testing** - Once the internal testing of the system is completed, it proceeds to undergo acceptance testing with the involvement of dedicated actors that act as end users. During this stage, if users encounter any issues or concerns, they are promptly addressed or recorded for future iterations.
- **Delivery** - Following the successful acceptance test phase, the system is deployed throughout the organization either through a small update package or a fresh installation. At the client's end, final testing is conducted after the software is delivered. If necessary (potentially due to a significant change in the systems functionality), training must be provided, while the user manual is updated to ensure smooth adoption and usage of the system.
- **Maintenance management** - Configuration management plays a vital role in system maintenance, ensuring the effective control and management of system configurations. To facilitate this process, version control tools are utilized, enabling the control of different versions, semi-versions, or patches within the system.

2.5 Maintenance plan best practices

This section reports on the best practices regarding the design of a maintenance plan and procedure, which will be included in HEIR's plan to ensure the operational success of the framework after the end of the project.

The maintenance plan must define all processes to handle software updates, patches, and new feature deployments. It defines versioning schemes and release cycles, considering factors such as bug fixes, feature enhancements, and compatibility requirements. The plan outlines the stages of release, including development, testing, quality assurance, and production deployment. It ensures that release packages are well-documented and thoroughly tested to minimize the risk of introducing new issues, among others.

2.5.1 Alignment of maintenance plan with business goals

The organization that adopts the HEIR framework clearly has specific expectations and goals that they aim to achieve through the framework adoption in their daily processes. Maintenance activities must first and foremost align with the organization's goals.

Different organizations adopting HEIR may have different goals; others prefer having a robust and always evolving cyber- security protection system that manages to track down new threats. Others need a system that is cost-effective and manages to achieve the same goals over time with little to none maintenance. Maintenance metrics and KPIs need to be carefully selected, aligned with the business goals and objectives, to track progress and measure success. Finally, the critical points of the system, be it software or infrastructure, need to be identified; tackling issues that these points present may be the key to successful maintenance, aligned with the business objectives and KPIs.

2.5.2 Regular updates, upgrades and patches

Regularly updating and upgrading software is a critical process that ensures its ongoing security, reliability, and performance. In addition to planned upgrades and updates, the system needs to be regularly scanned for emerging security threats vulnerabilities. Even though the HEIR framework attempts to tackle cybersecurity threats in environments where data access and management is critical, it is a software system of its own. Regular security scans on all its subcomponents and their interfaces, as well as on the infrastructure that HEIR is running on, with industry established tools (for example Snyk.io) can uncover many underlying security threats and vulnerabilities that are found every day. There are several different available open vulnerability registrars where system administrators or software architects and engineers can look up threats and their fixes, as well as mitigation strategies until a fix on a commercially available or open-source dependency is resolved and fixed. Identification of bugs and security vulnerabilities and their subsequent classification based on their criticality to the security, stability, or proper functionality of the system results in tightly timed update and upgrade plan.

A regular update schedule with well-defined timings needs to be established and followed by all stakeholders in HEIR, be it the original developer partners or the IT personnel in the organization where HEIR is deployed and used.

A manual upgrade or update plan on its own is neither time nor cost effective. Automated update mechanisms need to be established and adopted, streamlining the process, making it seamless for users and system administrators to receive and install updates.

2.5.3 Effective maintenance role assignments and processes

Early in the software development lifecycle, maintenance needs to be considered as if it were one of the system's original requirements. Building a software system and then attempting to create an appropriate maintenance plan is becomes a burden. Not only the maintenance aspects and goals need to be clearly defined, but the teams also that will be participating in the complete maintenance phase of the HEIR framework in production need to be clearly defined, along with their roles, their responsibilities, and their accountabilities. Maintenance can be performed by different stakeholders, including the original development team, an in-house maintenance team, or a third-party maintenance provider. It is therefore crucial that maintenance teams have clear goals and responsibilities, as well as all the necessary training materials of the system parts they need to maintain. Training materials include production level operation and deployment manuals, component specifications and configurations. For the maintenance tasks performed regularly by these teams, checklists and maintenance reports must be established.

Key roles assigned to a maintenance team include the maintenance manager that oversees the overall maintenance activities, sets priorities, and ensures adherence to the defined plans. Then, the maintenance team, comprised of skilled software developers, testers, system administrators, and support staff responsible for executing maintenance tasks. Finally, the users or user representatives that are responsible for reporting issues, providing feedback, and collaborating with the maintenance team.

2.5.4 Proactive monitoring

Proactive monitoring is a crucial aspect of software maintenance. Automated monitoring tools with embedded alerting capabilities can provide quick insights on the operational status of the system and help identify potential issues and optimizations that can be incorporated in the next maintenance cycle. The issues identified with an automated monitoring solution can range from performance bottlenecks under heavy load to unhandled exceptions. Informed decision-making on emergency, planned or future improvements can only stem from accurate monitoring of the running system.

Appropriate metrics need to be established for proactive monitoring to be effective. Performance hotspots, resource utilization, memory usage, response times that need to be kept to a minimum are such metrics. Metrics can also be established and tracked for the infrastructure on which the HEIR framework is operating. Key performance indicators (KPIs) are established to measure system reliability, responsiveness, uptime, and user satisfaction. Monitoring tools and automated alerts are employed to proactively identify and address performance bottlenecks, anomalies, and potential issues.

2.5.5 Quality assurance

Before deploying updates to production systems, testing them in a controlled environment is crucial. By testing updates in a staging or testing environment, any compatibility issues, conflicts, or unexpected behavior can be identified. This helps at the assessment of the impact of updates and the validation of their effectiveness before applying them to production environments. Thorough testing validates that the updated software functions as expected and doesn't introduce new problems or regressions. A complete quality assurance cycle must be defined and applied during all phases of the software development lifecycle, from design to release. Quality assurance must go beyond the mandatory unit testing that software developers alone can create and support. Complete end-to-end, integration and regression tests must be defined, executed and regularly updated to properly assess any change, minor or major in the component or the system under test.

2.5.6 Documentation

In software engineering, comprehensive documentation may be even more important than the system being developed. From the original system requirements and specifications that drove the design of the system, all the way down to the code developed and the operation/deployment manuals, everything must be documented. Yet today, even in large scale production systems, we are still a long way to go from having an established mentality for properly structured and maintained documentation, where people involved in the design and implementation of the system need not be persuaded on the documentation's importance.

Poorly documented systems make it very difficult to understand how they are working, what their pain points are and how they are orchestrated among others. For an effective maintenance plan for HEIR, all its subcomponents need to be documented thoroughly. Documentation must include user guides, i.e. detailed instructions and information to assist users in effectively utilizing the software's features and functionalities, technical manuals, i.e. technical documentation, including system architecture, API references, and configuration guides, release notes. Finally, documentation must highlight changes, bug fixes, and new features introduced in each software release, including the date, nature of the change, and individuals involved. The aforementioned are all critical aspects of components that need to be documented since their inception. Documenting changes, known issues, troubleshooting procedures and other critical aspects greatly simplifies the corrective or preventive processes of maintenance. As a component undergoes changes over time, so are the teams that are responsible for their management and maintenance. Knowledge transfer to newly appointed team members without comprehensive documentation is impossible to say the least.

The maintenance plan emphasizes the importance of comprehensive documentation to support maintenance activities.

As the components evolve, so should their accompanying documentation material be updated regularly to reflect all the different recent changes made. Outdated documentation is useful to none.

2.5.7 Version control and management

Apart from a solid documentation, version control and configuration management systems can be employed by teams to track changes, revert to previous versions if needed, and collaborate effectively, ensuring transparency and accountability throughout the maintenance process.

The establishment of formal procedures to manage and document changes is crucial. Proper version documentation, changes tracking, and approval procedures ensure that updates are applied in a controlled and consistent manner. This reduces the risk of unintended consequences or conflicts with existing configurations. Maintaining accurate and up-to-date documentation of the whole procedure, including applied updates, configuration changes, testing results, and known issues, is a good practice as it serves as a valuable resource for future updates and troubleshooting.

2.5.8 Bug tracking and defect management

A major channel facilitating the identification of the different system's defects is user feedback. Monitoring may prevent quite a lot of defects, but it is the users' perception of the system's operational status that provides valuable insights as to what kind of problems they are experiencing while using the HEIR system. The definition of support and escalation procedures is crucial to ensure timely assistance and issue resolution. Users can report software issues through designated channels, such as a help desk ticketing system or a dedicated support email. The plan outlines the process for issue triage, categorization, and prioritization. The critical or complex issues reported by users are escalated to higher-level support or development teams for expedited resolution. It is the maintenance plan that must define metrics pertaining to the maintenance team's response times versus the time a defect was reported via the defined reporting channels.

2.5.9 Risk management

A maintenance cycle is always a risky operation. All critical components, flows and operations and integration interfaces would need to be identified before the HEIR framework is released for production. It is only then that a risk registry can be compiled, allowing for the prioritization of bugs and defects during scheduled or emergency maintenance actions.

2.5.10 Data retention and backup

Data retention and backup schedules must be defined before the system launches into production. It is imperative that data retention policy rules are defined, as well as a strict database backup schedule, with clear rules on the timings, frequency, and backup retention windows. In the case of a catastrophic event, a tightly defined retention plan allows for a quick recovery from a data loss scenario, minimizing the effect of critical data being lost or corrupted.

2.5.11 Security compliance

The update/upgrading planning requires cautious steps regarding the existing data and configuration. Taking backups is a necessary precaution, as this ensures the restoration of the system to a previous working state in case of any issues during the update process, minimizing data loss and disruption. Moreover, compatibility and dependencies issues should be considered, as well as security-related improvements to address vulnerabilities that can be exploited by malicious actors. Vulnerabilities may come from a variety of different sources, from dependencies used during the development of a component to the access control schemes of the underlying infrastructure the system is running on.

It is crucial that security is at the top of priorities when a component is developed. Best practices on addressing well-known vulnerabilities are available and the development team must always make sure to align with these practices from the early days of the component. The interfaces of the components and the integration points must also be secured, eliminating the risk of data breaches or leaks.

Industry security standards exist that the development and maintenance teams need to adhere to. Adherence must be checked regularly on every minor or major change in the component. The initial security certification (base on available certifications and standards) provides the confidence to the end user that what they are operating is secure and their renewal solidifies that despite changes and reconfigurations the components of the system are still following best practices, guidelines, and industry standards. Each component owner must seek to acquire the security certifications which are relevant for their component and for the HEIR system.

The maintenance plan must also include a regular penetration tests schedule to be executed. It is preferable that the penetration testing processes are undertaken and initiated preferably by external to HEIR stakeholders. The penetration test process itself must be part of the maintenance plan. Their outcomes must be made quickly available to the development teams and the system administrators, so that not only corrective actions can be applied in the form of maintenance, but they are kept safely as part of the system's documentation for future reference.

Under perfect conditions, systems do not change, and a maintenance plan defined during the initial launch would run like clockwork. Unfortunately, this is never the case. As the software, technologies, infrastructure, business needs evolve, the same applies for maintenance plan; it must never be considered as being set in stone. The maintenance plan needs to be regularly revisited, evaluated and aligned with the software it serves.

3. Verification of framework usefulness

In terms of expanding the applicability of HEIR framework, an assessment of the framework is required for the verification of its usefulness. This part of the work is covered by T6.3 and is reported in D6.3. The key factors that were assessed in terms of usefulness are:

- Effectiveness: It involves evaluating whether the framework can solve the problems it was designed to address and achieve the desired outcomes.
- Practicality: Practicality involves examining how easily users can adopt the framework, its learning curve, and whether it streamlines existing processes.
- Suitability: It involves understanding if the framework's features align with the requirements and preferences of its intended users.
- Performance: This includes evaluating processing speed, resource utilization, and how well the framework scales with increased demand.
- Reliability: Reliability verification involves testing the framework under various scenarios to identify potential issues and ensure it can be trusted for critical tasks.
- Scalability: Scalability verification ensures that the framework can handle increased usage without a significant drop in performance.
- User Satisfaction: This involves obtaining insights into user experiences, feedback on usability, and the overall satisfaction with the solution.
- Real-world Validation: This involves deploying the framework in production or pilot environments to understand how it performs with real data and user interactions.

The aforementioned factors were evaluated via questionnaires and short interviews[4] that were provided to the pilots and third-party stakeholders (IT and non-IT experts) as well in terms of training days organized by HEIR consortium. Real-world validation is continuously monitored via the deployment of the framework's components on the 4 pilots of HEIR project.

In this section, a summary of the D6.3 results is presented.

3.1 End-used assessment results

3.1.1 User Experience

The utilized questionnaire was User Experience Questionnaire (UEQ[5]) to assess six usability and user experience aspects, namely Attractiveness, Perspicuity, Efficiency, Dependability, Stimulation, and Novelty. Seven responses were collected from personnel not adequately familiarized with HEIR solution. The highest scores were achieved for Stimulation, Novelty, and Dependability aspects signifying a tendency towards high satisfaction, while rather neutral score is achieved for Perspicuity. None of the items receives a negative rating, indicating the maturity and the robustness of HEIR solution. A more thorough investigation of the UEQ's responses shows that the respondents perceived HEIR framework as an inventive, supportive, good, meeting expectations, creative, exciting, interesting, organized and innovative solution, while, on the other hand, they pointed out difficulty to learn the functionalities of the framework, unpredictability, slowness, complicated, unlikability, usual, not secure, inefficient, confusing and unfriendly as variables with lower mean scores and, thus, as aspects that should be investigated more.

3.1.2 Technology & Operational Acceptance

Technology and operational acceptance were evaluated via short interviews – open questions by the four pilots based on their experience during the deployment and testing of the framework. It is worth mentioning that the whole HEIR system was not installed across all sites, permitting only the testing of specific and separated components per pilot. Thus, the reports confirmed that the tested components were easy to use and had no issues in integrating with the respective infrastructure, but it is problematic that a complete assessment of the whole system could not be achieved. However, having tested various

[4] D6.3. Assessment report and impact analysis
[5] https://www.ueq-online.org/

components and evaluating their functionalities and the ability to be integrated in various systems, the results are promising regarding the efficiency and applicability of HEIR framework as a whole.

3.1.3 Impact

The impact of the installed components was assessed via short interviews – open questions that were collected from people with some experience with the HEIR framework for a relatively short period of time. The results indicate that the HEIR framework modules would potentially have a great impact in:

- Enhancing cybersecurity and protection against cyberattacks
- Enhancing the protection and confidentiality of sensitive data protection
- Continuously monitoring the availability and reliability of medical systems and devices

In general, it was recorded a positive reaction regarding the potential impact of HEIR framework.

3.2 Third-party stakeholders feedback results

Third-party stakeholders' feedback was collected during dedicated training-info days from IT and non-IT experts in healthcare centres and hospitals via short, customized questionnaires. During these sessions, participants were tasked with evaluating the usefulness of particular HEIR modules, expressing their overall impressions, and identifying any lacking features. Additionally, they were requested to rate the importance and effectiveness of the presented content.

3.2.1 IT experts feedback on HEIR solution

The majority of the experts (62,5%) are not familiar with the management of cybersecurity systems or with actions to be taken as responses to cybersecurity incidents. Their answers per HEIR module/component tended to be positive:

- 59% of the participants reported that HEIR's Threat Hunting Module is better compared to their existing solution, with 33% reporting that they found it equivalent.
- 50% reported a supremacy of HEIR's Anomaly Detection Module, while 34% found it equivalent to their existing solution.
- HEIR's Observatory was assessed as quite useful for aiding risk assessment and determination of remedial actions with 69% answering over 4 in a likert-scale from 1 to 5.
- 75% of the respondents were in favour of HEIR's Privacy Aware Framework as a solution that contributes at defining who is entitled to the data of their department.

In general, only 17% of the participants considered that HEIR solution will not, moderately or significantly, improve their performance, while the overall perception of HEIR solution was positive or very positive for 65% of them. Some potential issues were reported, like the real-world communication with FHIR, the actual interoperability of FHIR, and the need for a push notification/alert mechanism.

3.2.2 Non-IT experts' feedback on HEIR solution

50% of non-IT experts reported to have an administrative role within their organization, with the remaining 50% belonging to the clinical staff. Even though not so familiar with cybersecurity issues and potential risks, 68% of the participants responded positively regarding the potential usefulness of Private Aware Framework. Furthermore, 72% expected a positive impact on their daily tasks with the use of the HEIR solution, while 83% expressed a positive perception of the HEIR solution.

4. HEIR framework legal, ethical and privacy concerns

As HEIR project progresses, it becomes crucial to address several key areas of concern to ensure its responsible and ethical implementation. Legal considerations play a pivotal role in ensuring compliance with regulations and safeguarding the rights of stakeholders involved. Ethical implications arise from the potential impact of the project on individuals and society as a whole, necessitating a thoughtful and principled approach. Privacy concerns are of paramount importance, especially when dealing with sensitive healthcare data, demanding robust measures to protect patient confidentiality and data security. This section discusses the legal, ethical and privacy related concerns regarding HEIR framework.

4.1 Legal aspect

Since the HEIR project deals with cybersecurity on hospital environments, the main legal issues that need to be addressed are the following:

- Data privacy and protection (laws and regulations include the GDPR, e-Privacy directive and other regional and national laws that apply to each specific partner country),
- Security measures with laws and regulations such as the NIS (2) Directive, the eIDAS regulation[6] etc.,
- Possible adherence to different Human Rights regulations such as the European Convention on Human Rights (ECHR) and the European Charter of Fundamental Rights (CFREU) also needs to be addressed.

4.2 Ethical aspect

The main ethical aspects that a project such as HEIR needs to address include:

- **Confidentiality and privacy:** Protecting confidentiality and privacy requires a commitment to robust security measures to shield sensitive information from unauthorized access. This involves continuous monitoring and updating of security protocols to align with the ever-evolving cybersecurity landscape.
- **Security:** The HEIR project implements state-of-the-art measures to ensure data security, while also ensuring that medical information and care remain accessible when needed.
- **Patient consent and autonomy:** The HEIR project prioritizes patient consent and autonomy by ensuring that all individuals are provided with clear, concise information about how their data will be used, and that consent is obtained voluntarily. Respecting patients' rights to control their personal information and acknowledging their ability to withdraw consent at any time is crucial.
- **Transparency:** Ethical conduct demands transparency in operations and a commitment to accountability. The HEIR project has measures in place to clearly communicate its data usage policies to all stakeholders and establish oversight mechanisms that include periodic audits or reviews.
- **Research Ethics:** The project has aligned itself with ethical guidelines for medical research. This includes obtaining proper ethical approvals where needed and maintaining transparency in research methodology.
- **Collaboration and Third-Party Relationships:** Ethical collaboration involves assessing third-party vendors and collaborators to ensure alignment with ethical principles. The HEIR project establishes clear agreements that do not compromise patient rights or confidentiality. Collaborative engagements have been built on shared values and a commitment to ethical conduct.

4.3 Privacy aspect

The main privacy aspects that a project such as HEIR needs to address are based on the 7 established principles of the GDPR:

[6] https://digital-strategy.ec.europa.eu/en/policies/eidas-regulation

- **Lawfulness, fairness and transparency:** Processing must be lawful, fair, and transparent to the data subject. This means the person whose data is being processed should understand how and why it is being used.
- **Purpose limitation:** Personal data must be collected for specified, explicit, and legitimate purposes and not further processed in a manner that is incompatible with those purposes.
- **Data minimization:** Processing of personal data must be adequate, relevant, and limited to what is necessary in relation to the purposes for which they are processed. This is about collecting only the data that is strictly required for the intended purpose.
- **Accuracy:** Personal data must be accurate and, where necessary, kept up to date. It must be corrected or deleted without delay when inaccurate.
- **Storage limitation:** Personal data must be kept in a form that permits identification of data subjects for no longer than is necessary for the purposes for which the personal data are processed.
- **Integrity and confidentiality:** Personal data must be processed in a manner that ensures appropriate security. This includes protection against unauthorized or unlawful processing and against accidental loss, destruction, or damage, using proper technical or organizational measures.
- **Accountability:** The data controller is responsible for and must be able to demonstrate compliance with the other six principles. This means having appropriate data protection policies, data protection impact assessments, relevant documentation on how data is processed, and relevant training for staff.

More details on how these Legal, Privacy and Ethical aspects are covered by the HEIR project can be found on deliverables D7.5 and D7.6.

5. End-user guidelines of HEIR framework components

This section includes installation and user manuals and guidelines for each HEIR component. The main objective of this section is to provide clear and complete guidelines not only ensuring the uncorrupted installation of HEIR framework, but also exposing the means of interaction with the framework in a comprehensible way.

5.1 HEIR facilitators

5.1.1 1st Layer GUI and Forensics Visualization Toolkit (FVT)

5.1.1.1 Installation guide

The 1st Layer GUI is a user-friendly web-based application that serves as the central interface for accessing and analysing information within the HEIR ecosystem. Built on an Angular framework, this GUI offers a comprehensive dashboard presenting aggregated security indicators, metrics, and outcomes derived from various HEIR components. Authorized users and security experts can gain valuable insights into their organization's security landscape.

On the other hand, the Forensics Visualization Toolkit (FVT) is a toolkit accessible through the 1st Layer GUI. It is also a web-based application that enables detailed investigations and analysis of departmental and device-specific security events and alerts. The FVT empowers authorized users and security experts with advanced functionalities, including timelines, charts, and interactive visualizations.

Both the 1st Layer GUI and FVT are deployed in a docker-compose setup, consisting of a front-end module, which represents the main GUI of the system built on an Angular-based framework, and a back-end & middleware module based on NodeJS. The back-end & middleware component facilitates data communication, analytics, and statistics generation, while the web server (nginx) ensures seamless access to the interfaces. This combination of front-end, back-end, and middleware components enables efficient querying, data composition, and user authentication within the HEIR ecosystem.

5.1.1.1.1 Prerequisites

Before installing and deploying the 1st Layer GUI and FVT, make sure you have the following pre-requisites:

- Docker & Docker Compose: Ensure that Docker and Docker Compose are installed on your host machine. You can refer to the official Docker Compose reference site for detailed instructions on installation and usage.

Internet access is also required for the installation process.

5.1.1.1.2 Pre-installation Tasks

Ensure that the necessary dependencies and services are accessible. Specifically, ensure that the HEIR's databases, such as ElasticSearch, are reachable through a specific URI.

5.1.1.1.3 Installation procedure

To install the 1st Layer GUI and FVT, follow these steps:

1. Acquire the necessary configuration files, including the docker-compose.yaml file and two config files for the web server (nginx). These files contain essential environment variables and settings for the installation.
2. Edit the docker-compose.yaml file and input the specific environment variables, such as the preferred exposed URL of the platform, available ports on your host machine for communication, and the URI for the HEIR Observatory's database (ElasticSearch). Similarly, update the nginx config files with the corresponding URLs and ports.
3. Transfer these files to the desired directory on your host machine. Open a terminal, navigate to the specified directory, and run the command "$ docker-compose up -d" to initiate the installation. The required images (frontend, backend, middleware, and web server) will be automatically downloaded from the HEIR's Docker Hub account.

5.1.1.1.4 Post-installation

After the installation is complete, you can access the 1st Layer GUI and FVT through a web browser. Enter the exposed URL in the browser's address bar to access the GUI and start exploring the features and functionalities.

5.1.1.1.5 Update procedure

If updates are available and you are notified by the HEIR's admin team, follow these steps to update the system:

1. Stop the running containers by running the command "$ docker-compose down".
2. Update the images' version information in the docker-compose.yaml file with the newer versions provided.
3. Run the command "$ docker-compose up -d" to restart the containers with the updated versions. The updates will be automatically incorporated into the system.

Note: It is recommended to clear the cache of both the Docker containers and your internet browser after performing any update actions.

5.1.1.1.6 Uninstall or Roll-back procedure

To uninstall or roll back to previous versions of the 1st Layer GUI and FVT, follow these steps:

1. Update the docker-compose.yaml file's image text fields with the desired previous versions.
2. Repeat the update procedure by running the command "$ docker-compose up -d" to restart the containers with the specified previous versions.

If you want to remove the containers entirely, stop them by running the command "$ docker-compose down" and then remove them with the command "$ docker-compose rm".

Note: It is recommended to clear the cache of both the Docker containers and your internet browser after performing any uninstall or roll-back actions.

5.1.1.1.7 Technical Tests

Adequate testing prior and after release to production is necessary to check not only functional aspects of the delivery but also not-functional ones.

Some tests might need to be performed before or after deployment of the software component, especially in cases when direct integration with other components is required.

5.1.1.2 User manual

5.1.1.2.1 Introduction

This User Manual (UM) provides comprehensive information on both the First Layer GUI (Graphical User Interface) and the Forensics Visualization Toolkit (FVT) of the HEIR ecosystem. The First Layer GUI is the main visualization dashboard for authorized users inside their organization, which provides organization’s overall security awareness by displaying aggregated security indicators, metrics and relevant outcomes of HEIR’s main components. The FVT offers a detailed representation of departmental and device specific security events & alerts, captured by HEIR’s sub-modules. Via FVT, security experts can do forensics investigations and further analyze abnormal insights. The manual is designed to help non-technical users effectively utilize these components of the HEIR system.

The First Layer GUI can be accessed within the organization's environment and serves as the HEIR's entry point for authorized users. It is a web application that provides advanced and interactive visualizations, empowering users with valuable insights into the organization's security landscape. With a user-friendly interface, the First Layer GUI offers key features such as key insights and statistics, enabling users to gain a comprehensive understanding of the organization's security status. Users can explore metrics and indicators, track performance, and identify potential vulnerabilities. Additionally, the First Layer GUI facilitates comparison with global indicators, providing a broader context for benchmarking security measures against industry standards. Furthermore, users can leverage auditing observation options, allowing them to delve into historical logs and events, aiding in incident analysis and proactive security management.

The FVT, an integral component of the HEIR system, is an advanced visualization toolkit accessible through the First Layer GUI. It caters specifically to authorized security experts, empowering them to delve deeper into security events, alerts, and detailed incidents within their respective departments and monitored devices. The FVT provides a rich set of functionalities for detailed investigation and analysis. Security experts can drill down into specific departments, explore timelines, charts, and other interactive visualizations, and filter data based on various criteria. With the FVT, users can gain comprehensive insights into security incidents, identify patterns, and respond effectively to potential threats. Leveraging its powerful visualization capabilities, the FVT facilitates data-driven decision-making and enhances incident response strategies.

Both the First Layer GUI and the FVT are web applications built using modern technologies.

5.1.1.2.2 Getting Started

1. Access the System:
   a. Request from the system administrator to create a valid user-account based on your role. (e.g., regulator, researcher etc.)
   b. Open a web browser and enter the URL provided by your system administrator.
   c. The login page will be displayed. Enter your credentials and click "Login" to access the system (Figure 3).

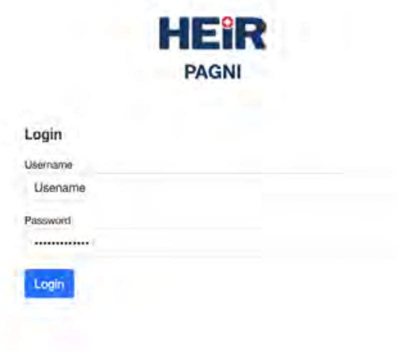


Figure 3: Login Screen

2. Dashboard Overview:
   The First Layer GUI dashboard provides a comprehensive overview of security-related information regarding both the hospital and the connected departments. It offers meaningful metrics presentation, auditing insights observation and comparison with Global indicators capabilities (HEIR Observatory live data fetching) (Figure 4). Security analysts can access the FVT through the dashboard for detailed departmental and device-specific investigations. The FVT offers security-relevant information, including ML

outcomes, SIEM reports, and RAMA score metadata. It provides analysts with tools to investigate potential security incidents and identify vulnerable areas across their system.

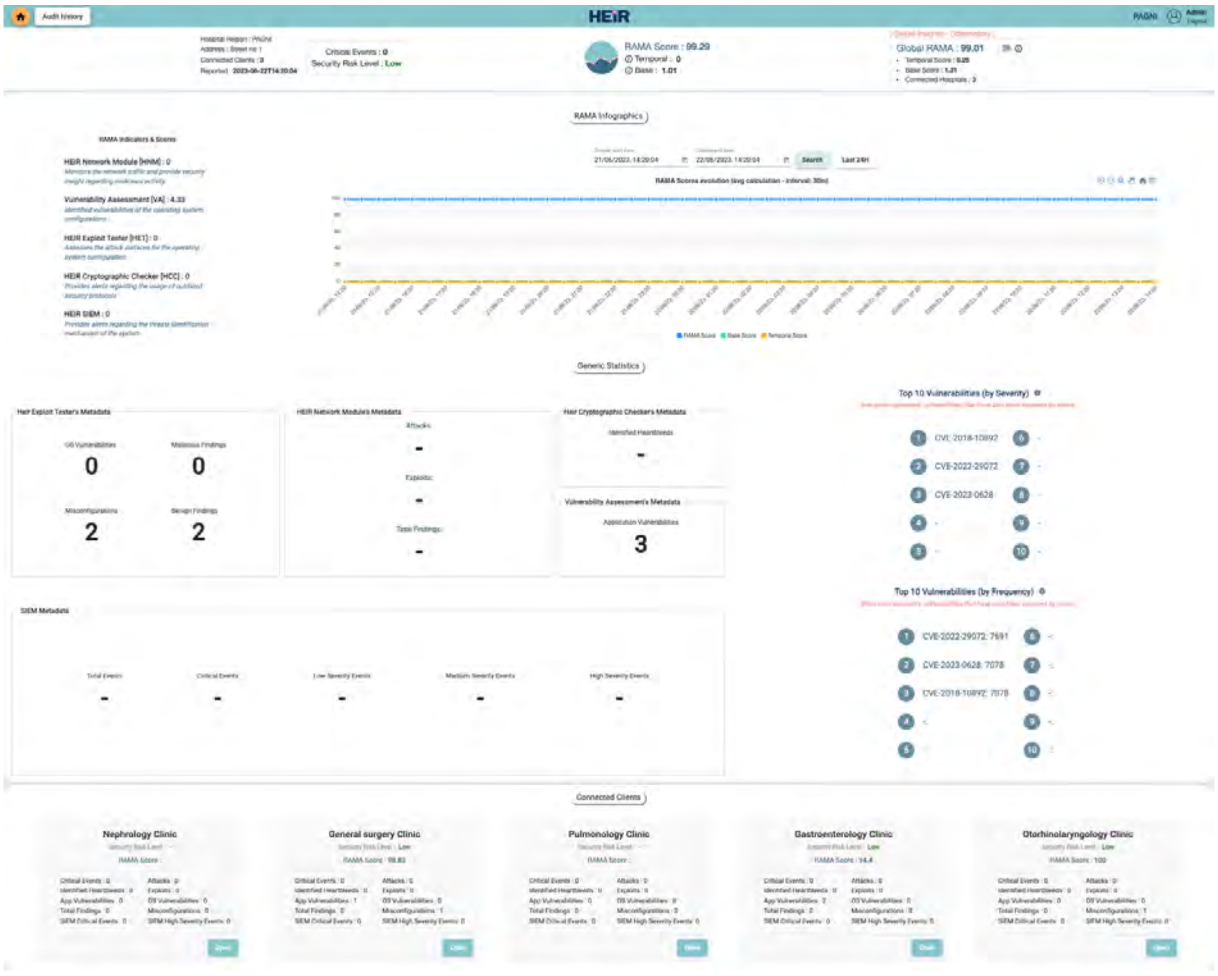
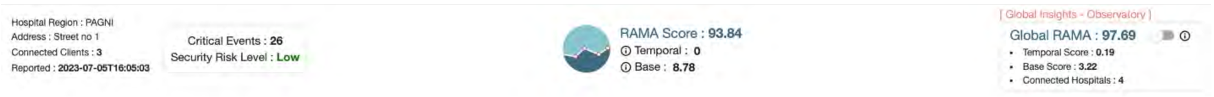


Figure 4: Complete page of HEIR Client GUI

a. Local and Global RAMA scores:
   On the upper part of the screen, users are able to see the aggregated RAMA scores, including RAMA, Base, and Temporal scores, as well as general information about the hospital's security status. On the right side of the panel, there are average statistical data and indicators for the Global RAMA Score, which are obtained from the Observatory. (Figure 5)



Hospital Region : PAGNI
Address : Street no 1
Connected Clients : 3
Reported : 2023-07-05T16:05:03

Critical Events : 26
Security Risk Level : Low

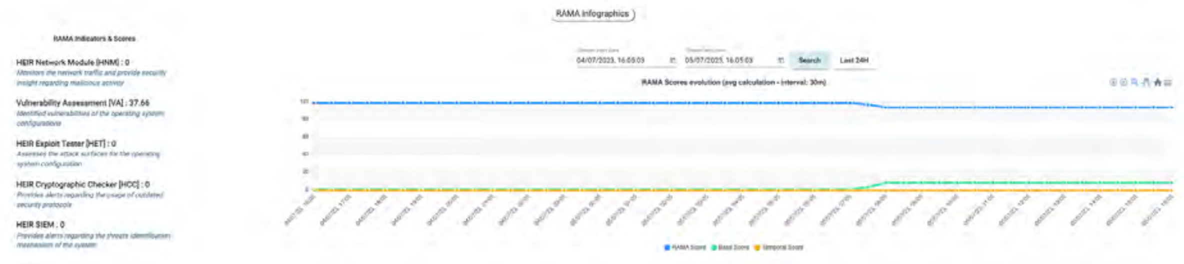
RAMA Score : 93.84
Temporal : 0
Base : 8.78

[Global Insights - Observatory]
Global RAMA : 97.69
- Temporal Score : 0.19
- Base Score : 3.22
- Connected Hospitals : 4

Figure 5: Local and Global RAMA scores

b. RAMA Infographics:
   The 'RAMA Indicators & Scores' provides a brief overview of the modules involved in the computation of RAMA scores, accompanied by their corresponding value indicators, is presented in Figure 11.
   Moreover, the 'RAMA Infographics' section features a multi-series line chart that

visually represents the progression of aggregated scores over time. This visual aid enhances user awareness and facilitates better monitoring of score fluctuations and variations (Figure 6)



RAMA Indicators & Scores

HEIR Network Module [HNM] : 0
Monitors the network traffic and provide security insight regarding malicious activity

Vulnerability Assessment [VA] : 37.66
Identified vulnerabilities of the operating system configurations

HEIR Exploit Tester [HET] : 0
Assesses the attack surfaces for the operating system configuration

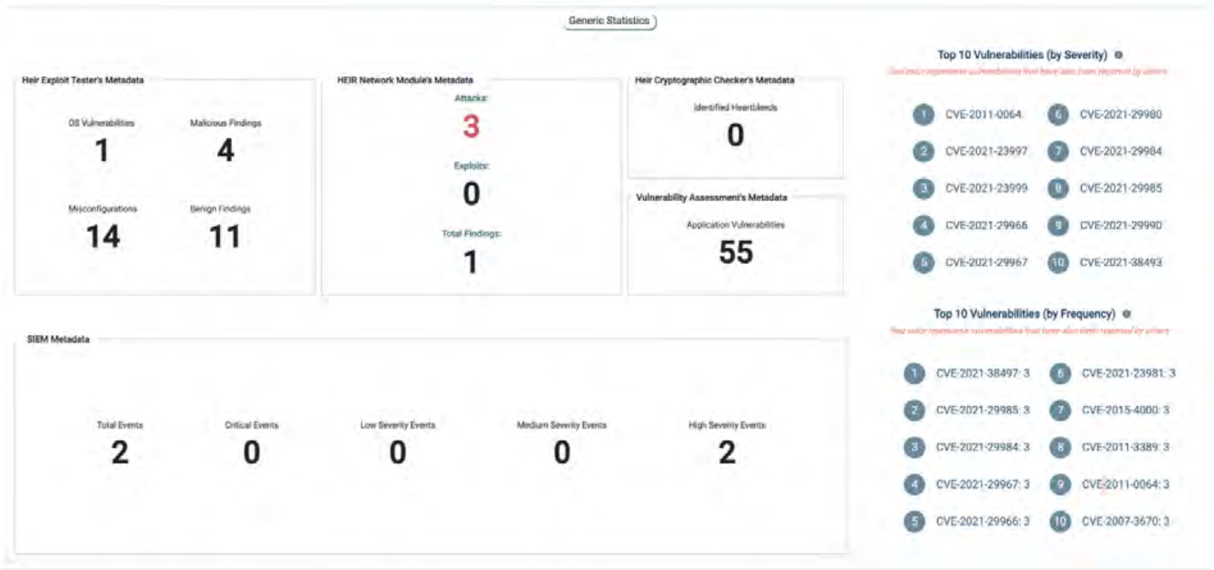
HEIR Cryptographic Checker [HCC] : 0
Provides alerts regarding the usage of outdated security protocols

HEIR SIEM : 0
Provides alerts regarding the threats identification mechanism of the system

04/07/2023, 16:05:03 – 05/07/2023, 16:05:03

Figure 6: RAMA infographics

c. Generic Statistics:
The metadata provided by the HEIR Aggregator encompasses valuable information regarding the embedded modules (HEIR facilitators) of the HEIR Clients across hospital. The displayed security insights include identified vulnerabilities in applications and operating systems, captured network malicious events, active misconfigurations, and results from event analysis tools, among others. Furthermore, the section presents the top 10 vulnerabilities, accompanied by a graphical representation that illustrates the order by severity in the upper section and by frequency in the lower section. This allows users to have a comprehensive overview of the most critical security risks within the organization (Figure 7)



Generic Statistics

Heir Exploit Tester's Metadata

OS Vulnerabilities	Malicious Findings
1	4
Misconfigurations	**Benign Findings**
14	11

HEIR Network Module's Metadata

Attacks: 3
Exploits: 0
Total Findings: 1

Heir Cryptographic Checker's Metadata

Identified Heartbleeds: 0

Vulnerability Assessment's Metadata

Application Vulnerabilities: 55

SIEM Metadata

Total Events	Critical Events	Low Severity Events	Medium Severity Events	High Severity Events
2	0	0	0	2

Top 10 Vulnerabilities (by Severity)

1	CVE-2011-0064	6	CVE-2021-29980
2	CVE-2021-23997	7	CVE-2021-29984
3	CVE-2021-23999	8	CVE-2021-29985
4	CVE-2021-29966	9	CVE-2021-29990
5	CVE-2021-29967	10	CVE-2021-38493

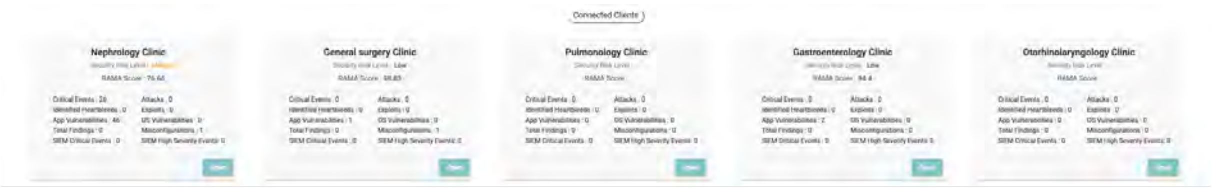
Top 10 Vulnerabilities (by Frequency)

1	CVE-2021-38497: 3	6	CVE-2021-23981: 3
2	CVE-2021-29985: 3	7	CVE-2015-4000: 3
3	CVE-2021-29984: 3	8	CVE-2011-3389: 3
4	CVE-2021-29967: 3	9	CVE-2011-0064: 3
5	CVE-2021-29966: 3	10	CVE-2007-3670: 3

Figure 7: Generic Statistics

d. Connected Clients:
Towards the bottom of the page, users have the ability to review the connected departments, which are additional HEIR Clients within the hospital. A concise summary of relevant information is presented, providing an overview of each connected department. Users also have the option to delve deeper into a selected department by clicking the 'Open' button. This grants them access to the home page of the FVT, where

they can explore detailed security-related data, investigate specific incidents, and gain deeper insights into the department's security status (Figure 8).



Nephrology Clinic
Security Risk Level : [illegible]
RAMA Score : 76.66
Critical Events : 28 — Attacks : 0
Identified Heartbleeds : 0 — Exploits : 0
App Vulnerabilities : 46 — OS Vulnerabilities : 0
Total Findings : 0 — Misconfigurations : 1
SIEM Critical Events : 0 — SIEM High Severity Events: 0

General surgery Clinic
Security Risk Level : Low
RAMA Score : 98.85
Critical Events : 0 — Attacks : 0
Identified Heartbleeds : 0 — Exploits : 0
App Vulnerabilities : 1 — OS Vulnerabilities : 0
Total Findings : 0 — Misconfigurations : 1
SIEM Critical Events : 0 — SIEM High Severity Events: 0

Pulmonology Clinic
Security Risk Level :
RAMA Score :
Critical Events : 0 — Attacks : 0
Identified Heartbleeds : 0 — Exploits : 0
App Vulnerabilities : 0 — OS Vulnerabilities : 0
Total Findings : 0 — Misconfigurations : 0
SIEM Critical Events : 0 — SIEM High Severity Events: 0

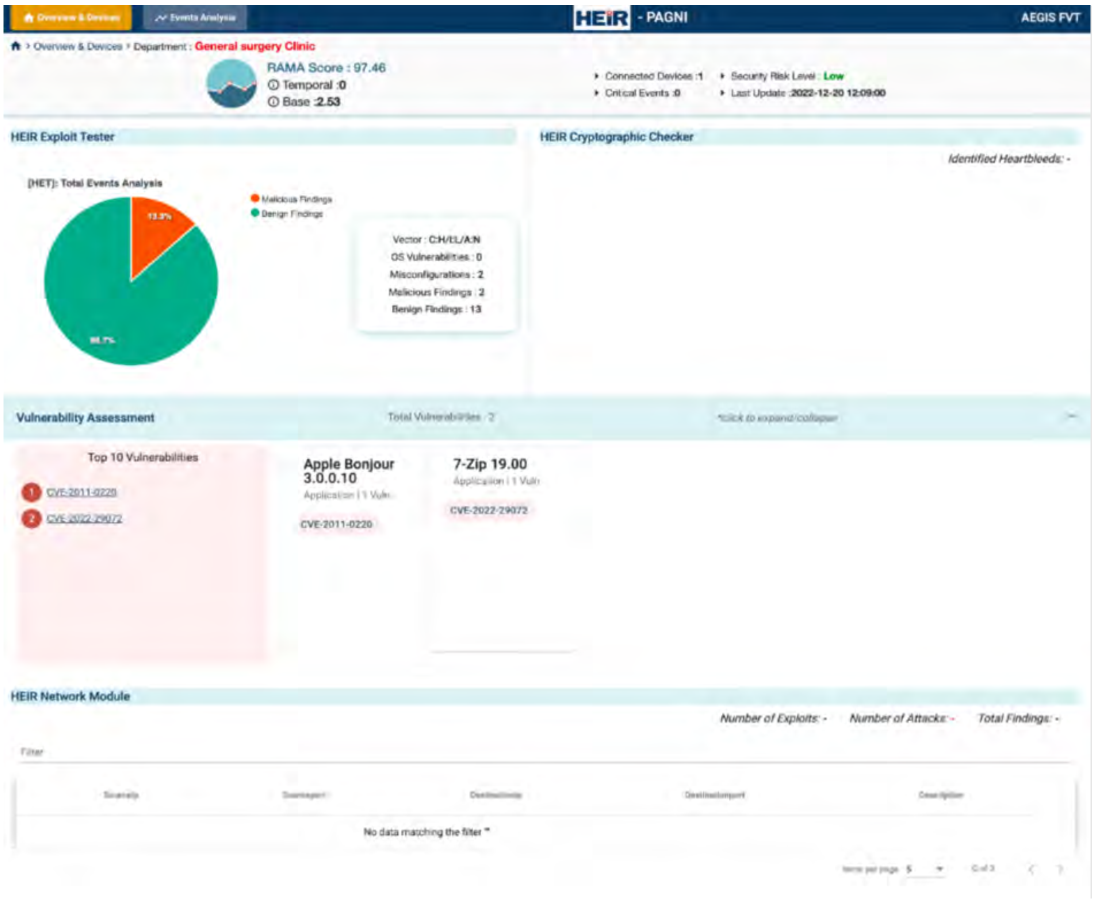
Gastroenterology Clinic
Security Risk Level : Low
RAMA Score : 94.4
Critical Events : 0 — Attacks : 0
Identified Heartbleeds : 0 — Exploits : 0
App Vulnerabilities : 2 — OS Vulnerabilities : 0
Total Findings : 0 — Misconfigurations : 0
SIEM Critical Events : 0 — SIEM High Severity Events: 0

Otorhinolaryngology Clinic
Security Risk Level :
RAMA Score :
Critical Events : 0 — Attacks : 0
Identified Heartbleeds : 0 — Exploits : 0
App Vulnerabilities : 0 — OS Vulnerabilities : 0
Total Findings : 0 — Misconfigurations : 0
SIEM Critical Events : 0 — SIEM High Severity Events: 0

Figure 8: Connected Clients

3. Audit History:
   Authorized auditors can access the Audit History screen, which is a navigation option at the top panel menu, where events are presented in both temporal and tabular representations. In the timeline, a hover functionality has been implemented, allowing users to hover over events and view additional information about each one.
4. Logout:
   Users are able to logout by clicking the "Logout" button placed on the top-right corner of the page, under the displayed username.
5. FVT:
   This toolkit provides a detailed representation of security events captured by the SIEM sub-module and processed by the ML Anomaly Detection Module. By utilizing the FVT, authorized users can investigate connected HEIR Clients and gain department-specific insights. This section will guide you through the main analysis dashboard, timeline visualization, event details, and line charts for device metrics.

   a. Overview & Devices
      In this screen, security experts can view the RAMA score, along with relevant metadata from the facilitators, similar to the First Layer GUI. However, this information refers specifically to the department that the user is examining (Figure 9).



Overview & Devices > Department : General surgery Clinic

RAMA Score : 97.46
Temporal :0
Base :2.53

- Connected Devices :1
- Critical Events :0
- Security Risk Level : Low
- Last Update :2022-12-20 12:09:00

HEIR Exploit Tester

[HET]: Total Events Analysis

Vector : C:H/EL/A:N
OS Vulnerabilities : 0
Misconfigurations : 2
Malicious Findings : 2
Benign Findings : 13

HEIR Cryptographic Checker

Identified Heartbleeds: -

Vulnerability Assessment

Total Vulnerabilities : 2

click to expand/collapse

Top 10 Vulnerabilities

1. CVE-2011-0220
2. CVE-2022-29072

Apple Bonjour 3.0.0.10
Application | 1 Vuln
CVE-2011-0220

7-Zip 19.00
Application | 1 Vuln
CVE-2022-29072

HEIR Network Module

Number of Exploits: - *Number of Attacks:* - *Total Findings:* -

Filter

Sourceip	Sourceport	Destinationip	Destinationport	Description
No data matching the filter ""				

Items per page: 5 0 of 0

Figure 9: Overview & Devices

Additionally, at the bottom of the screen, security experts can access the main analysis dashboard of the FVT by clicking the 'Inspect' button on any devices that is listed in the Connected Devices section (Figure 10).

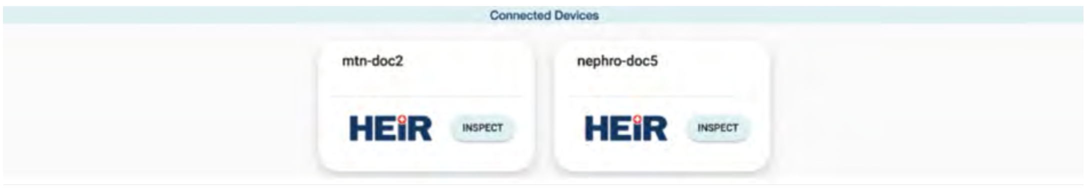
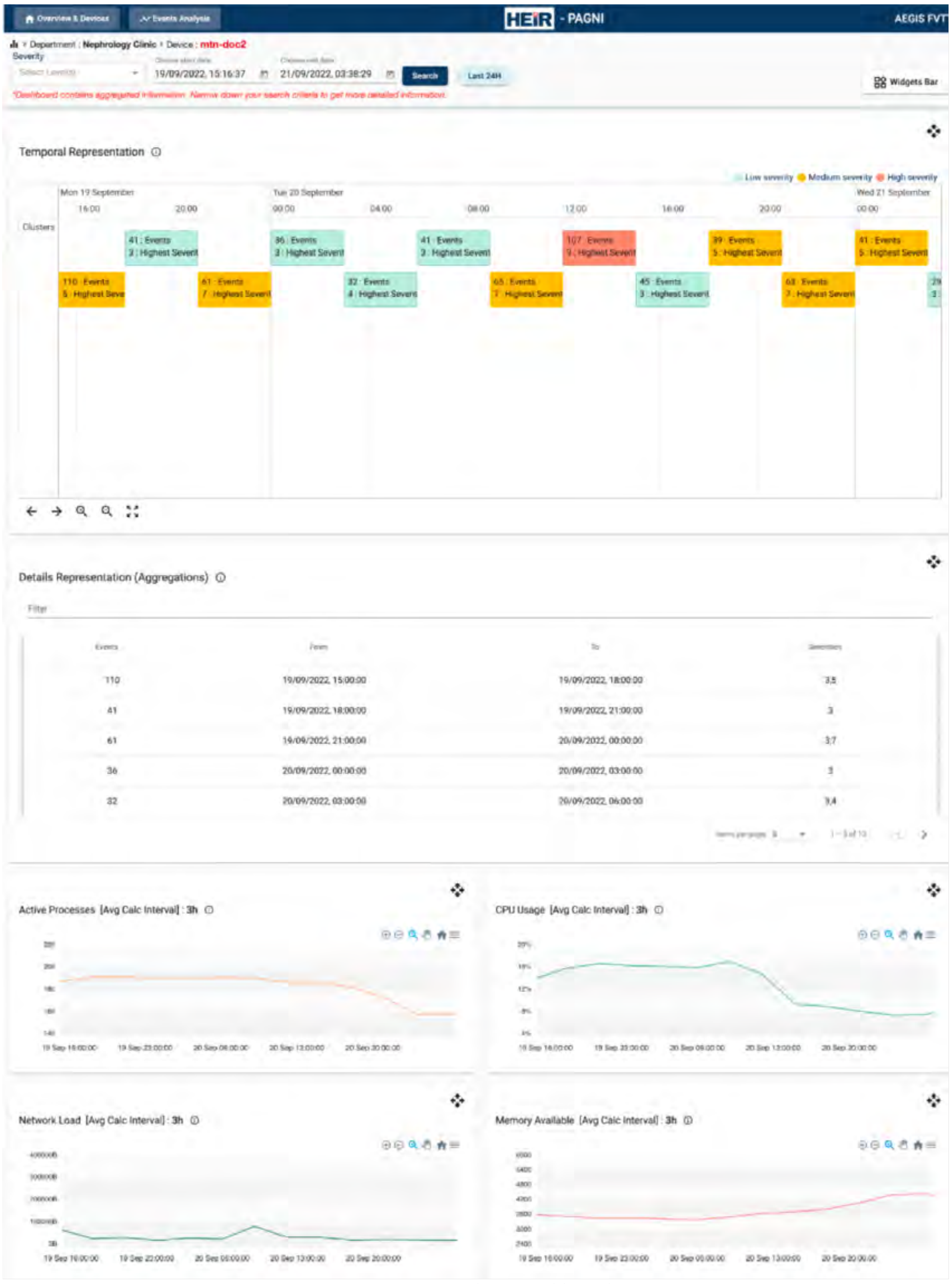


Figure 10: Connected Devices

b. Analysis Dashboard:
   In the 'Analysis Dashboard' screen, which is the main screen of the FVT, security experts can explore captured events and performance metrics specific to the device

they are inspecting, allowing for in-depth investigation of the selected device through several visualization widgets (Figure 11).

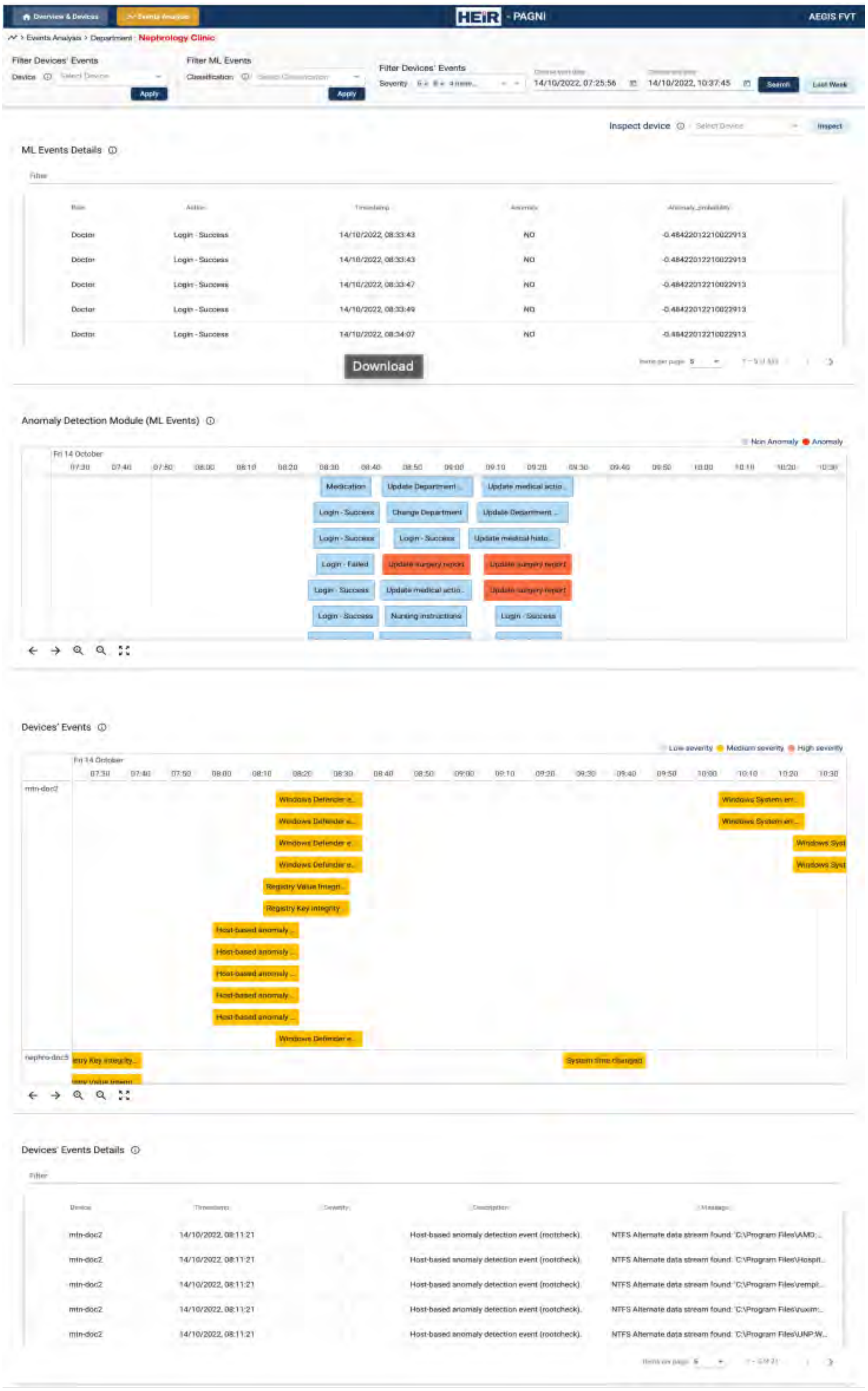

Overview & Devices
Events Analysis
HEIR - PAGNI
AEGIS FVT
Department : Nephrology Clinic › Device : mtn-doc2
Severity
Select Level(s)
19/09/2022, 15:16:37
21/09/2022, 03:38:29
Search
Last 24H
Widgets Bar
*Dashboard contains aggregated information. Narrow down your search criteria to get more detailed information.
Temporal Representation
Low severity
Medium severity
High severity
Mon 19 September
Tue 20 September
Wed 21 September
Clusters
110 : Events 5 : Highest Severity
41 : Events 3 : Highest Severity
61 : Events 7 : Highest Severity
36 : Events 3 : Highest Severity
32 : Events 4 : Highest Severity
41 : Events 3 : Highest Severity
65 : Events 7 : Highest Severity
107 : Events 9 : Highest Severity
45 : Events 3 : Highest Severity
39 : Events 5 : Highest Severity
68 : Events 7 : Highest Severity
41 : Events 5 : Highest Severity
Details Representation (Aggregations)
Filter
Active Processes [Avg Calc Interval] : 3h
CPU Usage [Avg Calc Interval] : 3h
Network Load [Avg Calc Interval] : 3h
Memory Available [Avg Calc Interval] : 3h

Events	From	To	Severities
110	19/09/2022, 15:00:00	19/09/2022, 18:00:00	3,5
41	19/09/2022, 18:00:00	19/09/2022, 21:00:00	3
61	19/09/2022, 21:00:00	20/09/2022, 00:00:00	3,7
36	20/09/2022, 00:00:00	20/09/2022, 03:00:00	3
32	20/09/2022, 03:00:00	20/09/2022, 06:00:00	3,4

Figure 11: Analysis Dashboard

c. Events Analysis:
In the 'Events Analysis' screen, accessible through FVT's navigation options, security experts can view events detected by the SIEM and ML component. These events, which are related to all the devices, are presented in both temporal and tabular representations. The screen also offers various functionalities, including filtering

capabilities, color-coded schemas, and the ability to select specific timeframes for detailed investigation (Figure 12).



ML Events Details

Filter

Role	Action	Timestamp	Anomaly	Anomaly_probability
Doctor	Login - Success	14/10/2022, 08:33:43	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:33:43	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:33:47	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:33:49	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:34:07	NO	-0.48422012210022913

Download

Items per page: 5

Devices' Events Details

Filter

Device	Timestamp	Severity	Description	Message
mtn-doc2	14/10/2022, 08:11:21		Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\AMD:...
mtn-doc2	14/10/2022, 08:11:21		Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\Hospit...
mtn-doc2	14/10/2022, 08:11:21		Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\rempl:...
mtn-doc2	14/10/2022, 08:11:21		Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\ruxim:...
mtn-doc2	14/10/2022, 08:11:21		Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\UNP:W...

Items per page: 5

Figure 12: Event Analysis

5.1.1.2.2.1 Cautions & Warnings

Unauthorized Access Prohibited

Both the First Layer GUI and the FVT are web applications intended for authorized use only. Any unauthorized access or attempts to breach the system's security are strictly prohibited and may result in legal action. Users are responsible for ensuring the confidentiality and integrity of their login credentials and must not share them with anyone.

5.1.1.2.2.2 Set-up Considerations

To access both the First Layer GUI and the FVT, you will need a computer or device with internet connectivity. There is no need for any specialized equipment or network configurations. Simply ensure that you have a working internet connection and a web browser installed.

5.1.1.2.2.3 User Access Considerations

The system distinguishes between two user groups: **security experts** and **hospital's auditors**.

1. Security experts:
   Authorized users can access the main page of the First Layer GUI, which includes a range of widgets and graphs, and they can also use the FVT for more detailed investigations.
2. Hospital's auditors:
   Hospital's auditors have can access the Audit History tab, where they can examine logs of activities performed by other components within the HEIR system.

5.1.1.2.2.4 Accessing the System

The FVT and First Layer GUI are accessible exclusively to authorized hospital staff with appropriate credentials. Furthermore, access to both applications is limited to within the hospital premises.

5.1.1.2.2.5 System Organization & Navigation

The HEIR's First Layer GUI and the FVT are thoughtfully organized and designed to provide users with intuitive navigation and seamless access to its various screens and functionalities. The system's main organization is implemented through the navigation panels. Below, you will find a short description of the navigation options.

First Layer GUI

1. Main Dashboard
   Upon successful login, users are landed into the main dashboard of First layer GUI. (see "Getting Started" section)

2. Audit History Screen
   Hospital's auditors can navigate to the audit history screen from the First Layer GUI. This screen provides a comprehensive view of the audit logs and data requests, provided by the Privacy Aware Framework of HEIR. Users can delve into the detailed logs to understand past events and user actions within the system. The audit history screen ensures transparency and accountability, allowing authorized users to review and monitor the full set of details, while user-roles with limited permissions to review and monitor PII redacted information.

FVT

1. Overview & Devices
   By clicking the 'Open' button on any Connected Device listed at the bottom of the Main Dashboard, the user gains access to the FVT. The first screen that appears is the Overview & Devices screen, where detailed information about the specific department is displayed (see "Getting Started" section).

2. Analysis Dashboard
   In this screen, users can examine events and metrics specific to the device they are inspecting

(see "Getting Started" section).

Events Analysis

The 'Events Analysis' screen allows security experts to view events detected by the SIEM and ML component. For more information about this screen, see "Getting Started" section.

5.1.1.2.2.6 Exiting the System

To properly exit the First Layer GUI, you can simply log out from the system and close the web browser. It is important to log out to ensure the security of your account and prevent unauthorized access.

5.1.1.2.3 Using the System

In this section, we will delve into the functionalities and features of both the First Layer GUI and the FVT, providing non-technical users with an understanding of how to effectively navigate and leverage these tools for security analysis and investigation.

5.1.1.2.3.1 First Layer GUI

This section provides a comprehensive guide on how to effectively utilize the features and functions available within the First Layer GUI. Whether you are a simple user or an authorized user, the following instructions will help you navigate through the system and make the most out of its capabilities (Figure 13).

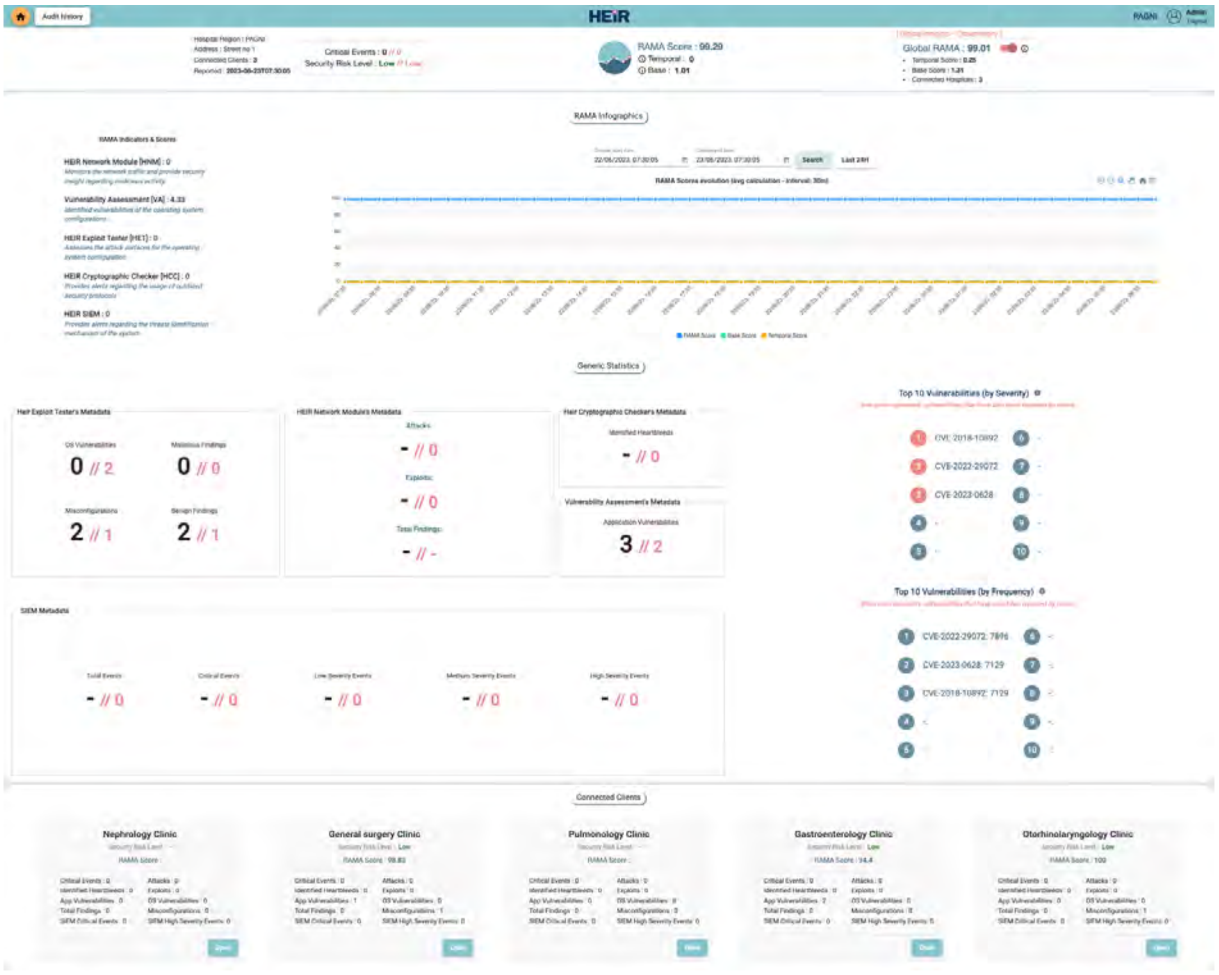
5.1.1.2.3.1.1 Changing Date and Time Filter

Within the First Layer GUI, you can adjust the visualized period in RAMA scores evolution histogram by adjusting the date and time filter above the chart. This feature allows you to focus on specific periods of interest and gain insights into the evolving security landscape over time.

5.1.1.2.3.1.2 Comparing Local and Global RAMA Scores metadata

End-users have the ability to compare their organization's RAMA scores and metadata with the ones calculated inside the HEIR Observatory. The RAMA score serves as an indicator of the security level within the specific hospital. By enabling the toggle button placed next to the Global RAMA score indicator (located in the top-right corner of the screen), you can initiate a comparative analysis.

Enabling this feature unlocks the capability to juxtapose your hospital's aggregated insights, including RAMA scores and corresponding metadata, with the global ones derived from the Observatory. This comparative analysis provides the ability to identify common vulnerabilities/ security findings between your environment and the broader healthcare domain (region-wise).



Audit history

HEIR

Hospital Region : PAGNI
Address : Street no 1
Connected Clients : 3
Reported : 2023-06-23T07:30:05

Critical Events : 0 // 0
Security Risk Level : Low // Low

RAMA Score : 99.20
Temporal : 0
Base : 1.01

Global RAMA : 99.01
- Temporal Score : 0.25
- Base Score : 1.21
- Connected Hospitals : 3

RAMA Infographics

RAMA Indicators & Scores

HEIR Network Module [HNM] : 0
Monitors the network traffic and provide security insight regarding malicious activity

Vulnerability Assessment [VA] : 4.33
Identified vulnerabilities of the operating system configurations

HEIR Exploit Tester [HET] : 0
Assesses the attack surfaces for the operating system configuration

HEIR Cryptographic Checker [HCC] : 0
Provides alerts regarding the usage of outdated security protocols

HEIR SIEM : 0
Provides alerts regarding the threats identification mechanism of the system

22/06/2023, 07:30:05 — 23/06/2023, 07:30:05 — Search — Last 24H

RAMA Scores evolution (avg calculation - interval: 30m)

Generic Statistics

Heir Exploit Tester's Metadata

OS Vulnerabilities	Malicious Findings
0 // 2	0 // 0
Misconfigurations	**Benign Findings**
2 // 1	2 // 1

HEIR Network Module's Metadata

Attacks	Exploits	Total Findings
- // 0	- // 0	- // -

Heir Cryptographic Checker's Metadata

Identified Heartbleeds: - // 0

Vulnerability Assessment's Metadata

Application Vulnerabilities: 3 // 2

SIEM Metadata

Total Events	Critical Events	Low Severity Events	Medium Severity Events	High Severity Events
- // 0	- // 0	- // 0	- // 0	- // 0

Top 10 Vulnerabilities (by Severity)

1	CVE-2018-10892	6	-
2	CVE-2022-29072	7	-
3	CVE-2023-0628	8	-
4	-	9	-
5	-	10	-

Top 10 Vulnerabilities (by Frequency)

1	CVE-2022-29072: 7896	6	-
2	CVE-2023-0628: 7129	7	-
3	CVE-2018-10892: 7129	8	-
4	-	9	-
5	-	10	-

Connected Clients

	Nephrology Clinic	General surgery Clinic	Pulmonology Clinic	Gastroenterology Clinic	Otorhinolaryngology Clinic
Security Risk Level	-	Low	-	Low	Low
RAMA Score		99.82		94.4	100
Critical Events	0	0	0	0	0
Attacks	0	0	0	0	0
Identified Heartbleeds	0	0	0	0	0
Exploits	0	0	0	0	0
App Vulnerabilities	0	1	0	2	0
OS Vulnerabilities	0	0	0	0	0
Total Findings	0	0	0	0	0
Misconfigurations	0	1	0	0	1
SIEM Critical Events	0	0	0	0	0
SIEM High Severity Events	0	0	0	5	0

Figure 13: Complete page of First Layer GUI

5.1.1.2.3.2 Forensics Visualization Toolkit (FVT)

This section provides a comprehensive guide on how to effectively utilize the features and functions available within the FVT. Whether you are a simple user or an authorized user, the following instructions will help you navigate through the system and make the most out of its capabilities.

5.1.1.2.3.2.1 Overview & Devices

5.1.1.2.3.2.1.1 Top 10 Vulnerabilities

In the Overview & Devices Screen, users can access a list of the top 10 vulnerabilities of the department. Each vulnerability is clickable, allowing users to obtain more detailed information about them from reputable sources like MITRE.

5.1.1.2.3.2.1.2 Sort and Filter Functionality

This screen provides a user-friendly sorting and filtering functionality for the tabular representation widget. Users can easily sort the table by any column to organize data in their preferred order. Additionally, they can filter the table by typing specific information at the top side of the widget, allowing them to quickly find relevant data (Figure 14).

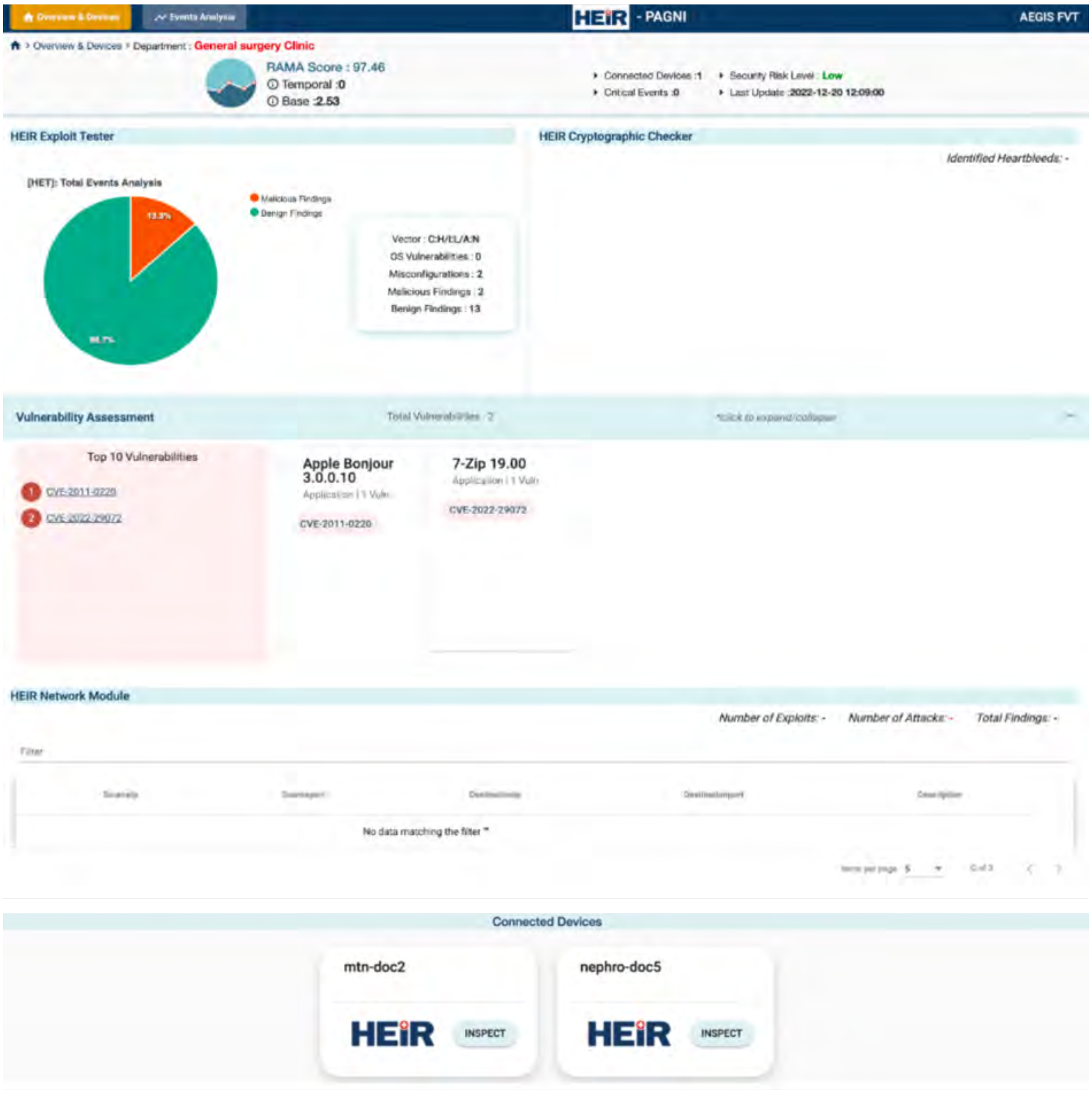


Figure 14: Overview & Devices - functionalities

5.1.1.2.3.2.2 Analysis Dashboard

5.1.1.2.3.2.2.1 Filtering by Date and Time

One of the key features of the "Analysis Dashboard" screen is the ability to filter events based on a specific date and time. By adjusting the filter, users can focus on events within a particular timeframe, allowing for a time-wise targeted analysis of security events.

5.1.1.2.3.2.2.2 Filtering by Severity

This screen also provides the option to filter events based on their severity. Users can prioritize events of higher severity, enabling them to quickly identify and investigate potential security risks or critical incidents.

5.1.1.2.3.2.2.3 Comprehensive Analysis Dashboard

The "Analysis Dashboard" within the FVT offers a set of widgets that provide different types of visualizations, covering system metrics and network-related information. These widgets can be customized and rearranged to cater to the investigation needs of the user.

5.1.1.2.3.2.2.4 Timeline Visualization

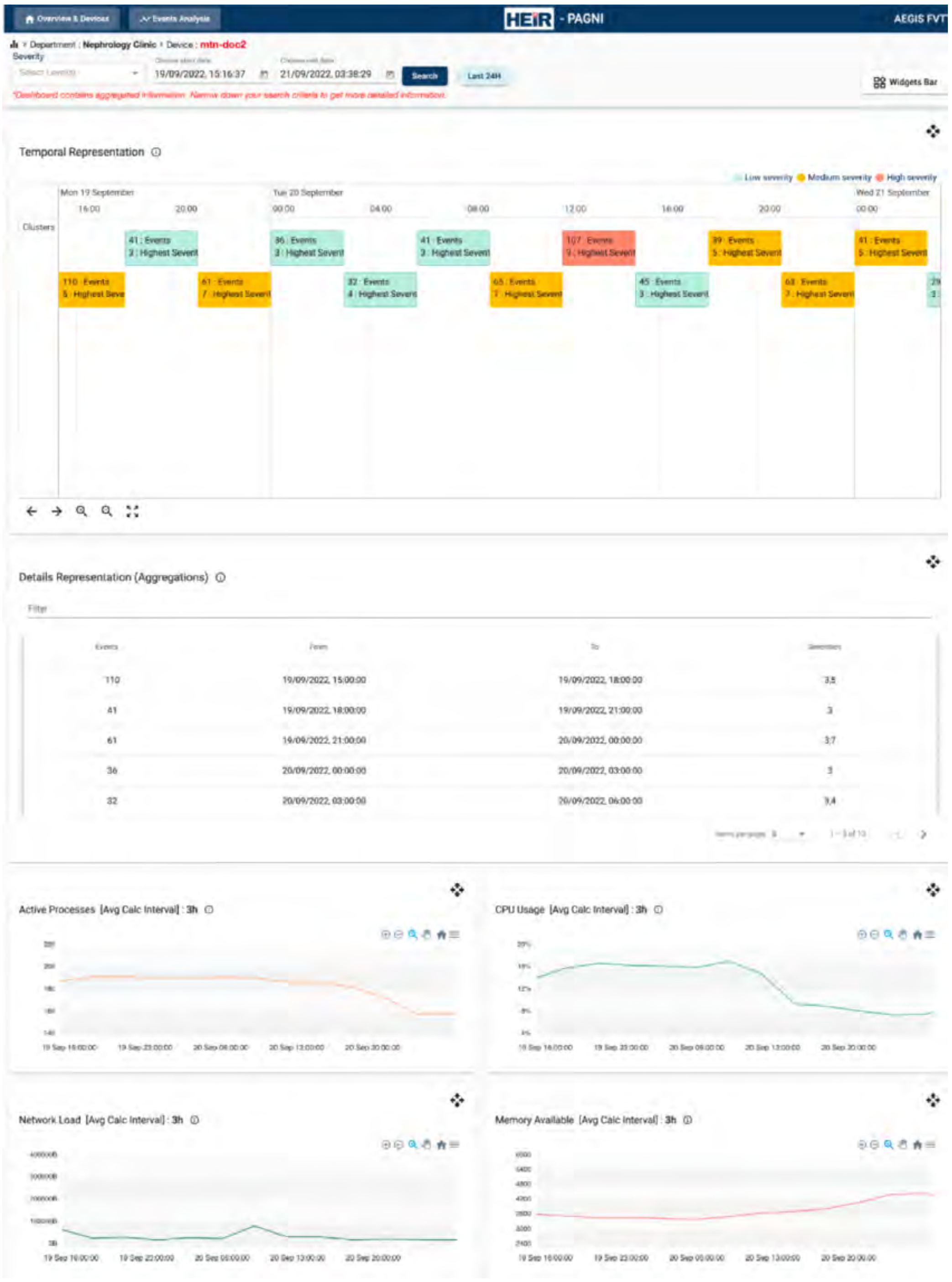
A timeline widget is also included, that offers a temporal representation of incoming logged events. Users can interact with the timeline, adjusting the range period to synchronize and update other widgets accordingly. Additionally, the timeline widget allows for annotations and adaptive visualization of events based on the volume of data.

5.1.1.2.3.2.2.5 Event Details

For a more detailed understanding of specific events, a details widget is provided. This widget presents comprehensive information about incoming events and supports text filtering capabilities, enabling users to search and locate specific information of interest.

5.1.1.2.3.2.2.6 Performance Metrics Line charts

The specific screen offers a valuable functionality where users can view performance metrics related to the specific device they are inspecting, presented in synchronized line charts. These line charts provide insights into various data points over time. Users can interact with the line charts by hovering over a line, which allows them to see detailed information for a particular date and time. As they hover over a line of a line chart, all other line charts on the dashboard automatically display information for the same specific date and time, providing a comprehensive and synchronized view of the device's performance metrics (Figure 15).



19/09/2022, 15:16:37 – 21/09/2022, 03:38:29

*Dashboard contains aggregated information. Narrow down your search criteria to get more detailed information.

Details Representation (Aggregations)

Events	From	To	Severities
110	19/09/2022, 15:00:00	19/09/2022, 18:00:00	3,5
41	19/09/2022, 18:00:00	19/09/2022, 21:00:00	3
61	19/09/2022, 21:00:00	20/09/2022, 00:00:00	3,7
36	20/09/2022, 00:00:00	20/09/2022, 03:00:00	3
32	20/09/2022, 03:00:00	20/09/2022, 06:00:00	3,4

Figure 15: Analysis Dashboard

5.1.1.2.3.2.3 Events Analysis

5.1.1.2.3.2.3.1 Filtering by Date and Time

This functionality allows users to filter events based on a specific date and time range.

5.1.1.2.3.2.3.2 Filtering by Severity

Users can filter events by severity in this screen to prioritize and investigate security incidents effectively.

5.1.1.2.3.2.3.3 Filtering by Devices

Users can filter events based on specific devices or equipment within the selected department. This functionality allows for a more granular analysis, focusing on events related to a particular device or group of devices.

5.1.1.2.3.2.3.4 Filtering ML Events

The “Events Analysis” screen incorporates machine learning capabilities, which enable the identification of anomalous events. Users can filter events specifically flagged by the ML Anomaly Detection Module, helping to pinpoint potential security breaches or irregular activities.

5.1.1.2.3.2.3.5 Comprehensive Analysis Dashboard

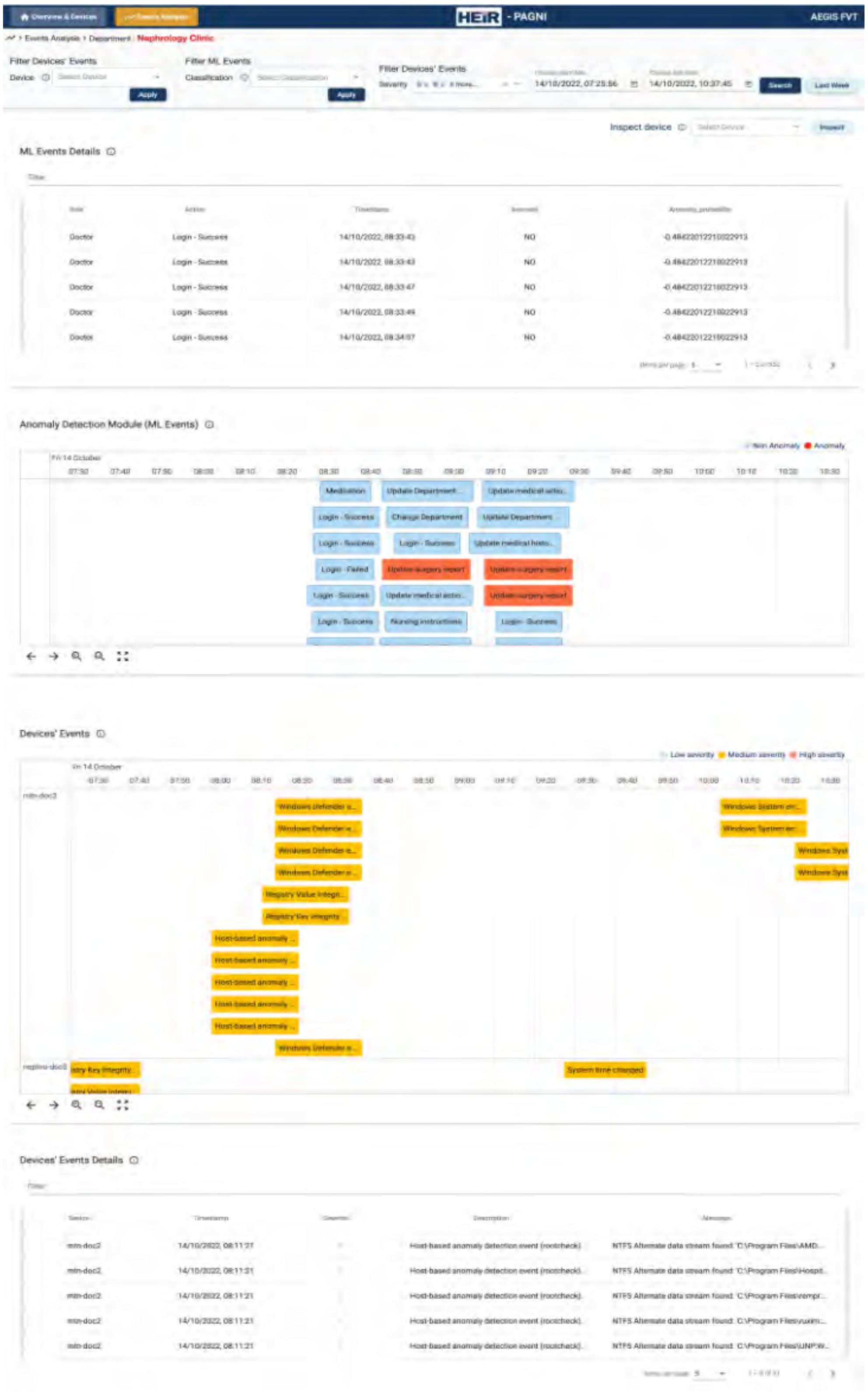
In this screen, we encounter once again the feature of a set of widgets that provide various visualizations and also can be customized and rearranged.

5.1.1.2.3.2.3.6 Timeline Visualization

In the Events Analysis screen, we can find again the functionality of the timeline widget.

5.1.1.2.3.2.3.7 Event Details

The functionality of a details widgets it is implemented in this screen too (Figure 16).



ML Events Details

Role	Action	Timestamp	Anomaly	Anomaly_probability
Doctor	Login - Success	14/10/2022, 08:33:43	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:33:43	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:33:47	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:33:49	NO	-0.48422012210022913
Doctor	Login - Success	14/10/2022, 08:34:07	NO	-0.48422012210022913

Devices' Events Details

Device	Timestamp	Severity	Description	Message
mtn-doc2	14/10/2022, 08:11:21	[illegible]	Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\AMD...
mtn-doc2	14/10/2022, 08:11:21	[illegible]	Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\Hospit...
mtn-doc2	14/10/2022, 08:11:21	[illegible]	Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: 'C:\Program Files\empl...
mtn-doc2	14/10/2022, 08:11:21	[illegible]	Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: C:\Program Files\vuxim...
mtn-doc2	14/10/2022, 08:11:21	[illegible]	Host-based anomaly detection event (rootcheck).	NTFS Alternate data stream found: C:\Program Files\UNP\W...

Figure 16: Event Analysis

5.1.2 HEIR Machine Learning (ML)-based Anomaly Detection & Threat Classification module

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 health 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 to represent the results tangibly.

The selected model/algorithm based on supervised/ unsupervised learning algorithms 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.

5.1.2.1 Installation guide

The machine learning component is being hosted in the docker for each use case. The resulting outcome is being pushed from HEIR VM to Elastic-Engine platform where the FVT (first layer GUI) is depicting the results. The model for both cases is already trained and deployed.

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"}]}

5.1.2.2 User manual

The user does not have direct communication with the ML component. More specifically, the output of the ML component is made available to the user through HEIR’s 1st Layer GUI.

5.1.3 Blockchain-based Privacy-Aware Framework (PAF)

5.1.3.1 Privacy Aware Framework (PAF)

The HEIR Privacy Aware Framework (PAF) allows for policy-driven control access to stored data. Built on top of the Fybrik Open Source platform, the Privacy Aware Framework may need to modify a number of files from the github Fybrik distribution according to the use case. In particular, modifications may be required to the:

- Fybrik Application yaml
- Fybrik Module yaml
- Rego policy definition
- The executable Fybrik module code (Python code)

5.1.3.1.1 Installation manual

Fybrik is installed based on the directions found in Quick Start. There are a large number of example implementation to show how to modify the files mentioned above found in: https://github.com/orgs/fybrik/repositories including an implementation used for third party data export in the Norwegian healthcare use case (https://github.com/fybrik/REST-read-example).

Extensive documentation describing the various Fybrik Custom Resource Definitions (CRDs) can be found at: https://fybrik.io/v1.3/reference/crds/

5.1.3.1.1.1 Prerequisites

Fybrik and the Privacy Aware Framework run on top of Kubernetes. We have used kind to implement our Kubernetes cluster, however other Kubernetes frameworks should work as well.

5.1.3.1.1.2 Pre-installation Tasks

N/A

5.1.3.1.1.3 Installation procedure

Steps to install Fybrik are explicitly given in https://fybrik.io/v1.3/get-started/quickstart/. Modifications to the Privacy Aware Framework YAML files required to implement use case specific scenarios are applied using standard Kubernetes methodology, e.g. kubectl apply -f <YAML file name>.

An example PAF implementation for reading and redacting information from a backend healthcare FHIR server can be found in: https://github.com/fybrik/REST-read-example.

5.1.3.1.1.4 Post-installation

All Control Plane PAF Kubernetes resources are installed in the **fybrik-system** namespace. After installation of Fybrik, one should check to make sure that all components in this namespace have been deployed, using standard Kubernetes commands, e.g.

```
kubectl get all -n fybrik-system
```

All pods should be deployed and in the running state. Output from the execution of code inside a pod can be viewed by using the following command:

```
kubectl logs <PODNAME> -n <namespace where the pod is executing>
```

5.1.3.1.1.5 Update procedure

The distributed Fybrik framework is stable, and it is not expected to be updated.

5.1.3.1.1.6 Uninstall or Roll-back procedure

Rollback of installed PAF YAML files follow the standard Kubernetes practice, e.g.:

```
kubectl delete -f <YAML file name>
```

In the event of a catastrophic corruption of the Fybrik framework or the Kubernetes cluster, the best recovery method is to delete the Kubernetes cluster, regenerate it and reapply Fybrik and PAF files.

5.1.3.1.1.7 Technical Tests

Once configured, the Privacy Aware Framework will automatically deploy a *data path* workload in the namespace specified in the Fybrik Application YAML file. Successful installation of a PAF configuration can be easily checked by verifying if an executing Kubernetes pod appears in this namespace.

In the event that the data path fails to deploy, standard Kubernetes debugging techniques, such as checking the log files in the fybrik-system namespace will need to be applied.

5.1.3.1.2 User manual

5.1.3.1.2.1 Introduction

At a high level, the Privacy Aware Framework uses policy, data schema classification and information relating to the data request (e.g., Requester name, Role, Organization …) to determine what transformations may need to be applied to fields in the requested data source.

Essentially the Privacy Aware Framework comes between the data requester and the data store and mediates the requests that are allowed to pass through to the backend, as well as data which is allowed to flow out.

Configuration of the PAF is done through human-readable YAML files, whereas the policy definition language is written in the Rego language used by the Open Policy Agent which serves as a Policy Decision Engine.

Request related parameters (e.g. user, role…) are encoded in a certificate (JWT) which are typically supplied to the PAF from an external identity management system, such as Keycloak.

The actual transformations supplied to the data are programmable and provided by the Fybrik module. Typically the Open Source examples in the Fybrik github repo (http://github.com/fybrik) implement redaction (masking) or deletion of columns containing Personal Identifiable Information (PII).

5.1.3.1.2.2 Getting Started

The Privacy Aware Framework (PAF) utilizes Kubernetes to provide a locked-down, secured environment and conceptually exists between the data requester and the data source. This can be seen in Figure 17.

In this example, Keycloak is an Identity Management (IM) system used to produce a JSON Web Token (JWT) which accompanies every REST request for FHIR data. The JWT authenticates the user and encodes data such as user, role, organization, which will be used by the PAF Policy Engine to determine what data transformations may be required on the returned data. The implementation of IM is outside of the scope of the Privacy Aware Framework.

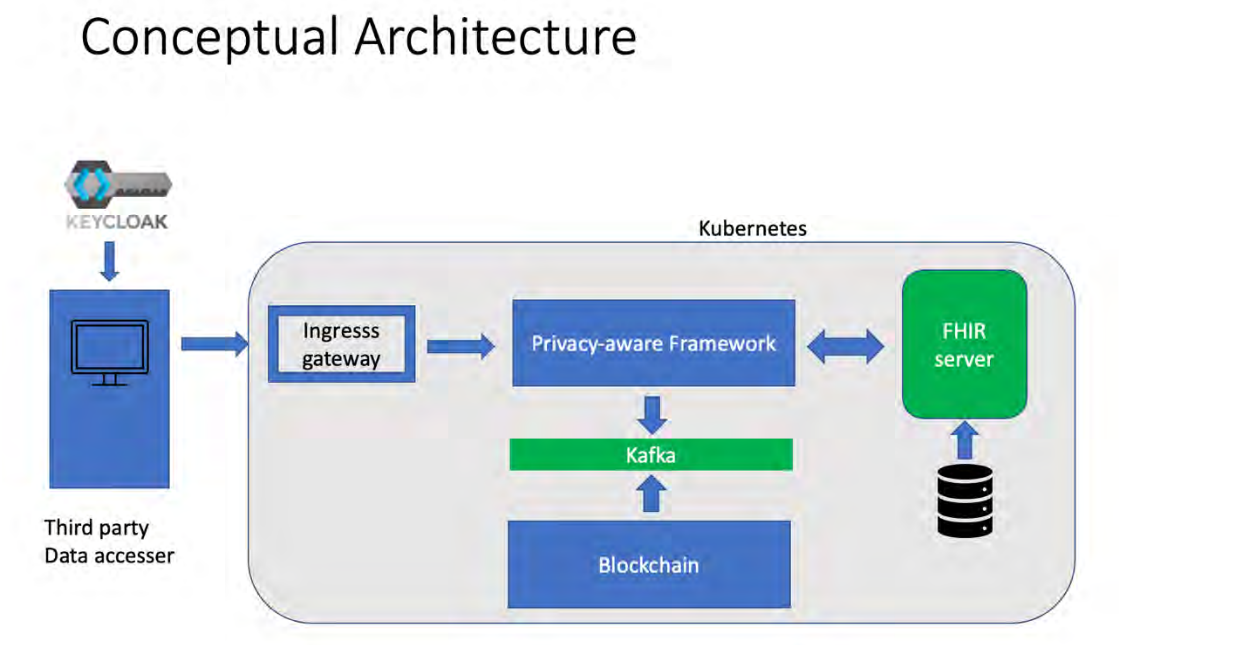


Figure 17: The Privacy Aware Framework Conceptual Architecture

5.1.3.1.2.2.1 Cautions & Warnings

The Privacy Aware Framework and Fybrik are released under the Apache 2.0 license.

5.1.3.1.2.2.2 Set-up Considerations

A Kubernetes cluster needs to be present before installing the Privacy Aware Framework following the instructions in https://fybrik.io/v1.3/get-started/quickstart/. This can be done following the directions found in: https://kind.sigs.k8s.io/. It is recommended that a dedicated cluster be set up for the Privacy Aware Framework.

In Figure 17, the FHIR server, representing the hospital's data store, is shown running inside of the same Kubernetes cluster as the PAF. This is one way of guaranteeing that all data requests must go through the Privacy Aware Framework via the Ingress Gateway. However, it is possible to configure a PAF solution keeping the data store outside of the Kubernetes cluster. In this case, the data store needs to be password-protected, and the access credentials to the data store reside in PAF as Kubernetes secrets, rather than made available to individual data requesters.

5.1.3.1.2.2.3 User Access Considerations

Requests for data are issued as REST commands and need to be accompanied by a JWT in the header of the request. The Privacy Aware Framework will determine whether or not a user can the stored data based on an evaluation of the identifying parameters in the JWT and the access policy defined for the PAF.

5.1.3.1.2.2.4 Accessing the System

As discussed, a JWT needs to identify each request to the PAF. Typically, the JWT will be produced by an Identity Management system. The details and implementation of this are outside the scope of PAF.

5.1.3.1.2.2.5 System Organization & Navigation

Detailed explanations of the Fybrik, which is the framework upon with the PAF is built, can be found in the extensive documentation in: https://fybrik.io/v1.3/reference/crds/

5.1.3.1.2.2.6 Exiting the System

The Privacy Aware Framework, once installed, will stay up and configured for as long as its services are required. The easiest way to totally remove PAF and all of its artifacts would be to delete the Kubernetes clusters upon which it was installed.

5.1.3.1.2.3 Using the System

Once configured, it is expected that only the data policy access could potentially change as a result of new legislation or internal policy. The policy (or, to be more exact, a list of rules), is defined in the Rego language as described in https://www.openpolicyagent.org/docs/latest/policy-language/.

5.1.3.1.2.3.1 Changing the data protection policy

To change the data protection policy, the original Rego file with the policy definitions needs to be modified. Once done, this file needs to be reapplied. If the PAF implemented was created following one of the examples found in the github repo (e.g. https://github.com/fybrik/REST-read-example), then simply reapply the updated Rego YAML file, following the original installation directions. For example, for the REST read example cited above, this would be to simply execute the script, applyPolicy.sh.

5.1.3.2 Blockchain – based Auditing Mechanism

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 behaviours 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[7], 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. The smart contract that defines the logic of the blockchain's

[7] https://hyperledger-fabric.readthedocs.io/en/release-2.5/

transactions is written in Java, while the REST API exposing the creation and filtering capabilities of the smart contract is a Spring Boot application written in Java.

5.1.3.2.1 Installation manual

5.1.3.2.1.1 Prerequisites

Hyperledger Fabric clearly defines the prerequisites required to install and test a docker-based network[8]. In addition, the blockchain-based auditing mechanism is containerised using Docker. Thus, the prerequisites are defined as follows:

- Git: Install git on your machine to be able to clone Hyperledger Fabric github project that contains all the required Docker images, binaries and testing samples.
- Docker: Install Docker on your machine. Docker provides a platform for containerization, allowing you to run applications in isolated environments.
- Docker Compose: Install Docker Compose, which is a tool used to define and run multi-container Docker applications. Docker Compose will be sed for the local, standalone installation and deployment of the Auditing mechanism
- cURL (optional): Install cURL as a helpful step towards the retrieval of all Hyperledger Fabric's components required to set up a blockchain network.

The provided docker-compose incorporates:

- The image per blockchain component (retrieved through WELLICS Gitlab Repository)
- A CouchDB database to store and retrieve the incoming logs.

It should be noted that the auditing mechanism is not a stand-alone component of HEIR, but it is part of PAF framework. In order to be properly installed, an installation of PAF and a dedicated Kubernetes cluster should be present in the system. For the integration with PAF in a Kubernetes environment (as is the case in the NSE pilot), Kubernetes configuration files have been created. While the majority of the present manual is for local, docker based deployment of the blockchain component itself, the available developed Kubernetes configuration files, make the deployment in such an environment a really simple task.

5.1.3.2.1.2 Pre-installation tasks

After the installation of the prerequisites, the following steps should be followed:

- Clone Hyperledger Fabric latest update from the project's Github repository[9]. This step sets up the environment by downloading the latest Hyperledger Fabric Docker images and, most importantly, by downloading the binary files required to set up a blockchain network.
- Retrieve the Docker images compiled by WELLICS. These images are tailored to the needs of the auditing mechanism and its specific requirements per blockchain's component.

The auditing mechanism is part of PAF framework, but its installation can be completed and tested independently of its integration with PAF with dummy data that resemble the expected income log from PAF framework. The integration with PAF is completed via Kubernetes and the interconnection between the two components is achieved via a dedicated Kafka network.

The developed Kubernetes configuration yaml files contain all the necessary configuration parameters so that the Auditing mechanism component is deployed, its API exposed internally to the Kubernetes cluster and integrated internally with the PAF.

5.1.3.2.1.3 Installation procedure (Docker based)

[8] https://hyperledger-fabric.readthedocs.io/en/release-2.5/prereqs.html
[9] https://github.com/hyperledger/fabric-samples

Once all prerequisite checks and controls are positively finished, the install process can start. The installation is as follows:

- Login to WELLICS Gitlab Container registry to be able to download the latest images of the blockchain's components.
- Start the Docker Containers: Use the docker-compose up command to start the Docker containers defined in the Docker Compose file. Docker Compose will pull the necessary images, create the containers, and wire them together based on the defined configuration.
- Install chaincode/smart contract on the network: Use the docker-compose-tools.yml file to install the smart contract that incorporates the transactions' logic. The heir-tools container that will be created will ensure the installation of the smart contract and the inclusion of a dedicated container for the chaincode functionalities. After the installation, the heir-tools container will stop automatically.

5.1.3.2.1.4 Installation procedure (Kubernetes based)

The same principles and order described in the docker based deployment procedure are used for the Kubernetes deployment of the Auditing mechanism. For the deployment process the Kubernetes specific yaml configuration files developed are applied in specific order.

The command to apply the yaml files is the following (provided one has command line interface access to the target Kubernetes cluster):

kubectl apply –f <component_name>.yml

kubectl apply –f <component_name>_svc.yml

The former applies, activates and rolls out each component independently, while the latter applies the component configuration as a Kubernetes service (along with networking details)

-
- The yaml files for the Orderer node are applied after.
- The yaml files for the peer nodes are applied after.
- The yaml file for the network configuration is then applied. The network configuration is applied as a a one time run service that manages the Auditing mechanism and underlying blockchain network configuration (smart contracts installation among other configuration steps)
- The yaml files for the Auditing mechanism Client

All Kubernetes pods can be checked for issues that appear in the respective pods' logs by issuing the command:

Kubectl logs –f <pod_name>

5.1.3.2.1.5 Post-installation

Following the successful installation, the end-user will need to perform a docker log inspection to:

- Identify if any exceptions/errors were raised during the initiation of the docker-compose.
- Ensure the correct initialization of the blockchain's internal network.
- Ensure the correct installation and initialization of the chaincode/smart contract.
- Make sure that the REST API application is running and is connected to Kafka network.

After the inspection of the logs, the user should connect to API's swagger user interface to inspect the correct display of the various endpoints created by the application and perform a test via dedicated test endpoints to inspect the procedure of storing and retrieving a log.

5.1.3.2.1.6 Update procedure

Updates on the auditing mechanism are being driven by the project requirements. More specifically, WELLICS is responsible to update the components and make them available to its Gitlab Container Registry. Upon update, an e-mail is sent notifying users about the need to pull the newest image.

If a major update occurs, the user is responsible for backup the database and the current deployment.

5.1.3.2.1.7 Uninstall or Roll-back procedure

The auditing mechanism is part of PAF framework, permitting its uncorrupted functionality. Though, a formal notification needs to be sent to the Technical Coordinator of PAF's implementation and the Technical Coordinator of HEIR, should an end-user wishes to uninstall them from its system. Following this, the TC will notify the corresponding partner, which, in turn, will be responsible to uninstall the components.

5.1.3.2.2 User manual

5.1.3.2.2.1 Introduction

The blockchain-based auditing mechanism is responsible for storing all data access requests performed by users that are caught by PAF framework, regardless of whether the requests were eventually authorized or not by PAF. The objectives of the auditing mechanism is to provide an immutable record of all data access requests, filtering mechanisms provided to the auditor in order to analyze and detect the stored logs and identify patterns of malicious behaviour, and the capability to develop timelines of events based on various filtering options.

The auditing mechanism is based on the creation of a blockchain network (with the utilization of Hyperledger Fabric framework), accessible via a dedicated client application developed using Spring Boot framework. The client application exposes a set of endpoints (API) enabling the end-user to interact with the storing and, mainly, filtering functionalities of the mechanism, incorporated by a Java-based smart contract. The client application is provided as an easy-to-use gateway to the logs stored in the blockchain network, helping the auditor extract useful information and observe the timeline of events.

This manual provides the necessary information for end-users to effectively interact and use the client application of the auditing mechanism.

5.1.3.2.2.2 Getting Started

5.1.3.2.2.2.1 Cautions & Warnings

Hyperledger Project source code files are made available under the Apache License, Version 2.0 (Apache-2.0).

5.1.3.2.2.2.2 Set-up Considerations

Set-up considerations are thoroughly discussed in the provided Installation manual.

5.1.3.2.2.2.3 User Access Considerations

An end-user can access and interact with the client application as soon as he/she has access to the system where PAF and the auditing mechanism are installed. The client application provides the option to create dedicated digital keys for each user, a procedure that requires the approval of an administrator of the system.

5.1.3.2.2.2.4 Accessing the System

No specific authentication/authorization requirements for accessing the application.

5.1.3.2.2.2.5 System Organization & Navigation

The application has two main functionalities: create (store) a new registration (log) to the blockchain network and filter (query) the existing logs. The client application is integrated with PAF via a dedicated Kafka network and topic and constantly listens for new messages. The messages contain extensive logs of the data access attempts caught by PAF. Specific information fields are retrieved from these messages and form the final log that will be stored in the blockchain via a creation/storing method that utilizes the Fabric Gateway framework[10]. The same framework is utilized for the development of various filtering methods. The storing logs procedure is automated and is based on listening to Kafka messages, while the filtering methods are exposed via endpoints. All endpoints are available via swagger User Interface on the installation environment and on port 8081.

5.1.3.2.2.2.6 Exiting the System

The auditing mechanism, as part of the PAF, once installed, will stay up and configured for as long as its services are required. The easiest way to totally remove PAF and all of its artifacts would be to delete the Kubernetes clusters upon which it was installed.

5.1.3.2.2.3 Using the System

The usage of the system is described as a set of interactions with the available endpoints that are presented on the next table:

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

[10] https://github.com/hyperledger/fabric-gateway

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

5.2 HEIR core framework components

5.2.1 HEIR Client's Processing system (HEIR Client)

5.2.1.1 Installation guide

HEIR Agent is the component providing the TDM, HET and VA modules. It runs periodic scans for misconfigurations and CVEs. It will also run scans on specific files when triggered, detecting malicious ones.

5.2.1.1.1 Prerequisites

Pre-requisites for the HEIR Agent to function properly:

- Windows 10 64bit or newer
- A Kafka broker (not necessarily on the same machine)
- A Heirclient instance (not necessarily on the same machine)

Required files, in addition to the installer:

- a `config.json` file
- a `signed.key` file containing the SSL key required by the Kafka broker
- a `signed.pem` file containing the SSL certificate required by the Kafka broker
- a `trusted_authority.cert` file containing the SSL ca required by the Kafka broker

The config.json file required by the HEIR Agent has the following format:

```
{
  "kafka_broker": "<ip and port of the Kafka broker>",
  "scan_id": "008f4070-154f-11ec-a544-f7bbc18a132a",
  "hospital_address": "<the address of the hospital>",
  "hospital_region": "<the region of the hospital>",
  "clientID": <the id of this client>,
  "hospitalID": <the id of the hospital>,
  "time_betwen_scans_minutes": 40,
  "scan_target": "<absolute path of the folder that will be monitored for malware files by TDM>"
}
```

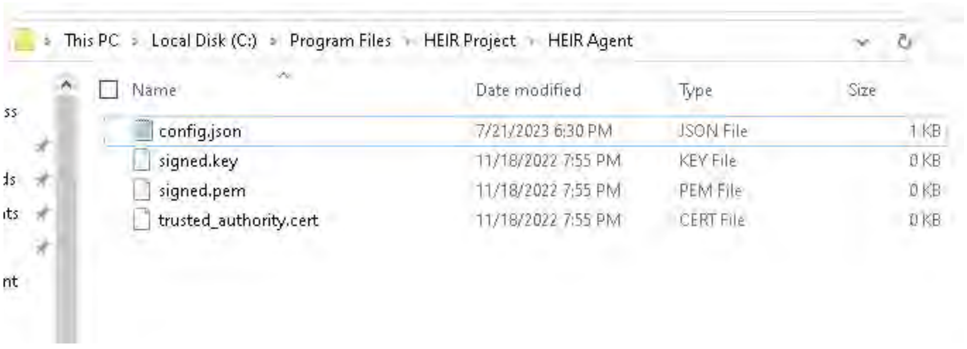
5.2.1.1.2 Pre-installation tasks

N/A

5.2.1.1.3 Installation procedure

Once all prerequisite checks and controls are positively finished, the install process can start.

First, we should prepare our configurations by creating the following folder: C:\Program Files\HEIR Project\HEIR Agent. Inside this folder place your configuration file and SSL key, certificate and ca files.

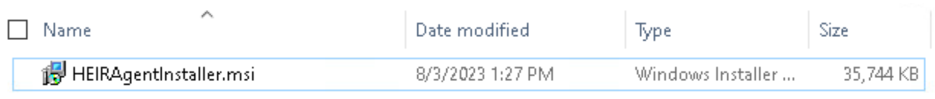


This PC > Local Disk (C:) > Program Files > HEIR Project > HEIR Agent

Name	Date modified	Type	Size
config.json	7/21/2023 6:30 PM	JSON File	1 KB
signed.key	11/18/2022 7:55 PM	KEY File	0 KB
signed.pem	11/18/2022 7:55 PM	PEM File	0 KB
trusted_authority.cert	11/18/2022 7:55 PM	CERT File	0 KB

Figure 18: Required files for HEIR agent installation

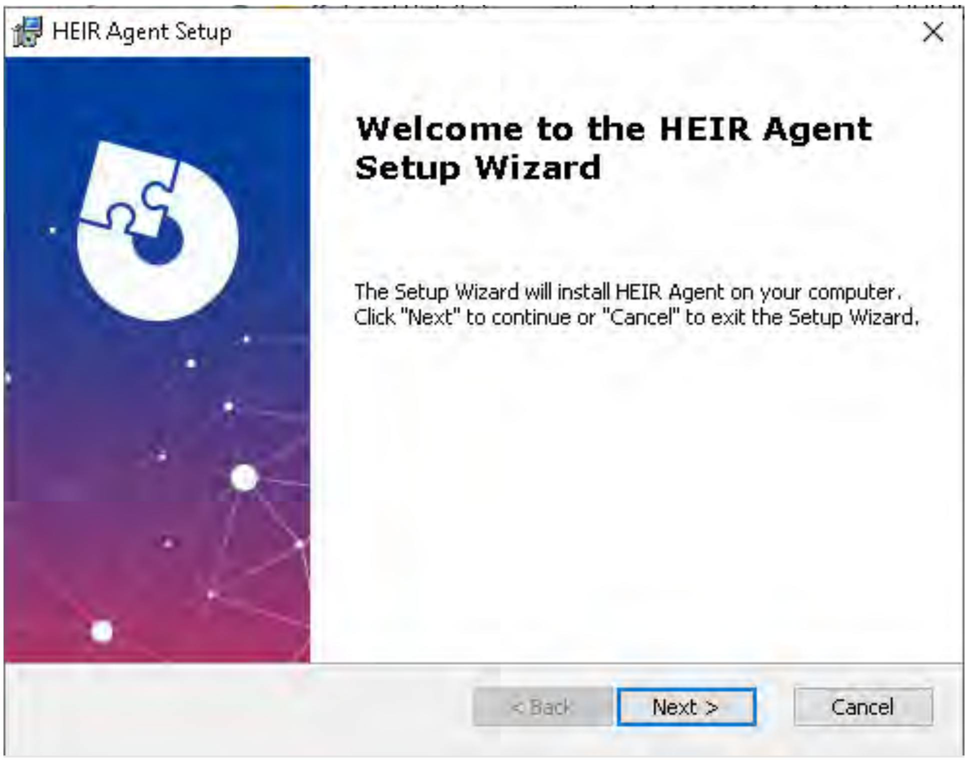
Next, we shall run the installer. Double click the MSI installer to run it.



Name	Date modified	Type	Size
HEIRAgentInstaller.msi	8/3/2023 1:27 PM	Windows Installer ...	35,744 KB

Figure 19: HEIR agent installer

The installer will start and prompt you to continue.



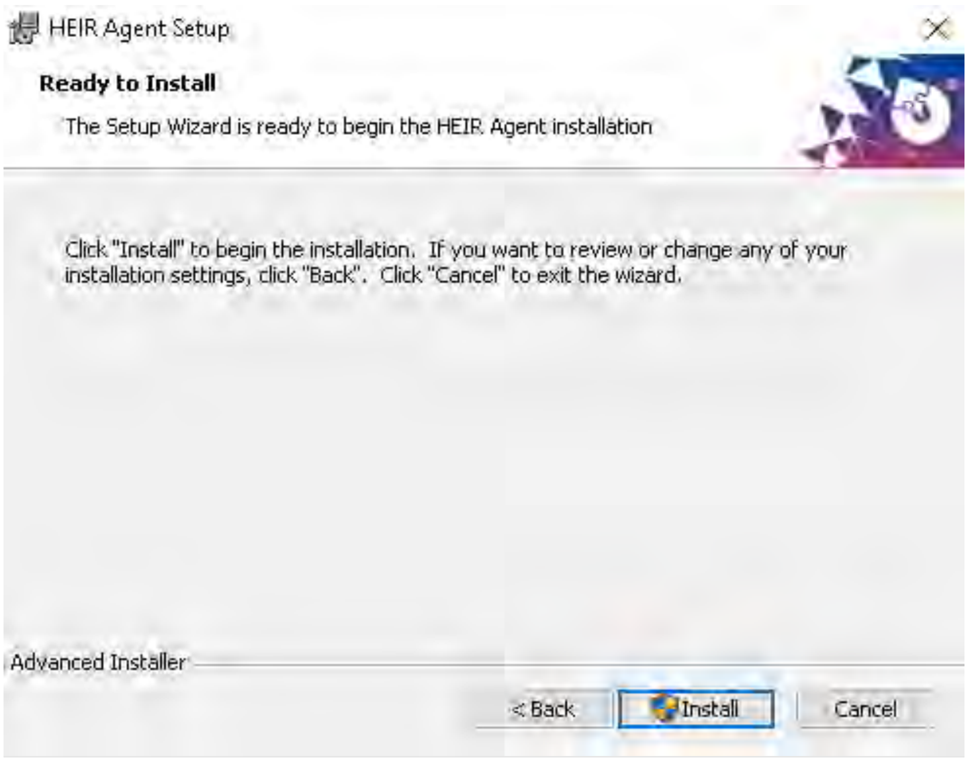
HEIR Agent Setup

Welcome to the HEIR Agent Setup Wizard

The Setup Wizard will install HEIR Agent on your computer. Click "Next" to continue or "Cancel" to exit the Setup Wizard.

Figure 20: First window of installation procedure

Next, it will prompt you to start the install process. This step will require administrator privileges.


HEIR Agent Setup

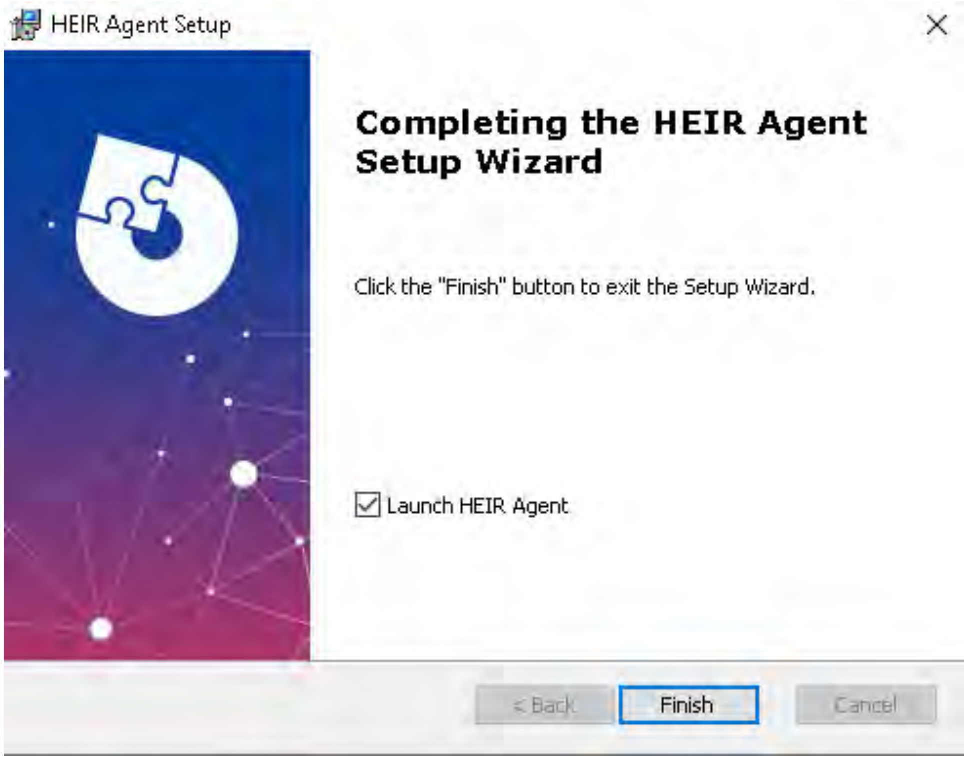
Ready to Install

The Setup Wizard is ready to begin the HEIR Agent installation

Click "Install" to begin the installation. If you want to review or change any of your installation settings, click "Back". Click "Cancel" to exit the wizard.

Figure 21: Second window of installation procedure - the user should press install button to continue

In the next step the installer will launch the HEIR Agent service. We recommend you leave the option ticked, otherwise you will have to launch the process manually.


HEIR Agent Setup

Completing the HEIR Agent Setup Wizard

Click the "Finish" button to exit the Setup Wizard.

☑ Launch HEIR Agent

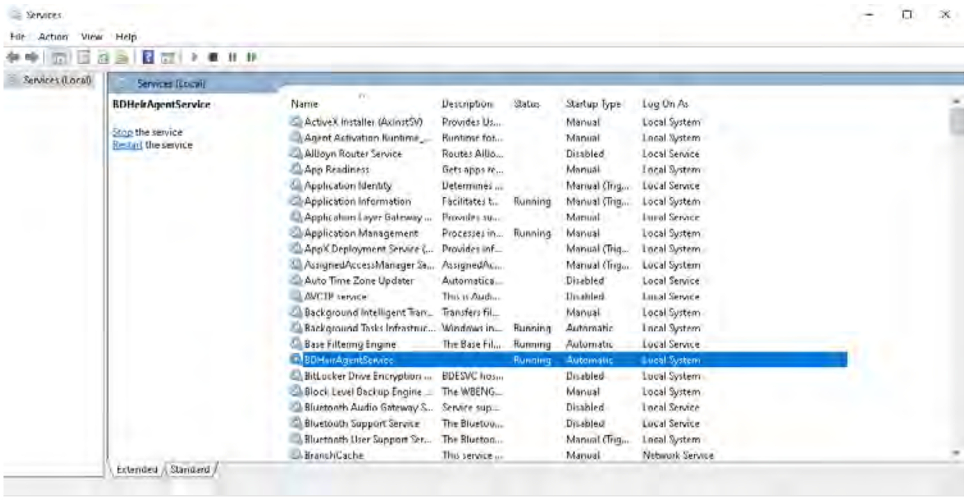
Figure 22: End of installation procedure. The HEIR agent has been successfully installed

Now, the HEIR Agent should be installed and running.

5.2.1.1.4 Post-installation

To verify that the application is running properly we can do the following checks:

1. Check that the HEIR Agent service is running. To do this open the Services app and look for the BDHeirAgentService. It's status should be: Running.



Services

File Action View Help

Services (Local)

BDHeirAgentService

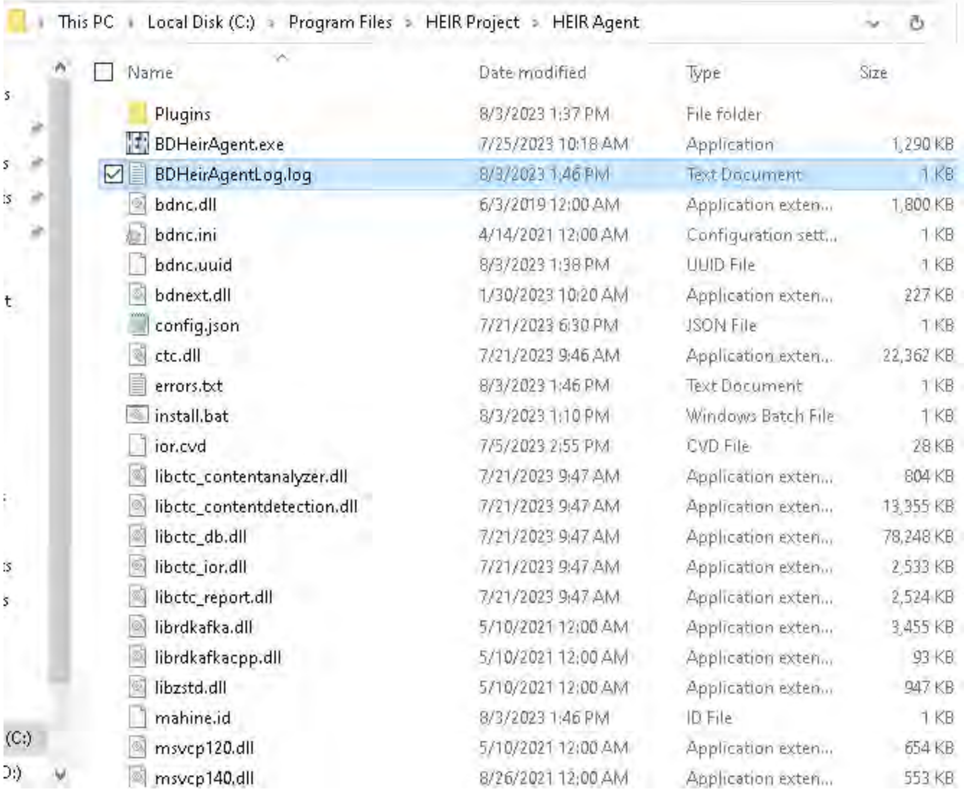
Stop the service
Restart the service

Name	Description	Status	Startup Type	Log On As
ActiveX Installer (AxInstSV)	Provides Us...		Manual	Local System
Agent Activation Runtime_...	Runtime for...		Manual	Local System
AllJoyn Router Service	Routes AllJo...		Disabled	Local Service
App Readiness	Gets apps re...		Manual	Local System
Application Identity	Determines ...		Manual (Trig...	Local Service
Application Information	Facilitates t...	Running	Manual (Trig...	Local System
Application Layer Gateway ...	Provides su...		Manual	Local Service
Application Management	Processes in...	Running	Manual	Local System
AppX Deployment Service (...	Provides inf...		Manual (Trig...	Local System
AssignedAccessManager Se...	AssignedAc...		Manual (Trig...	Local System
Auto Time Zone Updater	Automatica...		Disabled	Local Service
AVCTP service	This is Audi...		Disabled	Local Service
Background Intelligent Tran...	Transfers fil...		Manual	Local System
Background Tasks Infrastruc...	Windows in...	Running	Automatic	Local System
Base Filtering Engine	The Base Fil...	Running	Automatic	Local Service
BDHeirAgentService		Running	Automatic	Local System
BitLocker Drive Encryption ...	BDESVC hos...		Disabled	Local System
Block Level Backup Engine ...	The WBENG...		Manual	Local System
Bluetooth Audio Gateway S...	Service sup...		Disabled	Local Service
Bluetooth Support Service	The Bluetoo...		Disabled	Local Service
Bluetooth User Support Ser...	The Blueton...		Manual (Trig...	Local System
BranchCache	This service ...		Manual	Network Service

Extended / Standard

Figure 23: Checking HEIR agent service status

2. Check that the temporary files have been created. To do this go to the installation folder and look for the BDHeirAgentLog.log file. If it is present the application started correctly.



This PC > Local Disk (C:) > Program Files > HEIR Project > HEIR Agent

Name	Date modified	Type	Size
Plugins	8/3/2023 1:37 PM	File folder	
BDHeirAgent.exe	7/25/2023 10:18 AM	Application	1,290 KB
BDHeirAgentLog.log	8/3/2023 1:46 PM	Text Document	1 KB
bdnc.dll	6/3/2019 12:00 AM	Application exten...	1,800 KB
bdnc.ini	4/14/2021 12:00 AM	Configuration sett...	1 KB
bdnc.uuid	8/3/2023 1:38 PM	UUID File	1 KB
bdnext.dll	1/30/2023 10:20 AM	Application exten...	227 KB
config.json	7/21/2023 6:30 PM	JSON File	1 KB
ctc.dll	7/21/2023 9:46 AM	Application exten...	22,362 KB
errors.txt	8/3/2023 1:46 PM	Text Document	1 KB
install.bat	8/3/2023 1:10 PM	Windows Batch File	1 KB
ior.cvd	7/5/2023 2:55 PM	CVD File	28 KB
libctc_contentanalyzer.dll	7/21/2023 9:47 AM	Application exten...	804 KB
libctc_contentdetection.dll	7/21/2023 9:47 AM	Application exten...	13,355 KB
libctc_db.dll	7/21/2023 9:47 AM	Application exten...	78,248 KB
libctc_ior.dll	7/21/2023 9:47 AM	Application exten...	2,533 KB
libctc_report.dll	7/21/2023 9:47 AM	Application exten...	2,524 KB
librdkafka.dll	5/10/2021 12:00 AM	Application exten...	3,455 KB
librdkafkacpp.dll	5/10/2021 12:00 AM	Application exten...	93 KB
libzstd.dll	5/10/2021 12:00 AM	Application exten...	947 KB
mahine.id	8/3/2023 1:46 PM	ID File	1 KB
msvcp120.dll	5/10/2021 12:00 AM	Application exten...	654 KB
msvcp140.dll	8/26/2021 12:00 AM	Application exten...	553 KB

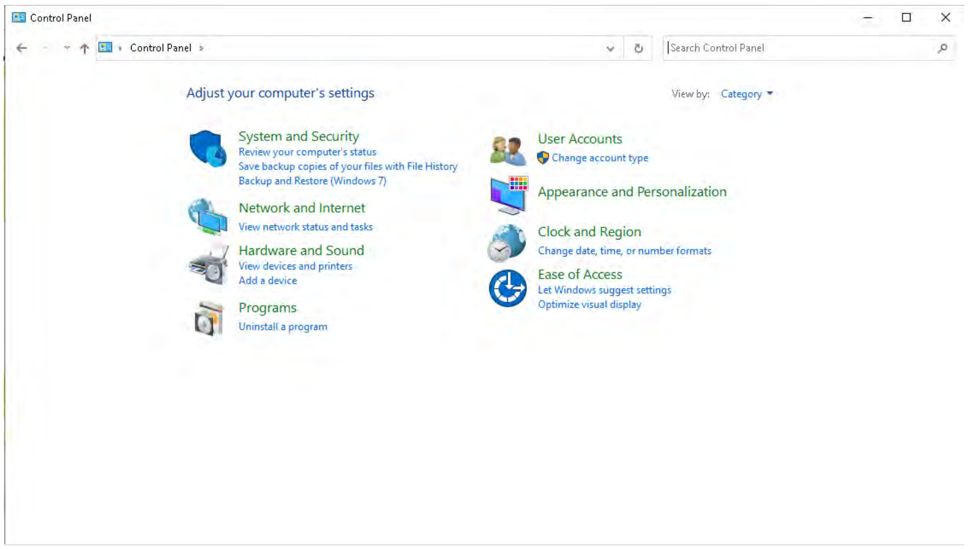
Figure 24: Checking the creation of temporary files in BDHeirAgentLog.log file

5.2.1.1.5 Update procedure

To update the HEIR Agent, you will have to uninstall the program by going through the steps outlined in the **Uninstall or Roll-back procedure**. After that you can proceed with the **Installation procedure** using the updated MSI installer.

5.2.1.1.6 Uninstall or Roll-back procedure

If you want to uninstall the HEIR Agent, either for good or just to re-install a newer/older version the procedure is the same. Start by going to the Control Panel:


Control Panel

Control Panel

Search Control Panel

Adjust your computer's settings

View by: Category

System and Security
Review your computer's status
Save backup copies of your files with File History
Backup and Restore (Windows 7)

Network and Internet
View network status and tasks

Hardware and Sound
View devices and printers
Add a device

Programs
Uninstall a program

User Accounts
Change account type

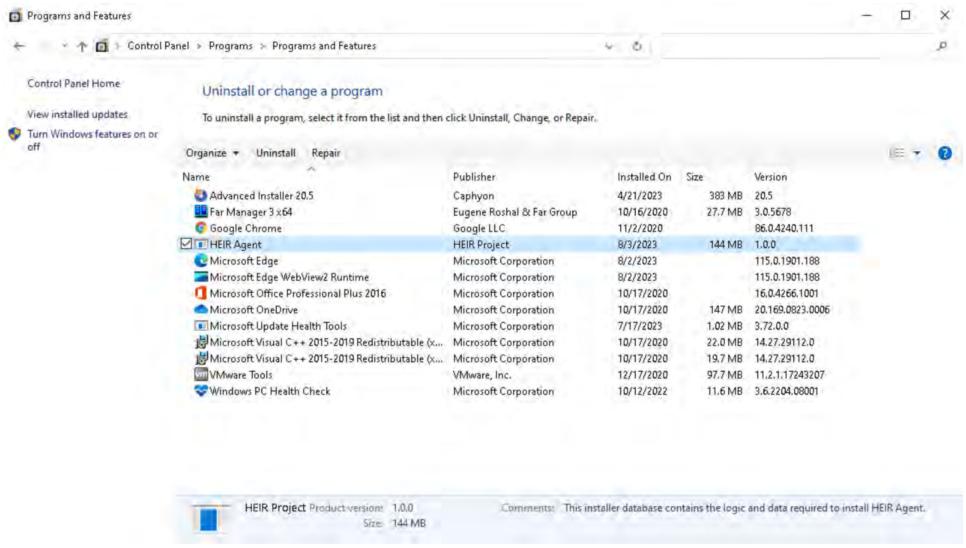
Appearance and Personalization

Clock and Region
Change date, time, or number formats

Ease of Access
Let Windows suggest settings
Optimize visual display

Figure 25: Access control panel

And open the Uninstall a program page under Programs.


Programs and Features

Control Panel > Programs > Programs and Features

Control Panel Home

View installed updates

Turn Windows features on or off

Uninstall or change a program

To uninstall a program, select it from the list and then click Uninstall, Change, or Repair.

Organize Uninstall Repair

Name	Publisher	Installed On	Size	Version
Advanced Installer 20.5	Caphyon	4/21/2023	383 MB	20.5
Far Manager 3 x64	Eugene Roshal & Far Group	10/16/2020	27.7 MB	3.0.5678
Google Chrome	Google LLC	11/2/2020		86.0.4240.111
HEIR Agent	HEIR Project	8/3/2023	144 MB	1.0.0
Microsoft Edge	Microsoft Corporation	8/2/2023		115.0.1901.188
Microsoft Edge WebView2 Runtime	Microsoft Corporation	8/2/2023		115.0.1901.188
Microsoft Office Professional Plus 2016	Microsoft Corporation	10/17/2020		16.0.4266.1001
Microsoft OneDrive	Microsoft Corporation	10/17/2020	147 MB	20.169.0823.0006
Microsoft Update Health Tools	Microsoft Corporation	7/17/2023	1.02 MB	3.72.0.0
Microsoft Visual C++ 2015-2019 Redistributable (x...	Microsoft Corporation	10/17/2020	22.0 MB	14.27.29112.0
Microsoft Visual C++ 2015-2019 Redistributable (x...	Microsoft Corporation	10/17/2020	19.7 MB	14.27.29112.0
VMware Tools	VMware, Inc.	12/17/2020	97.7 MB	11.2.1.17243207
Windows PC Health Check	Microsoft Corporation	10/12/2022	11.6 MB	3.6.2204.08001

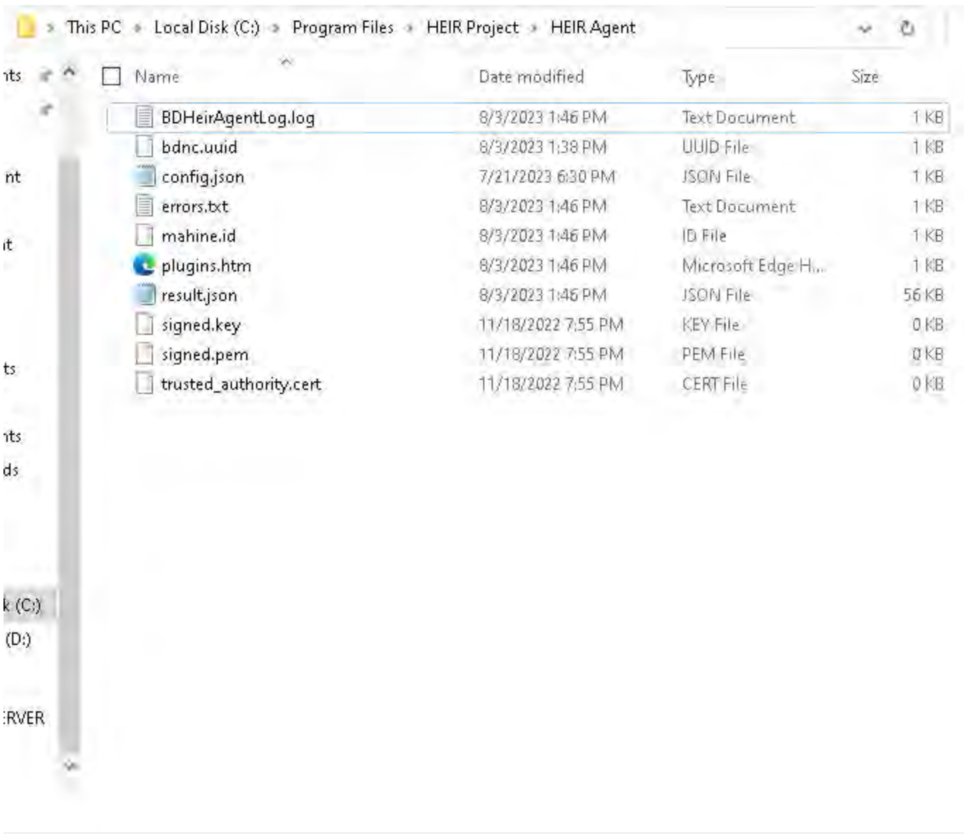
HEIR Project Product version: 1.0.0
Size: 144 MB
Comments: This installer database contains the logic and data required to install HEIR Agent.

Figure 26: Uninstall or change a program panel - find HEIR agent program

In the list of installed programs look for HEIR Agent, right click and press Uninstall.

This will open an Uninstall Wizard that will stop the service and delete most of the files added by the installer.

If you go to the install folder, you will see that only the files added in the first install step and a few temporary files are left.



This PC › Local Disk (C:) › Program Files › HEIR Project › HEIR Agent

Name	Date modified	Type	Size
BDHeirAgentLog.log	8/3/2023 1:46 PM	Text Document	1 KB
bdnc.uuid	8/3/2023 1:38 PM	UUID File	1 KB
config.json	7/21/2023 6:30 PM	JSON File	1 KB
errors.txt	8/3/2023 1:46 PM	Text Document	1 KB
mahine.id	8/3/2023 1:46 PM	ID File	1 KB
plugins.htm	8/3/2023 1:46 PM	Microsoft Edge H...	1 KB
result.json	8/3/2023 1:46 PM	JSON File	56 KB
signed.key	11/18/2022 7:55 PM	KEY File	0 KB
signed.pem	11/18/2022 7:55 PM	PEM File	0 KB
trusted_authority.cert	11/18/2022 7:55 PM	CERT File	0 KB

Figure 27: Verification of HEIR agent successful uninstall

If you want to delete the program for good, you can delete these manually. If you plan on re-installing the agent, they can be left in place.

5.2.1.2 *User manual*

N/A

5.2.2 Local and Global RAMA Score Calculator

5.2.2.1 *Installation guide*

Both the Local and the Global RAMA score calculators are software components that were written in Java, using the Spring Boot framework. The Local RAMA score calculator is responsible to communicate with the HEIR client, calculate the local RAMA score, construct the corresponding metadata, and forward them to the HEIR Aggregator. The Global RAMA score calculator communicates with the HEIR Aggregator, retrieves the aggregated scores and the corresponding metadata, calculates the Global RAMA score and then sends it – alongside the metadata- to the Observatory. All the communications are event-driven.

5.2.2.1.1 Prerequisites

Both the Local and Global RAMA score calculators are containerised using Docker Compose. Thus, the system must have the following:

- Docker: Install Docker on your machine. Docker provides a platform for containerization, allowing you to run applications in isolated environments.
- Docker Compose: Install Docker Compose, which is a tool used to define and run multi-container Docker applications.

The provided docker-compose incorporates:

- The image per component (retrieved through SPHYNX's Gitlab Repository)
- A Postgres database to store the results.

- The PgAdmin to examine the database.
- The configuration files related to the secrets (e.g., database passwords etc.) per docker image.

5.2.2.1.2 Pre-installation tasks

The local and the global RAMA score calculators depend on other HEIR-related components. Thus, to receive the local and global RAMA score, the following components needs to be installed:

- HEIR Client: The component responsible to provide the input to the Local RAMA Score calculator.
- HEIR Aggregator: The component responsible to receive the output of the Local RAMA Score Calculator and provide the input to the Global RAMA score calculator.

Moreover, the connection between these components (through HEIR's Message Broker) needs to be tested.

5.2.2.1.3 Installation procedure

Once all prerequisite checks and controls are positively finished, the install process can start. The installation is as follows:

- Login to SPHYNX's Gitlab Container registry to download the latest local and global RAMA score calculators.
- Start the Docker Containers: Use the docker-compose up command to start the Docker containers defined in the Docker Compose file. Docker Compose will pull the necessary images, create the containers, and wire them together based on the defined configuration.

5.2.2.1.4 Post-installation

Following the successful installation, the end-user will need to:

- Perform a docker log to identify if any exceptions/errors were raised during the initiation of the docker-compose.
- Examine the logs to make sure that the HEIR client/HEIR aggregator reports the expected input.
- Examine the logs to make sure that the Local/Global RAMA score calculator produces the results.

5.2.2.1.5 Update procedure

Updates on the Local and Global RAMA score calculators are being driven by the project requirements. More specifically, SPHYNX is responsible to update the components and make them available to its Gitlab Container Registry. Upon update, an e-mail is sent notifying users about the need to pull the newest image.

If a major update occurs, the user is responsible for backup the database and the current deployment.

5.2.2.1.6 Uninstall or Roll-back procedure

The Local and Global RAMA score calculators are core components of the project. Thus, a formal notification needs to be sent to the Technical Coordinator (TC), should an end-user wishes to uninstall them from its system. Following this, the TC will notify the corresponding partner, which, in turn, will be responsible to uninstall the components.

5.2.2.1.7 Technical Tests

During the implementation phase, SPHYNX employed a test methodology that involved using Junit5 and Mockito for unit and integration testing, respectively. In unit testing, individual code components or modules were tested independently to verify that each works as expected. JUnit5 is a popular testing framework for Java that provides annotations to define test methods, lifecycle methods, and assert statements. In this methodology, JUnit5 is used for writing and executing unit tests for each module of the Java application. Each test case is designed to validate the functionality of a specific module or class.

Mockito is a popular mocking framework that allows developers to create mock objects and use them to simulate dependencies in unit tests. Integration testing is the process of testing multiple modules together to verify their interactions and ensure that the application as a whole works as expected. Integration tests are designed to test the interfaces between modules and detect defects that arise when modules interact with each other. In this methodology, integration tests are designed to test the integration points between modules. Mockito was also used to create mock objects that simulate the behaviour of external dependencies.

Moreover, given that these components communicate with other HEIR-related components, the following end-to-end tests were performed.

R1	**GET Data from the HEIR Client**
Description	HEIR Client's output is being sent to the Local RAMA score calculator
Actors	SPHYNX, BitDefender
Preconditions	The HEIR client is up and running
Steps	
Step 1	Connect to pilots' VM
Step 2	Docker log the local RAMA calculator
Step 3	Examine the logs to see if a new entry was received
Step 4	Verify that the event was received
Expected Results	A log message showing the entry is shown.
Actual Results	{ "hospitalId":"1", "clientId":"5", "severity":"60", "client_name":"Some client", "scan_id":"a06beae7-f742-3285-e74e83b6a7008a72", "host_name":"HEIR-WIN10-2", "host_os":"Windows 10", "scan_status":"finished", "machine_id":"0102CC3601029F9E0102DE280102BB9F010280F30102E8B30102D9B8010280570102CF370102C0FB0102AA4A0102A1AA0102B6E20102EA600102C9D00102B962", "hospital_region":"PAGNI", "hospital_address":"Street no 1", "connected_clients":"6", "het":[{ "triggered":"true", "events":[]}], "vulnerabilityAssessment":[{ "triggered":"true", "events":[]}], "hcc":[{ "triggered":"true", "sslTest":null}], "hnm":[{ "triggered":"false", "events":[]}], "siem":[{ "triggered":"false", "events":[]}], "tdm":[{ "triggered":"false", "events":[]}]}

R2	**Local RAMA score output**
Description	Examine that, upon receiving an event, the local RAMA score is being produced
Actors	SPHYNX
Preconditions	The Local RAMA score is up and running

Steps	
Step 1	Connect to pilots' VM
Step 2	Docker log the local RAMA calculator
Step 3	Examine the logs to see if a new output exists.
Step 4	Verify that the output is correct.
Expected Results	A log message showing the entry is shown.
Actual Results	{ "hospitalId":"1", "clientId":"5", "host_name":"HEIR-WIN10-2", "machine_id":"0102CC3601029F9E0102DE280102BB9F010280F30102E8B30102D9B8010280570102CF370102C0FB0102AA4A0102A1AA0102B6E20102EA600102C9D00102B962", "hospital_region":"PAGNI", "hospital_address":"Street no 1", "connected_clients":"6", "severity":"60", "hccMetadata":null, "hetMetadata":null, "vulnerabilityAssessmentAggregatedMetadata":{ "vulnerabilityAssessmentMetadata":[{ "application_name":"Docker Desktop 4.6.1", "noOfVulnerabilities":"2", "machineId":"0102CC3601029F9E0102DE280102BB9F010280F30102E8B30102D9B8010280570102CF370102C0FB0102AA4A0102A1AA0102B6E20102EA600102C9D00102B962", "vulnerabilities":[CVE-2018-10892, CVE-2023-0628]}], "top10Vulnerabilities":{"CVE-2023-0628":77,"CVE-2018-10892":50}, "top10MostFrequentVulnerabilities":{"CVE-2023-0628":7066,"CVE-2018-10892":7066}, "totalNoOfVulnerabilities":"2", "lastTriggered":null}, "hnmMetadata":null, "siemMetadata":null, "indicators":{ "hnmScore":"0.0", "hccScore":"0.0", "hetScore":"0.0", "vaScore":"8.0", "siemScore":"0.0", "calculated":"null"}, "numberOfCriticalEvents":"0", "lastTriggered":null}

R3	**GET Data from the HEIR Aggregator**
Description	HEIR Aggregator output is being sent to the Global RAMA score calculator
Actors	SPHYNX, Siemens
Preconditions	The HEIR aggregator is up and running
Steps	
Step 1	Connect to the observatory VM
Step 2	Docker log the Global RAMA score calculator
Step 3	Examine the logs to see if a new entry was received
Step 4	Verify that the event was received
Expected Results	A log message showing the entry is shown.
Actual Results	{ "hospitalId":"1", "created":2023-06-22 10:20:03.0, "clientIdList":[5, 3, 6], "noOfClients":"3", "globalRamaScore":"99.29222", "cyberSecurityStatus":"null", "globalTemporalScore":"0.0", "globalBaseScore":"1.0111111",

	"numberOfCriticalEvents":"0", "numberOfIdentifiedHeartbleeds":"0", "noOfOSVulnerabilities":"0", "noOfMisconfigurations":"2", "numberOfBenignFindings":"2", "numberOfMaliciousFindings":"0", "noOfAppVulnerabilities":"3", "numberOfAttacks":"0", "numberOfExploits":"0", "totalHNMFindings":"0", "numberOfSIEMCriticalEvents":"0", "numberOfSIEMMediumEvents":"0", "numberOfSIEMHighEvents":"0", "numberOfSIEMLowEvents":"0", "totalNumberOfSIEMEvents":"0", "top10Vulnerabilities":{CVE-2023-0628=77, CVE-2022-29072=72, CVE-2018-10892=50}, "top10FreqVulnerabilities":{CVE-2022-29072=7643, CVE-2023-0628=7066, CVE-2018-10892=7066}}

R4	**Global RAMA score output**
Description	Examine that, upon receiving an event, the Global RAMA score is being produced
Actors	SPHYNX
Preconditions	The Global RAMA score is up and running
Steps	
Step 1	Connect to the observatory VM
Step 2	Docker log the Global RAMA score calculator
Step 3	Examine the logs to see if a new entry was produced.
Step 4	Verify that the Global RAMA score calculator produced an event.
Expected Results	A log message showing the entry is shown.

Actual Results	{ "id":"1", "numberOfHospitals":"3", "connectedHospitals":[1, 4, 3], "updated":"2023-06-22 10:22:15.419", "ramaScore":"99.00852", "cyberSecurityStatus":"Low", "temporalScore":"0.25", "baseScore":"1.3092594", "numberOfCriticalEvents":"0.0", "numberOfIdentifiedHeartbleeds":"0.0", "noOfOSVulnerabilities":"2.0", "noOfMisconfigurations":"1.0", "numberOfBenignFindings":"1.0", "numberOfMaliciousFindings":"0.0", "noOfAppVulnerabilities":"2.0", "numberOfAttacks":"0.0", "numberOfExploits":"0.0", "numberOfSIEMCriticalEvents":"0.0", "numberOfSIEMMediumEvents":"0.0", "numberOfSIEMHighEvents":"0.0", "numberOfSIEMLowEvents":"0.0", "totalNumberOfSIEMEvents":"0.0", "top10Vulnerabilities":{
	"CVE-2023-0628" : 77,
	"CVE-2022-29072" : 72,
	"CVE-2019-16294" : 67,
	"CVE-2022-31901" : 65,
	"CVE-2022-31902" : 55,
	"CVE-2018-10892" : 50
	}, "top10FreqVulnerabilities":{
	"CVE-2022-31902" : 13651,
	"CVE-2019-16294" : 13651,

	"CVE-2022-31901" : 13651, "CVE-2023-0412" : 11716, "CVE-2022-4345" : 11716, "CVE-2023-0413" : 11716, "CVE-2022-4344" : 11716, "CVE-2023-0416" : 11716, "CVE-2023-0417" : 11716, "CVE-2022-3724" : 11716 }, "hospitalInformation":[{ "id":"1", "ramaScore":"99.29222", "temporalScore":"0.0", "baseScore":"1.0111111", "connectedClients":"3", "criticalEvents":"0", "totalAttacks":"0"}, { "id":"3", "ramaScore":"98.55", "temporalScore":"0.75", "baseScore":"1.7500001", "connectedClients":"2", "criticalEvents":"0", "totalAttacks":"0"}, { "id":"4", "ramaScore":"99.183334", "temporalScore":"0.0", "baseScore":"1.1666667", "connectedClients":"1", "criticalEvents":"0", "totalAttacks":"0"}]}

5.2.2.2 User manual

The user does not have direct communication with the Local or Global RAMA score calculator. More specifically, the output of the Local RAMA score calculator is made available to the user through HEIR's 1st Layer GUI, whereas the output of the Global RAMA score calculator is made available through HEIR's Observatory.

5.2.3 HEIR Aggregator

5.2.3.1 Installation guide

The HEIR Aggregator is the component that makes the liaison between the HEIR client and RAMA calculator on the one hand and the HEIR's 1st layer of services GUI, and the Observatory on the other. The HEIR Aggregator was initially conceived for medical institutions with multiple departments for which individual HEIR clients and RAMA calculators were deployed, with the Aggregator collecting and producing combined scored and metadata statistics.
Over the course of the project the purpose of the Aggregator has evolved into a connecting component that performs the transfer of detected activity at the local level by the HEIR Clients and the RAMA calculator to the HEIR local GUI and to the HEIR Observatory, and it is now deployed to all participant Pilots of the HEIR project, regardless of the participating number of departments.

5.2.3.1.1 Prerequisites

The Heir Aggregator is deployed as a Docker container in Kubernetes.

The container includes a Python source code, a cron job file, as well as an .ini file to point to address for an ElasticSearch and a Kafka connection to communicate with the RAMA Calculator, the GUI and Observatory components.

To ensure a successful deployment and installation of the HEIR Aggregator component, several pre-requisites must be met. These pre-requisites encompass both hardware and software requirements to enable the proper functioning and integration of the Aggregator within the HEIR project ecosystem. Here are the key pre-requisites:

Hardware Requirements:

- Sufficient processing power and memory to handle the expected data load and processing requirements of the Aggregator.
- Stable and reliable network connectivity to facilitate communication with the RAMA calculator, HEIR's 1st layer of services GUI, and the Observatory.

Software Requirements:

- Kubernetes Environment: The deployment of the HEIR Aggregator relies on a Kubernetes environment. Ensure that a Kubernetes cluster is available and properly configured to host the Aggregator.
- Docker: The HEIR Aggregator is deployed as a Docker container. Ensure that Docker is installed and properly configured on the target deployment system.
- Elasticsearch storage
- Python Dependencies→automatically loaded in the Docker container:
    - elasticsearch: The Aggregator component requires the elasticsearch library for communication with the Elasticsearch database, where the collected data is stored.
    - pandas: The pandas library is necessary for data manipulation and processing within the Aggregator component.
    - configparser: The configparser library is used for reading and parsing the configuration file (ini file) of the Aggregator, allowing for easy customization and parameterization.
    - kafka-python: The kafka-python library is required for establishing connections and interactions with the Kafka messaging system used in the HEIR project.

Configuration Files:

- aggregator.ini File: The Aggregator utilizes an ini file for configuration purposes. This file contains various settings and parameters, such as the addresses for the Elasticsearch and Kafka connections. Ensure that the ini file is properly configured with the appropriate addresses and settings according to the specific deployment environment.

It is essential to ensure that all pre-requisites are met before proceeding with the deployment and installation of the HEIR Aggregator. Failure to meet these requirements may result in operational issues, performance degradation, or improper functioning of the Aggregator component. Therefore, adherence to hardware and software requirements, and careful configuration of the ini file are vital for a smooth and successful deployment of the HEIR Aggregator within the HEIR project.

5.2.3.1.2 Pre-installation tasks

The HEIR Aggregator relies on HEIR-related components. Thus, the following components need to already be present in the deployment environment:

- HEIR Client: The component responsible to provide the input to the Local RAMA Score calculator.
- Local RAMA Calculator: The component responsible to receive the output of the HEIR Client and provide the input to the HEIR Aggregator.

Moreover, the connection between these components needs to be tested.

5.2.3.1.3 Installation procedure

Once all prerequisite checks and controls are positively finished, the install process can start.

1. Retrieve the Docker Image:
    - Access the SIE GitLab repository where the Docker image for the HEIR Aggregator is hosted.
    - Clone the repository or download the Docker image file to your local environment.

2. Create a Kubernetes deployment manifest YAML file that specifies the configuration and resources for the HEIR Aggregator deployment.
3. Within the YAML file, reference the Docker image from the GitLab repository as the image for the HEIR Aggregator container.

Include the necessary configuration settings, environment variables, and resource requests/limits as per requirements.

5.2.3.1.4 Post-installation

Monitor Deployment:

- Use kubectl commands or Kubernetes dashboard to monitor the status of the HEIR Aggregator deployment.
- Verify that the Aggregator pod(s) are running and have successfully started within the specified namespace.
- Monitor logs and events to ensure proper initialization and connectivity to the required resources, such as Elasticsearch and Kafka.

Validate Aggregator Functionality:

- Use appropriate testing methods to validate the functionality of the HEIR Aggregator.
- Ensure that it is successfully receiving and processing data from the HEIR clients and RAMA calculator, and forwarding it to the HEIR local GUI and Observatory as expected.

5.2.3.1.5 Update procedure

1. Retrieve the Updated Docker Image
    - Access the SIE GitLab repository where the updated Docker image for the HEIR Aggregator is available.

2. Update Manifest File
    - Identify the Kubernetes deployment manifest YAML file that was used for the initial deployment of the HEIR Aggregator.
    - Modify the YAML file to reference the updated Docker image, ensuring that the latest version or tag were used.

3. Apply the Update
    - Use the kubectl command to apply the updated deployment manifest file, which will trigger the update process.

4. Monitor the Update Progress
    - Monitor the status of the update process using kubectl commands or Kubernetes dashboard.
    - Observe the rollout status, the termination of old pods, and the creation of new pods to ensure a smooth transition.

5. Validate Aggregator Functionality
    - Verify that the Aggregator is successfully receiving and processing activity data, forwarding it to the appropriate destinations (HEIR local GUI and Observatory), and operating as expected.

6. Rollback (if necessary)

- If any issues or unexpected behaviour arise during or after the update, consider rolling back to the previous version.
- Revert to the previous deployment manifest file or apply a backup to restore the previous state of the HEIR Aggregator.

5.2.3.1.6 Uninstall or Roll-back procedure

The HEIR Aggregator is a core component of the project. Thus, a formal notification needs to be sent to the Technical Coordinator (TC) should an end-user wish to uninstall it from their system. Following this, the TC will notify the corresponding partner, which, in turn, will be responsible for uninstalling the Aggregator component.

5.2.3.2 User manual

The user does not have direct communication with the HEIR Aggregator component. More specifically, the output of the Aggregator is made available to the user through HEIR's 1st Layer GUI, whereas the functionality and results of the Aggregator are accessible through the HEIR Observatory.

5.2.4 HEIR Observatory

5.2.4.1 Data Fusion Bus (DFB)

5.2.4.1.1 Installation guide

- The HEIR Observatory has been developed based on the ITML's product namely Data Fusion Bus (DFB) which is a real-time data platform. DFB consists of a stack of open source and custom-made components to support secure flows and offer high availability, scalability, and redundancy. It consists of the following components: Apache Kafka
- Apache Zookeeper
- DFB Admin: Java microservice for Apache Kafka administrative purposes
- Elastic Stack (Elasticsearch & Kibana)
- ES-Connector: Java microservice to store Apache Kafka topics to Elasticsearch.
- Grafana, JMX Exporter and Prometheus (Monitoring Stack for Apache Kafka)
- Web Server: NGINX

5.2.4.1.1.1 Prerequisites

Observatory DFB runs on top of Kubernetes.

Git, Docker, and a running Kubernetes cluster are required.

5.2.4.1.1.2 Pre-installation Tasks

DFB-admin and ES-Connector images should be built and present at the running Kubernetes Node.

5.2.4.1.1.3 Installation procedure

A shell script (*install.sh*) is implemented for the install procedure on a Kubernetes Cluster. Before running the shell script, a *variables.properties* file is provided with template values that need to be updated in order to match running environment.

5.2.4.1.1.4 Post-installation

DFB is deployed at its own Kubernetes namespace. After installation of DFB, Kubernetes standard commands can be executed, in order to make sure all components are running e.g. *kubectl -n dfb get pods*

5.2.4.1.1.5 Uninstall or Roll-back procedure

A shell script (*uninstall.sh*) is implemented to completely uninstall all DFB components from the Kubernetes cluster.

5.2.4.2 2nd Layer of Visualizations

5.2.4.2.1 Installation guide

The HEIR's 2nd layer of visualizations is one of the main components in HEIR Observatory and includes all the elements and methods to present information gathered by the HEIR Observatory means. Such information refers to the Global RAMA Score, relevant metadata, statistics and recommendations that are produced based on the local RAMA, but also to active data policy rules that were extracted from the Privacy Aware Framework of HEIR. Part of this information is transferred to Observatory through the local HEIR Aggregators that are located at hospital's premisses. The corresponding results are available through the visualisation dashboard.

Users accessing the HEIR Observatory have read-only access to data collected from the HEIR Clients. In order to support the addition of functionality that offers access to non-public information, authentication mechanisms and a login page has been developed.

The 2nd layer of visualizations has been developed and offered in a docker-compose setup. It consists of:

a. Front-end module: Angular-based framework
b. Back-end & Middleware: NodeJS
c. Web Server: ngnix

The front-end module refers to the main GUI of Observatory, while the back-end & middleware module provides all the mechanisms and techniques to query and communicate data between HEIR databases, sub-modules & clients, to generate analytics and statistics based on the derived data, to handle user-access requests and finally to compose the final data schemas that the front-end module requires.

5.2.4.2.1.1 Prerequisites

In order to install and deploy the 2nd layer of visualizations, you need to have installed the docker & docker compose on your host machine.

Refer to the official docker compose reference site, in order to examine the docker-compose available commands along with detailed descriptions and examples - link)

Internet access is also required.

5.2.4.2.1.2 Pre-installation Tasks

Ensure that the HEIR's database (ElasticSearch) is accessible through a specific URI.

5.2.4.2.1.3 Installation procedure

In order to install the 2nd layer of visualization, you need to acquire the necessary configuration files, that refer to one (1) docker-compose.yaml file and two (2) config files for the web-server (nginx).

In the docker-compose.yaml file, you need to input specific environment variables related to: a) your preferred exposed URL of the platform, b) available ports of your host machine that the docker images can use to communicate, c) the URI that the HEIR Observatory's database is exposed. (ElasticSearch)

Given information about the exposed URLs & Ports needs to be filled in the nginx config files as well.

(Placeholders are available in the files, in order to point the exact lines where user-input is required)

Next, you need to transfer these files into the desired directory of your host machine. Then, open a terminal, navigate inside the specified directory and type the " $ docker-compose up -d " command to start the installation. Note that necessary images (front-end, back-end & middleware, and web-server) will be automatically downloaded from HEIR's docker hub account.

5.2.4.2.1.4 Update procedure

If you get notified from the HEIR's admin team that updates are available, you'll need to stop the containers ($ docker-compose down), update the docker-compose.yaml's images text fields with the newer versions and re-enter the “ $ docker-compose up -d “ command. The updates will be automatically incorporated.

Note that clearing the cache of the dockers and the internet’s browsers of your machine, is strongly suggested after any update action.

5.2.4.2.1.5 Uninstall or Roll-back procedure

You can always rollback to previous versions by updating the docker-compose.yaml file’s images text fields accordingly and repeat the previous step. (e.g rolling back 1 version the server: image:serverv02 à image:server01).

If you want to remove the containers you need to stop the containers and then remove them. (docker-compose down , docker-compose rm).

Note that clearing the cache of the dockers and the internet’s browsers of your machine, is strongly suggested after any update action.

5.2.4.2.1.6 Technical Tests

Adequate testing prior and after release to production is necessary to check not only functional aspects of the delivery but also not-functional ones.

Some tests might need to be performed before or after deployment of the software component, especially in cases when direct integration with other components is required.

5.2.4.2.2 User manual

5.2.4.2.2.1 Introduction

This User Manual (UM) provides the information necessary for users of the HEIR Observatory’s 2nd layer of Visualizations to effectively utilize its functionalities and capabilities. The HEIR Observatory is a comprehensive web application designed to collect, analyze, and present the results of deployed HEIR Clients in healthcare environments. It serves as a central repository for storing and processing cybersecurity data from participating hospitals, enabling global insights into the level of security in healthcare environments. The manual is designed to help non-technical users effectively utilize these components of the HEIR system.

The HEIR Observatory is a cloud-based framework that consists of a set of components that offer data storage, processing and communication capabilities, as well as advanced visualizations. Its primary purpose is to provide global insights into the level of security in healthcare sector, enabling end-users to access global risk level indicators (e.g., Global RAMA scores), identified risks, vulnerabilities and enabled data policies.

The HEIR’s 2nd layer of visualizations, which is the end-users access point to Observatory, presents the results in an intuitive and user-friendly format, including charts, graphs, and advanced interactive widgets that provide historical and aggregated insights, such as the top vulnerabilities by severity and frequency. It follows a client-server architecture, where users access the application through a web-based interface. This interface provides a graphical user interface (GUI) that allows users to navigate and interact with the system efficiently. The web-based nature of the application ensures easy access from various devices and platforms with internet connection, promoting widespread usability. Additionally, a back-end component developed that enables analytics generation capabilities , seamless communication and dependency management within the HEIR environment.

The framework is protected via an identity management mechanism, requiring users to register and log into the system. Following a role-based access, navigation options are available only to authorized users.

5.2.4.2.2.2 Getting Started

1. Access the System

a. Request from the system administrator to create a valid user-account based on your role. (e.g., regulator, researcher etc.)

b. Open a web browser and enter the URL provided by your system administrator. c. The login page will be displayed. Enter your credentials and click "Login" to access the system (Figure 28).

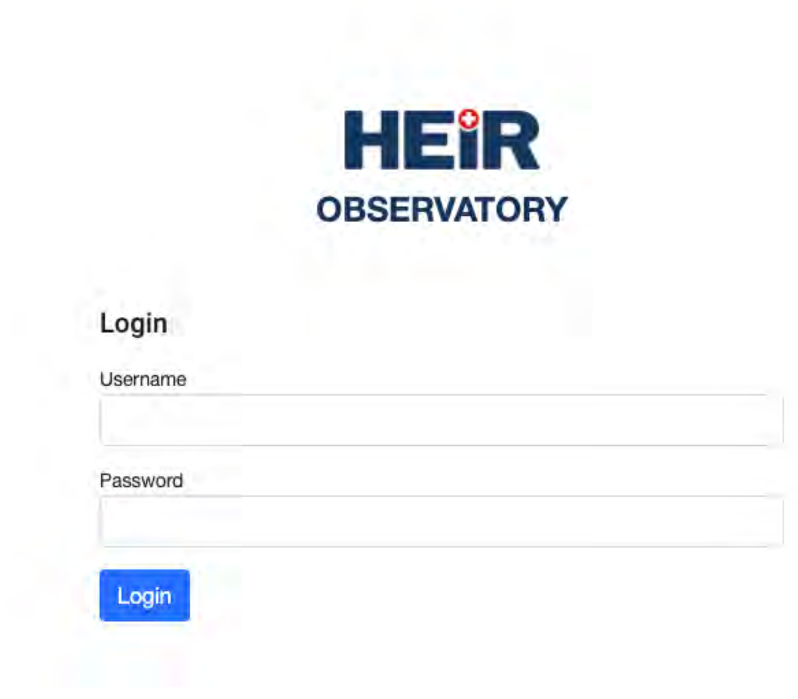


Figure 28: HEIR Observatory Login Page

2. Dashboard Overview
   Once logged in, you will be directed to the main dashboard, which provides an overview of the system's key information and insights. a. At the top of the page, you will find a summary of the number of connected hospitals, average critical events, and most frequent security reported levels (Figure 29).

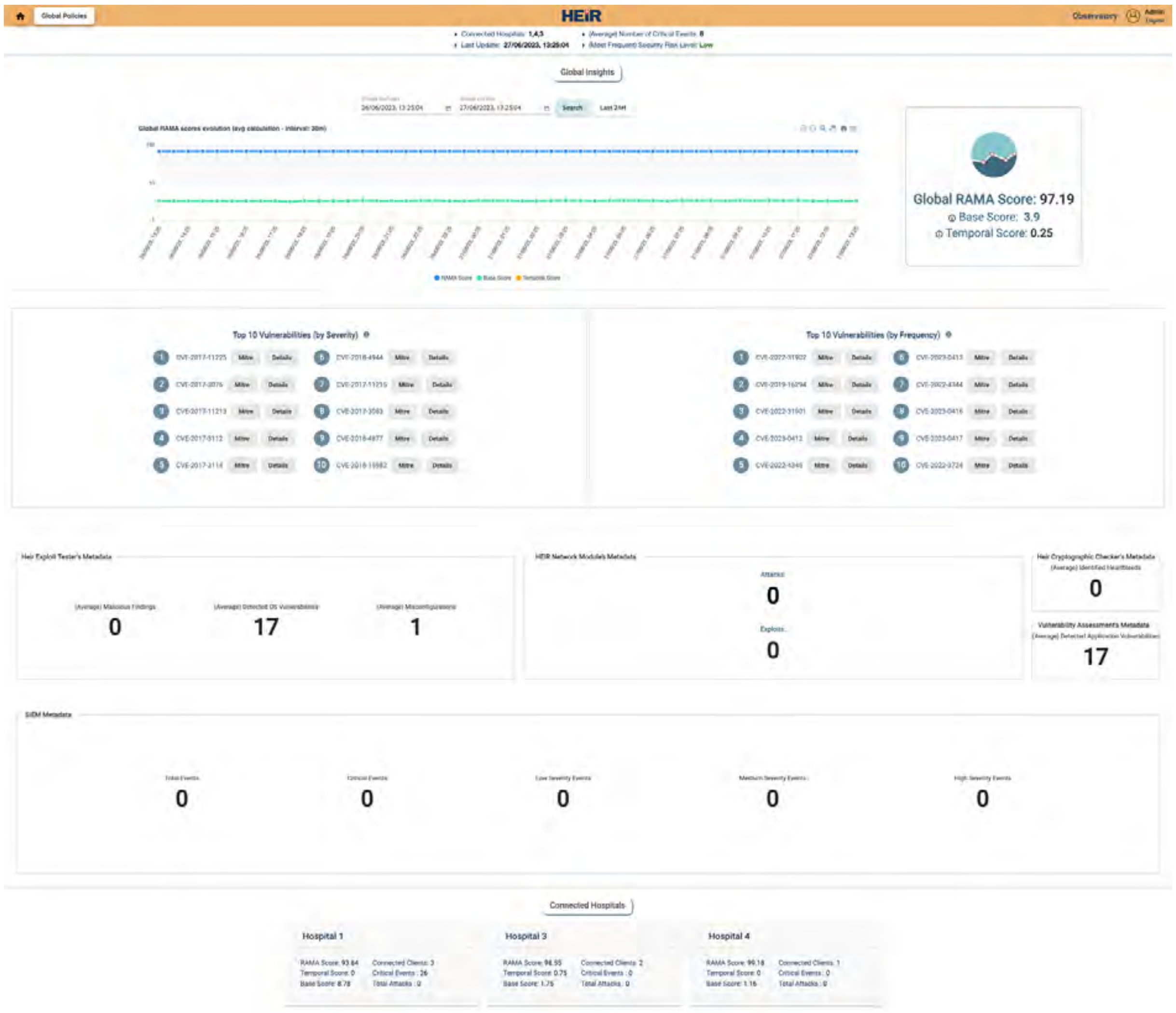
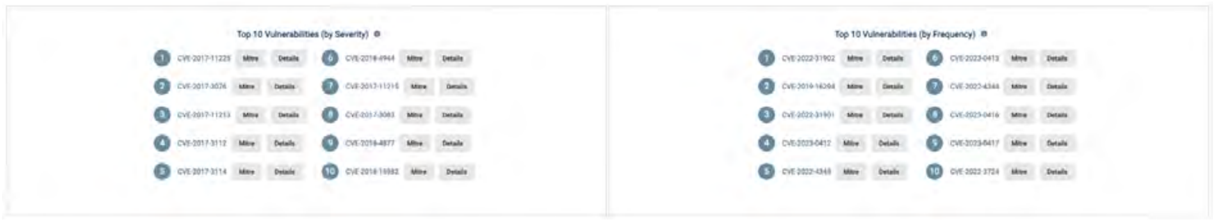


Figure 29: HEIR Observatory Main Dashboard

3. Global Insights
   The Global insights section displays the main results of the HEIR Observatory, including the Global RAMA Scores and relevant metadata. a. A line chart shows the evolution of Global RAMA Scores over time, where users can use the historical request functionality, as well as the zooming panning features to explore specific time periods (Figure 4).

4. Vulnerabilities
   The vulnerabilities section provides detailed information on the top 10 vulnerabilities by severity and frequency across all hospitals. Clicking on a specific vulnerability will provide additional details and navigation to global knowledge databases for further investigation (Figure 30).


Top 10 Vulnerabilities (by Severity)

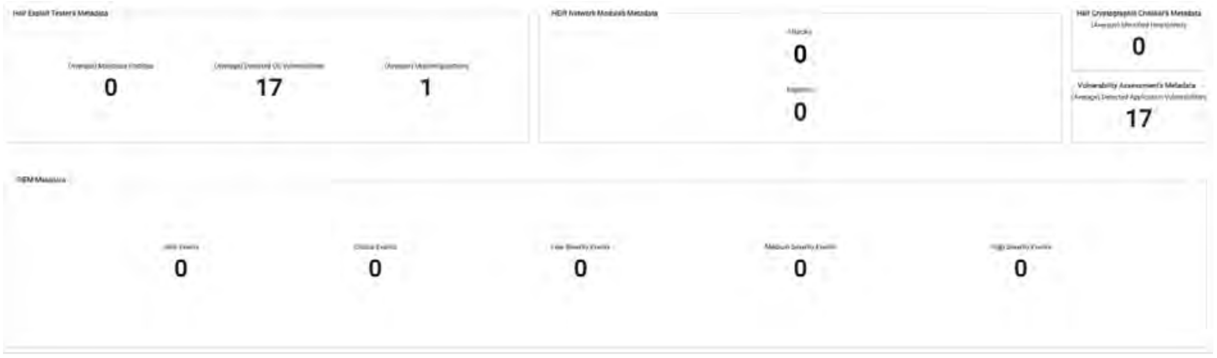
1 CVE-2017-11225 Mitre Details
2 CVE-2017-3076 Mitre Details
3 CVE-2017-11213 Mitre Details
4 CVE-2017-3112 Mitre Details
5 CVE-2017-3114 Mitre Details
6 CVE-2018-4944 Mitre Details
7 CVE-2017-11215 Mitre Details
8 CVE-2017-3083 Mitre Details
9 CVE-2018-4877 Mitre Details
10 CVE-2018-15982 Mitre Details

Top 10 Vulnerabilities (by Frequency)

1 CVE-2022-31902 Mitre Details
2 CVE-2019-16394 Mitre Details
3 CVE-2022-31901 Mitre Details
4 CVE-2023-0412 Mitre Details
5 CVE-2022-4345 Mitre Details
6 CVE-2023-0413 Mitre Details
7 CVE-2022-4344 Mitre Details
8 CVE-2023-0416 Mitre Details
9 CVE-2023-0417 Mitre Details
10 CVE-2022-3724 Mitre Details

Figure 30: HEIR Observatory Vulnerabilities

5. System Modules
   The system modules section displays generic statistics related to the embedded modules of the HEIR Clients in each hospital. Information is categorized by the source modules such as Exploit Tester's Metadata, Network Module's Metadata, Cryptographic Checker's Metadata, Vulnerability Assessment's Metadata, and SIEM Metadata (Figure 31).


HEIR Exploit Tester's Metadata
(Average) Malicious Profiles 0
(Average) Detected OS Vulnerabilities 17
(Average) Misconfigurations 1

HEIR Network Module's Metadata
Attacks 0
Isolations 0

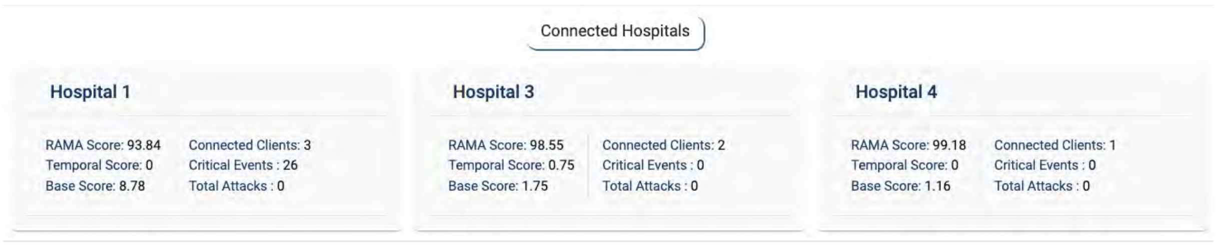
HEIR Cryptographic Checker's Metadata
(Average) Identified Insecurities 0

Vulnerability Assessment's Metadata
(Average) Detected Application Vulnerabilities 17

SIEM Metadata
Info Events 0
Critical Events 0
Low Severity Events 0
Medium Severity Events 0
High Severity Events 0

Figure 31: HEIR Observatory System Modules

6. Connected Hospitals
   At the bottom of the page, you can find a list of connected hospitals, presented in an anonymized manner. This allows you to monitor the Observatory's reach and impact without disclosing specific hospital identities, while it depicts the security status of the individual hospitals (Figure 32).


Connected Hospitals

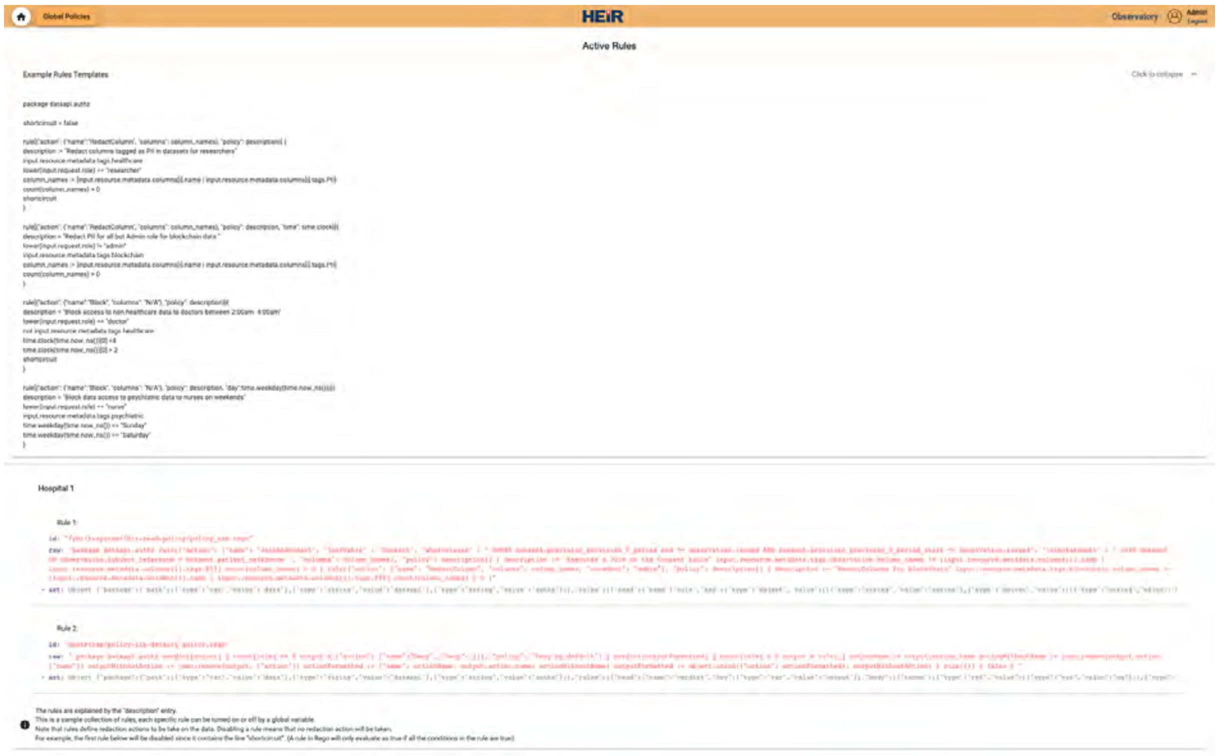
Hospital 1
RAMA Score: 93.84
Temporal Score: 0
Base Score: 8.78
Connected Clients: 3
Critical Events : 26
Total Attacks : 0

Hospital 3
RAMA Score: 98.55
Temporal Score: 0.75
Base Score: 1.75
Connected Clients: 2
Critical Events : 0
Total Attacks : 0

Hospital 4
RAMA Score: 99.18
Temporal Score: 0
Base Score: 1.16
Connected Clients: 1
Critical Events : 0
Total Attacks : 0

Figure 32: HEIR Observatory Connected Hospitals

7. Global Policies
   The Global Policies tab within the Observatory System is a dedicated section where authorized users can access and review the active data policy rules. Designed with user-friendliness in mind, the Global Policies tab presents these rules through easily understandable text and JSON visualized widgets. Users can navigate through the tab to observe the policies in place, accompanied by comprehensive descriptions and even sample data (Figure 33).
8. Logging Out
   To exit the HEIR Observatory, click on the "Logout" button located in the top menu bar.



Example Rules Templates

```
package datsapi.authz

shortcircuit = false

rule[{"action": {"name":"RedactColumn", "columns": column_names}, "policy": description}] {
description := "Redact columns tagged as PII in datasets for researchers"
input.resource.metadata.tags.healthcare
lower(input.request.role) == "researcher"
column_names := [input.resource.metadata.columns[i].name | input.resource.metadata.columns[i].tags.PII]
count(column_names) > 0
shortcircuit
}

rule[{"action": {"name":"RedactColumn", "columns": column_names}, "policy": description, "time": time.clock([
description = "Redact PII for all but Admin role for blockchain data"
lower(input.request.role) != "admin"
input.resource.metadata.tags.blockchain
column_names := [input.resource.metadata.columns[i].name | input.resource.metadata.columns[i].tags.PII]
count(column_names) > 0
}

rule[{"action": {"name":"Block", "columns": "N/A"}, "policy": description}]{
description = "Block access to non healthcare data to doctors between 2:00am 4:00am"
lower(input.request.role) == "doctor"
not input.resource.metadata.tags.healthcare
time.clock(time.now_ns())[0] <4
time.clock(time.now_ns())[0] > 2
shortcircuit
}

rule[{"action": {"name":"Block", "columns": "N/A"}, "policy": description, "day":time.weekday(time.now_ns())}]{
description = "Block data access to psychiatric data to nurses on weekends"
lower(input.request.role) == "nurse"
input.resource.metadata.tags.psychiatric
time.weekday(time.now_ns()) == "Sunday"
time.weekday(time.now_ns()) == "Saturday"
}
```

Hospital 1

Rule 1:

```
id: [illegible]
raw: [illegible]
ast: [illegible]
```

Rule 2:

```
id: [illegible]
raw: [illegible]
ast: [illegible]
```

The rules are explained by the "description" entry.
This is a sample collection of rules, each specific rule can be turned on or off by a global variable.
Note that rules define redaction actions to be take on the data. Disabling a rule means that no redaction action will be taken.
For example, the first rule below will be disabled since it contains the line "shortcircuit". (A rule in Rego will only evaluate as true if all the conditions in the rule are true).

Figure 33: HEIR Global Policies

5.2.4.2.2.2.1 Cautions & Warnings

Unauthorized Access Prohibited

The HEIR Observatory is a web portal intended for authorized use only. Any unauthorized access or attempts to breach the system's security are strictly prohibited and may result in legal action. Users are responsible for ensuring the confidentiality and integrity of their login credentials and must not share them with anyone.

5.2.4.2.2.2.2 Set-Up Considerations

To access the HEIR Observatory, you will need a computer or device with internet connectivity. There is no need for any specialized equipment or network configurations. Simply ensure that you have a working internet connection and a web browser installed.

5.2.4.2.2.2.3 User Access Considerations

The HEIR Observatory support role-based access. There are two main user roles: General Authorized Users and Authorized Policy Experts. Here's a brief description of each role and the corresponding restrictions placed on system accessibility:

1. General Authorized Users:
   General Authorized Users, like researchers or security experts, have access to the main page of the HEIR Observatory. They can explore the various visualizations, reports, and insights provided by the Observatory.
2. Authorized Policy Experts:
   Authorized Policy Experts have enhanced access privileges within the HEIR Observatory. They can access both the main dashboard screen and the Global Policies view. This expanded access allows policy experts to observe the active data policy rules across the connected organizations.

5.2.4.2.2.2.4 Accessing the System

In order to access the System, users need to request from the system administrator to create a corresponding account based on their role and needs. The admin will provide the necessary credentials to access the platform. In order, to change or retrieve missing credentials, users need to contact the system admin.

5.2.4.2.2.2.5 System Organization & Navigation

The HEIR’s 2nd layer of visualizations has been developed as to provide users with easy access to its main functions and features. The system's main organization is implemented through the menu panel and a home page button that serves as the main hub for navigation. Below, you will find a short descriptions of the navigation options.

1. Login Page:
   Upon accessing the system, users will be directed to the login page. Here, users are required to enter their unique user ID and password to gain access to the system. Once successfully logged in, users will be navigated to the main dashboard.
2. Home Page / Dashboard
   The home page serves as the main dashboard, containing the Observatory’s main results. It provides a comprehensive view of the system's data, statistics, and relevant information. It offers graphical representations and visualizations to help users understand and analyze the collected data.
3. Global Policies
   The Global Policies screen allows authorized policy experts to observe and review the currently active data policy rules across the HEIR’s connected hospitals. Users with relevant permissions can access this screen by selecting the " Global Policies" option from the main menu.

5.2.4.2.2.2.6 Exiting the System

To ensure proper exit from the HEIR Observatory system, follow the steps outlined below:

1. Logout: When you are ready to exit the system, locate the "Logout" option, found in the top-right corner of the screen. Click on the "Logout" option to initiate the logout process.
2. Session Termination: Once the logout process is finished, the system will terminate your current session. This ensures that your user account is securely logged out, preventing unauthorized access to your account and protecting your sensitive information.
3. Close the Browser: After successfully logging out, it is recommended to close the web browser entirely. This step ensures that no residual information or cached data related to your session remains accessible.

5.2.4.2.2.3 Using the System

5.2.4.2.2.3.1 Interacting with RAMA evolution chart

The following steps outline how users can interact with the widget effectively:

1. Select a specific date-time period from the calendar provided. Upon selection, the chart will update accordingly, displaying the relevant information within the chosen interval. This feature enables users to analyze the RAMA evolution over specific time frames.
2. The control panel, located at the top-right corner of the chart, offers several useful functionalities. Users can utilize this panel to perform various actions such as downloading the chart in different formats like SVG, PNG, or CSV. Additionally, users have the ability to zoom in and out of the chart, facilitating a closer examination of specific data points. Selection zoom allows users to focus on specific areas of interest, while panning enables seamless navigation across the chart. Furthermore, the control panel provides a reset zoom option, allowing users to revert to the default view.
3. For a more detailed analysis, users can leverage their mouse to interact directly with the chart. By holding down the left click and selecting a specific period, users can zoom in on their chosen interval, enabling a closer examination of the data. Moreover, hovering the mouse over the dots representing the RAMA scores provides additional information about each score.

5.2.4.2.2.3.2 Vulnerabilities buttons

Within the Observatory, users will discover two distinct lists that offer valuable insights into the top 10 vulnerabilities reported across all hospitals. These lists are categorized based on severity and frequency, providing users with a comprehensive understanding of the security landscape. For security experts seeking in-depth information about the displayed vulnerabilities, side buttons are available. By clicking on these buttons, users will be seamlessly redirected to globally recognized knowledge databases, such as MITRE.

5.2.5 HEIR SIEM

5.2.5.1 Installation guide

The HEIR SIEM has been developed based on the ITML's product namely Security Infusion (SI) which is an all-in-one solution, leveraging a plethora of state-of-the-art technologies and tools.

SI is an application that provides, in one single solution, basic features of SIEM, Monitoring and IDS systems, constituting a Control Baseline of ICT operations, with integrated risk mitigation and regulatory compliance capabilities.

Data form the underlying ICT infrastructure related with performance, services, network, and computing events, are monitored, collected, and analyzed in real time, with the capability of further storage for purposes of incident investigation and forensic analysis.

It consists of the following components:

- Apache Kafka
- Apache Zookeeper
- DFB Admin: Java microservice for Apache Kafka administrative purposes
- Elastic Stack (Elasticsearch & Kibana)
- ES-Connector: Java microservice to store Apache Kafka topics to Elasticsearch.
- Web Server: NGINIX
- Frontend Application: Laravel
- Wazuh & Filebeat
- Nagios
- SI agents

5.2.5.1.1 Prerequisites

Security Infusion runs on top of Kubernetes.

Prerequisites for the Security Infusion setup:

- Git
- Ansible
- Python
- Docker
- A running Kubernetes Cluster

5.2.5.1.2 Pre-installation Tasks

Security Infusion docker images should be present at the running Kubernetes node.

5.2.5.1.3 Installation procedure

A Python script (*templates.py*) is implemented to modify Security Infusion Kubernetes manifests to match the working environment using a text file (*csm-configuration.txt*) that contains all the required values for the Templates substitution which needs to be modified before running the python script.

The installation process is done using an implemented Ansible playbook (*playbook.yml*).

5.2.5.1.4 Post-installation

SI is deployed at its own Kubernetes namespace. After installation of SI, Kubernetes standard commands can be executed, in order to make sure all components are running e.g. *kubectl -n infusion get pods.*

5.2.5.1.5 Uninstall or Roll-back procedure

An Ansible script (*uninstall-playbook.yml*) is implemented to completely uninstall all SI components from the Kubernetes cluster.

6. Conclusion

The overall objective of this deliverable is to report all the commercialization activities of HEIR project by releasing a stable and reliable solution for the health-care industry and in turn for any end-to-end environment facing similar challenges. For this reason, the best practices for maintaining the framework are discussed in order to develop a plan tailored to HEIR needs and requirements. The maintenance plan adopts IEEE's framework for sequential maintenance activities with the following main objectives of this plan:

- Minimize system downtime by promptly addressing software issues and applying necessary fixes.
- Enhance system performance through performance optimization and tuning.
- Ensure the software remains compatible with evolving technology platforms and environments.
- Implement security updates and patches to protect against vulnerabilities and cyber threats.
- Continuously improve the software by incorporating user feedback and implementing new features.
- Adhere to regulatory and compliance requirements specific to the software domain.

Among the best practices identified that set the framework for the maintenance plan are its alignment with business goals, regular updates, upgrades and patches, effective maintenance role assignments and processes, proactive monitoring, quality assurance, documentation, version control and management, bug tracking and defect management, risk management, data retention and backup and security compliance.

Moreover, an overall report around the investigated usefulness of HEIR solution is presented based on the work conducted in terms of T6.3. The scope of this report is to verify and provide evidence regarding the usefulness of HEIR solution. The results, based on standardized (UEQ) or customized questionnaires and collected by employees that were and were not familiar with the solution as well as by IT and non-IT experts during HEIR training days, depict a promising picture regarding the acceptability, usefulness and usability of HEIR solution. All legal, ethical and privacy aspects, as reported on D7.5 and D7.6, are discussed in relation to real-world scenarios of the framework's utilization.

Finally, a deployment/installation guide and a user manual (if applicable) are collected and presented per component of HEIR framework to facilitate its deployment in real-world scenarios. The guides and the manuals are formatted in such a way that promote the applicability of the framework in any environment.