D2.1

The HEIR facilitators package: MVP

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

Deliverable type	Demonstrator
Deliverable reference no.	D2.1
Workpackage	WP2
Due date	**08-2021-M12**
Actual submission date	

Deliverable lead	BD
Editors	Ovidiu Mihăilă (BD)
Contributors	Dragoș Gavriluț, Bogdan Prelipcean (BD), Konstantinos Lampropoulos, Eftychia Lakka (FORTH), Leonidas Kallipolitis (AEGIS), Apostolis Zarras (TUD), Eliot Salant, Shlomit Koyfman (IBM)
Reviewers	Eftychia Lakka (FORTH), Eliot Salant (IBM)
Dissemination level	PU
Revision	FINAL / 1.0
Keywords	Facilitators, HEIR Agent, Vulnerability Assessment, SIEM, Threat Detection, Privacy-Aware Framework

Abstract

Deliverable D2.1 serves as the report which describes the initial version of the HEIR facilitators package MVP. The content is illustrating the development of the demonstrator based on the activities carried out within the context of WP2 and the relations with WP3 and WP4.

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

Executive Summary

The current deliverable presents the work that has been carried out towards the delivery of the HEIR facilitators package Minimum Viable Product - MVP. The development of the MVP demonstrates the effective integration of the facilitators' components into a simple, yet substantially integrated prototype with minimum functionality that showcases the potential of the proposed solution.

The HEIR facilitators package for the MVP includes (i) the vulnerability assessment module; (ii) the HEIR Agent: toolset used for submitting the information to the HEIR Client.

The demonstrator will be presented in a short report. The MVP will serve as the basis for further developments and will drive the implementation toward the release of complete prototypes (M18-M30). MVP release will act as proof-of-concept demonstrator.

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

Section 1 introduces the deliverable, Section 2 presents the MVP definition, the MVP development, and the subsequent steps towards the establishment of fully functional prototypes, Section 3 describes the HEIR MVP architecture and provides deployment details of the MVP infrastructure. The section also describes the integration actions and the communication mechanism used to connect the components. Section 4 presents the use case scenarios. Section 5 highlights the overall conclusions and plans.

Table of Contents

List of figures

1. Introduction

1.1 Scope and objectives

This document reports the work carried out to deliver the first inputs and guideline for the initial release of the HEIR facilitators package MVP. This work is accomplished in the context of the tasks T2.1 – T2.4 and it is the foundation for the further developments envisioned for M18 and M26.
The main takeaways of the document are enumerated below:

- Explanation of the components of the demonstrator, clearing what are the flows and relation. The explanation is made based on both functional and technical point of view.
- Matching of the use-cases security requirements and how HEIR solution covers it.

D2.1 (due M12, August 2021) will document the first internal design of the Facilitators package with details of the architecture, delivery modes, and connectivity of all the related components, taking as a starting point the solutions features documented in this deliverable. Related deliverables D2.2 (The HEIR facilitators package: 1st complete version) and D2.3 (The HEIR facilitators package: Final complete version) will use the current deliverable as a starting point.

1.2 Relation to other Tasks and Work Packages

This deliverable is related to all the Tasks and deliverables of WP2. Moreover, there is a close interrelation between this deliverable and the WP3 deliverables.

More specifically, this deliverable is strongly connected to (a) “D3.1 - The HEIR 1st layer of services package for the MVP” as the 1st layer of services use the results computed by the facilitators package. Furthermore, they share the same architecture for the MVP as them both are deployed locally for the MVP scenario.

2. MVP Definition

2.1 Methodology

MVP stands for Minimum Viable Product, and it is a version of the product where only the main features are completed. The purpose of the MVP is to present the basic components to early customers or end-users so that, with minimal effort, the development teams can collect the maximum amount of feedback. In addition, an MVP plays the role of ensuring that the features of the project are feasible and are ready for testing by the end-users. The feedback provided by the users can give insights regarding the customer's interests and is used as a guide by the developers for corrections, future developments, and improvements.

2.2 HEIR Facilitators MVP

The HEIR Facilitators MVP includes the basic functional components of the HEIR facilitators package system. It is based on specific predefined pilot scenarios to showcase the abilities of the overall system and ensure the functionalities of the proposed components.

For the HEIR Facilitators package MVP, data is collected at the endpoint level from PAGNI's environment. The main module that provides data is the Vulnerability Assessment module. The information collected and processed by the module is the operating system configurations and installed application information.

The results regarding misconfigurations and application vulnerabilities are forwarded to the HEIR core framework, more specific to the HEIR specific Client. Following this, the HEIR Client, a multi-modular application, conducts a set of tests and performs Risk Assessment for Medical Applications (RAMA), which is a score that depicts the security status of the hospital.

The results are then forwarded to the HEIR observatory where they get analysed and used to calculate the security status of the medical environment.

Besides providing the security information, the HEIR Facilitators package is aimed to be privacy-aware when processing healthcare sensitive data. To emphasize this, this deliverable also contains the status of the HEIR Collaborative privacy-aware framework as a complementary backbone for processing sensitive data for all stakeholders within the healthcare ecosystem.

3. HEIR Facilitators package MVP architecture

3.1 Overview

This subsection presents an overview of the HEIR Facilitators MVP Architecture. Figure 1 illustrates a high-level overview of the MVP architecture and the different components within the facilitators package. More specifically, the components are (a) Vulnerability Assessment and (b) HEIR Agent.

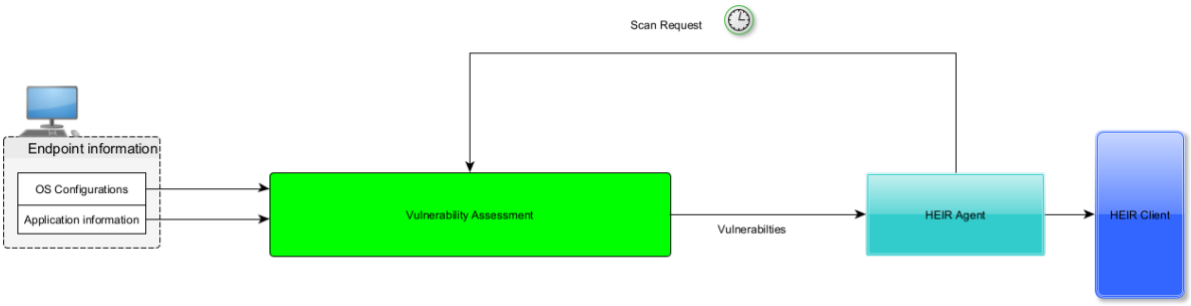

Figure 1: HEIR Facilitators package MVP Architecture

3.2 Vulnerability Assessment Module

The Dynamic Vulnerability Assessment and Monitoring module (also denoted as the Vulnerability Assessment module) extracts information about the operating system configurations and application information. In case that these points of interest are not properly configured, or the applications are outdated they can pose a security risk for the endpoint and after that to the entire medical environment. In Figure 2, we have an overview of the data flow

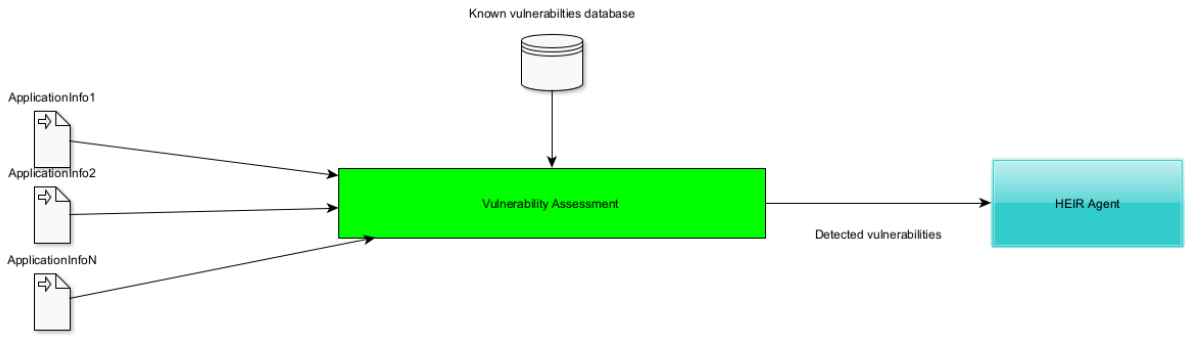

Figure 2: Vulnerability Assessment Module - Data flow

The collected information is:

- Application Name
- Version

For the MVP the module contains a set of rules that asses the application information and if a potential vulnerability is detected the alert is then stored in the results. The full results from this module are collected by the Local Correlation Component forwarded to the HEIR Client (for the MVP the HEIR Client is also deployed locally). The component has a database with known CVE[1] that are matched against the existing information on the installed applications (name and version). The output of the module will contain:

- the CVE number

[1] Common Vulnerabilities and Exposures

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

An example of such output is as follows:


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


3.3 HEIR Agent

The HEIR Agent has the management role for the Facilitators package. It communicates and controls the integrated modules. For the Vulnerability Assessment it submits a scan request as scheduled task.

For the MVP the HEIR Agent shares the architecture and instance with the HEIR Client described in the deliverable D3.1.

Further this component will have the role to integrate the other modules from the WP2.

It will serve as a backbone for the Threat Hunting Module, more specific SIEM, Forensics and ML-based Anomaly Detection and Threat Classification modules as shown in Figure 3.

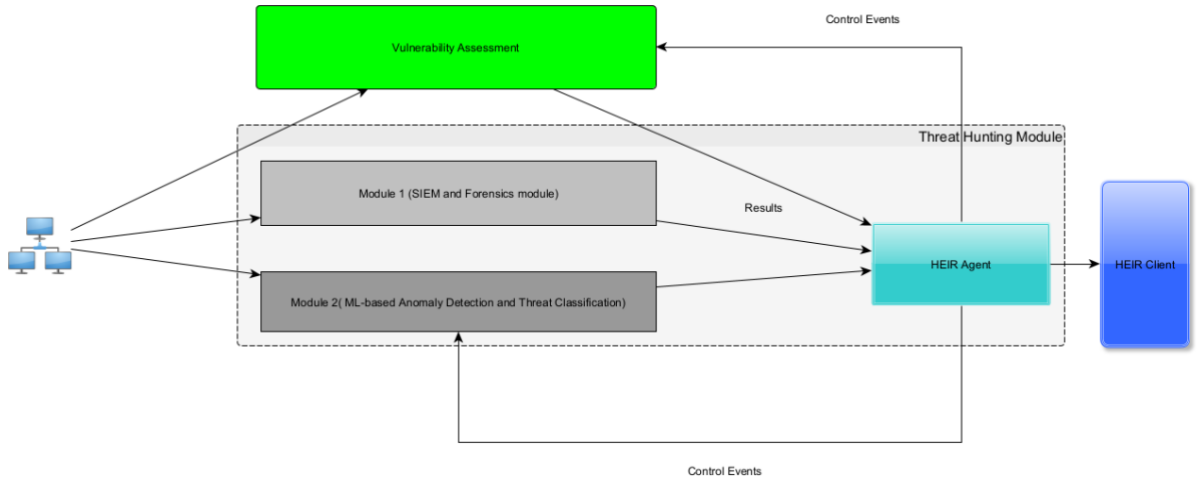

Figure 3: HEIR Agent – High Level Architecture

The HEIR Agent has an interface on which the other module must implement and the communication between modules (data and control) is event based. For the Vulnerability Assessment the scheduled monitoring task is activated by emitting an event. The results from the modules (i.e. Vulnerability Assessment) is also encapsulated in events.

This component also has the role to centralize and normalize the events that are further submitted the HEIR Client. For the MVP the submission is made directly inside the HEIR Agent.

3.4 The HEIR Interactive Forensics module

3.4.1 SIEM

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

The events reported by the Wazuh agents are the outcome of a wide range of tasks such as

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

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

[2] https://wazuh.com/

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

In addition to logs and events deriving from the operating system, Wazuh can collect and integrate logs deriving from network devices such as routers, firewalls etc. either by monitoring the log files themselves or via forwarding log messages in through Rsyslog[5]. This can potentially facilitate the collection of logs from medical devices that will need to be monitored within the hospital use-case environments.

The Wazuh event flow management is depicted in Figure 4.

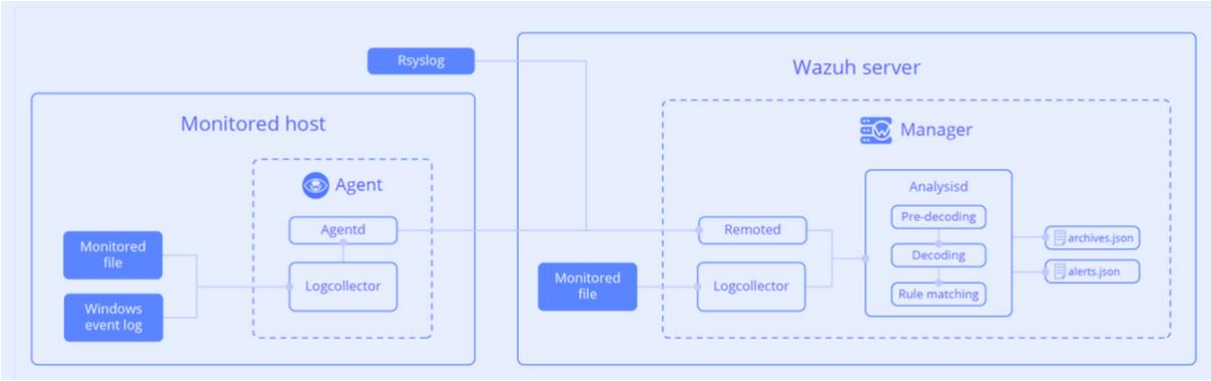

Figure 4: Wazuh event flow management

3.4.2 Forensics Visualization Toolkit

The Forensics Visualization Toolkit (FVT) provides users with a timeline-based representation of the security events captured by the SIEM sub-module. It is accessed through the 1st layer of visualisations and is meant to represent the captured events in a more detailed way. Authorized users who belong to the hospital staff and have access to the HEIR Client GUI (HCG) can further investigate any of the connected HEIR Clients of the hospital through the FVT.

Users accessing the FVT will firstly choose from the devices overview (Figure 4) the device they want to investigate.

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

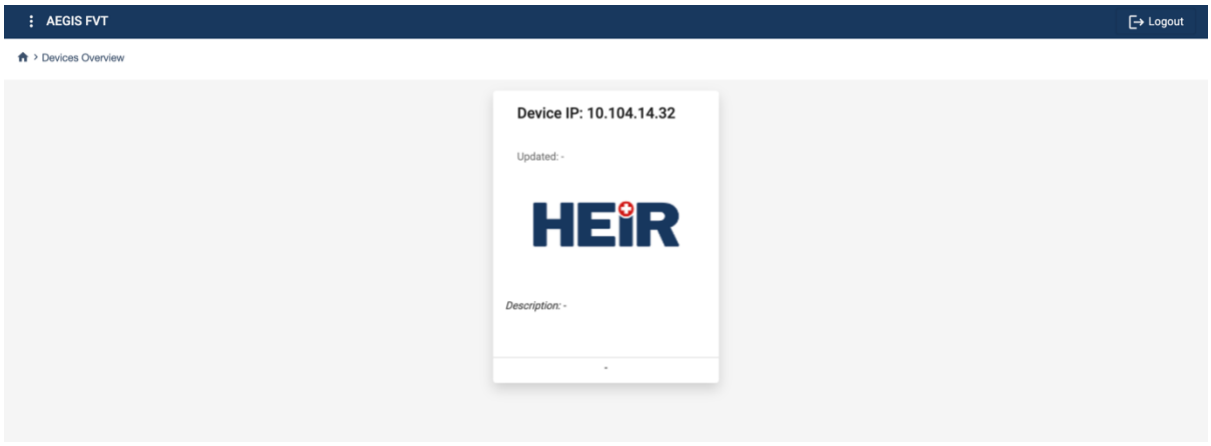

Figure 5: FVT - Device Selection

In the context of MVP, the FVT dashboard (Figure 5) allows users to choose from a set of widgets containing different types of visualizations, that refer to different system metrics and network related information. The widgets could be standalone and support discrete input sources, but their combination offers a comprehensive depiction and meaningful visualization of the data. The user will also be able to filter the incoming data by different meaningful parameters and to request historical data.

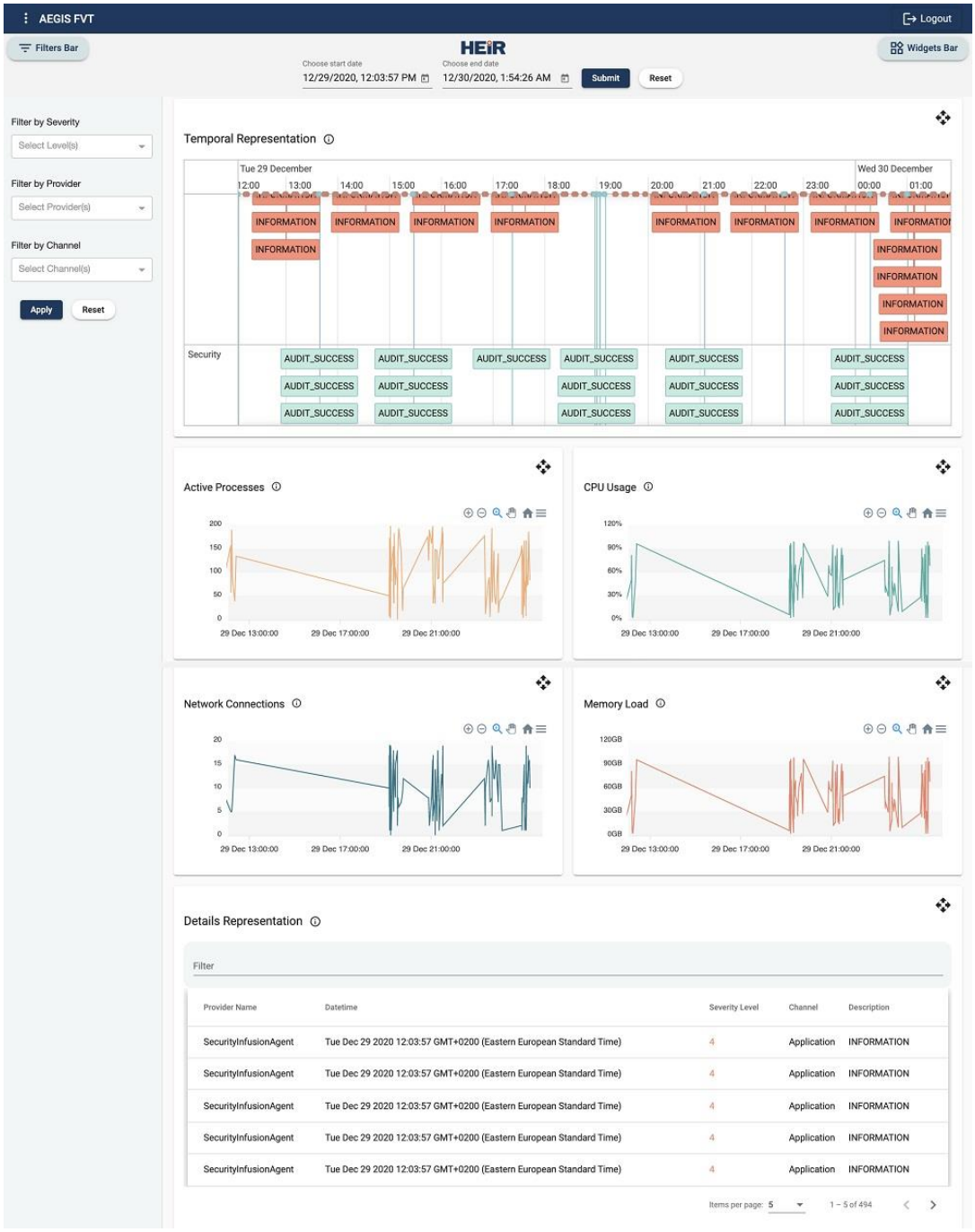

Figure 6: FVT - Dashboard

Temporal representation of incoming logged events captured by the SIEM will be available through the Timeline widget (Figure 7). By changing the range period in the timeline all the available widgets will be timewise synchronized and updated accordingly.

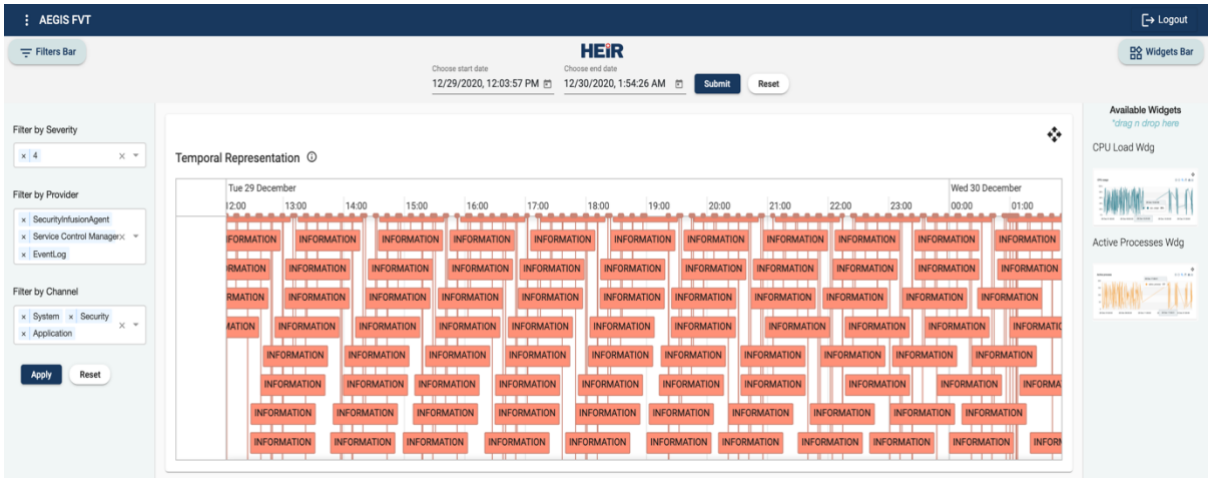

Figure 7: FVT - Timeline widget

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

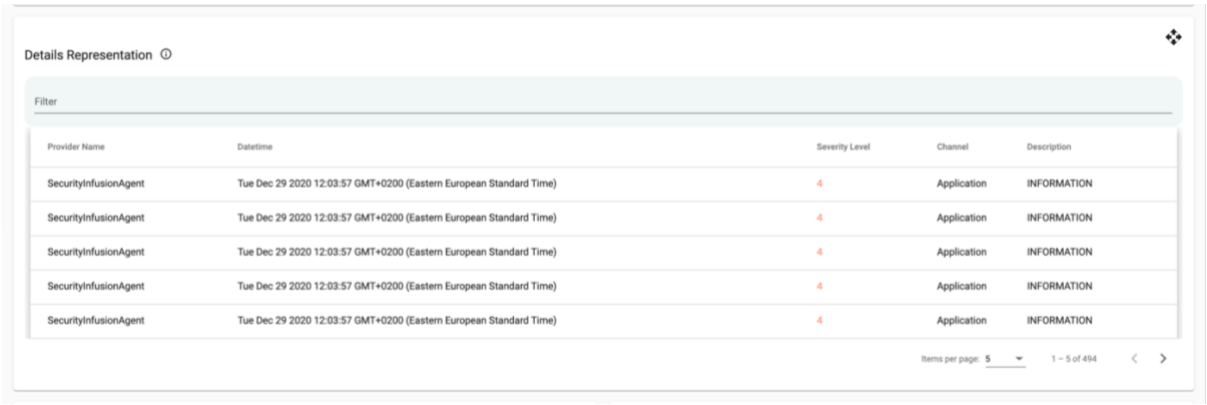

Figure 8: FVT - Details widget

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

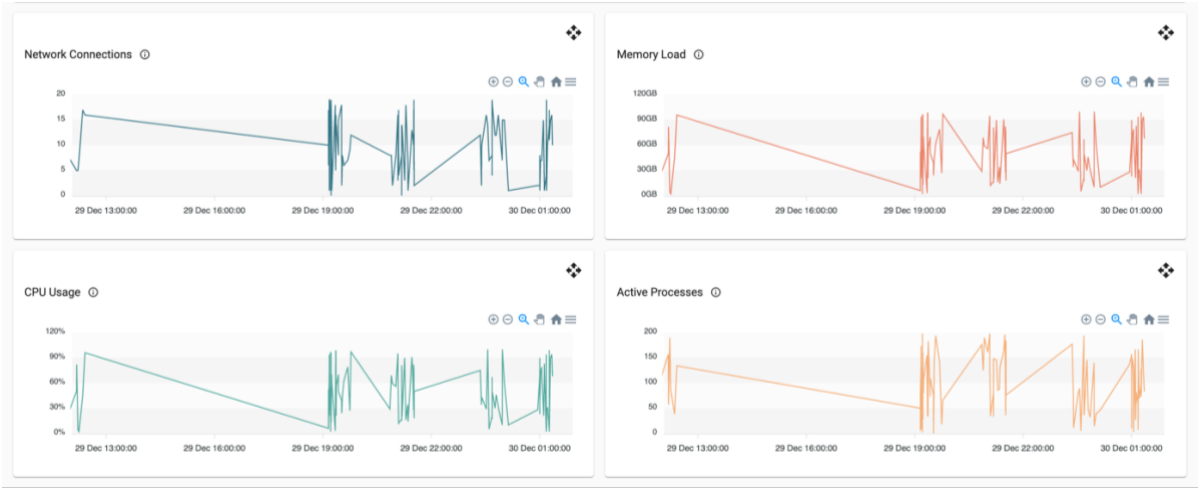

Figure 9: FVT - Line Chart Widget

Although events coming to FVT are only the ones produced by the SIEM for the MVP, FVT can support multiple sources. This feature is necessary for the next rounds of implementation in order to accommodate events generated by the rest of the HEIR facilitators, i.e. the ML anomaly detection and the Collaborative privacy-aware framework.

3.5 ML-based Anomaly Detection and Threat Classification Module

Anomaly detection is a step-in data mining that identifies data points, events, and observations that deviate from a dataset's normal behavior. The ML-based Anomaly Detection and Threat Classification module is responsible for deploying novel machine learning (ML) algorithms to provide efficient event and threat data classification. One of the most efficient ML algorithms for this task is the Random Forest Regression (RFR). RFR is a supervised learning algorithm that uses the ensemble learning method for regression. The ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. Figure 10 illustrates the structure of the Random Forest algorithm.

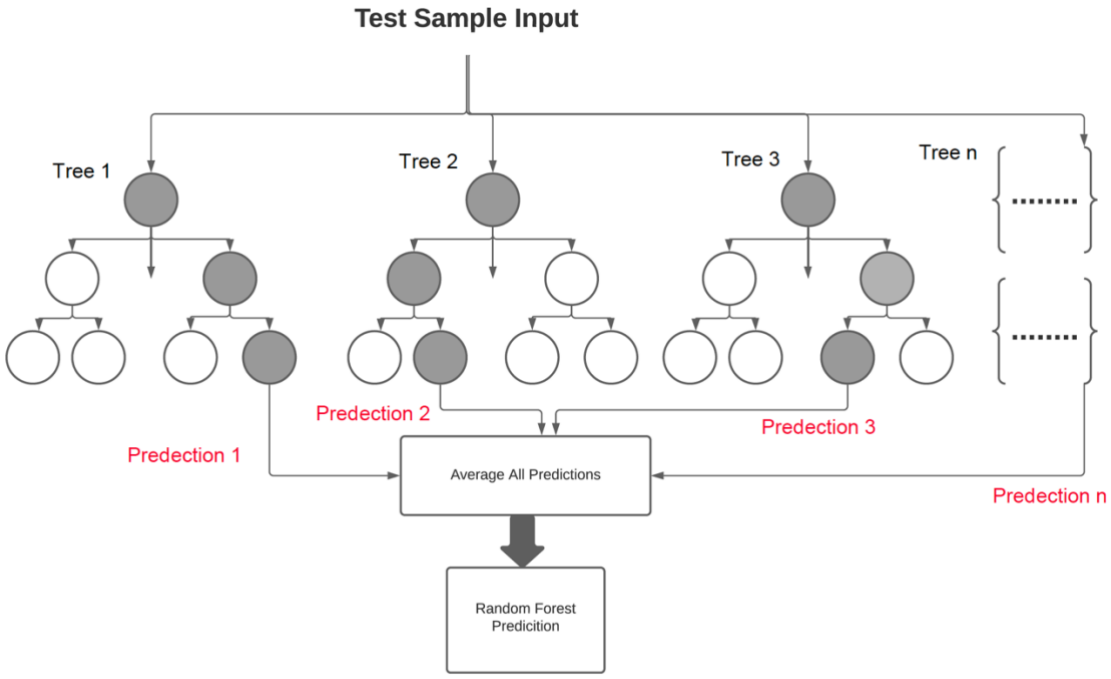

Figure 10: Random Forrest Algorithm

RFR works as follows:

1. Pick random k data points from a training set.
2. Build a decision tree associated with these *k* data points.
3. Choose the number *N* of trees you want to build and repeat steps 1 and 2.
4. For a new data point, make each of your *N*-tree trees predict the value of *y* for the data point in question and assign the new data point to the average across all predicted *y* values.

Random Forest Regression is a powerful and accurate model. It performs excellently on many problems, including features with non-linear relationships. Our anomaly detection framework implemented RFR as its first deployed ML algorithm. On top of that, it was trained on research data to prove its efficiency and fix any potential shortcomings. The next step involves the deployment of the framework in real-world medical data.

3.6 The HEIR Collaborative privacy-aware framework

3.6.1 Introduction

The goal of Task 2.4, "The *HEIR* Collaborative privacy-aware framework", is to create the technology for a dynamically composable data encapsulation which will provide security and privacy guarantees for a given workload. In particular, T2.4 aims to provide a secured workflow which will:

- Restrict access to the entry point of the workload (ingress) based on defined policies supported by token authentication for user identity.
- Restrict access of data in the workload based on defined security policies and token authentication.
- Provide a secure environment for component communication within a workflow, preventing both external attacks on components and accidental data leakage by components within a workflow.
- Allow for an easily configurable and modifiable solution.

To this end, IBM, the leader of this task, is both leveraging and contributing to the *Fybrik* framework (https://github.com/fybrik/) which is being contributed to the Open Source community and is expected to become a significant asset for IBM's cloud technologies. The results of HEIR research into secure data containers and use case requirements are being used to enhance development of Fybrik, while at the same time, T2.4 is incorporating technologies, architectural concepts and code developed in Fybrik into the HEIR solution as a facilitator.

T2.4 decided to build on top of leading, proven technologies, such as Docker containers and the Open Source Kubernetes (k8s) platform (http://kubernetes.io). Docker containers allow for creation of a portable execution environment. Kubernetes creates a secure and resilient environment for the deployment and management of application containers; however, neither of these technologies are directly concerned with data, data policies or role-based access. T2.4 extends these building blocks to cover the world of policy-driven data protection, incorporating such leading OpenSource frameworks as the power of the Open Policy Agent (OPA) framework (https://www.openpolicyagent.org/) to provide declarative policy support which tightly couples with the k8s environment and Istio (http://istio.io) which implements a *service mesh* on top of Kubernetes.

3.6.2 Fybrik

Fybrik is a cloud native platform to unify data access and governance, enabling business agility while securing enterprise data. By providing access and use of data only via a secured platform, Fybrik brings together access control and governance for data, greatly reducing risk of data loss or leakage.

Fybrik allows:

- **Data users** to use data in a self-service model without manual processes and without dealing with credentials. Fybrik uses common tools and frameworks for reading from and exporting data to data lakes or data warehouses to conform to latest industrial standards.
- **Data stewards** to control data access and data usage by applications. Use the organization's policy manager and data catalog of choice and lets Fybrik enforce data usage policies when data is requested, but before it is returned to the user.
- **Data operators** to automate data lifecycle management via implicit data copies, eliminating the need for manual versioning and copying of data.

The inputs to Fybrik are declarative definitions with a separation of aspects:

- Data stewards input definitions related to data governance and security.

- Data users input definitions related to the use of the data by the business logic of their applications.
- Data operators input definitions related to infrastructure and available resources.

Upon creation or change of any definition, Fybrik compiles together relevant inputs into a blueprint (or "Plotter" for a multi-cloud configuration) of the data path performing any policy-dictated transformations required on the requested data (per application) and transparently injecting access keys, such as to the cloud object store.

The blueprint augments the application workload and data sources with additional services and functions packed as pluggable modules. This creates a data path that:

- Integrates business logic with non-functional data centric requirements such as enabling data access regardless of its physical location, caching, lineage tracking, etc.
- Enforces governance on the usage of data, including limiting what data the business logic can access, performing transformations as needed, and controlling what the business logic can export and to where.

Fybrik is an open solution that can be extended to work with a wide range of tools and data stores and is both contributing to HEIR, as well as benefiting from HEIR development and use cases. As such, we expect Fybrik to be a major vehicle of exploitation for HEIR T2.4 results.

3.6.3 Design Considerations for a Privacy-aware Framework in HEIR

A central concept behind the design of HEIR's T2.4 Collaborative Privacy-aware Framework is to secure the data flow in user-supplied application code – which may include untrusted third-party sources, or coding errors that leak data - while supplying policy-driven control of the data being transferred from a data store to a Data Requestor. In particular, the aim of the HEIR, through Fybrik, is to provide an easily utilizable framework for data governance and workflow isolation; further protection can be achieved by the cloud provider supplying protection for data-in-process through hardware-enabled Trusted Execution Environments such as AMD SEV or Intel SGX. It is considered the responsibility of the workflow owner to protect data-at-rest, for example, using encryption.

The following points were taken as design goals for a privacy-aware framework:

- Data to be made accessible to a data user (requester) is regulated by a modifiable set of policies.
- For structured data, the governance of data should be fine-grained, i.e., at the attribute level.
- Governance policy needs to be evaluated at run-time.
- Industrial standards should be used wherever possible.
- Application owners wishing to secure the path between their application and data source should be able to do this with Fybrik without requiring expertise in Kubernetes.
- Prevent data from leaking out of the workflow defined in Fybrik.

4. Facilitators package MVP use case scenario

PAGNI's PANACEA is a patient management information system (bed management system), providing the health professional with the right information, when needed, in a way that can easily monitor the hospitalization incident, ensuring a "paperless" environment. At the same time, it enables the treating physician to have a complete picture of his/her patient, as he/she can gather information from all the hospitals of Crete. In more detail, PANACEA is a complete hospital electronic file, accessible from any computer system (PC, tablet, smartphone, etc.) PANACEA servers are located at the hospital's server room running.

For the MVP, we will deploy the facilitators package in PAGNI's local environment to identify potential issues. The Local RAMA Score will be calculated based on these findings.

The flow of the MVP scenarios is as follows:

- The Endpoint component for Vulnerability Assessment is deployed into the endpoints from the hospital infrastructure.
- A scheduled task will run the Vulnerability Assessment tool that will generate an event (report) application vulnerability assessment.
- The generated event will be normalized and aggregated by the HEIR Client (at the endpoint level in this point as it shares the instance with the HEIR Agent) and then emitted to the message broker (Kafka[6]).

[6] https://kafka.apache.org/

5. Conclusion

The current deliverable describes the implementation and rationale behind the HEIR facilitators package Minimum Viable Product - MVP. It analyses the MVP use case scenarios and explains how and why they were selected. The main objective of these use cases is to validate and test the HEIR facilitators package architecture and to initiate the integration between the components and 1st layer of services components. For each component, the work performed to support the operation of the MVP is being described. Lastly, we also present the communication methods and the integration elements that enabled the delivery of the MVP, based on the proposed use cases.

The next actions will focus on the further development and refinement of HEIR facilitators package. This will be based on the common activities within the technical work packages, as well as the feedback to be obtained from the MVP. The 1st completed version will be reported in D2.2 - "The HEIR facilitators package: 1st complete version".

6. Appendix A – Vulnerability Assessment Module - sample output


```
        {
            "application_name": "Docker Desktop",
            "cves": [
                {
                    "cve": "CVE-2018-10892",
                    "description": "The default OCI linux spec in oci/defaults{_linux}.go in Docker/Moby from 1.11 to current does not block /proc/acpi pathnames. The flaw allows an attacker to modify host's hardware like enabling/disabling bluetooth or turning up/down keyboard brightness.",
                    "publish_date": "2018-07-06T16:29Z",
                    "score": 50
                },
                {
                    "cve": "CVE-2020-11492",
                    "description": "An issue was discovered in Docker Desktop through 2.2.0.5 on Windows. If a local attacker sets up their own named pipe prior to starting Docker with the same name, this attacker can intercept a connection attempt from Docker Service (which runs as SYSTEM), and then impersonate their privileges.",
                    "publish_date": "2020-06-05T14:15Z",
                    "score": 72
                }
            ],
            "version": "2.2.0.4"
        },
        {
            "application_name": "git-scm git",
            "cves": [
                {
                    "cve": "CVE-2019-1387",
                    "description": "An issue was found in Git before v2.24.1, v2.23.1, v2.22.2, v2.21.1, v2.20.2, v2.19.3, v2.18.2, v2.17.3, v2.16.6, v2.15.4, and v2.14.6. Recursive clones are currently affected by a vulnerability that is caused by too-lax validation of submodule names, allowing very targeted attacks via remote code execution in recursive clones.",
                    "publish_date": "2019-12-18T21:15Z",
                    "score": 67
                },
                {
                    "cve": "CVE-2019-1353",
                    "description": "An issue was found in Git before v2.24.1, v2.23.1, v2.22.2, v2.21.1, v2.20.2, v2.19.3, v2.18.2, v2.17.3, v2.16.6, v2.15.4, and v2.14.6. When running Git in the Windows Subsystem for Linux
```

(also known as \\\\\\\\\\\\\\\"WSL\\\\\\\\\\\\\\\") while accessing a working directory on a regular Windows drive, none of the NTFS protections were active.",

"publish_date": "2020-01-24T22:15Z",

"score": 75

},

{

"cve": "CVE-2019-1348",

"description": "An issue was found in Git before v2.24.1, v2.23.1, v2.22.2, v2.21.1, v2.20.2, v2.19.3, v2.18.2, v2.17.3, v2.16.6, v2.15.4, and v2.14.6. The --export-marks option of git fast-import is exposed also via the in-stream command feature export-marks=... and it allows overwriting arbitrary paths.",

"publish_date": "2020-01-24T22:15Z",

"score": 36

},

{

"cve": "CVE-2019-19604",

"description": "Arbitrary command execution is possible in Git before 2.20.2, 2.21.x before 2.21.1, 2.22.x before 2.22.2, 2.23.x before 2.23.1, and 2.24.x before 2.24.1 because a \\\\\\\\\\\\\\\"git submodule update\\\\\\\\\\\\\\\" operation can run commands found in the .gitmodules file of a malicious repository.",

"publish_date": "2019-12-11T00:15Z",

"score": 93

},

{

"cve": "CVE-2020-11008",

"description": "Affected versions of Git have a vulnerability whereby Git can be tricked into sending private credentials to a host controlled by an attacker. This bug is similar to CVE-2020-5260(GHSA-qm7j-c969-7j4q). The fix for that bug still left the door open for an exploit where _some_ credential is leaked (but the attacker cannot control which one). Git uses external \\\\\\\\\\\\\\\"credential helper\\\\\\\\\\\\\\\" programs to store and retrieve passwords or other credentials from secure storage provided by the operating system. Specially-crafted URLs that are considered illegal as of the recently published Git versions can cause Git to send a \\\\\\\\\\\\\\\"blank\\\\\\\\\\\\\\\" pattern to helpers, missing hostname and protocol fields. Many helpers will interpret this as matching _any_ URL, and will return some unspecified stored password, leaking the password to an attacker's server. The vulnerability can be triggered by feeding a malicious URL to `git clone`. However, the affected URLs look rather suspicious; the likely vector would be through systems which automatically clone URLs not visible to the user, such as Git submodules, or package systems built around Git. The root of the problem is in Git itself, which should not be feeding blank input to helpers. However, the ability to exploit the vulnerability in practice depends on which helpers are in use. Credential helpers which are known to trigger the vulnerability: - Git's \\\\\\\\\\\\\\\"store\\\\\\\\\\\\\\\" helper - Git's \\\\\\\\\\\\\\\"cache\\\\\\\\\\\\\\\" helper - the \\\\\\\\\\\\\\\"osxkeychain\\\\\\\\\\\\\\\" helper that ships in Git's \\\\\\\\\\\\\\\"contrib\\\\\\\\\\\\\\\" directory Credential helpers which

are known to be safe even with vulnerable versions of Git: - Git Credential Manager for Windows Any helper not in this list should be assumed to trigger the vulnerability.",

"publish_date": "2020-04-21T19:15Z",

"score": 50

},

{

"cve": "CVE-2020-5260",

"description": "Affected versions of Git have a vulnerability whereby Git can be tricked into sending private credentials to a host controlled by an attacker. Git uses external \\\\\\\\\\\\\\\\"credential helper\\\\\\\\\\\\\\\\" programs to store and retrieve passwords or other credentials from secure storage provided by the operating system. Specially-crafted URLs that contain an encoded newline can inject unintended values into the credential helper protocol stream, causing the credential helper to retrieve the password for one server (e.g., good.example.com) for an HTTP request being made to another server (e.g., evil.example.com), resulting in credentials for the former being sent to the latter. There are no restrictions on the relationship between the two, meaning that an attacker can craft a URL that will present stored credentials for any host to a host of their choosing. The vulnerability can be triggered by feeding a malicious URL to git clone. However, the affected URLs look rather suspicious; the likely vector would be through systems which automatically clone URLs not visible to the user, such as Git submodules, or package systems built around Git. The problem has been patched in the versions published on April 14th, 2020, going back to v2.17.x. Anyone wishing to backport the change further can do so by applying commit 9a6bbee (the full release includes extra checks for git fsck, but that commit is sufficient to protect clients against the vulnerability). The patched versions are: 2.17.4, 2.18.3, 2.19.4, 2.20.3, 2.21.2, 2.22.3, 2.23.2, 2.24.2, 2.25.3, 2.26.1.",

"publish_date": "2020-04-14T23:15Z",

"score": 50

},

{

"cve": "CVE-2021-21300",

"description": "Git is an open-source distributed revision control system. In affected versions of Git a specially crafted repository that contains symbolic links as well as files using a clean/smudge filter such as Git LFS, may cause just-checked out script to be executed while cloning onto a case-insensitive file system such as NTFS, HFS+ or APFS (i.e. the default file systems on Windows and macOS). Note that clean/smudge filters have to be configured for that. Git for Windows configures Git LFS by default, and is therefore vulnerable. The problem has been patched in the versions published on Tuesday, March 9th, 2021. As a workaound, if symbolic link support is disabled in Git (e.g. via `git config --global core.symlinks false`), the described attack won't work. Likewise, if no clean/smudge filters such as Git LFS are configured globally (i.e. _before_ cloning), the attack is foiled. As always, it is best to avoid cloning repositories from untrusted sources. The earliest impacted version is 2.14.2. The fix versions are: 2.30.1, 2.29.3, 2.28.1, 2.27.1, 2.26.3, 2.25.5, 2.24.4, 2.23.4, 2.22.5, 2.21.4, 2.20.5, 2.19.6, 2.18.5, 2.17.62.17.6.",

"publish_date": "2021-03-09T20:15Z",

"score": 50

```
                }
            ],
            "version": "2.21.0"
        }
    ]
}
```