,

D5.3 Second HORSE Release: HORSE IT-2 version

Revision: v.1.0

Work package	WP 5
Task	T5.2, T5.3, T5.4
Due date	31/08/2025
Submission date	31/08/2025
Deliverable lead	UPC
Version	1.0
Authors	Eva Rodríguez (UPC), Xavi Masi-Bruin (UPC), Judit Cerdà (UPC), Fabrizio Granelli (CNIT), Alessandro Carrega (CNIT), Paul Paixão (EFACEC), Pedro Elísio (EFACEC), Stefanos Venios (Suite5), Thodoris Velmachos (Suite5), Andreas Petrou (Suite5), Panagiotis Gkonis (NKUA), Panagiotis Trakadas(NKUA), Nikolaos Nomikos (NKUA), Michail Danousis (8BELLS), Nikolaos Androulikakis (8BELLS), Alice Piemonti (MAR), Vito Cianchini (MAR), Emilio García de la Calera Molina (UMU), Anthony Joel Pogo Medina (UMU), Jose Manuel Manjón (TID), Iulisloi Zacarias (TUBS), Leesa Joyce (HOLO), George Xylouris (ZORTE), Alexandros Katsarakis (STS)
Reviewers	Alice Piemonti (MAR), Paulo Paixão (EFACEC), Pedro Elísio (EFACEC)

Abstract	This document presents the second release of the HORSE cybersecurity platform, developed during the IT-2 iteration and refined using insights gained from the IT-1 validation phase. It provides a comprehensive account of the development, integration, and testing activities that shaped this second and final release, detailing the methodologies applied to system integration and functional validation. The deliverable reports on the IT-2 validation results, highlighting the practical implementation of platform capabilities through demonstrators. Each demonstrator illustrates the

	validation of key components and interfaces, supported by integration test matrices that assess interactions and confirm operational consistency. Designed to ensure robustness and reliability, the demonstrators validate both individual module functionality and their seamless integration within the overall platform. It is worth noting that Demonstrators 9 and 10 showcase the functionality of the HORSE platform in the two real-world use cases considered in the project: Secure Smart LRT (Light Rail Transit) or Metro Operation Systems, and Remote Rendering to Power XR (Extended Reality) in industrial environments.
Keywords	Functional Validation; Integration; Cybersecurity; Mobile Networks

DOCUMENT REVISION HISTORY

Version	Date	Description of change	List of contributor(s)
V0.1	31/03/2025	1st version of the template for comments	Eva Rodríguez (UPC)
V0.2	07/04/2025	Revised version of the template for comments	Xavi Masip (UPC) Fabrizio Granelli (CNIT)
V0.3	14/04/2025	Final version of the template for comments	Eva Rodríguez (UPC)
V0.4	13/06/2025	Contributions to Section 2	All partners
V0.5	07/07/2025	Contributions to Section 3	All partners
V0.6	07/07/2025	Contributions to Section 4	All partners
V0.7	07/07/2025	Contribution to Section 5	All partners
V0.8	18/07/2025	Updates in the integration tables in Section 3	All partners
V0.9	18/07/2025	Updates in the testing tables in Section 4	All partners
V0.10	18/07/2025	Updates in Section 5	All partners
V0.11	21/07/2025	Contributions to section 1 and section 6	UPC
V0.12	28/07/2025	Version for internal review	UPC
V0.13	04/08/2025	Revision from MARTEL	Alice Piemonti (MAR)
V0.14	04/08/2025	Revision form EFACEC	Paulo Paixão (EFACEC), Pedro Elísio (EFACEC)
V0.15	26/08/2025	Version for QA	Eva Rodríguez (UPC), Xavi Masip (UPC)
V1.0	29/08/2025	Quality assessment and final version to be submitted.	Fabrizio Granelli (CNIT)

Disclaimer

Co-funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the other granting authorities. Neither the European Union nor the granting authority can be held responsible for them.

Copyright notice

Project co-funded by the European Commission in the Horizon Europe Programme		
Nature of the deliverable:	OTHER	
Dissemination Level		
PU	*Public, fully open, e.g. web*	**X**
SEN	*Sensitive, limited under the conditions of the Grant Agreement*	
Classified R-UE/ EU-R	*EU RESTRICTED under the Commission Decision No2015/ 444*	
Classified C-UE/ EU-C	*EU CONFIDENTIAL under the Commission Decision No2015/ 444*	
Classified S-UE/ EU-S	*EU SECRET under the Commission Decision No2015/ 444*	

* *R: Document, report (excluding the periodic and final reports)*
DEM: Demonstrator, pilot, prototype, plan designs
DEC: Websites, patents filing, press & media actions, videos, etc.
DATA: Data sets, microdata, etc
DMP: Data management plan
ETHICS: Deliverables related to ethics issues.
SECURITY: Deliverables related to security issues
OTHER: Software, technical diagram, algorithms, models, etc.

Executive summary

This document provides a comprehensive account of the second release of the HORSE cybersecurity platform, a robust and modular solution designed to enhance the security of 6G/5G networks. The second release marks a significant milestone in the project, delivering fully integrated software components that have been validated against the project's requirements and performance objectives.

This deliverable, D5.3, is intended to accompany the software release of the HORSE project components and serve as a practical guide for navigating the source code available on GitHub. For each software component, the document outlines its core functionalities, system requirements, dependencies, and a reference to the installation guidelines to support deployment and integration. The second release demonstrates the successful integration of all HORSE components into a cohesive architecture, confirming their ability to interoperate seamlessly. Validation activities show that the platform meets its key performance indicators. To provide tangible evidence of these achievements, ten dedicated demonstrators were developed, each designed to validate specific aspects of the HORSE framework. These demonstrators illustrate the platform's effectiveness in preventing, detecting, and mitigating cybersecurity threats while providing clear workflows, inputs, and outputs to support straightforward deployment.

In addition, the document provides the validation matrix elaborated to assess the interactions and performance of the integrated components. The integration matrix presents the final status of integration efforts and highlights the HORSE partners' ability to interconnect individual components within a modular and highly integrated architecture, as envisioned for next-generation network infrastructures. Section 3 is supported by the description of the ten demonstrators designed to validate the HORSE framework. Each demonstrator includes the workflow defined for the prevention, detection and mitigation of network threats, as well as a set of integration tests, along with the expected inputs and outputs, to support straightforward end-to-end component integration.

Finally, the document presents the HORSE compliance matrix, which maps the final HORSE requirements to the validation activities and test results. This ensures full traceability and demonstrates alignment between the project's goals and the verified outcomes of the second release, underscoring the consortium's success in delivering a secure, modular, and high-performing cybersecurity solution for future communication infrastructure.

Table of contents

List of figures

List of tables

Abbreviations

3D CAD	3 Dimensional Computer Aided Design
5G	Fifth Generation of Mobile Networks
6G	Sixth Generation of Mobile Networks
AI	Artificial Intelligence
API	Application Programming Interface
AR	Augmented Reality
cKB	Common Knowledge Base
CN	Core Network
DDoS	Distributed Denial of Service
DEME	Detector and Mitigation Engine
DN	Data Network
DNS	Domain Name System
DRL	Deep Reinforcement Learning
DT	Digital Twin
DTE	Distributed Trustable AI Engine
EM	Early Modeling
ES	Elastic Search
ePEM	End-to-End Secure Connectivity Manager
FAR	Forwarding Action Rule
gNB	Next Generation Node B
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IA-DT	Impact Analysis Digital Twin
IBI	Intent-based Interface
INFR	Infrastructure
IP	Internet Protocol
IT-X	Project Iteration X
JSON	JavaScript Object Notation
k8s	Kubernetes
LRT	Light Rail Transit
ML	Machine Learning

NDT	Network Digital Twin
NEF	Network Exposure Function
NF	Network Function
NTP	Network Time Protocol
P&P	Prediction and Prevention
P&P NDT	Prediction and Prevention Network Digital Twin
PAG	Policies and Data Governance
PCAP	Packet CAPture
PDU	Protocol Data Unit
PFCP	Packet Forwarding Control Protocol
PreProc	Pre-Processing
QoS	Quality of Service
RAN	Radio Access Network
REST	Representational State Transfer
RTR	Reliability, Trust and Resilience
SEID	Session Endpoint Identifier
SM	Smart Monitoring
SMF	Session Management Function
TEID	Tunnel Endpoint Identifier
TX.Y	Task X.Y
UE	User Equipment
UI	User Interface
UPF	User Plane Function
URL	Uniform Resource Locator
VM	Virtual Machine
WebUI	Web-based User Interface
WPX	Work Package X
XML	Extensible Markup Language
XR	Extended Reality
VR	Virtual Reality

1 Introduction

The HORSE project represents a unified effort to advance cybersecurity within the rapidly evolving 6G network landscape. This document, D5.3, presents the second release of the HORSE cybersecurity platform, a solution designed to improve the security of 5G/6G networks. It serves as a comprehensive companion to the final software release, offering a structured walkthrough of all platform components and their completed implementations. This report is intended for a broad audience of stakeholders, including developers, integrators, cybersecurity professionals, and policymakers.

The document summarizes the outcomes of Tasks T5.2, T5.3, and T5.4, and documents the successful integration of all developed components into a unified framework. It builds upon the work from the first HORSE release (D5.2) and follows the final architectural guidelines established in D2.4. The development aligns with the final requirements defined in D2.3 and reflects the integration of components reported in D3.2 and D4.2.

Additionally, this deliverable provides detailed software specifications and practical guidance for deployment, including installation procedures and dependency requirements. It revisits and consolidates the ten demonstrators, which are designed to validate the full operational scope of the platform and confirm its capacity to support comprehensive, modular, and scalable cybersecurity strategies. The final version of all software and tools is available on a dedicated GitHub repository: https://github.com/HORSE-EU-Project.

1.1 Purpose of the document

The primary objective of this deliverable is to serve as a comprehensive companion document to the second release of the HORSE platform, providing a complete and well-structured walkthrough of all platform components and their finalized implementations. Its scope is focused not only on describing the finalized software components, but more importantly on highlighting their integration and deployment within a unified cybersecurity framework. It is intended to guide stakeholders through the full suite of HORSE components, offering detailed descriptions of their functionalities, installation and configuration parameters procedures, and operational behaviour in realistic deployment scenarios.

This document represents the culmination of the consortium's collective efforts, marking the transition from early-stage development and iterative prototyping to a fully integrated cybersecurity solution. It incorporates the lessons learned throughout the implementation phase and outlines refinements made in response to evolving requirements, resulting in targeted architectural improvements. Central to this release is the validation of interoperability and seamless coordination among all components, achieved through a rigorous integration strategy executed under real-world conditions.

A key element of the document is the presentation of the ten demonstrators, which were developed as essential proof points for the HORSE framework. Each demonstrator is designed to validate specific aspects of the platform’s capabilities, ranging from prevention and detection to mitigation of security threats, while demonstrating the platform’s ability to address diverse challenges in mobile and next-generation networks. These demonstrators validate the full operational scope of the HORSE platform and confirm its capacity to support comprehensive, modular, and scalable cybersecurity strategies.

1.2 Relation to other HORSE deliverables and project work

This document summarizes the final design, development, and integration efforts undertaken for the HORSE platform components developed across work packages WP3, WP4, and WP5. Within the scope of WP5, it details the completion of the Intent-Based Interface, as specified

in Task T5.1. Task T5.2 focuses on the integration of the final versions of the components developed under WP3 and WP4, ensuring interoperability and alignment with the overall system architecture. Additionally, the document presents the results of Task T5.3, which defines the demonstration strategy, and Task T5.4, which addresses continuous validation.

This deliverable builds upon and extends the work previously presented in D5.2 First HORSE Release: HORSE IT-1 version [1], and follows the final architectural guidelines established in D2.4 HORSE Landscape and Architectural Design [2]. These developments are aligned with the final set of requirements defined in D2.3 HORSE Landscape: Technologies, State of the Art, AI Policies, and Requirements (IT-2) [3].

The document also reflects the final integration for the HORSE components reported in D3.2 HORSE Platform Intelligence Developed (IT-2) [4] and D4.2 HORSE AI-assisted Human-centric Secure and Trustable Orchestration Developed (IT-2) [5]. Together, these contributions form the foundation of the final, integrated HORSE platform, supporting its deployment as a comprehensive cybersecurity solution for next-generation network environments.

1.3 Structure of the document

The document is structured as follows:

- Section 2 presents the final developments of the HORSE platform components, consolidating the results of the work conducted under WP3, WP4, and WP5. It includes concise descriptions of each component's functionality, along with links to the corresponding source code repositories on GitHub. Additionally, the section provides details on component dependencies and comprehensive instructions for installation and execution, enabling seamless deployment and testing of the components.
- Section 3 outlines the validation strategy adopted for the second release of the HORSE platform. It introduces the integration matrix, which maps the directional interactions between HORSE components and the underlying infrastructure. Each interaction is uniquely identified by a reference ID, which is consistently used throughout the document for traceability. This section also describes the tools and methodologies employed to monitor and manage integration activities during the project. Additionally, it provides detailed documentation of each interaction, including associated data types, communication protocols, and the current implementation status.
- Section 4 focuses on the testing procedures applied to the final HORSE framework. For every interaction defined in Section 3, the corresponding test case, along with its outcome and status, is systematically presented. This ensures a clear view of the testing coverage and the overall readiness of the integrated platform.
- Section 5 presents the Compliance Matrix for HORSE Iteration IT-2, which maps the final HORSE requirements defined in Deliverable D2.3 HORSE Landscape: Technologies, State of the Art, AI Policies, and Requirements (IT-2) [3] to the validation activities conducted throughout the project. This matrix links each requirement to the relevant validation tests and demonstrators from both iterations of the project (IT-1 and IT-2). This structured mapping not only ensures clear traceability between the defined requirements and their practical validation, but also supports systematic evaluation, and provides evidence of requirement fulfillment across the project's use cases and demonstrators.

2 HORSE Release 2

This section presents an overview of the components included in the second release of the HORSE platform. Each component is described in detail, including its core functionalities, system requirements, dependencies, and step-by-step installation guidelines to support deployment and integration. In addition to these technical details, this section also provides direct access to the project's GitHub repository, where the full source code, documentation, and development history are maintained for reference and collaboration.

The design and implementation of these components are based on the HORSE architectural framework designed in WP2 and reported in Deliverable D2.4 HORSE Landscape and Architectural Design [2]. Figure 1 provides a visual representation of the main components of the HORSE framework, highlighting their interfaces and the communication flows between them. This diagram serves as a navigational aid, helping users contextualize each component within the framework and better comprehend how they interact.

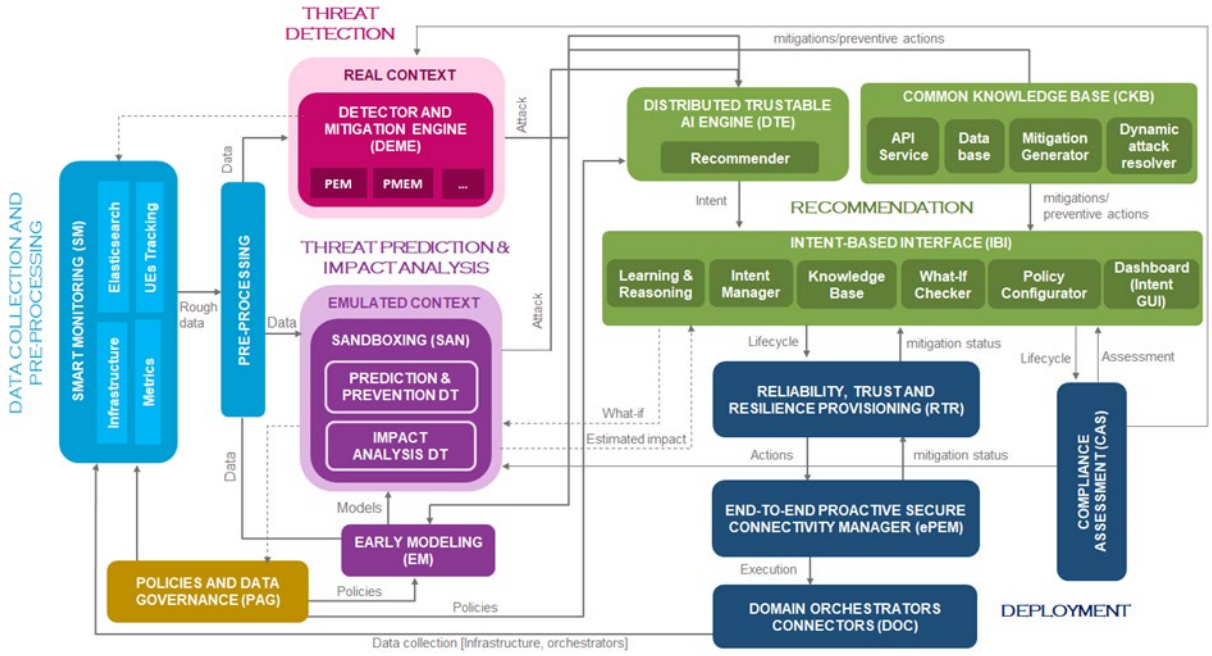

Figure 1: HORSE architecture.

2.1 Smart Monitoring

The Smart Monitoring (SM) component is responsible for the continuous collection, processing, and storage of critical network and system data within the HORSE framework. It serves as the foundational data provider for various HORSE components by offering processed data streams and persistent storage.

The SM component integrates with several components, including Pre-Processing, Policies and Data Governance (PAG), and others, by exposing interfaces for data access and storage. Pre-Processing utilizes the collected data to extract relevant features for further analysis, while PAG and other components can store or retrieve data.

Smart Monitoring provides a wide range of information such as network traffic data (e.g., pcap files), user equipment (UE) location data, and other metadata related to network behavior. This data enables HORSE components to make informed decisions and coordinate actions based on the current and historical state of the network.

- Data collection (e.g. network traffic, system metrics) and persistence in Elasticsearch[1]
- Data querying via Elasticsearch
- Data visualization via Kibana[2]

Version: v1.0

Link to repository:

https://github.com/HORSE-EU-Project/Sphynx-Data-Collection

Installation Instructions:

https://github.com/HORSE-EU-Project/Sphynx-Data-Collection/blob/main/elastic/README.md

https://github.com/HORSE-EU-Project/Sphynx-Data-Collection/blob/main/smartPcapIngestion/README.md

https://github.com/HORSE-EU-Project/Sphynx-Data-Collection/blob/main/UELocator/README.md

List of dependencies and third-party software Included in the current release

- Ubuntu version 22.04.3
- Git version 2.34.1
- Elasticsearch version 7.17.13
- Kibana version 7.17.18
- Docker version 24.0
- Docker Compose version 2.27.0
- Python version 3.12.3
- Flask version 3.1

2.2 Pre-Processing

The Pre-processing component acts as a critical middleware between the Smart Monitoring (SM) component and several key HORSE components, including the Detector and Mitigation Engine (DEME), the Network Digital Twin (NDT), and Early Modeling (EM). These components depend on a continuous and reliable flow of network traffic to operate effectively. The main role of the Pre-processing component is to extract raw network traffic data from the SM's Elasticsearch instance, clean and restructure it, and filter the useful information required by each consuming component. The pre-processing procedure incorporates data validation against predefined JSON schemas to ensure integrity, robust error-handling mechanisms to address malformed or incomplete records, and performance optimizations adapted from the Data Fusion Framework (DFF) to support real-time and high-volume data processing. After processing, the component forwards the relevant data to the appropriate HORSE component (such as DEME) for real-time decision making, while also storing the results in a specialized analytics index within Elasticsearch. Its containerized deployment enhances security, ensures isolation, and simplifies integration across various system environments.

---

[1] *Elasticsearch, https://www.elastic.co/elasticsearch*

[2] *Kibana, https://www.elastic.co/kibana*

Version: v1.0

Link to repository:

https://github.com/HORSE-EU-Project/Pre-Processing

Installation Instructions:

https://github.com/HORSE-EU-Project/Pre-Processing/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Ubuntu 22.04.3
- Docker, version 26.1.1
- Docker Compose version v2.27.0
- Git version 2.34.1

2.3 Prediction and Prevention Network Digital Twin

The Prediction and Prevention (P&P) Network Digital Twin (NDT) is one of the components of the Sandboxing module. The component is aimed to build an emulated replica of the actual 5G/6G infrastructure in order to perform predictions about potential events that might happen in the future and enable prevention strategies. This component receives information from the Early Modelling (EM) about the models of the attacks and the mitigations of each of them, and it produces a list of potential threats and mitigation actions to submit to the Distributed Trustable AI Engine (DTE).

The component is provided in the form of a Virtual Machine integrating the network emulator environment and the associated software to build and run the Digital Twin. This version includes:

- input and output REST interfaces for NDT control
- stable interfaces with EM and DTE components
- algorithms to perform prediction and detection of attacks
- a web interface to check the received input files and produced outputs, and to test the actions and predictions of the P&P DT.

Version: v2.3

Link to repository:

https://github.com/HORSE-EU-Project/PredictionDigitalTwin

Installation Instructions:

https://github.com/HORSE-EU-Project/PredictionDigitalTwin/blob/main/README.md

List of main dependencies and third-party software included in the current release

- Virtualbox 7.0 (VM hosting environment)
- Vagrant v2.4.1 (VM provisioning)
- Ubuntu 20.04
- Comnetsemu: v0.3.0
- UERANSIM: v3.2.6

- Open5gs: v2.4.4
- Docker v28.3.0 or newer
- Python 3.13 or newer
- FASTApi
- Pymongo
- Uvicorn
- Redis
- Streamlit
- Dnspython
- Matplotlib
- NetworkX
- Pandas
- Scapy
- Statsmodels
- Tabulate
- Pyfiglet
- Java openjdk-11-jdk
- Wireshark / Termshark
- Ntfyù
- Flask
- Python xml library
- Python json library

2.4 Impact Analysis Network Digital Twin

The final version of the Impact Analysis Network Digital Twin (IA-NDT) includes different sub-modules that collectively deliver the component's full set of functionalities. These sub-modules are:

- Network Digital Twin: is the sub-module that contains the pods representing the topology agreed with a Prometheus instance for monitoring.
- Policy Translator: this sub-module translates the outputs of the HORSE components into a valid format for the orchestrator.
- Security Orchestrator: enables mitigation policies to be applied to the NDT.

The IA-NDT communicates with the Smart Monitoring for receiving the topology information, the Early Modelling for the attack information and the Intent-Based Interface for the What-If loop.

Version: v1.0

Link to repository:

https://github.com/HORSE-EU-Project/Impact-Analysis-Digital-Twin

Installation Instructions:

https://github.com/HORSE-EU-Project/Impact-Analysis-Digital-Twin/blob/main/README.md

List of dependencies and third-party software Included in the current release

- KIND (Kubernetes cluster)
- Golang, version 1.21 or newer
- Kubectl, version 1.27.3 or newer
- Make
- Docker, version 20.10.16 or newer
- Kubernetes Network Emulator (KNE)
- Open5GS, version 2.7.0
- UERANSIM, version 3.2.6

2.5 Early Modelling

The current version of the Early Modeling (EM) component is responsible for the characterization of three attacks considered in HORSE including DNS amplification, NTP DDoS, and DDoS Downlink attacks. The EM component analyses the information of these aforementioned threats by characterizing assets, threat actor, Tactics, Techniques, and Procedures (TTP), cyber security vulnerabilities, threats, and impact assessment of the attack and control strategy. The current version of the EM is demonstrated with the help of key inputs and outputs. More specifically, in HORSE, we are following a fixed network topology, so rather than incorporating the information of organization assets by their ID, type, and criticality, we have incorporated the topology information as network features, which include information about nodes and ports respectively. In a similar way, we have incorporated threat information received by different partners such as source node, destination node, threat identifier, number of packets, protocol and flag information as well as the information of threat execution time. The input attack information is provided in the form of JSON. Moreover, the mitigation actions are provided by the common knowledge base (cKB) and the outcome of the component is provided in the form of an XML file. This XML file is used by the sandboxing/ network digital twin (NDT) for the prediction of threats and preliminary assessment of the attack impact.

Version: v2.0

Link to repository:

https://github.com/HORSE-EU-Project/Earlymodeling_v1

Installation Instructions:

https://github.com/HORSE-EU-Project/Earlymodeling_v1/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Docker version 24.0.7

2.6 Policies and Data Governance

The PAG component is a data processing service built with NestJS[3]. It is designed to harvest Packet Capture (PCAP) data in JSON format from the Smart Monitoring component. Subsequently, the PAG handles and anonymizes sensitive information, and finally it stores the anonymized PCAP data back into the storage indexes of the Smart Monitoring component. The service uses Redis and Bull for job queue management and scheduling, ensuring efficient and scalable processing.

Version: v0.2

Link to repository:

https://github.com/HORSE-EU-Project/PAG/tree/main/v0.2

Installation Instructions:

https://github.com/HORSE-EU-Project/PAG/blob/main/v0.2/README.md

List of dependencies and third-party software Included in the current release:

- Nestjs - v11.0.17
- Nestjs Scheduler - v5.0.1
- Docker - v4.26.1+
- Bull - v4.16.5 (library used for queuing the anonymisation jobs)
- Redis - v6.2.17
- elastic/elasticsearch - v8.17.1 (library used for the communication with the Elastic of the Smart Monitoring component)

2.7 Detector and Mitigation Engine

The component known as DEME (Detection and Mitigation Engine), operating within the "real context" of the HORSE framework, applies innovative Machine Learning-based techniques to deliver timely and accurate detection of potential threats to the physical infrastructure.

DEME leverages a hierarchical tree architecture, where first-stage nodes (Ingress nodes) focus on specific monitored features. These nodes perform historical data analysis, learning, and deviation detection between real-time data and predictions, enabling a narrow focus and thus maximizing performance.

Subsequent stages in the tree progressively learn from the outputs of previous layers, gaining increasing visibility as they approach the egress node. At the top of the hierarchy, the egress node achieves a comprehensive 360-degree view of the system, effectively overcoming the limiting “silo effect” that typically constrains current state-of-the-art solutions.

Through its interfaces DEME:

- receives, in real time and via an Application Programming Interface (API), a snapshot of the monitored network features for each instance within the topology and then generates the corresponding output.

---

[3] *NestJS, https://nestjs.com/*

- returns, in real time and via an API, the detected threats and the relative detection confidence to DTE.

Version: v 1.0

Link to repository:

https://github.com/HORSE-EU-Project/DEME-apis

List of dependencies and third-party software Included in the current release

- Docker, version 20.10.16 or newer

2.8 Common Knowledge Base

The Common Knowledge Base (cKB) is a component of the HORSE project, designed as a centralized database to store and provide essential information on attack mitigations. The uniqueness of this database lies in its enhancement with AI during the data population phase.

The cKB is composed by:

- Database: stores detailed information on various attacks and their corresponding mitigation actions.
- Web Server: exposes APIs to retrieve essential information from the database without direct access.

Here are listed the current functionalities of the cKB:

- The web server's API allows retrieval of a list of mitigations for a specified attack and the complete list of all attacks present in the database.
- Mitigations are provided in JSON format as an ordered list. Each mitigation includes an execution priority to effectively counteract the specified attack, along with a description of its impact and the rationale for its assigned priority.
- Artificial Intelligence, and specifically, Generative AI, is utilized in different ways:
- To merge and, when needed, enhance data gathered from different heterogeneous sources, to build a comprehensive database of attacks and mitigations.
- To assign priorities to each mitigation related to specific attacks.
- To create detailed descriptions for each mitigation and explain why each priority was assigned.

Version: v 2.0

Link to repository:

https://github.com/HORSE-EU-Project/KnowledgeBase-proto

Installation Instructions:

https://github.com/HORSE-EU-Project/KnowledgeBase-proto/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Docker, version 26.0.7

2.9 Distributed Trustable AI Engine

The Distributed and Trustable Engine (DTE) receives inputs from the DEME in the form of advises, for different types of attacks in the network. DTE maps the outcome of ML detection

and classification to appropriate recommendations that are expressed through intents. These intents are later forwarded to the IBI and can be expressed in two forms: a) mitigative, which describes the objectives the system should accomplish in the short term (e.g., within a few seconds) so that the impact of ongoing security incidents is reduced and b) preventive, which describes the objectives the system should accomplish in the longer term so that is better protected against future potential threats.

Version: v 1.0

Link to repository:

https://github.com/HORSE-EU-Project/NKUA-DTE

Installation Instructions:

https://github.com/HORSE-EU-Project/NKUA-DTE/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Docker version 25.0.3
- FastAPI version

2.10 Intent-based Interface

The Intent-Based Interface (IBI) is a component of the HORSE project that functions as an intelligent intermediary for cybersecurity operations. It is designed to receive and interpret high-level security objectives, known as "intents," and translate them into actionable mitigation and prevention strategies. Following the proposed HORSE architecture, the IBI receives security intents from the DTE or from the network administrators. Integration with external tools is a key aspect for the IBI component, for example, allowing the component to receive inputs from additional components or even external tools, such as a GUI. Therefore, the component adopts the RESTful services architectural style. The security intents accepted by the IBI are mitigation or prevention intents, used to mitigate detected attacks that are already in place or prevent attacks predicted by the HORSE sandbox components. Internally, the IBI maps intents to actions using a recommender system that combines technologies from expert systems and machine learning techniques. The IBI regularly synchronizes with the cKB component to obtain the most up-to-date mitigation configurations.

The workflow for processing intents varies slightly within the IBI component, depending on the type of intent (mitigation or prevention). When receiving an intent of type "prevention", before the actions are applied in the network, the IBI first checks with the IA-DT whether the outcomes of the changes align with the expectations. For "mitigation" intents, this approach is not adopted because it will incur additional processing time to apply the mitigations. A new functionality was implemented in the second release of HORSE, which is the validation of configuration workflows through the CAS component. Hence, IBI only applies configurations after they are checked against a set of policies by CAS. If some configurations are deemed invalid by CAS, the IBI component either adjusts the previously tried configurations or attempts a new set of migration techniques. After all configurations are validated, the set of proposed mitigation configurations is sent to RTR for their implementation in the infrastructure.

The component is provided in the form of a "Dockerized" Python microservice application for easy deployment. Application configuration and setup are simplified by following best practices for developing containerized applications. This version of the application includes:

- And API that allows for the submission of intents to either mitigate an ongoing attack or prevent a potential one

- An internal threaded loop that processes intents through a multi-step pipeline that includes validation, enrichment with data from the cKB, generating recommendations
- Validation and assessment of configurations with the IA-DT and the CAS
- Tracking of status of intents
- Execution of actions/configurations via the RTR component

Version: v1.0

Link to repository:

https://github.com/HORSE-EU-Project/IBI

Installation Instructions:

https://github.com/HORSE-EU-Project/IBI/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Elasticsearch, version 9.0
- Docker, version 26.1.2
- Docker Compose, version 2.27.0
- Python, version 3.13
- annotated-types, version 0.7.0
- anyio, version 4.9.0
- blinker, version 1.7.0
- certifi, version 2023.11.17
- charset-normalizer, version 3.3.2
- click, version 8.1.7
- elastic-transport, version 8.17.1
- elasticsearch, version 9.0.2
- fastapi, version 0.115.13
- flask, version 3.0.1
- h11, version 0.16.0
- idna, version 3.6
- itsdangerous, version 2.1.2
- jinja2, version 3.1.3
- markupsafe, version 2.1.4
- mysql-connector, version 2.2.9
- numpy, version 2.3.1
- pandas, version 2.3.0
- prompt-toolkit, version 1.0.14
- psutil, version 7.0.0
- pydantic, version 2.11.7
- pydantic-core, version 2.33.2
- pygments, version 2.19.2
- pyinquirer, version 1.0.3
- python-dateutil, version 2.9.0.post0
- pytz, version 2025.2
- pyyaml, version 6.0.2

- regex, version 2024.11.6
- requests, version 2.31.0
- six, version 1.17.0
- sniffio, version 1.3.1
- starlette, version 0.46.2
- typing-extensions, version 4.14.0
- typing-inspection, version 0.4.1
- tzdata, version 2025.2
- urllib3, version 2.1.0
- uvicorn, version 0.34.3
- wcwidth, version 0.2.13
- werkzeug, version 3.0.1

2.11 Compliance Assessment

The Compliance Assessment (CAS) component is responsible for ensuring that all actions initiated by HORSE comply with policies and regulatory requirements before execution outside the HORSE framework. Its primary role is to verify whether proposed actions align with compliance constraints, based on the current operational status of the system.

The CAS component is actively engaged by the Intent-Based Interface (IBI) during the decision-making process for mitigation actions. Upon receiving a request, CAS gathers relevant system state information from other HORSE components and evaluates the proposed action against a set of compliance policies.

Based on this evaluation, CAS returns one of three possible outcomes:

- **Compliant**: the action fully aligns with policies and can proceed.
- **Partially Compliant**: the action requires minor adjustments to meet compliance; CAS provides guidance on necessary modifications.
- **Non-Compliant**: the action is rejected due to misalignment with current policies or system state.
- **Unable to make a decision**: if the mitigation action doesn't correlate with the current attack predicted / detected currently on HORSE.

This assessment mechanism ensures that all external interactions initiated by HORSE remain within legal and operational boundaries, enhancing trustworthiness and accountability.

Version: v0.1

Link to repository:

https://github.com/HORSE-EU-Project/CAS

Installation Instructions:

https://github.com/HORSE-EU-Project/CAS/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Ubuntu version 22.04.3
- Java 17
- OPA version 1.5.0
- Docker version 26.1.2

- Docker Compose version 2.27.0
- Git version 2.34.1

2.12 Reliability, Trust and Resilience

The Reliability, Trust, and Resilience (RTR) component is a core component of the mitigation enforcement workflow within the HORSE architecture. Positioned between the Intent-Based Interface (IBI) and the End-to-End Proactive Secure Connectivity Manager (ePEM), the RTR is responsible for interpreting high-level mitigation instructions that describe the security actions to be applied to specific areas of the network topology. These instructions can be provided either in a structured JSON format or in natural language. The RTR validates the input and, when necessary, leverages advanced Natural Language Processing (NLP) techniques to translate free-text commands into structured mitigation actions. Once interpreted, the RTR generates applicable rules in the form of Ansible playbooks, which are then forwarded to the ePEM for enforcement. Additionally, the RTR includes mechanisms to monitor the status of each mitigation action, tracking whether it is pending, successfully enforced, or has failed, and logging relevant details to support real-time observability and post-event analysis.

Version: v 0.1

Link to repository:

https://github.com/HORSE-EU-Project/RTR

Installation Instructions:

https://github.com/HORSE-EU-Project/RTR/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Ubuntu 22.04.3
- Docker, version 26.1.1
- Docker Compose version v2.27.0
- git version 2.34.1

2.13 End-to-End Secure Connectivity Manager

In the HORSE project, the End-to-End Proactive Secure Connectivity Manager (ePEM) plays a crucial role in enabling a secure and resilient 6G wireless and computing ecosystem. Its primary responsibility lies in service orchestration, facilitating the recursive deployment of various functional components.

The ePEM supports the following key aspects.

- **Multi-tenancy and Device Heterogeneity**: it enables the flexible deployment of services for multiple users and accommodates a wide range of diverse devices through virtualization.
- **End-to-End Resource Self-Configuration**: it ensures that resources are automatically configured across the entire network, from end to end.
- **Secure Framework Provision**: it provides a secure environment for all deployed services and applications, working in conjunction with other security components within the HORSE architecture. This includes integrating actions from reliability, trust, and resilience provisioning to ensure a robust and secure performance.

Key Features

- Deploys blueprints that build 5G, Kubernetes (k8s), and VyOS2 services
- Manages the lifecycle of blueprints
- Manages k8s cluster and Machines/VMs

Version: v1.0.1

Link to repository:

- https://github.com/HORSE-EU-Project/ePEM

Installation Instructions:

- https://github.com/HORSE-EU-Project/ePEM/blob/HORSE/README.md
- https://nfvcl-ng.readthedocs.io/en/latest/index.html#getting-started

List of dependencies and third-party software Included in the current release

- An OpenStack instance (you can use an all-in-one installation here).
- An Ubuntu (22.04 LTS) instance where the NFVCL will be installed and run.
- Having OSM 14 running on a reachable machine, in the following installation procedure, all the services will be installed on the same Ubuntu instance.
- Python 3.11 (Installation performed in setup.sh).
- If the NFVCL and OSM are running on the same machine:
  - **RECOMMENDED**: 4 CPUs, 16 GB RAM, 80GB disk and a single interface with Internet access.
  - **MINIMUM**: 2 CPUs, 8 GB RAM, 50GB disk and a single interface with Internet access.

2.14 Domain Orchestrator Connectors

The Domain Orchestrator Connectors (DOC) component is the final component in the HORSE mitigation workflow, playing a critical role in managing multi-cluster and multi-domain environments. Its primary purpose is to orchestrate and enforce mitigation actions across diverse and geographically distributed network infrastructures. The DOC is responsible for adapting standardized mitigation actions generated within the HORSE context, typically originating from the RTR and passed through the ePEM, into actionable commands tailored for specific orchestrators. Each mitigation action in JSON format is translated into a corresponding enforcement operation, depending on the requirements and communication protocols of the target infrastructure segment. By enabling this dynamic translation and orchestration, the DOC allows the HORSE platform to support seamless deployment and policy enforcement across heterogeneous environments, including RAN, Core, Transport, Edge, and Cloud segments. This facilitates scalable and flexible cross-geographical network management.

Version: v1.0

Link to repository:

https://github.com/HORSE-EU-Project/Domain-Orchestrator-Connectors

Installation Instructions:

https://github.com/HORSE-EU-Project/Domain-Orchestrator-Connectors/blob/main/README.md

List of dependencies and third-party software Included in the current release

- Ubuntu 22.04.3
- Docker, version 26.1.1
- Docker Compose version v2.27.0
- git version 2.34.1

3 Final HORSE Framework Integration

This section outlines the validation done for the HORSE intermediate release (IT-2). More specifically, it presents the approach for managing and monitoring the integration of HORSE components, along with the implementation progress of the specified endpoints. The HORSE integration process is supported by comprehensive documentation and systematic monitoring framework.

As detailed in D5.2 “First HORSE Release: HORSE IT-1 version” [1], we have developed an integration matrix, as depicted in Table 1, encapsulating the interactions between HORSE framework components, and also including the infrastructures, emulated through the CNIT, UMU and UPC testbeds, where the PoCs and use cases are validated.

The rows and columns of the integration matrix include the testbed and HORSE components, illustrating the interactions as specified in the PoCs and UCs workflows. In this matrix, each interaction is identified using a unique code in the format [component_A_ID].[component_B_ID], denoting a directed link from Component A to Component B. These unique component IDs are consistently referenced across all validation tables.

	1. SM	2. PreProc	3. DEME	4. DTE	5. P&P NDT	6. IA NDT	7. EM	8. PAG	9. IBI	10. RTR	11. ePEM	12. DOC	13. CAS	14. CKB	15. INFR
1. SM		1.2			1.5	1.6	1.7	1.8							
2. PreProc	2.1		2.3												
3. DEME				3.4									3.13		
4. DTE									4.9						
5. P&P NDT				5.4										5.14	
6. IA NDT									6.9						
7. EM					7.5	7.6									
8. PAG	8.1														
9. IBI						9.6				9.10			9.13	9.14	
10. RTR											10.11				
11. ePEM												11.12			
12. DOC															12.15
13. CAS			13.3						13.9						
14. CKB									14.9						
15. INFR	15.1														

Table 1: HORSE Framework intermediate release (IT-1) validation matrix

Based on the final HORSE framework validation matrix (Table 1), and the workflows defined in D2.4 HORSE Landscape and Architectural Design [2], the HORSE components’ interfaces have been developed, which are presented in Table 2 to Table 14. The information they encapsulate includes:

- ID: Unique identifier as set in the HORSE integration matrix

- CompA and CompB: Names of the HORSE components (output from component A, input to component B)
- Responsible: Partner(s) responsible for the implementation and documentation of the respective integration endpoint
- Data Type and Protocol used for the information exchange between component A and component B

Next, each of the ten HORSE demonstrators is described in detail, including the considered scenario, the associated workflow, and the corresponding integrations, along with the data types and protocols used in the integration process.

3.1 DEMO 0: Hello World

3.1.1 Description

The attack scenario "*Attack 0: Synthetic Detection and Prevention Test*" is designed to validate the communication and functionality of the HORSE framework components under controlled conditions. Synthetic data is injected into the system to simulate an attack, specifically by generating ping traffic toward the DEME and the Prediction and Prevention Digital Twin (DT) components. The test evaluates the detection and prediction capabilities of the security system across multiple components, including Pre-processing, DEME, DTE, IBI, CKB, RTR, ePEM, CAS, and DOC. Two sub-tests are performed: the first focuses on pure threat detection, while the second evaluates proactive threat prediction. Mitigation actions are triggered according to predefined policies within the CAS component, ensuring that detection (Policy 1) and prediction (Policy 2) mechanisms operate as expected.

3.1.2 Testbed and Context Environment

The testbed, depicted in Figure 2, consists of a 5G communication infrastructure, with all components deployed as Docker containers.

At the lowest layer of the architecture, there are 10 User Equipments (UEs). These UEs are evenly distributed: five are connected to one gNodeB and the remaining five to a second gNodeB. All the components are emulated using UERANSIM.

The gNodeBs are connected to an edge router that links them to a Multi-access Edge Computing (MEC) server. The MEC server is currently idle but prepared for the deployment of services. It is connected to a User Plane Function (UPF) deployed at the edge, operating within a different network than the central 5G core.

This edge router is also connected to the main router of the architecture. The main router interconnects with the primary user plane and the central 5G core, both built using Open5GS. Additionally, it connects to another internal router that provides access to a DNS server. From this DNS router, traffic is routed through a gateway router that enables access to the Internet and to the HORSE core network. The gateway router also hosts an application server for service execution at the perimeter of the infrastructure.

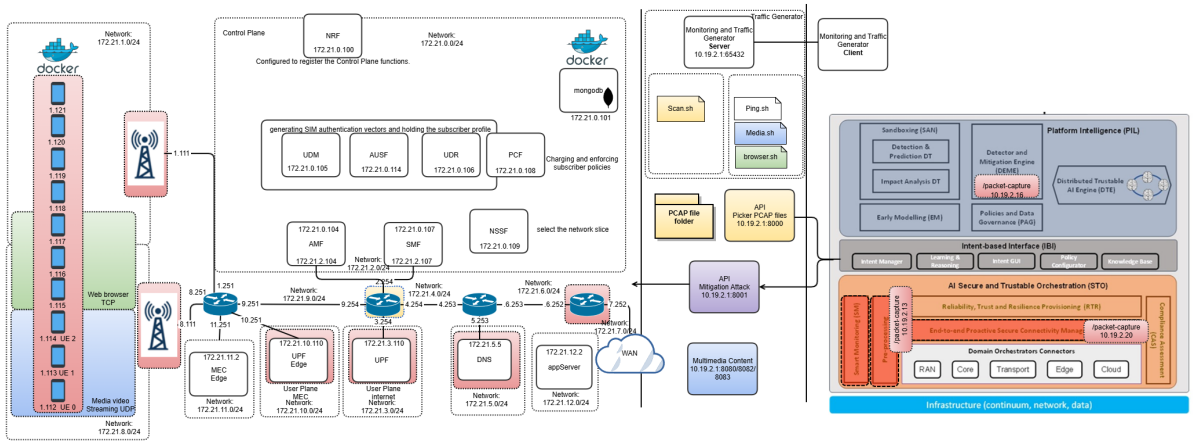

Figure 2: Demo 0 UPC testbed configuration

In addition to the overall testbed architecture, a set of dedicated APIs has been implemented to enhance experiment control. These APIs enable real-time traffic monitoring, PCAP file management, attack simulation, and automated mitigation actions across the 5G infrastructure.

Monitoring and Traffic Generation Tools

Embedded Monitoring (TCPdump)

Several nodes in the infrastructure are equipped with **TCPdump** for comprehensive traffic monitoring. These include:

- User Equipments (UEs)
- gNodeBs
- UPF nodes
- The DNS server
- The gateway router

TCPdump is configured to capture traffic in PCAP files on all available network interfaces, enabling in-depth packet-level analysis and data collection during experiments. The files are stored in a shared folder with the nodes of HORSE architecture.

Traffic Control Interface

The testbed includes dedicated nodes that support testbed interaction and traffic generation. A central **TCP/IP server** allows remote command execution through a **TCP/IP client**, enabling the following operations:

- *monitoring <time> <steps>*: Runs the monitoring script for the specified duration and number of steps.
- *trafgen ping <ID_UE_0-9> <time>*: Triggers ICMP traffic generation on a specific UE for a given time.
- *trafgen media <time>*: Generates media-like traffic on UE -0, UE -1, and UE -2.
- *trafgen web <time>*: Simulates web traffic on UE -3, UE -4, and UE -5.

These tools provide a flexible mechanism for conducting controlled experiments across different traffic scenarios.

API for PCAP File Management

A dedicated **FastAPI-based service** has been developed to manage .pcap packet capture files generated within the 5G environment. This API provides the following endpoints:

- **POST /uploadPCAPfiles**
  Upload a .pcap file with a unique timestamp identifier.
  Parameters:
  - timestamp (str): Unique temporal identifier.
  - file (UploadFile): PCAP file to upload.

  Response: Confirmation of successful upload.

- **GET /downloadPCAPfiles/{file_id}**
  Download a specific .pcap file by its identifier.
  Parameters:
  - file_id (str): Identifier of the file.

  Response: Returns the file or a 404 error if not found.

- **GET /getPCAPfiles**
  Retrieve the list of all stored .pcap files.
  Response: List of filenames.

- **DELETE /deletePCAPfile/{file_id}**
  Delete a specific .pcap file.
  Parameters:
  - file_id (str): File identifier.

  Response: Confirmation of deletion or a 404 error.

This API enables flexible and centralized access to traffic capture data for post-analysis and performance validation.

Mitigation API

An additional API has been developed to **mitigate security threats** by dynamically interacting with affected nodes. The available endpoints and operations include:

Test Execution

POST /execute_test

Run predefined test scenarios using specific components.

- **Test 1** components: Pre-processing, DEME, DTE, IBI, CKB, RTR, ePEM, CAS
- **Test 2** components: Pre-processing, EM, P&P DT, DTE, IBI, CKB, IA DT, RTR, ePEM, CAS

Network-Level Mitigation Actions

POST /dns_rate_limiting:

- Apply rate-limiting on the DNS server for selected IPs.
- Fields: rate, duration, source_ip_filter

POST /router_rate_limiting

- Limit request rate from the router on port 53.
- Fields: device, rate, duration

POST /block_ip_addresses

- Block specific malicious IP addresses.
- Fields: blocked_ips

POST /block_ues_multidomain

- Apply global rate-limiting on all UEs at DNS level (port 53).
- Fields: rate_limiting

POST /define_dns_servers

- Register DNS servers used in the testbed.
- Fields: dns_servers with name and ip

POST /firewall_pfcp_requests

- Drop a percentage of PFCP packets on port 8805.
- Fields: drop_percentage, request_types

POST /validate_smf_integrity

- Verify and restart the SMF container if compromised.
- Fields: check, action

POST /discard_unauthorized_packets

- Discard unauthorized packets and optionally trigger external alarms.
- Fields: filter, alarm

These capabilities allow for real-time detection, response, and enforcement mechanisms across the 5G architecture, enabling validation of resilience and security policies.

3.1.3 Workflow Demo 0 - Threat Detection

Figure 3 presents the workflow for demo 0, specifically for threat detection, including all the HORSE components involved, as well as their interactions.

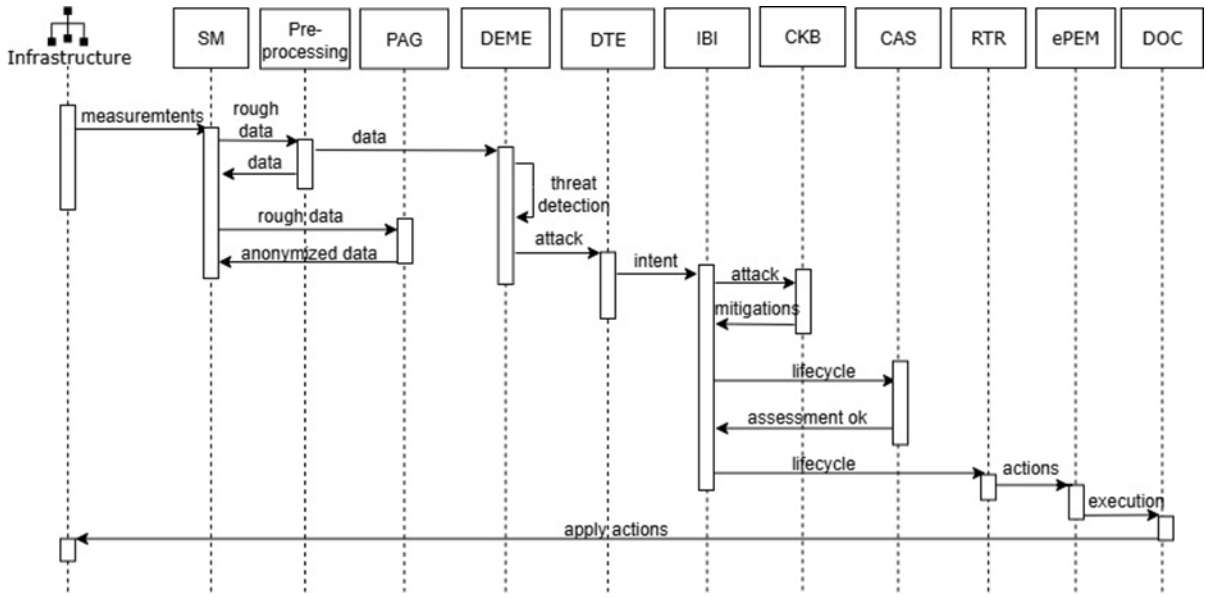

Figure 3: Demo 0 – Threat Detection Workflow

3.1.4 Integration Demo 0 - Threat Detection

Table 2 presents the integration done in Demonstrator 0 for the threat detection workflow. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
15.1	INFR	SM	UPC	STS	PCAP file in Shared Folder.
1.2	SM	PreProc	STS	8BELLS	PCAP in JSON format.
2.1	PreProc	SM	8BELLS	STS	JSON formatted record, sent over HTTP protocol using REST APIs
1.8	SM	PAG	STS	S5	PCAP data in JSON format from ES index.
8.1	PAG	SM	S5	STS	PCAP data in JSON format to ES index.
2.3	PreProc	DEME	8BELLS	ETI	JSON format returned via dedicated API calls.
3.4	DEME	DTE	ETI	NKUA	JSON format returned via dedicated API calls.
4.9	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigating an attack in JSON format sent over HTTP protocol using REST APIs.
9.14	IBI	CKB	TUBS	MAR	Description of the attack in JSON format sent over HTTP protocol using REST APIs
14.9	CKB	IBI	MAR	TUBS	List of possible mitigation actions in JSON format sent over HTTP protocol using REST APIs.
9.13	IBI	CAS	TUBS	STS	Actions to be validated JSON over RESTful/HTTP.
13.9	CAS	IBI	STS	TUBS	JSON formatted answer on the action over RESTful/HTTP.
9.10	IBI	RTR	TUBS	8BELLS	List of actions to be enforced in JSON format sent over HTTP protocol using REST APIs.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
10.11	RTR	ePEM	8BELLS	CNIT	List of commands for each action to be enforced in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
11.12	ePEM	DOC	CNIT	8BELLS	List of command for each action not enforced directly by ePEM in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
12.15	DOC	Infrastructure	8BELLS	UPC	Mitigation actions JSON using REST APIs. The JSON follows a predefined schema that the infrastructure can parse

Table 2: HORSE components integration endpoints – Demonstrator 0 – Threat detection workflow

3.1.5 Workflow Demo 0 - Threat Prediction

Figure 4 presents the threat prediction workflow for demo 0. It includes all the HORSE components involved, as well as their interactions.

Threat Prediction Workflow

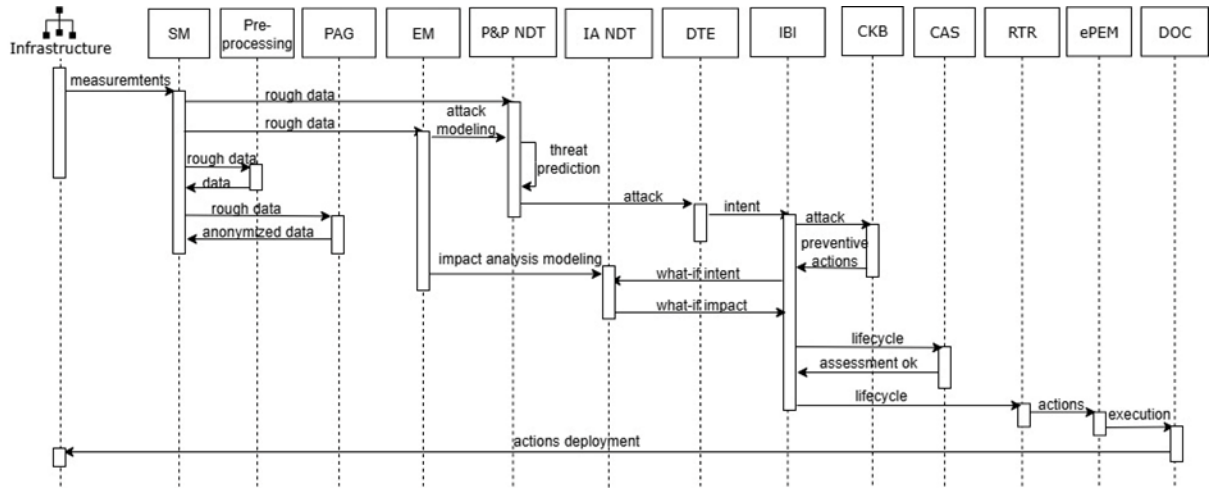

Figure 4: Demo#0 – Threat Prediction Workflow

3.1.6 Integration Demo 0 - Threat Prediction

Table 3 presents the integration done in Demonstrator 0 for the threat prediction workflow. It includes the integration endpoints of the different components involved, the partner

responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
15.1	INFR	SM	UPC	STS	PCAP file in Shared Folder.
1. 5	SM	P&P NDT	STS	CNIT	XML file Rest API/HTTP.
1.7	SM	EM	STS	UPC	XML over HTTP.
1.2	SM	PreProc	STS	8BELLS	PCAP in JSON format.
2.1	PreProc	SM	8BELLS	STS	JSON formatted record, sent over HTTP protocol using REST APIs
1.8	SM	PAG	STS	S5	PCAP data in JSON format from ES index.
8.1	PAG	SM	S5	STS	PCAP data in JSON format to ES index.
7.5	EM	P&P NDT	UPC	CNIT	Threat Model in API REST
5.4	P&P NDT	DTE	CNIT	NKUA	JSON format sent over HTTP protocol using REST APIs.
4.9	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigate an attack in JSON format sent over HTTP protocol using REST APIs.
9.14	IBI	CKB	TUBS	MAR	Description of the attack in JSON format sent over HTTP protocol using REST APIs.
14.9	CKB	IBI	MAR	TUBS	List of possible mitigation actions in JSON format sent over HTTP protocol using REST APIs.
9.6	IBI	IA NDT	TUBS	TID /UMU	Attack and prevention or mitigation action to be tested by the digital twin in JSON format sent over HTTP.
6.9	IA NDT	IBI	UMU	TUBS	Result the test with collected KPIs in the digital twin encoded in JSON format sent over HTTP.
9.13	IBI	CAS	TUBS	STS	Actions to be validated in JSON over RESTful/HTTP.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
13.9	CAS	IBI	STS	TUBS	Evaluation in JSON format HTTP Request.
9.10	IBI	RTR	TUBS	8BELLS	List of actions to be enforced in JSON format sent over HTTP protocol using REST APIs.
10.11	RTR	ePEM	8BELLS	CNIT	List of commands for each action to be enforced in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
11.12	ePEM	DOC	CNIT	8BELLS	List of command for each action not enforced directly by ePEM in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
12.15	DOC	INFR	8BELLS	UPC	Mitigation actions JSON using REST APIs. The JSON follows a predefined schema that the infrastructure can parse

Table 3: HORSE components integration endpoints – Demonstrator 0 – Threat prediction workflow.

3.2 DEMO 1: NTP DDoS attack detection and mitigation

3.2.1 Description

This attack scenario is designed to validate the framework's ability to effectively detect and initiate appropriate mitigation actions against an NTP-based DDoS amplification attack. To simulate the attack, synthetic data—crafted in accordance with relevant scientific literature and field-tested methodologies—is injected into the system. The simulation specifically involves the generation of NTP MONLIST requests toward the NTP server, which are particularly dangerous as they can disrupt network synchronization, severely degrade performance, or even cause a complete network outage. Moreover, amplification attacks are especially problematic due to their low cost, as they exploit the target's infrastructure to generate large volumes of traffic through the amplification mechanism.

The test assesses the detection and mitigation effectiveness of the security framework across multiple components, including SM, Pre-processing, DEME, DTE, and IBI.

3.2.2 Testbed and Context Environment

The testbed, see Figure 5, consists of a 5G communication infrastructure deployed on an Openstack instance. Each HORSE component is deployed in an Openstack VM. The only exception is the SAN components (NDTs) that are deployed in a docker container. Instead, the 5G Core is deployed in a k8s cluster with one control plane node and one worker node.

At the lowest layer of the architecture, there are 10 User Equipments (UEs). These UEs are evenly distributed: five are connected to one gNodeB and the remaining five to a second gNodeB. All the components are emulated using UERANSIM.

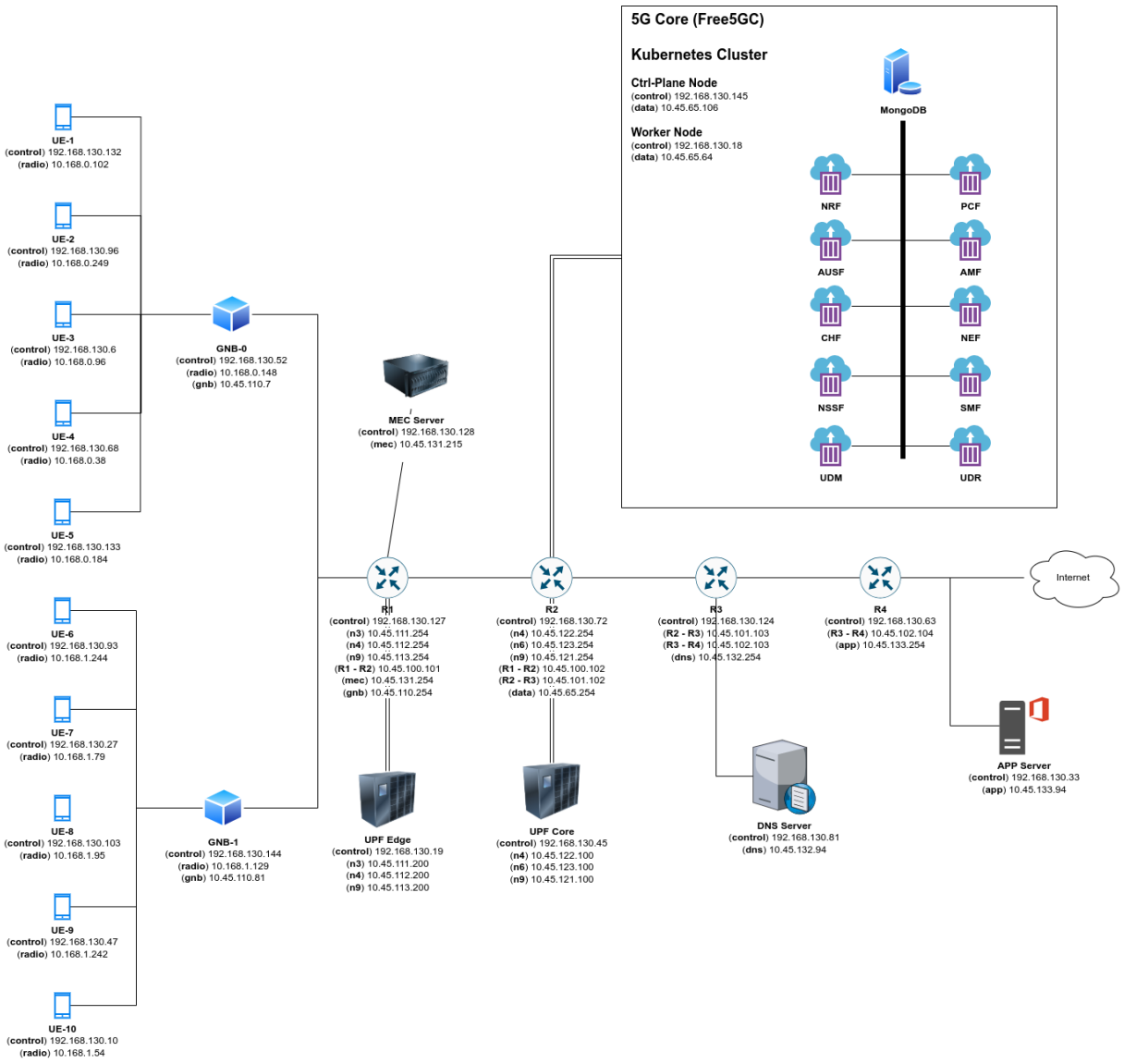

Figure 5. CNIT Testbed.

The gNodeBs are connected to an edge router that links them to an MEC server. The MEC server is currently idle but prepared for the deployment of services. It is connected to a UPF deployed at the edge, operating within a different network than the central 5G core.

This edge router is also connected to the main router of the architecture. The main router interconnects with the primary user plane and the central 5G core, both built using Free5GC. Additionally, it connects to another internal router that provides access to a DNS server. Finally, the last router (R4) is connected to the Application (APP) Server, and it enables access to the

Internet and to the HORSE core network. The APP server allows the service execution at the perimeter of the infrastructure.

3.2.3 Workflow Demo 1

Figure 6 presents the threat detection workflow for demo 1. It includes all the HORSE components involved, as well as their interactions.

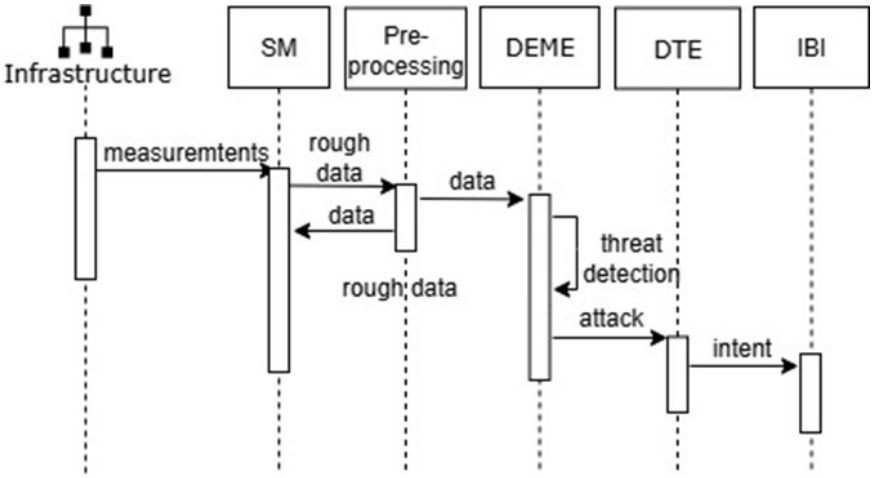

Figure 6: Demo 1 Workflow

3.2.4 Integration Demo 1

Table 4 presents the integration done in Demonstrator 1. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
15.1	INFR	SM	UPC	STS	PCAP file in Shared Folder.
1.2	SM	PreProc	STS	8BELLS	PCAP in JSON format.
2.3	PreProc	DEME	8BELLS	ETI	JSON format returned via dedicated API calls.
3.4	DEME	DTE	ETI	NKUA	JSON format returned via dedicated API calls.
4.9	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigating an attack in JSON format sent over HTTP protocol using REST APIs.

Table 4: HORSE components integration endpoints – Demonstrator 1.

3.3 DEMO 2: DNS DDoS Prediction and Impact Analysis

3.3.1 Description

This demo consists of the evaluation of the impact of a denial of service (DoS or DDoS) Reflection Amplification attack in uplink, i.e. originated from one or several UEs within the 5G network and directed towards an external DNS server. In this scenario, the impact of the attack is intended to be validated in the NDT Impact Analysis using two possible countermeasures: a laxer one that consists of lowering the bandwidth associated with the malicious UEs, and a more rigid one in which the UEs' traffic is completely filtered out. Both countermeasures are applied on any router within the infrastructure. In addition, to validate the impact, the IBI requests some metrics, such as packets per second and bytes per second.

3.3.2 Testbed and Context Environment

The demo will be executed in UMU testbed where all the components involved in the demo are already deployed.

3.3.3 Workflow Demo 2

Figure 7 presents the threat detection workflow for demo 1. It includes all the HORSE components involved, as well as their interactions.

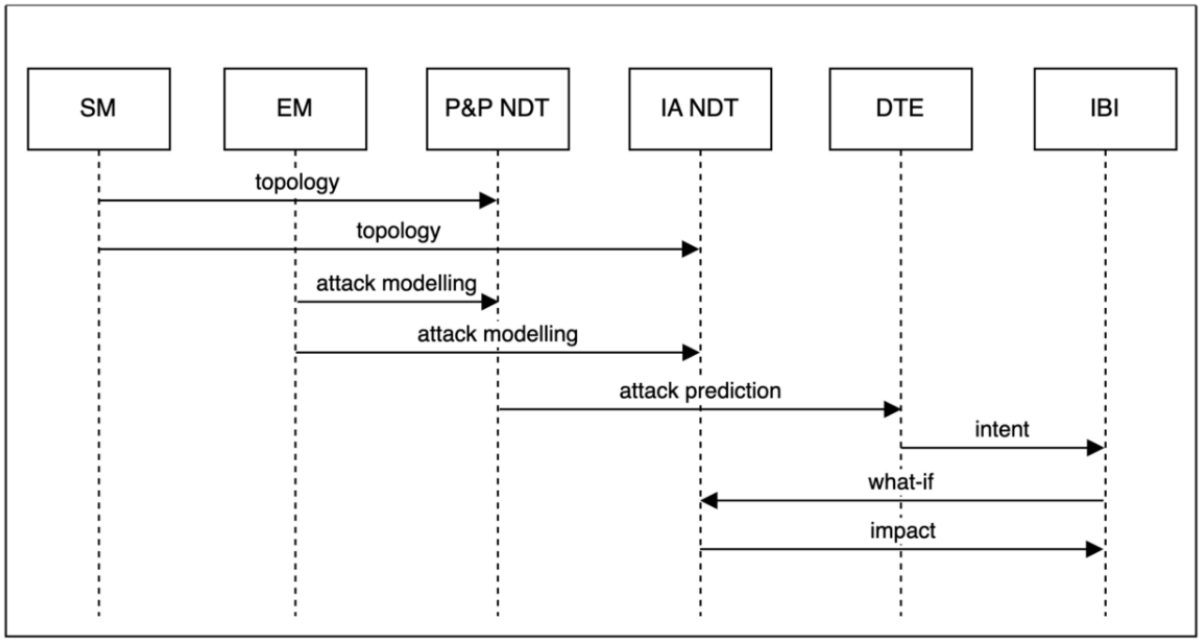

Figure 7: Demo 2 Workflow

3.3.4 Integration Demo 2

Table 5 presents the integration done in Demonstrator 2. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
1.5	SM	P&P NDT	STS	CNIT	Topology in API REST.
1.6	SM	IA-NDT	STS	TID/UM	Topology in API REST.
5.4	P&P NDT	DTE	CNIT	NKUA	Detected Threat in API REST.
7.5	EM	P&P NDT	UPC	CNIT	Threat Model in API REST.
7.6	EM	IA-NDT	UPC	TID/UMU	Threat Model in API REST.
4.10	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigate an attack in JSON format sent over HTTP protocol using REST APIs.
9.6	IBI	IA-NDT	TUBS	TID/UMU	What-if in API REST.
6.9	IA-NDT	IBI	TID/UMU	TUBS	What-if in API REST.

Table 5: HORSE components integration endpoints – Demonstrator 2.

3.4 DEMO 3: DNS DDoS Prediction, Impact and Compliance

3.4.1 Description

This demo consists of the evaluation of the impact of a denial of service (DoS or DDoS) attack in downlink, i.e. from an external attacker directed to a UE of the 5G network. In this scenario, the impact of an attack is intended to be validated in the NDT Impact Analysis using two possible countermeasures: a laxer one that consists of lowering the bandwidth of the attacker's address and a more rigid one in which the attacker's traffic is filtered out completely. Both countermeasures are performed on any router in the infrastructure. In addition, to validate the impact, the IBI asks for some metrics, such as packets per second, bytes per second.

3.4.2 Testbed and Context Environment

The demo will be executed in UMU testbed where all the components involved in the demo are already deployed.

3.4.3 Workflow Demo 3

Figure 8 presents the threat prediction workflow for demo 3. It includes all the HORSE components involved, as well as their interactions.

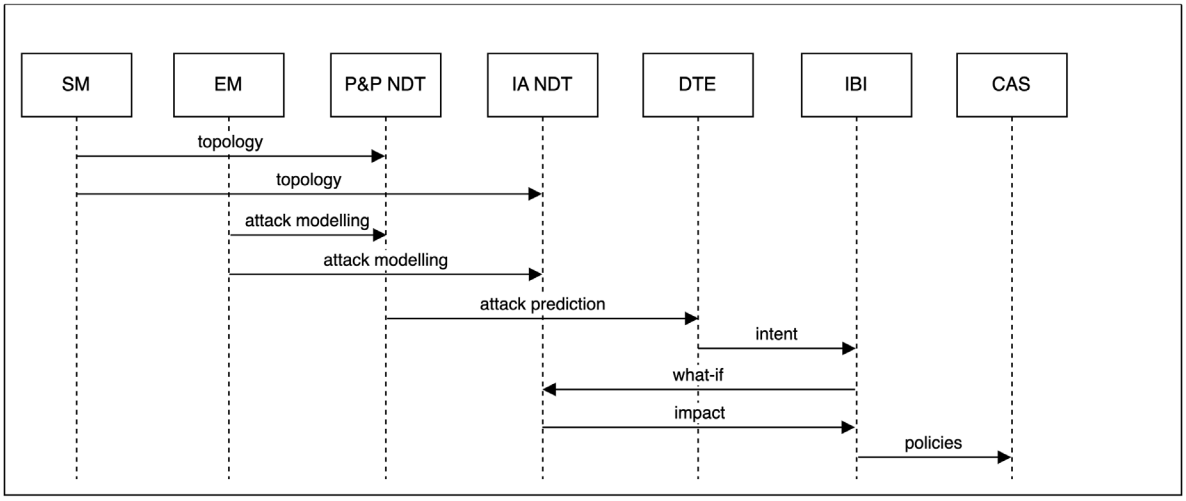

Figure 8: Demo 3 Workflow.

3.4.4 Integration Demo 3

Table 6 presents the integration done in Demonstrator 3. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
1.5	SM	P&P NDT	STS	CNIT	Topology in API REST.
1.6	SM	IA-NDT	STS	TID/UMU	Topology in API REST.
5.4	P&P NDT	DTE	CNIT	NKUA	Detected Threat in API REST.
7.5	EM	P&P NDT	UPC	CNIT	Threat Model in API REST.
7.6	EM	IA-NDT	UPC	TID/UMU	Threat Model om API REST.
4.10	DTE	IBI	NKUA	TUBS	Prevention Intent in JSON over RESTful API.
9.6	IBI	IA-NDT	TUBS	TID/UMU	What-if in API REST.
6.9	IA-NDT	IBI	TID/UMU	TUBS	What-if result in API REST.
9.13	IBI	CAS	TUBS	STS	Actions to be validated in JSON over RESTful/HTTP.

Table 6: HORSE components integration endpoints – Demonstrator 3.

3.5 DEMO 4: DT data poisoning detection

3.5.1 Description

This demo is focused on the detection of anomalies or poisoning in the inputs provided to the P&P NDT. This situation would generate wrong predictions and reduce the performance of the P&P NDT.

In this scenario, the data poisoning is obtained by overwriting the pcap files provided by the infrastructure to the P&P NDT. Such action is performed via a manual script.

The detection is performed in two ways: (i) by checking the consistency of the pcap file using a hashing function, and (ii) by detecting anomalies in the pcap content (e.g. “unusual” IP addresses).

As a result, the P&P NDT will notify the DTE component about inconsistencies in the input data.

3.5.2 Testbed and Context Environment

This demo will be implemented in CNIT testbed and will involve mainly the P&P NDT component and the DTE component of the HORSE architecture.

3.5.3 Workflow Demo 4

Figure 9 presents the threat prediction workflow for demonstrator 4. It includes all the HORSE components involved, as well as their interactions.

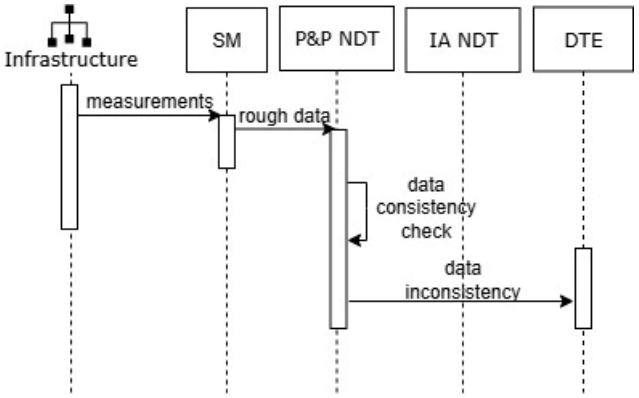

Figure 9: Demo 4 Workflow.

3.5.4 Integration Demo 4

Table 7 presents the integration done in Demonstrator 4. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
0	Attacker script	P&P NDT	CNIT	CNIT	Manual poisoning of the PCAP file traces through a Linux script.
1	P&P NDT	DTE	CNIT	NKUA	Inconsistency notification via REST interfaces.

Table 7: HORSE components integration endpoints – Demonstrator 4.

3.6 DEMO 5: Multidomain DDoS DNS

3.6.1 Description

Attack 5: DDoS DNS Attack on Multidomain Infrastructure with Enforcement of Mitigation Actions

This attack scenario simulates a Distributed Denial of Service (DDoS) attack targeting the DNS servers across two separate domains: the UPC and CNIT testbeds. The DEME component detects abnormal DNS traffic, triggering a coordinated response across the HORSE security components. The DTE, CKB, and IBI components collaboratively define the mitigation workflow, which includes blocking malicious UEs in both domains and generating an updated list of DNS servers. The RTR component automatically generates Ansible playbooks to deploy these countermeasures, while ePEM specifies the technical actions to be enforced. Finally, the DOC component applies the mitigation policies across both infrastructures. The actions include port-based rate limiting and selective blocking of compromised UEs, guided by policies that ensure service continuity, limit collateral impact, and safeguard DNS functionality.

3.6.2 Testbed and Context Environment

UPC and CNIT testbeds are interconnected to demonstrate the multidomain functionality of the HORSE framework, as depicted inFigure 10 In the proposed scenario, UEs from the CNIT and UPC testbeds initiate a multidomain DDoS DNS attack to the DNS server in the CNIT testbed.

The UPC 5G testbed is a containerized infrastructure designed to validate advanced monitoring, detection, and mitigation mechanisms for 5G networks. It includes UEs and gNodeBs emulated with UERANSIM, a MEC server, UPF instances, and an Open5GS Core, all monitored with TCPdump across critical nodes. The testbed incorporates APIs for traffic generation, PCAP management, and automated threat response. UPC testbed is interconnected with the CNIT testbed to enable cross-domain experiments, for the execution of the multidomain DDoS DNS attack.

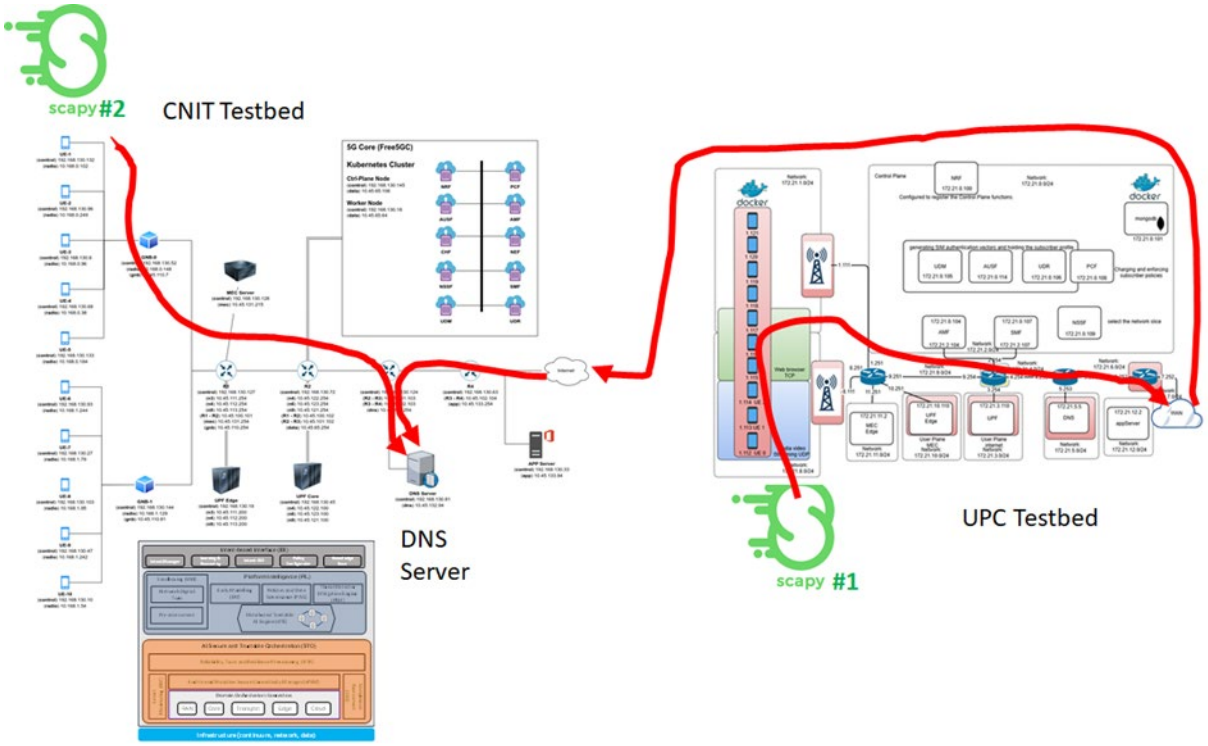

Figure 10: Demo 5 Scenario.

3.6.3 Workflow Demo 5

Figure 11 presents the workflow for demo 5. It includes all the HORSE components involved, as well as their interactions. Note that in this workflow two infrastructures are involved, the CNIT and the UPC testbeds. Moreover, mitigations are enforced in the two testbeds, in this multidomain scenario.

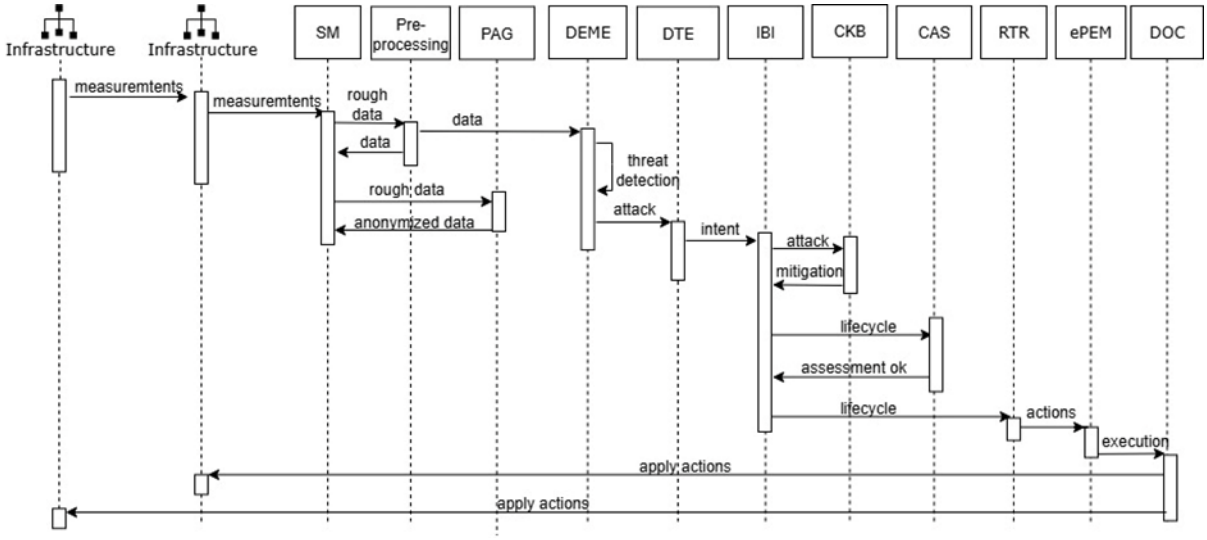

Figure 11: Demo 5 Workflow.

3.6.4 Integration Demo 5

Table 8 presents the integration done in Demonstrator 5. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
15.15	INFR (UPC)	INFR (CNIT)	UPC	CNIT	PCAP file in Shared Folder.
15.1	INFR (CNIT)	SM	CNIT	STS	PCAP file in Shared Folder.
1.2	SM	PreProc	STS	8BELLS	JSON formatted PCAP data sent over HTTP protocol using REST APIs.
2.1	PreProc	SM	8BELLS	STS	JSON formatted record, sent over HTTP protocol using REST APIs
2.3	PreProc	DEME	8BELLS	ETI	JSON format returned via dedicated API calls.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
3.4	DEME	DTE	ETI	NKUA	JSON format sent over HTTP protocol using REST APIs.
4.9	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigate an attack in JSON format sent over HTTP protocol using REST APIs.
9.14	IBI	CKB	TUBS	MAR	Description of the attack in JSON format sent over HTTP protocol using REST APIs.
14.9	CKB	IBI	MAR	TUBS	List of possible mitigation actions in JSON format sent over HTTP protocol using REST APIs.
9.13	IBI	CAS	TUBS	STS	Actions to be validated by CAS encoded in JSON sent over HTTP.
13.9	CAS	IBI	STS	TUBS	JSON formatted answer on the action over HTTP protocol using REST APIs.
9.10	IBI	RTR	TUBS	8BELLS	List of actions to be enforced in JSON format sent over HTTP protocol using REST APIs.
10.11	RTR	ePEM	8BELLS	CNIT	List of commands for each action to be enforced in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
11.12	ePEM	DOC	CNIT	8BELLS	List of command for each action not enforced directly by ePEM in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
12.15	DOC	INFR (CNIT)	8BELLS	CNIT	Mitigation actions JSON using REST APIs
12.15	DOC	INFR (UPC)	8BELLS	UPC	Mitigation actions JSON using REST APIs. The JSON follows a predefined schema that the infrastructure can parse

Table 8: HORSE components integration endpoints – Demonstrator 5

3.7 DEMO 6: MitM on IBI intents

3.7.1 Description

This demo evaluates the resilience of the system against a Man-in-the-Middle (MitM) attack targeting the interfaces between DEME and DTE. The assumption is that an attacker can take over one of the interfaces between DEME and DTE, or between DTE and IBI, and inject malicious or fake mitigation or prevention intents in the HORSE framework. The system will react blindly to the new intent and attempt to fulfill the attacker's requirements without verifying whether the action is necessary. The demo consists of detecting the manipulation of an intent sent to the IBI, thereby preventing the deployment of incorrect mitigation actions.

The process begins with the Detector and Mitigation Engine (DEME) identifying an initial attack, like a DDoS, and reporting it to the Distributed Trustable AI Engine (DTE). The DTE, in turn, will process the detection and convert it into an intent to the IBI. Simultaneously, a malicious actor intercepts and sends a fake intent to the IBI to mitigate a completely different and non-existent attack. The IBI processes the fraudulent intent and proposes corresponding mitigation actions, which are then sent to the CAS for validation. The CAS queries the DEME to get the original attack information, and according to that, assesses the mitigation actions proposed by IBI. In case of fraudulent intent, the CAS does not validate the actions and sends this information back to IBI. The IBI component stops its operation because the information it receives from DTE may no longer be reliable.

3.7.2 Testbed and Context Environment

The demo will be executed in the UPC testbed, since the core components of this demo are already deployed in the testbed facility. It does not involve any new component or advanced measurement tool or software, since it relies on default operations of components. The information required to determine whether the demo was successful can be extracted from the logs of the components involved. The core components involved in this attack scenario are the DEME, the DTE, IBI, and CAS.

3.7.3 Workflow Demo 6

Figure 12 presents the workflow for demo 6. It includes all the HORSE components involved, as well as their interactions.

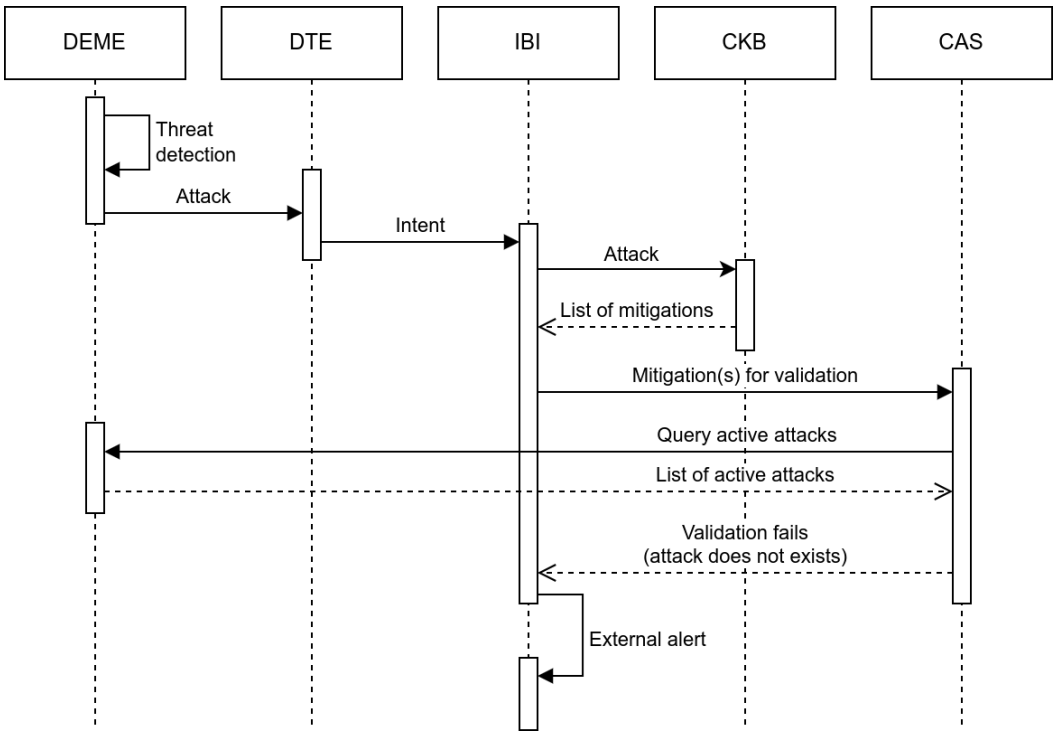

Figure 12: Demo 6 Workflow.

3.7.4 Integration Demo 6

Table 9 presents the integration done in Demonstrator 6. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
3.4	DEME	DTE	ETI	NKUA	JSON format sent over HTTP protocol using REST APIs.
4.9	DTE	IBI	NKUA	TUBS	Intent in JSON format sent over RESTful endpoints using HTTP protocol.
9.13	IBI	CAS	TUBS	STS	Mitigation actions in JSON format sent over RESTful endpoints using HTTP protocol.
13.3	CAS	DEME	STS	ETI	Request the list of ongoing attacks JSON documents over RESTful APIs.
3.13	DEME	CAS	ETI	STS	List of ongoing/current attacks encoded in JSON over HTTP.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
13.9	CAS	IBI	STS	TUBS	Result of the validation of the actions in JSON format sent over RESTful endpoints using HTTP protocol.

Table 9: HORSE components integration endpoints – Demonstrator 6.

3.8 DEMO 7: API and Network Exposure

3.8.1 Description

A malicious actor may attempt to arbitrarily access NEF services in order to retrieve sensitive information such as user traffic routing, device location, and mobility events. This type of information gathering is typically the first phase of an eavesdropping attack, which is especially relevant and dangerous in industrial XR scenarios, such as our second use case.

Brute-force-based attacks can be identified by DEME, which leverages its learning and predictive capabilities to detect anomalies. In this scenario, the Security Manager (SM) captures network traffic, the Pre-processing component extracts and processes NEF API access attempts by IP address, DEME detects suspicious behavior by comparing real-time activity against its predictions, DTE generates the appropriate intent, and IBI orchestrates the corresponding response actions.

3.8.2 Testbed and Context Environment

The CNIT testbed is the same as described in Section 3.2.2 for Demo 1.

3.8.3 Workflow Demo 7

Figure 13 presents the workflow for demo 7. It includes all the HORSE components involved, as well as their interactions.

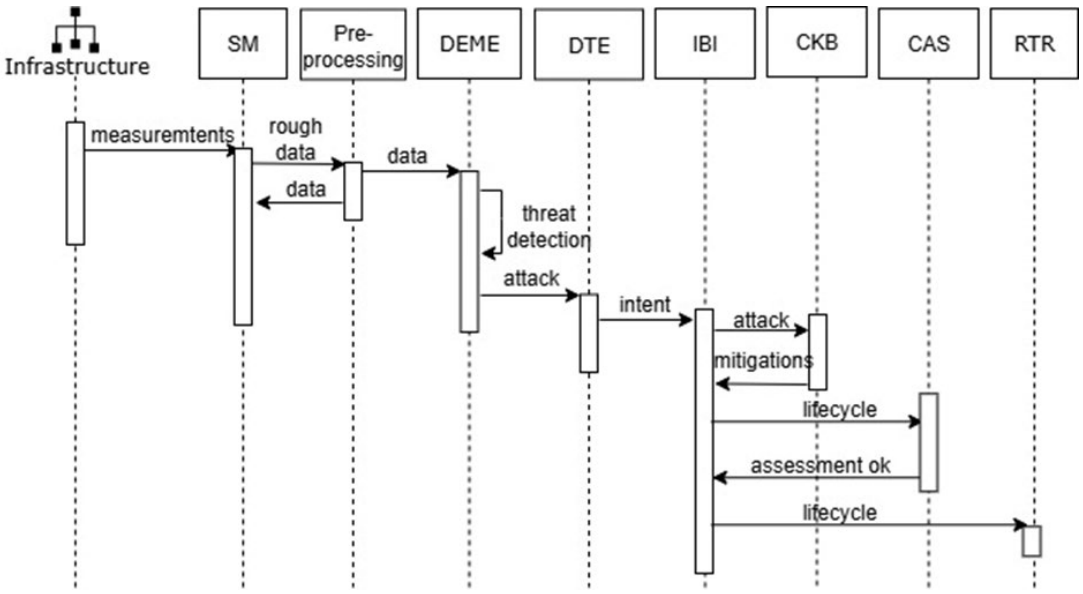

Figure 13: Demo 7 Workflow.

3.8.4 Integration Demo 7

Table 10 presents the integration done in Demonstrator 7. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
15.1	INFR	SM	CNIT	STS	PCAP files in Shared folder.
1.2	SM	PreProc	STS	8BELLS	PCAP data in a JSON format over HTTP.
2.3	PreProc	DEME	8BELLS	ETI	JSON format returned via dedicated API calls.
3.4	DEME	DTE	ETI	NKUA	JSON format sent over HTTP protocol using REST APIs.
4.9	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigate an attack in JSON format sent over HTTP protocol using REST APIs.
9.14	IBI	CKB	TUBS	MAR	Description of the attack in JSON format sent over HTTP protocol using REST APIs.
14.9	CKB	IBI	MAR	TUBS	List of possible mitigation actions in JSON format sent over HTTP protocol using REST APIs.
9.10	IBI	RTR	TUBS	8BELLS	List of actions to be enforced in JSON format sent over HTTP protocol using REST APIs.

Table 10: HORSE components integration endpoints – Demonstrator 7.

3.9 DEMO 8: Attack on signaling PFCP traffic (NKUA)

3.9.1 Description

This demo showcases the detection and mitigation of attacks on the N4 interface of the 5G core network using ML and the DTE processing for creating different types of intents that are sent to the IBI, in order to implement appropriate mitigation and prevention measures. This process involves various steps, aiming to show: 1) the DEME-DTE API where ML attack detection and classification results are given as input to the DTE, 2) the DTE processing of this information and creation of intents, and 3) the DTE-IBI API where these intents are forwarded to the IBI. The demo includes three steps, which are highlighted in Figure 14.

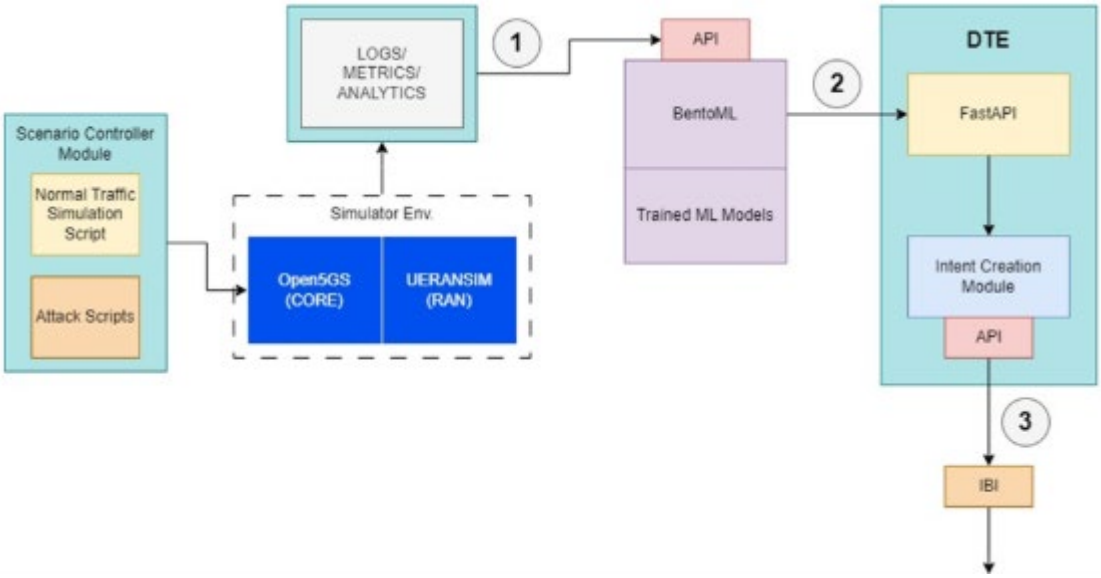

Figure 14: The three-step process of the PFCP attack demonstrator.

More specifically, these steps are given in detail below:

- Step 1 - Execution of PFCP attacks: To create a realistic threat environment, this demo employs Scapy [6], a Python-based packet manipulation tool, to simulate various types of attacks on the N4 interface. A script is developed to prompt a malicious Session Management Function (SMF) to randomly execute an attack every X minutes, where X is a random number between 1 and 30 minutes. In greater detail, the following types of attacks are simulated:
  - Unauthorized PFCP Session Modification Request: For this scenario, the adversary aims to manipulate the UPF to discard packet handling settings. The malicious user achieves this by sending a PFCP Session Modification Request that includes a DROP flag in the Apply Action field of the FAR rules. This action results in the deletion of the Tunnel Endpoint Identifier (TEID) and the IP address of the gNB from the UPF. Consequently, while the connection between the UE and the gNB remains active, the client is unable to access the data network (DN).
  - Unauthorized PFCP Session Deletion Request: This attack sends unauthorized PFCP session deletion requests from the SMF to the UPF, aiming to disconnect a specific UE from the DN without disrupting its connection to the 5G Radio Access Network (RAN) or Core Network (CN). It targets PDU sessions on the N4 interface, affecting connectivity observed on the N3 interface. The only remedy for the affected UE is to restart its session or connect to another gNB, which generates a new SEID, halting the attack's effect.
  - Unauthorized PFCP Session Establishment: This attack is instantiated from the SMF of the 5G CN. The target of this attack is the UPF, which handles processes and forwards user data to the DN. The goal of this attack is the exhaustion of the UPF's resources to handle legitimate Session Establishment Requests and Heartbeat Requests. This will potentially hinder the capability of the 5G CN to successfully formulate new PDU sessions between clients and DN.
- Step 2 - Data collection and monitoring: To monitor and analyze the network operation and data traffic, two tools are mainly used:
  - Wireshark, a network protocol analyzer, captures network traffic and generates datasets with the proper features that are exploited to update the ML model stored in BentoML.

- Prometheus, an open-source monitoring and alerting toolkit, acquiring logs from various NFs, such as the AMF, SMF, and UPF.

- Step 3 - ML Models performance improvement: The next step focuses on using the captured datasets to improve the performance of ML models. The datasets generated by Wireshark are used to enhance the detection and classification of the ML models deployed via BentoML. These models are continuously updated with new data collected from the testbed, ensuring their adaptation to evolving threat patterns and achieving high detection rates.

3.9.2 Testbed and Context Environment

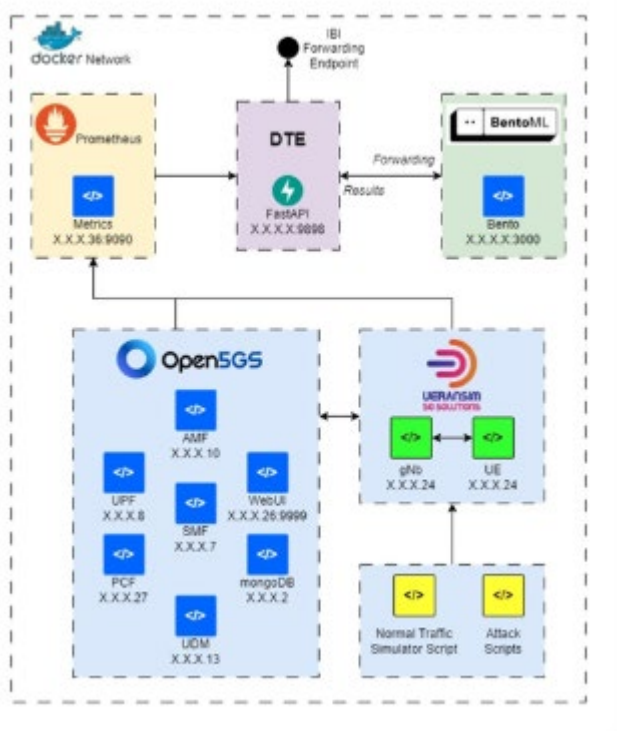

Figure *15: Test-bed architecture for the PFCP attacks demonstrator.*

This demo is executed in a fully operational 5G CN and RAN environment, based on Open5GS and UERANSIM. The system is entirely containerized using Docker, thus ensuring scalability, modularity, and easy deployment. The overall architecture is shown in Figure 15. In addition, this setup includes a user-friendly WebUI for UE registration and configuration, along with advanced network monitoring and ML tools, i.e. Prometheus, Wireshark, and BentoML. In this way, the developed demo environment allows testing, development, and validation of the DEME-DTE and DTE-IBI APIs and interactions, mainly focusing on the use of ML capabilities and the creation of appropriate mitigation and prevention intents. More specifically, below the different tools and frameworks used in the demonstrator and their corresponding role is discussed:

- Open5GS: An open-source project providing a complete 5G core network, including the AMF, SMF, and UPF. Open5GS forms the backbone of our 5G core network, enabling comprehensive network functionalities and services. Open5GS also provides a useful WebUI to register/deregister network users.

- UERANSIM: A robust UE and RAN simulator that interacts with the 5G CN. UERANSIM provides a realistic simulation environment for testing and development, allowing the evaluation of the network's performance and different security measures under varying numbers of UEs.

- Docker: The entire 5G CN and RAN are containerized using Docker, ensuring a modular, scalable, and easily deployable infrastructure. This approach simplifies the management and orchestration of NFs, facilitating updates and maintenance to accommodate more complex attack scenarios and corresponding intents.
- BentoML: This platform is utilized for storing, deploying, and managing ML models. In this demonstrator, BentoML plays a crucial role in the deployment of models designed for attack detection and classification, enhancing the security and reliability of the network. BentoML Utilizes an easy-to-use API in order to serve and output the inference results.
- Prometheus: An open-source monitoring and alerting toolkit, enabling the collection and monitoring of various metrics from all NFs within the 5G core network. Prometheus enables real-time visibility into the network's performance and health, facilitating proactive management and troubleshooting.
- Wireshark: A powerful network protocol analyzer employed for monitoring network traffic, creating datasets, and updating ML models. Wireshark's capabilities enable detailed inspection and analysis of the data traffic in the network, providing valuable insights for improving network performance and security.

3.9.3 Workflow Demo 8

This demo provides an efficient way to simulate PFCP attacks on the 5G core networks and showcases how ML-based attack detection and classification is leveraged by the DTE to create proper intents. At this stage, the following innovations are provided:

- Integration of ML capabilities, deployed through BentoML to detect and classify PFCP attacks in the 5G core network in real-time, enhances the network's responsiveness to threats. At the initial implementation of DTE for IT-1, classification models were used to identify network anomalies. However, deep reinforcement learning (DRL) approaches will also be integrated in DTE in the next iteration, to leverage more dynamic responses and real-time updates on the adopted mitigation strategies.
- Adoption of intent-based networking principles for dynamic and adaptive network configurations to mitigate attacks. In this respect, the DTE creates two types of intents:
  - Mitigative intents, describing objectives that should be accomplished in the short term (e.g., within a few seconds) to reduce the impact of ongoing security incidents. For instance, an intent can recommend to “Mitigate PFCP Session Deletion attack against UPF X” with the following requirements: “SMF downtime less than 1 sec, rate of PFCP Session Deletion requests reaching UPF X equal to 0”. In this case, the mitigation could be supported by configuring a firewall to drop all PFCP Session Deletion requests.
  - Preventive Intents, describing objectives that should be achieved in the longer term to ensure protection against future potential threats. For example, an intent can advise to “Prevent all PFCP-related attacks against UPF with the following requirement: “SMF downtime less than 150 sec” could be supported by checking the integrity of the SMF and, if found compromised, redeploying the SMF, creating a new slice, or deploying a new core network.

The complete workflow of the demonstrator is depicted in

Figure 16 below.

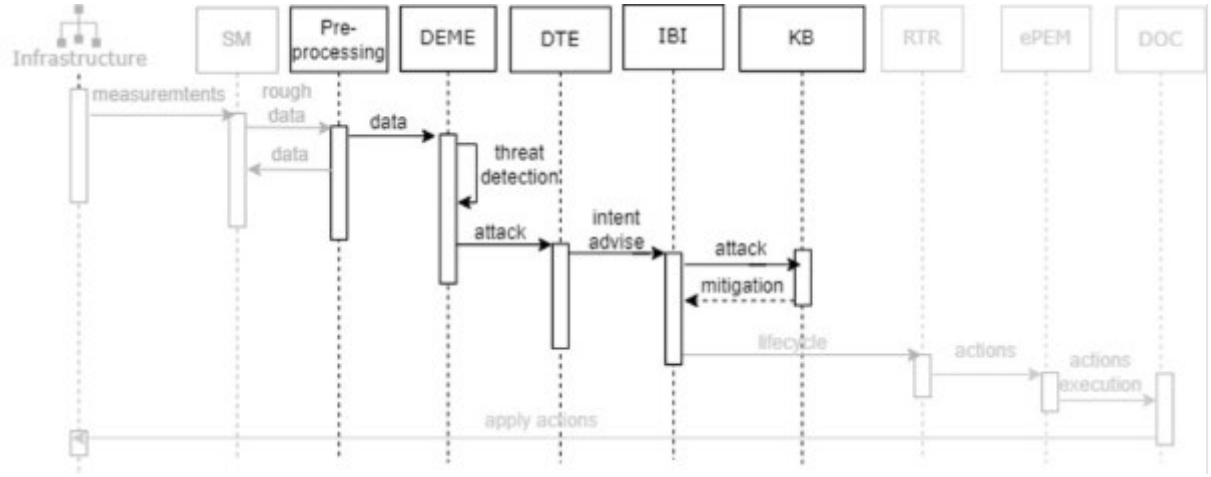

Figure 16: Demonstrator 8 Workflow.

3.9.4 Integration Demo 8

Table 11 presents the integration done in Demonstrator 8. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
2.3	PreProc	DEME	8BELLS	NKUA	JSON over HTTP/REST API.
3.4	DEME	DTE	NKUA	NKUA	PCAP in JSON over HTTP/REST API.
4.9	DTE	IBI	NKUA	TUBS	JSON over HTTP/REST API.
9.14	IBI	CKB	TUBS	MAR	JSON over HTTP/REST API.

Table 11: HORSE components integration endpoints - Demonstrator 8.

3.10 DEMO 9: UC1: Secure Smart LRT Systems (SS-LRT)

3.10.1 Description

UC1 intends to take advantage of using the future 6G networks supported by secure and resilient capabilities provided by HORSE solutions when applicable to Metro systems (Light Rail Transit).

A turnkey LRT solution, similar to those provided by EFACEC, involves the integration of several subsystems such as Public Information, network and communications, video surveillance, AVLS – Automatic Vehicle Location, Regulation, SCADA, Signalling, etc.

The network services in such systems play a key role and malfunctions in the network must be avoided since it can impact on the normal Metro's operation leading to interruptions and loss of Quality of Service that may result in penalties.

In this Use Case testbed, the main goal consists of simulating congestion in the network by amplifying the volume of traffic through Amplification Attacks (DNS attack), verifying the impact in the Metro's subsystems as well validating the HORSE policies and mitigation actions.

Therefore, a DNS attack similar to that described in section 3.4 will be generated for Demo 9.

3.10.2 Testbed and Context Environment

Related to Use Case 1, it was defined as two testbeds. The first one involves the integration of UMU environment and EFACEC and the second one involves the integration of UPC and EFACEC.

The following figures show the reference architecture for each testbed.

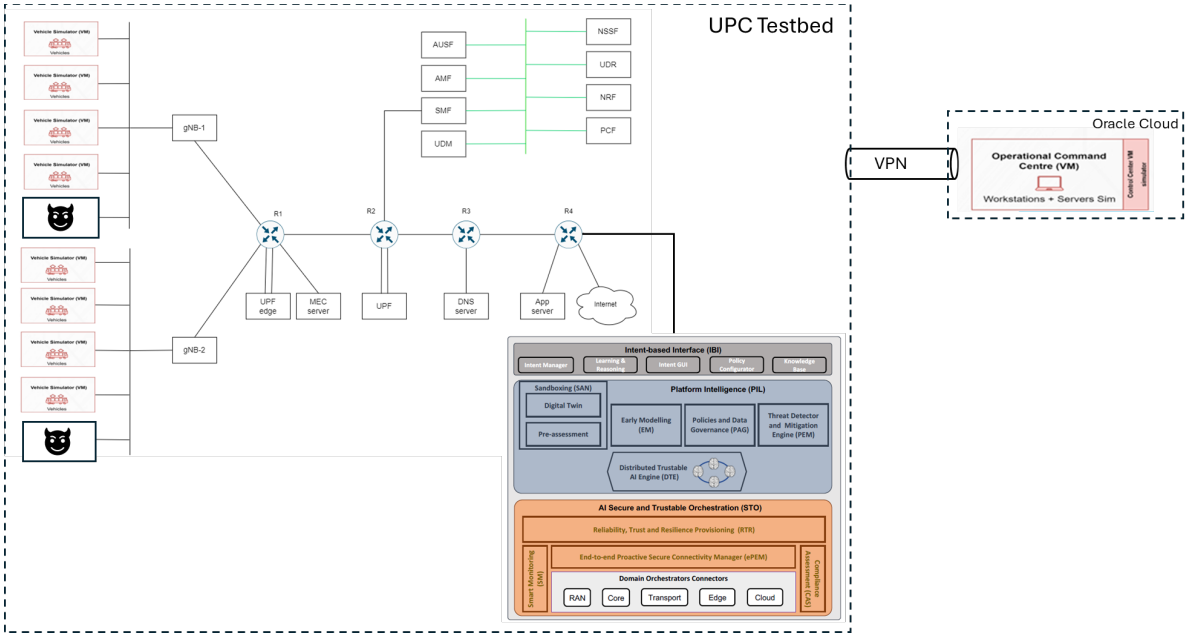

Figure 17: UC1 - UPC Testbed

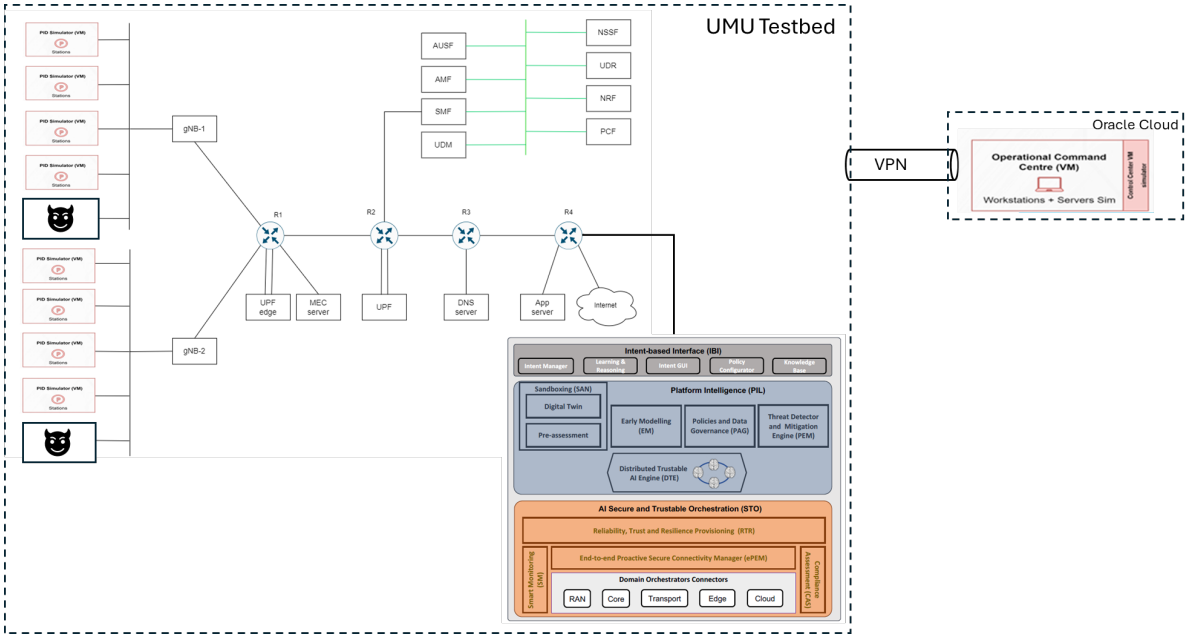

Figure 18: UC-1 UMU Testbed.

The UPC and UMU testbed support all the network infrastructure as well as the HORSE components. Additionally, it provides communication to EFACEC Labs (Cloud and premises) where a representative scenario of a LRT system is deployed. Typically, a LRT system involves an Operational Command Centre (OCC) is used, with all application servers, database servers and workstations for the operation. It involves the tramstops/stations and the Metro line that supports the trams/vehicles. Concerning demonstration 9, the OCC is deployed in EFACEC labs, the tramstop components are deployed in UMU/UPC, and the Vehicle's simulator can be also deployed in both UMU/UPC environments. In this scenario, the OCC and the tramstops/metro Line are connected by the 5G/6G UMU/UPC infrastructure. The goal of this demo is to measure the impact in the LRT operation, when network congestion occurs, caused by an amplification attack, disturbing communication between the OCC and the tramstops and the vehicles.

Three LRT subsystems were defined to be deployed in the tramstops/Line: i)Video Surveillance ii) Passenger Information and Automatic Vehicle Localization (AVLS) and for this purpose EFACEC provides a simulator system of PIDs (Passenger Information Displays) and a simulator system of Vehicles.

Next Figure 19 shows the integration scenario for demonstration 9, illustrating the communication between the OCC and the tramstops/Line.

Figure 19 also illustrates the virtualization tools used to support the VM (Virtual Machines) both at UMU and EFACEC. The environment for UPC use an architecture similar to UMU.

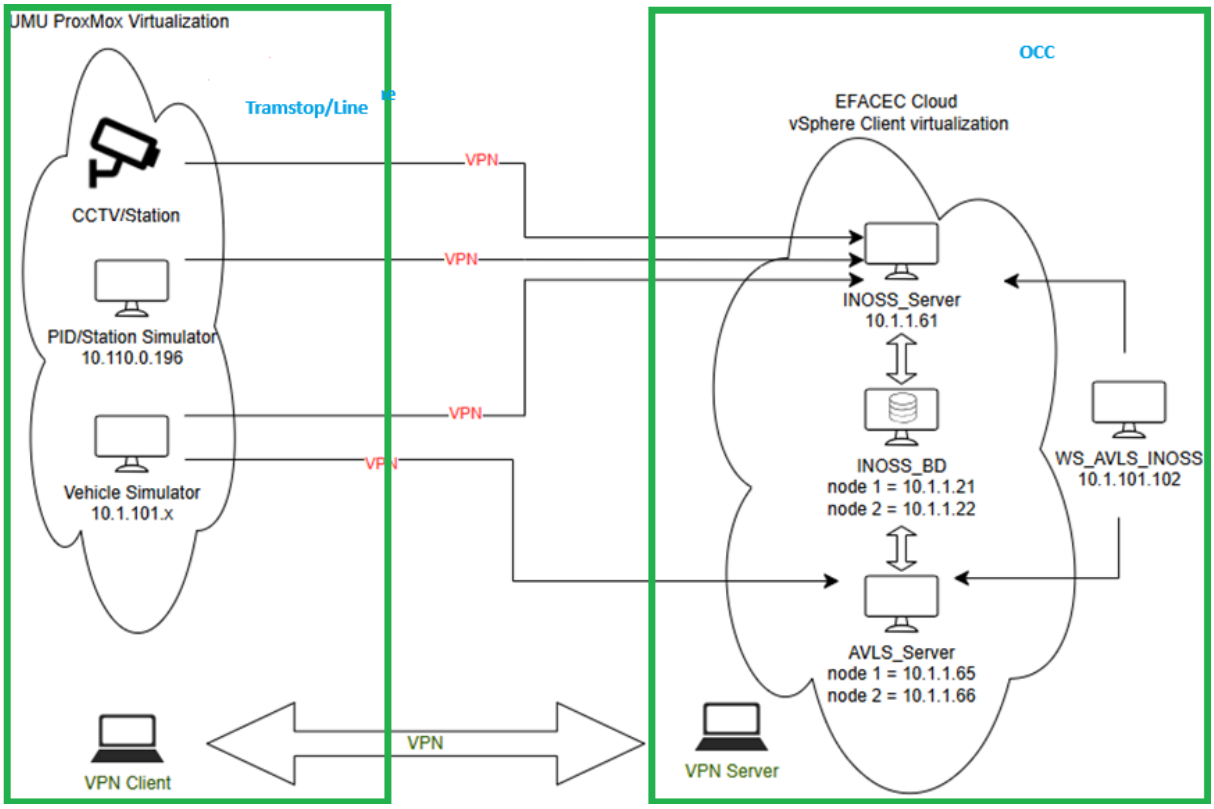

Figure 19: OCC/Tramstops-Line integration

For the UMU testbed, demonstration 9 will emphasize the integration between the OCC and the Passenger Information system and for the UPC it will emphasize the integration between the OCC and the AVLS system.

Figure 20 illustrates the Metro Line synoptic representation in OCC, using a Vehicle simulation to allow the visualization of the vehicles performing their metro services. In the figure, the vehicles are shown in green or red, representing that they are due or delayed. As explained previously, communication between the OCC and the vehicles is performed over the network, and the attack will impact the operation. In the presence of a network disturbance, the vehicles will be represented and yellow meaning that they are not able to communicate with the OCC and they can even be removed from the synoptic if the congestion persists. The localization of the trams is critical for the operation and demonstration 9 intends to allow the attack, mitigate the attack, and contribute to its resolution.

Figure 20: OCC synoptic – Vehicle localization

Figure 21 shows the PID simulator. The simulator is a software application that emulates a tramstop display, allowing to see text messages sent by the OCC operator or sent automatically by the OCC with the estimated arrivals times at the tramstop (forecast information).

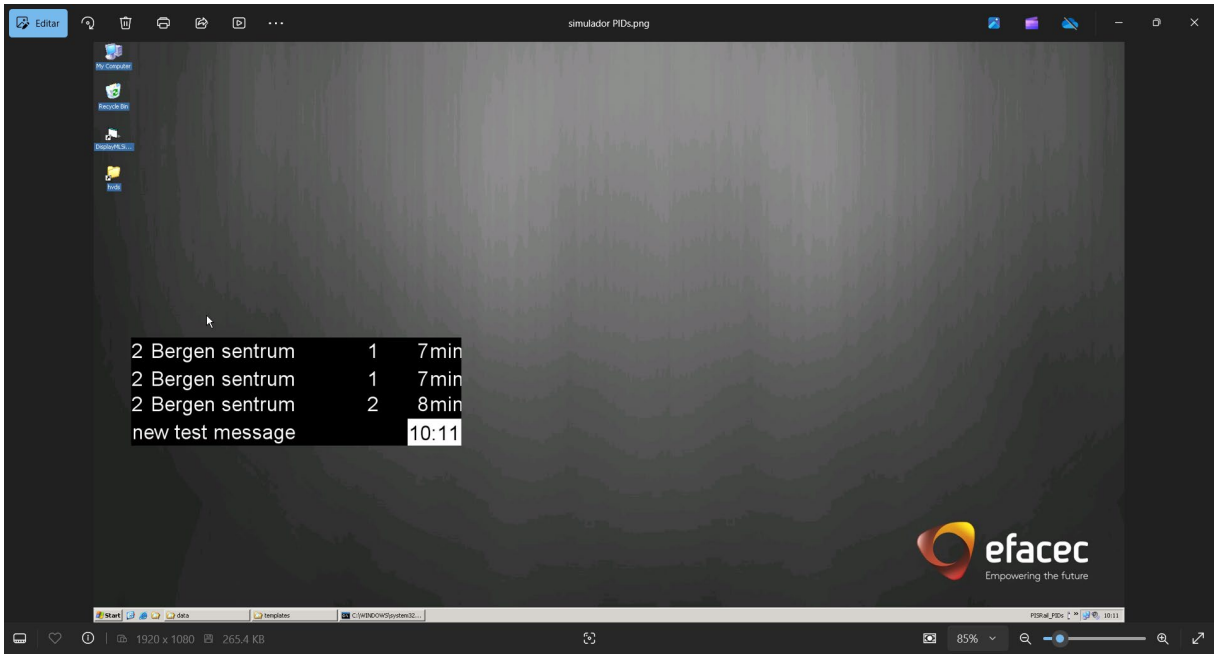

Figure 21: PID simulator.

Figure 22 shows the Passenger Information at OCC

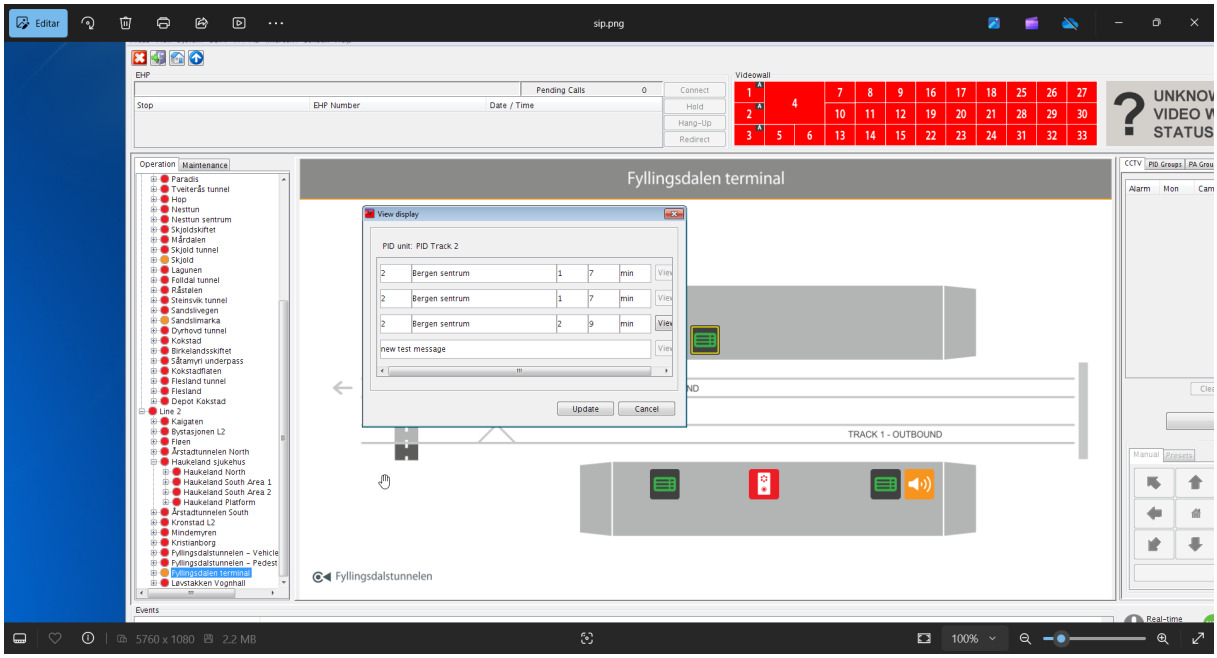

Figure 22: OCC – Passenger Information.

In the presence of disturbance in the network, the communication between the OCC and the Display will be affected meaning after a long period of interruption the forecast information will be removed from the equipment and the OCC operator will not be able to send text messages to the tramstop.

3.10.3 Workflow Demo 9

Demonstrator 9 focuses on the DNS Amplification Attack, Figure 23 illustrates the threat detection workflow that was considered. Notably, Demonstrator 9 highlights the critical roles of both IBI and CAS. Additional steps were introduced into the workflow to address policy management (via CAS) and the generation of external alarms, which are intended to be received and processed by the OCC.

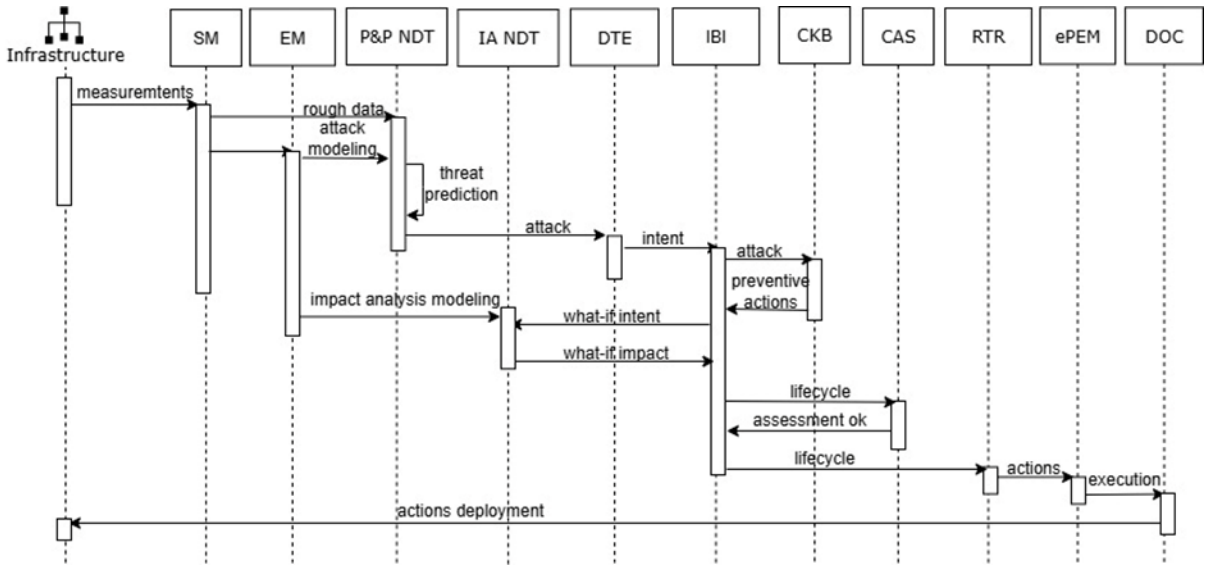

Figure 23: Demonstrator 9 Workflow.

3.10.4 Integration Demo 9

Table 12 presents the integration done in Demonstrator 9. It includes the integration endpoints of the different components involved, the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
1.5	INFR	SM	UMU UPC	STS	PCAP file in Shared Folder.
1.5	SM	P&P NDT	STS	CNIT	Topology in API REST.
7.5	EM	P&P NDT	UPC	CNIT	Threat Model in API REST.
5.4	P&P NDT	DTE	CNIT	NKUA	Detected Threat in API REST.
7.6	EM	IA-NDT	UPC	TID/UMU	Threat Model om API REST.
4.10	DTE	IBI	NKUA	TUBS	Prevention Intent in JSON over RESTful API.
9.6	IBI	IA-NDT	TUBS	TID/UMU	What-if in API REST.
6.9	IA-NDT	IBI	TID/UMU	TUBS	What-if result in API REST.
9.13	IBI	CAS	TUBS	STS	Result of the validation of the actions against CAS policies.
9.10	IBI	RTR	TUBS	8BELLS	List of actions to be enforced in JSON format sent over HTTP protocol using REST APIs.
10.11	RTR	ePEM	8BELLS	CNIT	List of commands for each action to be enforced in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
11.12	ePEM	DOC	CNIT	8BELLS	List of command for each action not enforced directly by ePEM in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
12.15	DOC	INFR	8BELLS	UMU UPC	Mitigation actions JSON using REST APIs

Table 12: HORSE components integration endpoints - Demonstrator 9.

3.11 DEMO 10: UC2: Remote Rendering to Power XR Industrial (R22XRI)

3.11.1 Description

The HORSE project is pioneering the integration of resilient, secure 6G infrastructure into Extended Reality (XR) environments, transforming the way industrial collaboration and design are performed at scale. A key demonstration of this potential is the Remote Rendering for XR Industrial use case, featuring a real-time, multi-user collaboration scenario powered by Hololight SPACE.

Hololight SPACE is a leading XR engineering platform that enables engineers and designers to interact with original 3 dimensional computer aided design (3D CAD) data in immersive Augmented Reality (AR) and Virtual Reality (VR) environments. Its underlying Hololight Stream technology offloads heavy rendering processes to local servers or cloud infrastructure, ensuring high-fidelity performance across a range of XR devices. However, this architecture also presents challenges: as the platform scales and handles increasingly sensitive or mission-critical data, it becomes more vulnerable to network disruptions and cyber threats.

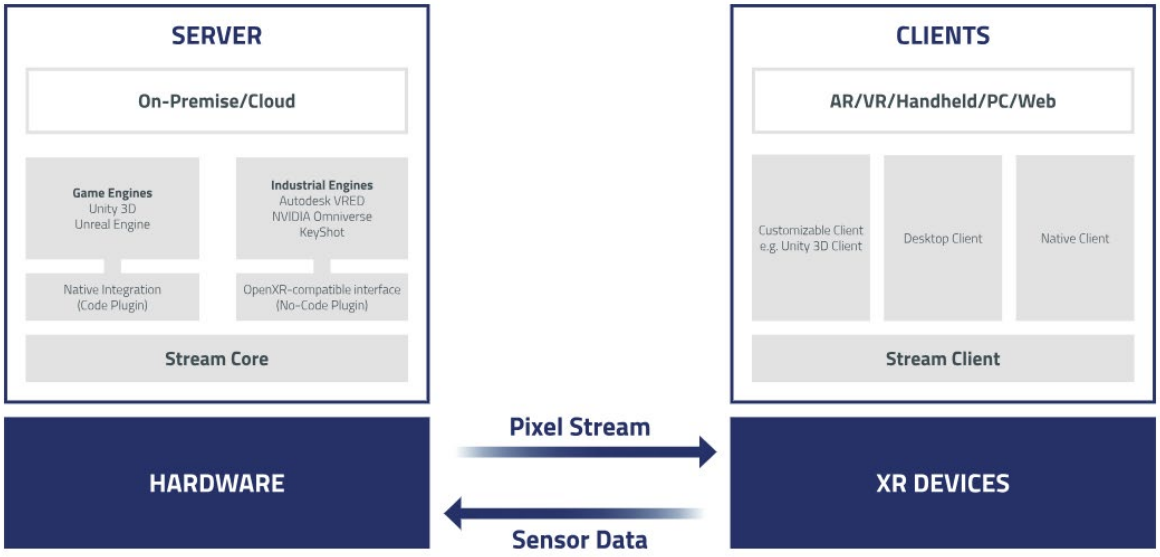

Figure 24: Hololight Stream workflow. Hololight Stream offloads the rendering of the application Hololight Space to a server. The data from the server is pixel streamed to the AR Glasses.

To address this, HORSE integrates advanced security and monitoring components directly into the network layer. In this use case, two remote users collaborate in Hololight SPACE, manipulating shared 3D models and communicating via avatars in real time. The session demands ultra-low latency and high bandwidth—conditions that are severely impacted in the presence of cyberattacks such as packet loss, latency spikes, or suspicious access to network elements like the Network Exposure Function (NEF). HORSE actively secures the XR session through:

1. Smart Monitoring that continuously logs every NEF API request to detect anomalies,
2. Intelligent Preprocessing which aggregates and analyzes access patterns from IP addresses,
3. The DEME Engine, which performs real-time behavioral analysis to flag threats,
4. Swift Firewall Action, which dynamically blocks and unblocks risky sources based on real-time assessments.

This real-time threat detection and mitigation ensures the session remains uninterrupted, with smooth avatar motion and synchronized scene updates, even under simulated attacks. The system adapts dynamically, minimizing false positives while protecting users and data integrity. With HORSE, Hololight demonstrates not only a secure and stable XR streaming solution, but also the future of industrial XR collaboration at scale. This combination of remote rendering and AI-enhanced network defense transforms how industries can safely deploy XR technologies across distributed teams, laying the groundwork for resilient, high-performance 6G services.

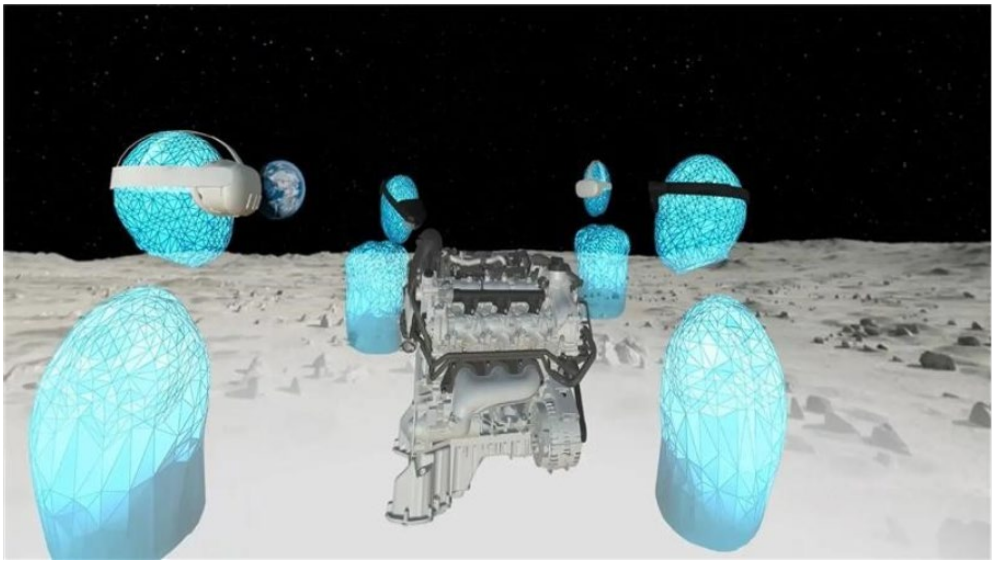

Figure 25: Multiuser VR session. Multiple users collaborating in a Virtual Reality session to inspect a 3D CAD model of an engine.

Figure 26: Multi-user AR session. Two users collaborating in an Augmented Reality session for virtual prototyping.

3.11.2 Testbed and Context Environment

The demo wil run on the CNIT infrastructure shown in Figure 27. There are 10 different UEs used to generate traffic related to the normal and attack behavior. The 5G Core is a Free5GC deployment on a k8s cluster.

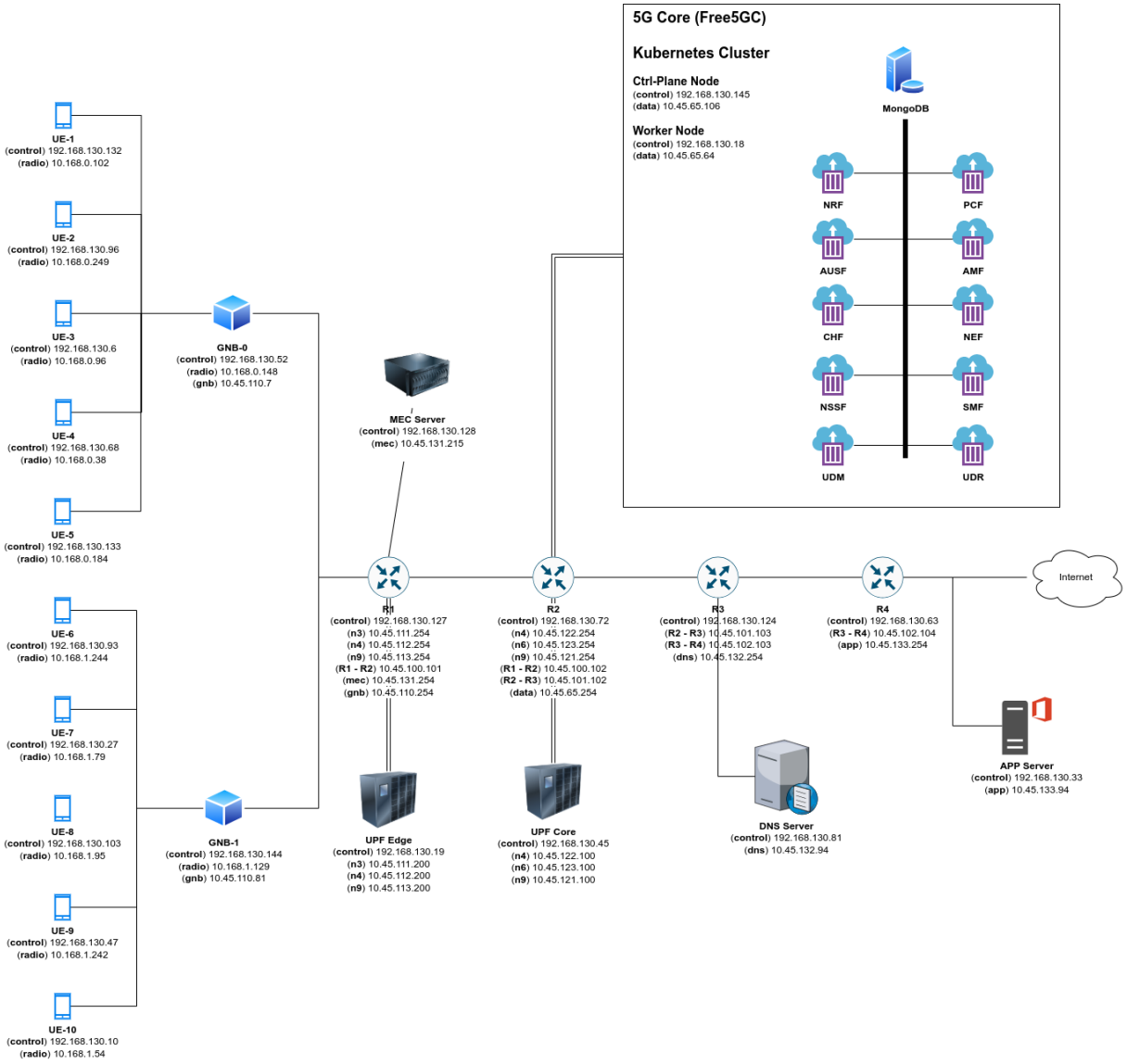

Figure 27: CNIT infrastructure for Demo 10.

3.11.3 Workflow Demo 10 - Threat Prediction

In this demo we have two different types of attacks: *(i)* API Exposure (APIExp) and *(ii)* DDoS for network congestion. For this reason, we must consider two different workflows:

1. Threat (APIExp/DDoS) prediction
2. APIExp and DDoS detection

The first workflow is the first phase of each of the two types of attacks (APIExp and DDoS). This workflow is shown in Figure 28. The output of this workflow is to identify and assess policies (send in the *lifecycle* message by IBI) for a possible threat (in this case APIExp or DDoS).

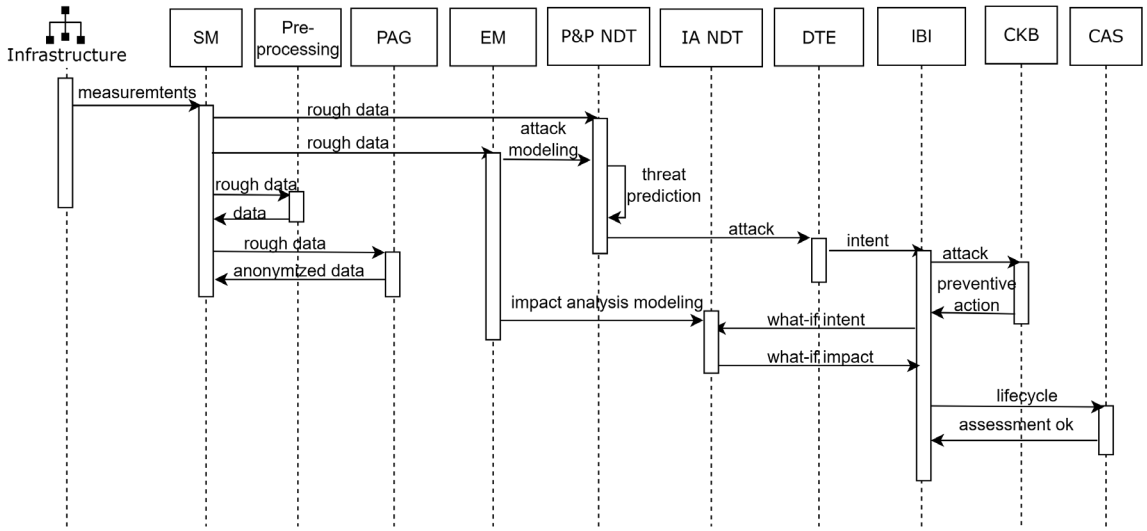

Figure 28: Workflow for threat prediction Demo 10.

3.11.4 Integration Demo 10 - Threat Prediction

Table 13 presents the integration done in Demonstrator 10 for the prediction of threats. It includes the integration endpoints of the different components involved threat prediction as described in Figure 28, as well as the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
15.1	INFR	SM	CNIT	STS	Log data for APIExp PCAP from Shared Folder for DDoS
1.7	SM	EM	STS	UPC	Log data for APIExp PCAP from Shared Folder for DDoS
1.5	SM	P&P NDT	STS	CNIT	Log data for APIExp PCAP from Shared Folder for DDoS
1.2	SM	PreProc	STS	8BELLS	Log data for APIExp PCAP from Shared Folder for DDoS
2.1	PreProc	SM	8BELLS	STS	JSON formatted ES record, sent over HTTP protocol using REST APIs
1.8	SM	PAG	STS	S5	JSON format from ES index Log data for APIExp PCAP data for DDoS
8.1	PAG	SM	S5	STS	JSON format from ES index Log data for APIExp PCAP data for DDoS
5.4	P&P NDT	DTE	CNIT	NKUA	Detected Threat - API REST

7.5	EM	P&P NDT	UPC	CNIT	Threat Model over API REST.
7.6	EM	IA-NDT	UPC	CNIT	Threat Model over API REST.
4.9	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigate an attack in JSON format sent over HTTP protocol using REST APIs.
9.14	IBI	CKB	TUBS	MAR	Description of the attack in JSON format sent over HTTP protocol using REST APIs.
14.9	CKB	IBI	MAR	TUBS	List of possible preventive actions in JSON format sent over HTTP protocol using REST APIs.
9.6	IBI	IA-NDT	TUBS	CNIT	Attack and prevention or mitigation action to be tested by the digital twin in JSON format sent over HTTP
6.9	IA-NDT	IBI	TID/UMU	TUBS	Result the test with collected KPIs in the digital twin encoded in JSON format sent over HTTP.
9.13	IBI	CAS	TUBS	STS	Actions to be validated by CAS encoded in JSON sent over HTTP.
13.9	CAS	IBI	STS	TUBS	JSON formatted answer on the action against CAS policies.

Table 13: HORSE framework interactions endpoints - Demonstrator 10 – Threat Prediction.

3.11.5 Workflow Demo 10 - Threat Detection

For UC2, Remote Rendering to Power XR Industrial, also demonstrated the detection of threats. The detection path for the APIExp and DDoS attacks is shown in *Figure 29*. It uses the policies identified in the prediction part. Obviously, even with a previously identified and assessed policies, it is necessary to access again the policies by CAS considering that the status of the HORSE framework and the underlying infrastructure can be changed from the prediction execution to the detection one.

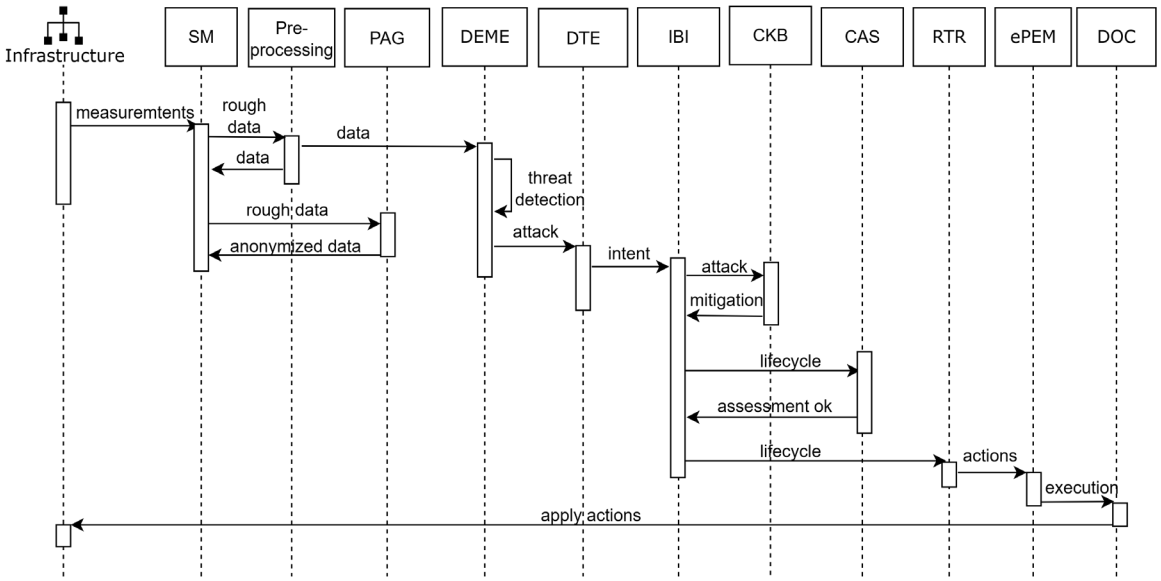

Figure 29. Workflow for APIExp and DDoS detection attacks.

3.11.6 Integration Demo 10 - Threat Detection

Table 14 presents the integration done in Demonstrator 10 for the detection of threats. It includes the integration endpoints of the different components involved threat detection as described in Figure 29, as well as the partner responsible for each component, and the data types and protocols used for information exchange between the HORSE framework components participating in this demonstrator.

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
15.1	INFR	SM	CNIT	STS	Log data for APIExp PCAP from Shared Folder for DDoS
1.2	SM	PreProc	STS	8BELLS	JSON format from ES index. Log data for APIExp PCAP data for DDoS
2.1	PreProc	SM	8BELLS	STS	JSON formatted ES record, sent over HTTP protocol using REST APIs
1.8	SM	PAG	STS	S5	JSON format from ES index. Log data for APIExp PCAP data for DDoS
8.1	PAG	SM	S5	STS	JSON format from ES index. Log data for APIExp PCAP data for DDoS
2.3	PreProc	DEME	8BELLS	ETI	JSON data with amount of … received by the target in each interval. API Request for APIEpx Packets for DDoS

Integration Endpoints			Responsible For		Data Type & Protocol
ID	CompA	CompB	CompA	CompB	
3.4	DEME	DTE	ETI	NKUA	JSON data with the accuracy level of an attack (from 0 = no attack to 1 = attack) for the target in each interval.
4.9	DTE	IBI	NKUA	TUBS	Intent to prevent/mitigate an attack in JSON format sent over HTTP protocol using REST APIs.
9.14	IBI	CKB	TUBS	MAR	Description of the attack in JSON format sent over HTTP protocol using REST APIs.
14.9	CKB	IBI	MAR	TUBS	List of possible mitigation actions in JSON format sent over HTTP protocol using REST APIs.
9.13	IBI	CAS	TUBS	STS	List of mitigation actions to be validated by CAS encoded in JSON sent over HTTP.
13.9	CAS	IBI	STS	TUBS	JSON formatted answer on the action (OK/Partially OK/Not OK).
9.10	IBI	RTR	TUBS	8BELLS	List of actions to be enforced in JSON format sent over HTTP protocol using REST APIs.
10.11	RTR	ePEM	8BELLS	CNIT	List of commands for each action to be enforced in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
11.12	ePEM	DOC	CNIT	8BELLS	List of command for each action not enforced directly by ePEM in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.
12.15	DOC	INFR	8BELLS	CNIT	Action applied to the CNIT testbed.

Table 14: HORSE components integration endpoints – Demonstrator 10 – Threat Detection Workflow.

4 Final HORSE Framework Validation and Testing

To test the interactions between both, the HORSE components, and the UC testbeds and the HORSE framework, validation tests have been developed for each interaction pair. Specifically, a test case has been defined for every interaction pair, including the expected results of the interaction. Each test case contains the following information:

- Unique identifier, as specified in the HORSE integration matrix;
- CompA and CompB: Names of the HORSE components involved;
- Test description;
- Test Result;
- Current implementation status for the test.

The technical validation tests for each demonstrator are summarized in Table 15 to Table 27. These tables present the actual tests conducted for each interaction, along with their corresponding results and current testing status.

Following this, each of the ten HORSE demonstrators involved in the validation process is described, along with the related interfaces and corresponding validation tests.

4.1 Demo 0

4.1.1 Demo 0 - Threat Detection

Table 15 summarizes the validation tests conducted for the Threat Detection Workflow in Demonstrator 0. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.1	INFR	SM	Read PCAP files and translate them to JSON.	Saved data in ES.	Passed
1.2	SM	PreProc	Provide the PCAP data in JSON format	Successfully retrieved and answered to PreProc.	Passed
8.1	PAG	SM	Check the PCAP file using Kibana UI.	The IP addresses and the payload are anonymized.	Passed
2.3	PreProc	DEME	Provide the extracted and pre-processed PCAP data in JSON format.	Successfully received by DEME.	Passed

3.4	DEME	DTE	Provide the threat detection and relative confidence indication in JSON format.	Successfully received by DTE.	Passed
4.9	DTE	IBI	Provide an intent to mitigate the detected thread.	Received by IBI.	Passed
9.14	IBI	CKB	Request the list of possible mitigation for detected/predicted attack.	Request for the list of mitigations received.	Passed
14.9	CKB	IBI	Provide the list of possible mitigation actions for the requested attack.	List of mitigation received by IBI.	Passed
9.13	IBI	CAS	Validate the proposed actions with CAS before being sent for enforcement.	Request received by CAS.	Passed
13.9	CAS	IBI	Provide an answer on the query based on the proposed action from IBI.	Evaluated the action based on the status of HORSE,	Passed
9.10	IBI	RTR	Sent the list of actions and configuration to be enforced in the infrastructure.	Mitigation actions and configurations received by RTR.	Passed
10.11	RTR	ePEM	Sent the list of commands (with Ansible playbooks) for each action to be enforced.	List of commands received by ePEM,	Passed
11.12	ePEM	DOC	Sent the list of commands (with Ansible playbooks) for each action not enforced directly by ePEM.	List of commands received by DOC	Passed
12.15	DOC	INFR	Execute the list of commands.	List of commands executed,	Passed

Table 15: HORSE framework interactions testing – Demonstrator 0 – Threat detection workflow

4.1.2 Demo 0 - Threat Prediction

Table 16 summarizes the validation tests conducted for the Threat Prediction Workflow in Demonstrator 0. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.1	INFR	SM	Read PCAP files and translate them to JSON.	Save data in ES.	Passed
1. 5	SM	P&P NDT	SM sends topology to the P&P NDT.	Topology stored in the P&P NDT	Passed
1.7	SM	EM	SM sends PCAP files to EM.	Stored data in EM.	Passed
1.2	SM	PreProc	Read Index records and translate them to specific JSON schema.	Read index and translated data successfully	Passed
2.1	PreProc	SM	Send JSON formatted record to SM for storage	stored data to the SM index successfully	Passed
8.1	PAG	SM	Check the PCAP file using Kibana UI.	The IP addresses and the payload are anonymized.	Passed
7.5	EM	P&P NDT	Check that the required threat model instance is transferred to the P&P DT.	The threat model instance is available at the P&P DT.	Passed
5.4	P&P NDT	DTE	Analyzes the PCAP file and performs attack prediction, then informs the DTE.	Successfully received by DTE.	Passed
4.9	DTE	IBI	Provide an intent to mitigate the detected thread.	Received by IBI.	Passed

9.14	IBI	CKB	Request the list of possible mitigations for detected/predicted attack.	Request for the list of mitigations received.	Passed
14.9	CKB	IBI	Provide the list of possible mitigation actions for the requested attack.	List of mitigation received by IBI.	Passed
9.6	IBI	IA NDT	Request to simulate an attack and a mitigation action.	Description of the attack, the prevention action and KPIs to be collected.	Passed
6.9	IA NDT	IBI	Provide the results of the simulation.	Values or the measured KPIs.	Passed
9.13	IBI	CAS	Validate the proposed actions with CAS before being sent for enforcement.	Request received by CAS.	Passed
9.10	IBI	RTR	Sent the list of actions and configuration to be enforced in the infrastructure.	Mitigation actions and configurations received by RTR.	Passed
10.11	RTR	ePEM	Sent the list of commands (with Ansible playbooks) for each action to be enforced.	List of commands received by ePEM.	Passed
11.12	ePEM	DOC	Sent the list of commands (with Ansible playbooks) for each action not enforced directly by ePEM.	List of commands received by DOC.	Passed
12.15	DOC	INFR	Execute the list of commands.	List of commands executed.	Passed

Table 16: HORSE framework interactions testing – Demonstrator 0 – Threat prediction workflow.

4.2 Demo 1

Table 17 summarizes the validation tests conducted in Demonstrator 1. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.1	INFR	SM	Read PCAP files and translate them to JSON.	Save data in ElasticSearch	Passed
1.2	SM	PrePRoc	Provide the PCAP data in a JSON format.	Successfully retrieved and answered to PreProc.	Passed
2.3	PreProc	DEME	Check that the extracted and pre-processed PCAP data in JSON format is transferred to the DEME.	Required extracted and pre-processed data available at DEME.	Passed
5.4	DEME	DTE	Check that the detection and relative confidence indication in JSON format is transferred to DTE.	Required detection and relative confidence indication available at DTE.	Passed
4.9	DTE	IBI	Provide an intent to mitigate the detected thread.	Intent received by IBI.	Passed

Table 17: HORSE framework interactions testing - Demonstrator 1.

4.3 Demo 2

Table 18 summarizes the validation tests conducted in Demonstrator 2. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
1.5	SM	P&P NDT	SM sends topology to the P&P NDT.	P&P NDT can emulate the topology.	Passed
1.6	SM	IA NDT	SM sends topology to the IA NDT.	IA NDT can emulate topology.	Passed

Interactions			Test Description	Result	Status
ID	CompA	CompB			
1.5	EM	P&P NDT	EM sends threat model to the P&P NDT.	P&P NDT can detect the attack.	Passed
7.6	EM	IA NDT	EM sends threat model to simulate attack (DDoS downlink) over IA-NDT.	IA NDT can simulate the attack.	Passed
5.4	P&P NDT	DTE	P&P NDT sends the predicted attack to DTE.	Successfully received by DTE.	Passed
4.9	DTE	IBI	DTE ask IBI to generate a mitigation over IA-NDT.	DTE can trigger IBI to generate a mitigation action.	Passed
9.6	IBI	IA-NDT	IBI sends an Intent with a mitigation action (QoS or Filtering) over an IA-NDT component. Also, it asks for a certain metric to be collected.	IA NDT can test the mitigation.	Passed
6.9	IA-NDT	IBI	IA-NDT sends the results of the collected metrics to the IBI for analysis.	IBI receives the impact of the attack.	Passed

Table 18: HORSE framework interactions testing - Demonstrator 2.

4.4 Demo 3

Table 19 summarizes the validation tests conducted in Demonstrator 3. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
1.5	SM	P&P NDT	SM sends topology to the P&P NDT.	P&P NDT can emulate the topology.	Passed
1.6	SM	IA NDT	SM sends topology to the IA NDT.	IA NDT can emulate topology.	Passed
1.5	EM	P&P NDT	EM sends threat model to the P&P NDT.	P&P NDT can detect the attack.	Passed

Interactions			Test Description	Result	Status
ID	CompA	CompB			
7.6	EM	IA NDT	EM sends threat model to simulate attack (DDoS downlink) over IA-NDT.	IA NDT can simulate the attack.	Passed
5.4	P&P NDT	DTE	P&P NDT sends the predicted attack to DTE.	DTE can trigger IBI to generate a mitigation action.	Passed
4.9	DTE	IBI	DTE asks IBI to generate a mitigation over IA-NDT.	IBI receives the intent.	Passed
9.6	IBI	IA-NDT	IBI sends an Intent with a mitigation action (QoS or Filtering) over an IA-NDT component. Also, it asks for a certain metric to be collected.	IA NDT can test the mitigation.	Passed
6.9	IA-NDT	IBI	IA-NDT sends the results of the collected metrics to the IBI for analysis.	IBI receives the impact of the attack.	Passed
9.13	IBI	CAS	IBI sends the set of configurations to prevent the predicted attach for policy checking by CAS.	CAS run the validation and return the results to IBI.	Passed

Table 19: HORSE framework interactions testing - Demonstrator 3.

4.5 Demo 4

Table 20 summarizes the validation tests conducted in Demonstrator 4. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.5	Attacker script	P&P NDT	Detection of anomalies or hashing fail.	Anomaly detected.	Passed
5.4	P&P NDT	DTE	Detection of anomalies or hashing fail.	Anomaly signaled.	Passed

Table 20: HORSE framework interactions testing - Demonstrator 4

4.6 Demo 5

Table 21 summarizes the validation tests conducted in Demonstrator 5. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.15	INFR (UPC)	INFR (CNIT)	Sent the PCAP files from INFR (UPC) to INFR (CNIT).	Save the PCAP files in INFR (CNIT).	Passed
15.1	INFR (CNIT)	SM	Read PCAP files and transform them to JSON format and save them.	Save the PCAP file data in a json format in ES.	Passed
1.2	SM	PreProc	Send the correct PCAP data to PreProc a JSON format.	Successfully retrieved and answered to PreProc.	Passed
2.3	PreProc	DEME	Check that the extracted and pre-processed PCAP data in JSON format is transferred to the DEME.	Required extracted and pre-processed data available at DEME.	Passed
3.4	DEME	DTE	Check that the detection and relative confidence indication in JSON format is transferred to DTE.	Required detection and relative confidence indication available at DTE.	Passed
4.9	DTE	IBI	Provide an intent to mitigate the detected thread.	Intent received by IBI.	Passed
9.14	IBI	CKB	Request the list of possible mitigation for detected/predicted attack.	Request for the list of mitigations received.	Passed
14.9	CKB	IBI	Provide the list of possible mitigation actions for the requested attack.	List of mitigation received by IBI.	Passed
9.13	IBI	CAS	Validate the proposed actions with CAS before being sent to enforcement.	Request received by CAS.	Passed
13.9	CAS	IBI	Provide an answer to the query based on the proposed action from IBI.	Evaluated the action based on the status of HORSE.	Passed
9.10	IBI	RTR	Sent the list of actions and configuration to be enforced in the infrastructure.	Mitigation actions and configurations	Passed

Interactions			Test Description	Result	Status
ID	CompA	CompB			
				received by RTR.	
10.11	RTR	ePEM	Sent the list of commands (with Ansible playbooks) for each action to be enforced.	List of commands received by ePEM.	Passed
11.12	ePEM	DOC	Sent the list of commands (with Ansible playbooks) for each action not enforced directly by ePEM.	List of commands received by DOC.	Passed
12.15	DOC	INFR (CNIT)	Execute the list of commands.	List of commands executed.	Passed
12.15	DOC	INFR (UPC)	Execute the list of commands.	List of commands executed.	Passed

Table 21: HORSE framework interactions testing - Demonstrator 5.

4.7 Demo 6

Table 22 summarizes the validation tests conducted in Demonstrator 6. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
3.4	DEME	DTE	Provide the threat detection and relative confidence indication in JSON format.	Successfully received by DTE.	Passed
4.9	DTE	IBI	Generate the mitigation intent based on the information received from DTE.	IBI receives the intent.	Passed
9.13	IBI	CAS	Generate the list of mitigation actions to be assessed by CAS.	CAS receives the list of mitigation actions.	Passed
13.3	CAS	DEME	Request the list of ongoing/current attacks.	DEME receives the request.	Passed

3.13	DEME	CAS	DEME answers with the list of ongoing/current attacks.	CAS receives the response.	Passed
13.9	CAS	IBI	Validation result of the mitigation actions.	IBI receives the validation result.	Passed

Table 22: HORSE framework interactions testing - Demonstrator 6.

4.8 Demo 7

Table 23 summarizes the validation tests conducted in Demonstrator 7. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.1	INFR	SM	Read PCAP files and translate them to JSON.	Save data in ElasticSearch	Passed
1.2	SM	PreProc	Provide the PCAP data in a JSON format.	Successfully retrieved and answered to PreProc.	Passed
2.3	PreProc	DEME	Check that the extracted and pre-processed PCAP data in JSON format is transferred to the DEME.	Required extracted and pre-processed data available at DEME.	Passed
5.4	DEME	DTE	Check that the detection and relative confidence indication in JSON format is transferred to DTE.	Required detection and relative confidence indication available at DTE.	Passed
4.9	DTE	IBI	Provide an intent to mitigate the detected thread.	Intent received by IBI.	Passed
9.14	IBI	CKB	Request the list of possible mitigations for detected/predicted attack.	Request for the list of mitigations received.	Passed
14.9	CKB	IBI	Provide the list of possible mitigation actions for the requested attack.	List of mitigation received by IBI.	Passed
9.10	IBI	RTR	Sent the list of actions and configuration to be	Mitigation actions and	Passed

			enforced in the infrastructure.	configurations received by RTR.	

Table 23: HORSE framework interactions testing - Demonstrator 7.

4.9 Demo 8

Table 24 summarizes the validation tests conducted in Demonstrator 8. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
2.3	PreProc	DEME	Check that the collected data (PCAP files and logs) are processed and transferred to the DEME component.	The collected and processed data (PCAP files and logs) is available at the DEME component.	Passed
3.4	DEME	DTE	Check if the DTE correctly receives the DEME's detection data, being sent over the REST API.	PCAP, JSON over HTTP/REST API.	Passed
4.9	DTE	IBI	Check if the IBI can correctly receive security intents regarding the detected attack from DTE.	An HTTP response with code 200 to acknowledge the reception of the intent.	Passed
9.14	IBI	KB	Check if the IBI can correctly receive mitigation lists given the detected attack.	JSON over HTTP/REST API.	Passed

Table 24: HORSE framework interactions testing – Demonstrator 8

4.10 Demo 9 – UC1

Table 25 summarizes the validation tests conducted in Demonstrator 9 related to UC1. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
12.1	INFR	SM	Read PCAP files and translate them to JSON.	Saved data in ES.	Passed
1.5	SM	P&P NDT	SM sends topology to the P&P NDT.	P&P NDT can emulate the topology.	Passed
7.5	EM	P&P NDT	EM sends threat model to the P&P NDT.	P&P NDT can detect the attack.	Passed
5.4	P&P NDT	DTE	P&P NDT sends the predicted attack to DTE.	DTE can trigger IBI to generate a mitigation action.	Passed
7.6	EM	IA-NDT	EM sends threat model to simulate attack (DDoS downlink) over IA-NDT.	IA NDT can simulate the attack.	Passed
4.10	DTE	IBI	DTE asks IBI to generate a mitigation over IA-NDT.	IBI receives the intent.	Passed
9.6	IBI	IA-NDT	IBI sends an Intent with a mitigation action (QoS or Filtering) over an IA-NDT component. Also, it asks for a certain metric to be collected.	IA NDT can test the mitigation.	Passed
6.9	IA-NDT	IBI	IA-NDT sends the results of the collected metrics to the IBI for analysis.	IBI receives the impact of the attack.	Passed
9.13	IBI	CAS	IBI sends the set of configurations to prevent the predicted attach for policy checking by CAS.	CAS run the validation and return the results to IBI.	Passed

9.10	IBI	RTR	Sent the list of actions and configuration to be enforced in the infrastructure.	Mitigation actions and configurations received by RTR.	Passed
10.11	RTR	ePEM	Sent the list of commands (with Ansible playbooks) for each action to be enforced.	List of commands received by ePEM,	Passed
11.12	ePEM	DOC	Sent the list of commands (with Ansible playbooks) for each action not enforced directly by ePEM.	List of commands received by DOC,	Passed
12.15	DOC	INFR	Execute the list of commands.	List of commands executed,	Passed

Table 25: HORSE framework interactions testing - Demonstrator 9

4.11 Demo 10 – UC2

4.11.1 Demo 10 - Threat Prediction

Table 26 summarizes the validation tests conducted for the Threat Prediction Workflow in Demonstrator 10. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.1	INFR	SM	Translate in JSON • Log data for APIExp • PCAP files for DDoS	Saved data in EL.	Passed
1.2	SM	PreProc	Provide the JSON format • Log for APIExp • PCAP for DDoS	Successfully retrieved and answered to PreProc	Passed
8.1	PAG	SM	Check data using Kibana UI • Log for APIExp • PCAP for DDoS	The IP addresses and the payload are anonymized.	Passed

Interactions			Test Description	Result	Status
ID	CompA	CompB			
7.5	EM	P&P NDT	Store the information received by EM in a local XML file.	Attack description file successfully stored	Passed
5.5	P&P NDT	P&P NDT	Periodically sync with the physical infrastructure and detect attacks.	Internal prediction loop in operation through log messages on the terminal.	Passed
5.4	P&P NDT	DTE	Transfer information about predicted attack to DTE.	Attack description successfully received.	Passed
7.6	EM	IA-NDT	Store the information received by EM in a local XML file.	Attack description file successfully stored	Passed
4.9	DTE	IBI	DTE sends an intent to mitigate the predicted threat.	Intent received by IBI.	Passed
9.14	IBI	CKB	Request the list of possible mitigations for detected/predicted attack.	Request for the list of mitigations received.	Passed
14.9	CKB	IBI	Provide the list of possible mitigation actions for the requested attack.	List of mitigations received by IBI.	Passed
9.13	IBI	CAS	Send the list of proposed actions to CAS for validation before being sent for enforcement.	CAS receives the request from IBI.	Passed
13.9	CAS	IBI	Provide an answer to the query based on the proposed action from IBI.	Evaluated the action based on the status of HORSE.	Passed

Table 26: HORSE framework interactions testing - Demonstrator 10 – Threat Prediction.

4.11.2 Demo 10 - Threat Detection

Table 27 summarizes the validation tests conducted for the Threat Detection Workflow in Demonstrator 10. It outlines the components involved, provides a description of each test, presents the results, and indicates the overall status.

Interactions			Test Description	Result	Status
ID	CompA	CompB			
15.1	INFR	SM	Translate in JSON • Log data for APIExp • PCAP files for DDoS	Saved data in EL.	Passed
1.2	SM	PreProc	Provide the JSON format • Log for APIExp • PCAP for DDoS	Successfully retrieved and answered to PAG.	Passed
2.1	PreProc	SM	Store JSON formatted record into SM index	Successfully stored the JSON formatted record to the SM index	Passed
8.1	PAG	SM	Check data using Kibana UI • Log for APIExp • PCAP for DDoS	The IP addresses and the payload are anonymized.	Passed
2.3	PreProc	DEME	Provide the extracted and pre-processed data in JSON format • Log for APIExp • PCAP for DDoS	Successfully received by DEME.	Passed
3.4	DEME	DTE	Provide the threat detection and relative confidence indication in JSON format.	Successfully received by DTE.	Passed
4.9	DTE	IBI	Provide an intent to mitigate the detected thread.	Intent received by IBI.	Passed
9.14	IBI	CKB	Request the list of possible mitigations for detected/predicted attack.	Request for the list of mitigations received.	Passed
14.9	CKB	IBI	Provide the list of possible mitigation actions for the requested attack.	List of mitigation received by IBI.	Passed
9.13	IBI	CAS	Validate the proposed actions with CAS before being sent for enforcement.	Request received by CAS.	Passed
13.9	CAS	IBI	Provide an answer on the query based on the proposed action from IBI	Evaluated the action based on the status of HORSE	Passed
9.10	IBI	RTR	Send the list of actions and configuration to be enforced in the infrastructure.	Mitigation actions and configurations received by RTR.	Passed

10.11	RTR	ePEM	List of commands for each action to be enforced in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.	List of commands received by ePEM.	Passed
11.12	ePEM	DOC	List of commands for each action not enforced directly by ePEM in JSON format with embedded Ansible playbooks sent over HTTP protocol using REST APIs.	List of commands received by DOC	Passed
12.15	DOC	INFR	Execute the list of commands.	List of commands executed	Passed

Table 27: HORSE framework interactions testing – Demonstrator 10 – Detection Workflow.

5 Compliance Matrix for HORSE Iteration IT-2

This section presents the compliance matrix that maps the final HORSE requirements, defined in Deliverable D2.3 [3], to the corresponding validation activities conducted throughout the project .

The matrix details each requirement by its identifier, name, and description. It also identifies the specific HORSE component (s) that address the requirement, along with the demonstrators in which the requirement is implemented, validated, and tested.

This deliverable focuses on the mapping of requirements completed under the IT-2 iteration. Requirements addressed in the earlier IT-1 iteration were previously reported in Deliverable D5.2 [1].

By establishing this mapping, see Table 28, the compliance matrix ensures clear traceability between the defined requirements and their practical verification through the project's use cases and demonstrators.

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
REQ-F-01	Multi-device connectivity monitoring	The HORSE platform must monitor the connectivity of devices connected to the managed network.	SM	Covered IT-1	
REQ-F-02	Multi-device protection	The HORSE platform must monitor the cybersecurity, including authentication, authorization, threat detection and secure connectivity, of the devices on extreme edge connected to the monitored network.	SM DEME P&P DT	Covered IT-2	4
REQ-F-03	Auditing of messages	The HORSE platform could offer auditing capabilities per subsystem.	IBI PIL STO	Covered IT-2	0 to 10
R9EQ-F-04	Auditing of device connections	The HORSE platform should produce auditing logs of devices connecting to the network managed by the platform.	SM	Covered IT-1	
REQ-F-05	Unauthorized device attempt detection	The HORSE platform must be able to detect when an unauthorized device attempts to connect to network and stop it from	SM	Covered IT-2	4

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
		harming the HORSE platform.			
REQ-F-06	Detection of connection error on devices	The HORSE platform must detect connection errors of authorized devices trying to join the HORSE network / slice.	SM	Covered IT-2	4
REQ-F-07	Network Performance monitoring	The HORSE platform must ensure efficient monitoring mechanisms to timely identify network performance degradation regarding reliability, latency, and bandwidth.	SM	Covered IT-1	
REQ-F-08	Authentication support	The HORSE platform should be able to supervise different authentication and authorization mechanisms (Ie: OAuth2.0, DIDs, digital signatures).	ALL	Covered IT-1	
REQ-F-09	Threat detection	The HORSE platform must be able to detect potential threats from external entities.	DEME	Covered IT-1	
REQ-F-10	Data integrity	The HORSE platform should be able to verify the integrity of information exchanged between different HORSE components.	ALL	Covered IT-2	4
REQ-F-11	Notification	The HORSE platform must be able to notify the network operator about actions enforced on detected threats or foreseen threats.	IBI ePEM RTR DOC	Covered IT-2	0 to 10
REQ-F-12	Detection of attacks	The HORSE platform should detect attacks (such as DDoS attacks) on the network.	DEME	Covered IT-1	

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
REQ-F-13	Mitigation of attacks	The HORSE platform should propose network and system reconfigurations to mitigate attacks.	IBI DTE	Covered IT-1	
REQ-F-14	Prediction of attacks	The HORSE platform must implement a methodology to predict an attack.	DEME	Covered IT-1	
REQ-F-15	Policy enforcement	The HORSE platforms should be able to enforce reconfiguration of the infrastructure in case of threat or attack detection.	RTR ePEM DOC	Covered IT-1	
REQ-F-16	Sandboxing of reconfiguration	The HORSE platform should be able to test and evaluate new configurations before deploying them to the infrastructure.	IBI IA-DT EM	Covered IT-1	
REQ-F-17	Persistence of information	The HORSE platform must save the intents entered by the user thought the Intent GUI in a persistent manner.	IBI	Covered IT-1	
REQ-F-18	Reporting of Intent-based decisions	The HORSE platform should report to the network administrator the decisions taken by the intent-based component.	IBI	Covered IT-1	
REQ-F-19	Anomal detection	The HORSE platform must provide the appropriate mechanisms for anomaly detection in the transmitted messages.	DEME	Covered IT-1	
REQ-F-20	Attack modelling	The HORSE platform must be able to model attacks.	EM PEM	Covered IT-1	
REQ-F-21	Attacks impact modelling	The HORSE platform should model the impact of attacks in a SAN scenario.	EM	Covered IT-2	3

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
REQ-F-23	Access management	The user must be able to define and then the HORSE platform must enforce access policies in real time and ensure that access to the collected data assets is only granted to the authorized entities (users or components).	PAG	Covered IT-2	10
REQ-F-26	Multi-tier orchestration	The HORSE platform should be able to manage services in a multi-tier environment, comprising cloud, edge and far edge.	SM RTR ePEM DOC	Covered IT-1	
REQ-F-27	Notification of provisioning	The HORSE platform must notify the user about reconfiguration of the network to mitigate or avoid an attack.	IBI ePEM DOC RTR	Covered IT-2	2 to 5
REQ-F-28	Real-time monitoring of attacks	The HORSE platform must continuously monitor the network for possible attacks and notify the user about them.	SM DEME SAN DTE ePEM	Covered IT-2	0 to 10
REQ-F-29	Policies definition	The HORSE platform should provide a way to define policies and actions to be performed in the network when certain conditions are met.	IBI RTR ePEM DOC	Covered IT-1	
REQ-F-31	Use of anonymized data for AI training	The HORSE platform should use anonymized data to train AI models to detect threats and attacks.	DTE	Covered IT-1	
REQ-F-32	AI-based policies definition	The HORSE platform should be able to define set of optimum policies using AI/ML models to guarantee the system security against potential attacks.	DTE	Covered IT-2	1, 5

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
REQ-F-33	Reproducibility and repeatability of certain behaviors of digital twin in the network	The HORSE Digital Twin should be able to consistently repeat specific experiments, and to incorporate controlled variations to experiment execution as requested by its users.	SAN	Covered IT-1	
REQ-F-34	Different granularity of control plane functions for DT	The HORSE Digital Twin should be capable of deploying different network functions to test them independently or to model whole functionality sets or planes as a single entity, according to specific experiment.	SAN	Covered IT-1	
REQ-F-35	Data anonymization	The HORSE platform must execute data anonymization operations on collected data assets.	PAG	Covered IT-2	0 to 10
REQ-F-36	Data encryption	The HORSE platform must support end-to-end data encryption for data in transit.	PAG	Covered IT-2	0 to 10
REQ-F-37	Observability	The HORSE platform could allow the user to monitor the status (successful or failed execution) and view an incident summary of all AI pipelines.	PAG DTE	Covered IT-2	0 to 10
REQ-F-38	Data retention	The user should be able to define and then the HORSE platform should execute data retention operations (e.g., automated deletion after a certain due date) on collected data assets.	PAG	Covered IT-2	0 to 10
REQ-F-39	Data ingestion 1	The HORSE platform must allow the ingestion of	SM PreProc	Covered IT-1	

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
		data at rest (for example, from a file or from an API).			
REQ-F-40	Data ingestion 2	The HORSE platform must allow the ingestion of real-time data (for example, streaming data).	SM PreProc	Covered IT-1	
REQ-F-41	Data pre-processing	The user must be able to define data pre-processing rules (for handling outliers, for handling missing data values, etc.) on the data assets and the HORSE platform must pre-process the collected data assets according to these rules.	SM PreProc	Covered IT-1	
REQ-F-42	Network and service status information	The PIL must be able to periodically gather information about the status of the network and running services.	PIL	Covered IT-2	4
REQ-F-43	Threat Detection Time	The Horse platform must be able to provide early attack detection within three ROPs (Note: The ROP period with which monitoring data are collected from the network is dimensioned according to the network size. A typical value is in the order of some minutes).	DEME	Covered IT-2	1, 5, 7
REQ-F-44	Threat Detection Rate	The detection rate, i.e., the number of successfully detected attacks over the total, should be above 90%, considering the application of ML and the SoA benchmarks.	DEME	Covered IT-2	1
REQ-F-45	Monitoring the latency between	The SM should be able to provide the IBI at any time the average latency	INFR	Covered IT-2	0, 5 and 8

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
	network endpoints	between two network end-points.			
REQ-F-46	Monitoring the throughput between network endpoints	The SM should be able to provide the IBI at any time the average throughput between two network end-points.	INFR	Covered IT-2	0, 5 and 8
REQ-F-47	Monitoring of packet loss between two network endpoints	The SM should be able to provide the IBI how reliable is a network connection between two network endpoints in a scale from 0 to 1.	INFR	Covered IT-2	0, 5 and 8
REQ-F-48	Visual information of intent's lifecycle	The IBI should be able to show the lifecycle of received intents and its execution.	IBI	Covered IT-2	0 to 3 and 5 to 10
REQ-F-49	Learning and reasoning about human input	When a decision is escalated to human operators, the IBI should be able to learn the decision taken from the operator to apply the same reasoning when same situation repeats.	IBI	Covered IT-2	0 to 3 and 5 to 10
REQ-F-50	Store attacks/mitigations	The HORSE platform must be capable of storing detailed records of known attacks and their corresponding mitigation strategies within the CKB.	CKB	Covered IT-2	0, 1, ,5, 7, 9, 10
REQ-F-51	Access to Attacks Mitigations information	The HORSE platform must enable its components to access the Knowledge Base (KB) to retrieve attacks and mitigations. data via REST-API.	CKB	Covered IT-2	0, 1, ,5, 7, 9, 10
REQ-F-52	Generation of mitigation strategies with GenAI	The HORSE platform should leverage advanced generative AI techniques to automatically generate new mitigation strategies for	CKB	Covered IT-2	0, 1, ,5, 7, 9, 10

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
		identified attacks, enhancing the content of the CKB.			
REQ-F-53	Prioritize mitigation actions	The HORSE platform should provide mechanisms for ranking, thus prioritizing mitigation actions based on their severity and impact.	CKB	Covered IT-2	0, 1, ,5, 7, 9, 10
REQ-F-54	Location awareness	The HORSE platform should have a means to determine the location of UE in the network and share this information through a centralized API.	LOCATION API (SM)	Covered IT-2	3
REQ-F-55	Network status visualization	The HORSE platform could show the status of the network, including the status of the detected or predicted attacks or anomalies and the status of the mitigations and preventive actions.	Dashboard (INFR)	Covered IT-2	0 to 10
REQ-F-56	Decentralized ML training	The HORSE platform should support decentralized training of the involved ML models.	DTE	Covered IT-2	8
REQ-F-57	ML models repository	The HORSE platform should support dynamic update of various ML models stored in the corresponding repository.	DTE	Covered IT-1	
REQ-F-58	Intents	The HORSE platform should cater for different types of intent depending on short- and long-term goals (mitigation and prevention intents).	IBI DTE	Covered IT-1	
REQ-F-59	Monitoring of packet flows	The SM must provide the SAN	SM	Covered IT-2	2,3

REQ ID	Name	Description	HORSE Component	Status	Demonstrator #
	through networked interfaces	data flows in PCAP format to enable the analysis and replica of flows between the Physical and the DT of the Network.			
REQ-F-60	Policy Collection	The CAS must collect policies from the policy configurator using REST APIs.	CAS	Covered IT-2	0, 3, 5, 6, 9, 10
REQ-F-61	Compliance Verification	The CAS must assess collected policies against compliance criteria defined by 3GPP and ENISA standards.	CAS	Covered IT-2	0, 3, 5, 6, 9, 10
REQ-F-62	Flagging Non-Compliant Policies	The CAS must flag policies that do not meet compliance standards for review.	CAS	Covered IT-2	3, 6, 9
REQ-F-63	Reporting	The CAS should generate detailed compliance reports.	CAS	Covered IT-2	3, 6, 9

Table 28: Compliance matrix between the functional requirements and functional validation.

As Table 28 shows, the HORSE validation process addresses all the requirements defined across both iterations of the project. In this second iteration, 35 requirements are covered, ensuring that the entire set of requirements specified in deliverables D2.1 [7] and D2.3 [3] has been fully satisfied.

6 Conclusion

This document serves as the accompanying report for the second and final release of the HORSE framework, which integrates multiple components into a unified cybersecurity platform designed to meet the evolving requirements of 6G networks. This milestone represents a major advancement in addressing the cybersecurity challenges of next-generation communications.

This deliverable provides an overview of the HORSE components integrated in the final version of the framework, along with the ten demonstration scenarios developed to showcase their capabilities. These scenarios illustrate their interactions both within the framework and with the underlying infrastructure. Additionally, the document outlines the integration methodology, supporting tools, and the detailed planning activities that guided the deployment process. Based on this foundation, a comprehensive list of component interactions is documented, including their implementation status, the data types exchanged, and the communication protocols used. A suite of test cases has been meticulously designed and executed to validate functionality of the HOSE framework. These test cases ensure that all interactions operate as intended under realistic conditions, thereby strengthening confidence in the overall system.

In conclusion, the second and final release of the HORSE framework demonstrates a fully integrated, stable, and mature cybersecurity platform. The successful execution of the ten demonstration scenarios validates its ability to address complex 5G/6G security requirements while ensuring interoperability across diverse components. The positive outcomes of the test cases and integration efforts confirm not only the technical soundness of the platform but also its readiness for large-scale demonstration, cross-use case validation, and evaluation. Overall, this release establishes HORSE as a strong foundation for enabling secure and resilient next-generation communications.

7 References

[1] HORSE Project, “D5.2 First HORSE Release: HORSE IT-1 version”. June 2024. Available at: https://horse-6g.eu/deliverables/, Retrieved on 24/07/2025.

[2] HORSE Project, “D2.4 HORSE Landscape and Architectural Design”. December 2024. Available at: https://horse-6g.eu/deliverables/, Retrieved on 24/07/2025.

[3] HORSE Project, “D2.3 HORSE Landscape: Technologies, State of the Art, AI Policies, and Requirements (IT-2)”. November 2024. Available at: https://horse-6g.eu/deliverables/, Retrieved on 24/07/2025.

[4] HORSE Project, “D3.2 HORSE Platform Intelligence Developed (IT-2)”. June 2025. Available at: https://horse-6g.eu/deliverables/, Retrieved on 24/07/2025.

[5] HORSE Project, “D4.2 HORSE AI-assisted Human-centric Secure and Trustable Orchestration Developed (IT-2)”. June 2025. Available at: https://horse-6g.eu/deliverables/, Retrieved on 24/07/2025.

[6] Scapy, https://scapy.net/, Retrieved on 24/07/2025.

[7] HORSE Project, “D2.1 HORSE Landscape: Technologies, State of the Art, AI Policies and Requirements”. June 2023. Available at: https://horse-6g.eu/deliverables/, Retrieved on 24/07/2025.