



AI-DRIVEN CYBERSECURITY FOR FINANCIAL SERVICE PROVIDERS

D2.2 – REFERENCE ARCHITECTURE AND BLUEPRINTS-FIRST VERSION

Lead Beneficiary	UBITECH LIMITED (UBI)
Work Package Ref.	WP2-LLM-Powered Reference Architecture, Specification and Systems Integration
Task Ref.	T2.2-Reference Architecture and Systems' Specifications
Deliverable Title	Reference Architecture and Blueprints-First version
Due Date	30/09/2026
Delivered Date	30/09/2026
Revision Number	v1.0
Dissemination Level	PU - Public
Type	R — Document, report
Document Status	Final
Review Status	Reviewed
Document Acceptance	Acceptable
EC Project Officer	Rodica MORARIU

DIGITAL-ECCC-2024-DEPLOY-CYBER-07-KEYTECH



Co-funded by
the European Union



ECCC 
EUROPEAN CYBERSECURITY
COMPETENCE CENTRE

The project funded under Grant Agreement No 101249596 is supported by the European Cybersecurity Competence Centre

DISCLAIMER

The sole responsibility for the content of this publication lies with the authors. It does not necessarily reflect the opinion of the European Union. Neither the ECCC nor the European Commission is responsible for any use that may be made of the information contained therein.

COPYRIGHT MESSAGE

This report, if not confidential, is licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0); a copy is available here: <https://creativecommons.org/licenses/by/4.0/>. You are free to share (copy and redistribute the material in any medium or format) and adapt (remix, transform, and build upon the material for any purpose, even commercially) under the following terms: (i) attribution (you must give appropriate credit, provide a link to the license, and indicate if changes were made; you may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use); (ii) no additional restrictions (you may not apply legal terms or technological measures that legally restrict others from doing anything the license permits).

CONTRIBUTING PARTNERS

Partner Acronym	Role	Name Surname
UBI	Editor, Contributor	Nikos Papageorgopoulos, Giannis Ledakis
INNOV	Contributor, Internal Reviewer	George Fatouros
UNP	Contributor, Internal Reviewer	Ricardo Ladeira, Márcio Mateus, Pedro Maló
UPRC	Contributor	Efterpi Paraskevoulakou, Konstantinos Mavrogiorgos
HUA	Contributor	George Kousiouris, Panagiotis Rizomiliotis, Christos Diou
IQB	Contributor	Cosmina Stalidi
K3Y	Contributor	Panagiotis Radoglou Grammatikis, Evangelia Pantraki, Christos Dalamangas, Nikolaos Ntampakis, Christos Vasilakis, Dimitrios Balompas, Antonios Sarigiannidis
QUBO	Contributor	Ernstjan de Gooyert, Amin Babazadeh, Tom Schubert
UBI, GFT	Quality Assurance	Louiza Kachrimani, Michele Canteri

REVISION HISTORY

Version	Date	Partner(s)	Description
v0.1	26/06/2026	UBI	Creation of deliverable document and initial Table of Contents.
v0.2	13/07/2026	UBI	Drafting of the Introduction and Section 2.1, including the architecture modelling principles and development workflow.
v0.3	07/08/2026	UBI, K3Y, IQB	Contributions to the tool and component landscape, development of the blueprint modelling approach and drafting of Section 4 on the CyberAid reference architecture.
v0.4	28/08/2026	UBI, UPRC, INNOV	Further contributions to the tool and component landscape and development of the blueprint instances presented in Section 5.
v0.5	11/09/2026	UBI, HUA, QUBO, UNP	Contributions from HUA , QUBO and UNP on Section 3. Finalization of Section 4 and 5.
v0.6	15/09/2026	INNOV, UNP	Internal review of the complete deliverable, covering contextual, technical and editorial consistency.
v0.7	23/09/2026	UBI	Changes based on review.
v1.0	30/09/2026	UBI	Final quality check and submission to the EC

LIST OF ABBREVIATIONS

Acronym	Definition
AI	Artificial Intelligence
AML	Anti-Money Laundering
API	Application Programming Interface
ATT&CK	Adversarial Tactics, Techniques and Common Knowledge
CEP	Complex Event Processing
CLI	Command-Line Interface
CTI	Cyber Threat Intelligence
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DKIM	DomainKeys Identified Mail
DMARC	Domain-based Message Authentication, Reporting and Conformance
DORA	Digital Operational Resilience Act
eBPF	Extended Berkeley Packet Filter
ECSMAF	ENISA Cybersecurity Market Analysis Framework
EMTD	Email Threat Detection
EMNLP	Empirical Methods in Natural Language Processing
ENISA	European Union Agency for Cybersecurity
GUI	Graphical User Interface
HITL	Human-in-the-Loop
IBAN	International Bank Account Number
ID	Identifier
ISO/IEC/IEEE	International Organization for Standardization / International Electrotechnical Commission / Institute of Electrical and Electronics Engineers
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
KYC	Know Your Customer
LLM	Large Language Model
MCP	Model Context Protocol
MITRE	MITRE Corporation
NIS2	Network and Information Security Directive 2
NVD	National Vulnerability Database
OAuth	Open Authorization
OLISTIC	Risk Assessment and Management Engine
OWASP	Open Worldwide Application Security Project
PQC	Post-Quantum Cryptography
PSD2	Revised Payment Services Directive
Q2FA	Quantum Two-Factor Authentication
QTPP	Quantum Trusted Token Provider

QTU	Quantum Token User
RAG	Retrieval-Augmented Generation
REST	Representational State Transfer
SIEM	Security Information and Event Management
SOC	Security Operations Centre
SPF	Sender Policy Framework
SSH	Secure Shell
STIX	Structured Threat Information Expression
TAXII	Trusted Automated Exchange of Intelligence Information
UC	Use Case
URL	Uniform Resource Locator
XDR	Extended Detection and Response

EXECUTIVE SUMMARY

Deliverable D2.2, Reference Architecture and Blueprints: First Version, presents the initial reference architecture and blueprint development approach for the CyberAId project. The deliverable builds on the requirements baseline established in D2.1 and provides a common structure for organising the cybersecurity tools, AI enabled capabilities, shared services, information flows and interaction mechanisms that constitute the CyberAId framework.

The methodology followed in D2.2 uses the requirements baseline as the primary architectural driver, supported by the user stories and the initial CyberAId component landscape. The requirements define the needs and constraints that must be addressed, while the user stories inform the actors, interactions and expected outcomes represented within the reference architecture. The component landscape identifies the available and planned capabilities that must be positioned and connected. The methodology is iterative and allows both the architecture and component landscape to be refined when gaps, overlaps or unclear responsibilities are identified.

The deliverable presents the CyberAId tool and component landscape through five architectural groups: Presentation and HITL, LLM Orchestration and Agentic Analysis, Security Capability Services, Integration and Messaging Fabric, and Knowledge and Data Fabric. In the Functional View, the combined LLM Orchestration and Agentic Analysis group is represented as two distinct layers, separating workflow coordination from specialist analytical capabilities. These groups provide a consistent basis for describing the role of each component and for positioning the components within the wider platform. The landscape includes user facing interfaces, AI based analytical capabilities, deterministic security tools, messaging and integration mechanisms, and shared knowledge and data services. The landscape also includes contextual risk and compliance assessment, report composition, and digital forensics and post-incident analysis.

The CyberAId reference architecture is presented through three complementary views. The System Context View defines the platform boundary and its relationships with users, monitored financial infrastructures and external entities. The Functional View provides the principal structural representation of the platform and organises its components and shared services according to their responsibilities. The Data and Knowledge View describes how telemetry, threat intelligence, findings, evidence and risk information move through the platform. Together, these views establish the initial logical and integration baseline of CyberAId without prescribing a single physical deployment model.

The deliverable also establishes a lightweight methodology and common template for developing the CyberAId blueprints. The methodology connects a defined cybersecurity objective with the applicable requirements, selected architecture elements, CyberAId components, interactions, controls and expected validation evidence. The initial blueprint portfolio is organised around the four CyberAId pilot scenarios. At this stage, the blueprints are treated as evolving design artefacts that identify the required capabilities and relationships while allowing detailed configurations, protocols, deployment arrangements and testing procedures to be refined as the corresponding technical work matures.

Overall, D2.2 provides the first common architectural baseline for the CyberAId framework and establishes how this baseline can be applied to financial sector cybersecurity scenarios through the blueprints. The reference architecture, component landscape and blueprint instances will continue to evolve as component specifications, interfaces, information flows, integration arrangements and pilot environments become more mature. Their consolidated versions will provide the basis for the final reference architecture and blueprints.

TABLE OF CONTENTS

1.	Introduction	10
1.1	Scope of Deliverable	10
1.2	Relation to other Work Packages and Work Plan	11
1.3	Deliverable Structure	11
2.	Methodology	13
2.1	Architecture Modelling Approach.....	13
2.1.1	Modelling Principles.....	13
2.1.2	Architecture Development Workflow	14
2.2	Blueprint Modelling Approach.....	15
2.2.1	Blueprint Purpose and Structure	16
2.2.2	Blueprint Methodology and Template	16
2.2.3	Mapping from Requirements to Blueprint Elements	17
3.	CyberAid Tool and Component Landscape	19
3.1	Presentation and HITL.....	19
3.1.1	Security Dashboards - GUI	19
3.1.2	Operations & Administration CLI	22
3.1.3	Learning Centre	22
3.2	LLM Orchestration and Agentic Analysis	24
3.2.1	Email Threat Detection	24
3.2.2	Anomaly Detection.....	26
3.2.3	Automated Log Analysis & Event Correlation	27
3.2.4	AI-Driven Pentesting & Attack Generator	28
3.2.5	Code Analysis	29
3.2.6	Pentesting Outcomes Analysis & Reporting	29
3.2.7	Risk & Compliance Assessment	30
3.2.8	Reporting.....	30
3.3	Security Capability Services	30
3.3.1	SIEM / XDR Platform.....	30
3.3.2	Kernel-Level Monitor (eBPF) + Connection Tracer	31
3.3.3	Graph Analytics Engine	32
3.3.4	Digital Twin Service.....	32
3.3.5	Risk Assessment Engine	33
3.3.6	Federated Analytics & Compliant Data Sharing.....	33
3.3.7	Quantum Token 2FA Service	35

- 3.3.8 Digital Forensics & Post-Incident Analysis 37
- 3.4 Integration and Messaging Fabric..... 37
 - 3.4.1 Event & Alert Bus 37
 - 3.4.2 Tool Adapter Layer & Registry..... 38
 - 3.4.3 Log Collector & Forwarding..... 38
 - 3.4.4 Authentication & Access Control 39
- 3.5 Knowledge and Data Fabric..... 39
 - 3.5.1 Security Knowledge Model & CTI Platform..... 39
 - 3.5.2 Semantic Retrieval & Vector Store..... 40
 - 3.5.3 Fraud Property Graph 40
 - 3.5.4 Security Event & Log Store..... 41
- 4. CyberAid Reference Architecture..... 42
 - 4.1 System Context View 42
 - 4.2 Functional View 43
 - 4.3 Data and Knowledge View 46
 - 4.4 Integration and Deployment Considerations 47
- 5. CyberAid Blueprints 50
 - 5.1 BP-01: Tiered Client Impersonation Detection for Private Banking and Trading Systems..... 50
 - 5.2 BP-02: Advanced Anti-Money Laundering Detection for Payment Service Providers 52
 - 5.3 BP-03: AI-Enhanced Anomaly Detection and Incident Response for Banking Cybersecurity 54
 - 5.4 BP-04: Cyber Resilience for High-Frequency Trading..... 55
- 6. Conclusions And Next Steps..... 58

LIST OF FIGURES

- Figure 1: Modelling workflow for the CyberAid reference architecture 14
- Figure 2: Requirements to blueprint traceability mapping 18
- Figure 3: The analyst conversation, with a Tier 3 finding held for review 20
- Figure 4: A request declined for want of access, with the required input named 21
- Figure 5: EMTD pipeline architecture 24
- Figure 6: Overview of Annotation Mechanisms Approach for the CyberAid Platform 34
- Figure 7: Event and alert interactions..... 38
- Figure 8: CyberAid System Context View 42

Figure 9: CyberAid Functional View 45

Figure 10: CyberAid Data and Knowledge View..... 47

LIST OF TABLES

Table 1: CyberAid blueprint development methodology 17

Table 2: Baseline dashboard views..... 20

Table 3: Deterministic pre-checks executed on every message. 25

Table 4: Q2FA input and operation naming across use cases 36

Table 5: Principal CyberAid component interactions and interface requirements 48

Table 6: Initial CyberAid blueprints 50

Table 7: BP-01 Tiered Client Impersonation Detection 52

Table 8: BP-02 Advanced Anti-Money Laundering Detection..... 54

Table 9: BP-03 AI-Enhanced Anomaly Detection and Incident Response 55

Table 10: BP-04 Cyber Resilience for High-Frequency Trading..... 57

1. INTRODUCTION

The increasing digitalisation of financial services, together with the growing complexity of cyber threats, creates a need for cybersecurity solutions that can operate across heterogeneous financial infrastructures, security tools, operational workflows and regulatory obligations. Financial organisations rely on interconnected systems, cloud and on-premises environments, APIs, payment systems, trading platforms, third-party providers and data-intensive services. These environments require cybersecurity capabilities that are not only technically effective, but also interoperable, traceable, auditable and aligned with the operational needs of financial-sector stakeholders.

Within this context, the CyberAId project aims to develop an AI-driven cybersecurity framework that supports proactive defence, real-time monitoring, incident response, reporting and decision support for financial-sector environments. The project combines several cybersecurity tools and services under a common architectural approach, supported by AI-enabled analysis, knowledge-driven reasoning, human-in-the-loop interaction and compliance-oriented reporting. The reference architecture and blueprints defined in this deliverable provide the initial structural basis for organising these capabilities into a coherent CyberAId framework.

Deliverable D2.2, Reference Architecture and Blueprints, constitutes one of the core outputs of WP2. It builds on the requirements baseline established in D2.1 and uses the user stories and initial component landscape to inform the reference architecture. The requirements provide the formal traceability basis for the blueprint instances. While D2.1 focused on identifying and structuring the cybersecurity requirements for the financial sector, D2.2 focuses on how these requirements are reflected in the CyberAId tool landscape, platform architecture, component interactions and blueprint methodology.

The architecture and blueprint methodology presented in this document constitute the first version of the CyberAId reference architecture and blueprints. They will be refined based on the requirements baseline and the progressive development of blueprint instances, as the CyberAId component landscape matures. The blueprint methodology will be progressively applied to relevant financial cybersecurity scenarios, supporting the iterative development of blueprint instances and the consolidation of the reference architecture, which will be further revised and extended in D2.3.

1.1 Scope of Deliverable

The scope of this deliverable is to present the first version of the CyberAId reference architecture and blueprint development methodology for financial cybersecurity environments. The deliverable defines the initial architectural baseline of the CyberAId framework by describing the main tools, components, services, interaction models and cross-cutting mechanisms that support the project's cybersecurity lifecycle.

The deliverable also presents the first version of the CyberAId reference architecture. This includes the main architecture principles, the platform context, the key actors and interaction points, the high-level role of the orchestration layer, the agent-based processing model, security and observability mechanisms, and the integration and messaging model. The aim is to provide sufficient architectural structure to support technical alignment and integration of components.

In addition, this deliverable defines the methodology for developing the CyberAId blueprints. The initial blueprint instances connect cybersecurity objectives with applicable requirements, selected components, principal interactions and controls, expected outcomes, validation evidence and open issues. At this stage of the project, the blueprints are treated as structured and evolving design artefacts rather than final implementation packages. The deliverable therefore focuses on the purpose,

structure, common template and mapping approach that will be used to connect requirements, tools, architecture elements and expected outcomes.

This first version includes initial interface requirements and deployment considerations that guide subsequent implementation and integration. Detailed endpoint definitions, protocol configurations, physical deployment topologies, integration-test procedures and final validation results will be refined through the corresponding technical activities.

1.2 Relation to other Work Packages and Work Plan

This deliverable is closely connected to the overall CyberAid work plan and acts as a bridge between the requirements and user stories established in D2.1, the CyberAid component landscape, and the subsequent technical, integration and validation activities of the project. The requirements and user stories define the needs, constraints, actors and expected outcomes that the architecture must address, while the component landscape identifies the functional elements that must be organised and connected within the platform.

The regulatory, standards-related, technical, operational and pilot-derived considerations relevant to the architecture are already represented through the requirements baseline established in D2.1. They are therefore not treated as separate parallel inputs to the architecture modelling process. D2.2 translates these consolidated requirements, together with the user stories and component landscape, into a common architectural baseline defining component responsibilities, interfaces, information flows and cross-cutting concerns.

D2.2 also supports the technical work performed in the subsequent work packages. The tool and component landscape described in this deliverable provides an initial alignment point for the development of proactive defence, monitoring, reporting, privacy-preserving analytics, digital twin, red teaming, CTI and AI-supported decision-support capabilities. In this sense, the deliverable helps ensure that the individual technical components remain consistent with the broader CyberAid architectural direction.

The deliverable is also relevant to pilot and validation activities, as the architecture and blueprint methodology provide a first view of how CyberAid capabilities may be applied in financial-sector contexts. The blueprint approach is intended to support the translation of requirements into practical guidance that can later be refined through pilot feedback and validation activities.

The detailed LLM orchestration design, including the internal logic of agents, workflows, prompt strategies and human-in-the-loop mechanisms, is treated in the dedicated LLM orchestration deliverable. D2.2 only positions the orchestration layer within the wider reference architecture and describes its role at a high level. Similarly, the detailed implementation and operational integration of cybersecurity components will be addressed in later integration-oriented deliverables.

1.3 Deliverable Structure

This deliverable is organised into five main sections following the present introduction.

Section 2 presents the methodology for defining the reference architecture and developing the blueprints. It explains how requirements and user stories, together with the CyberAid component landscape, are used to define the platform boundary, identify stakeholders and concerns, organise components and interactions, and assess cross-cutting concerns. It also introduces the blueprint approach, template, and mapping between requirements and blueprint elements.

Section 3 presents the CyberAid tool and component landscape. It provides an overview of the main tools, services and technologies considered in the first architectural baseline, organised into the presentation and human-in-the-loop components, the LLM orchestration and agentic analysis

components, the security capability services, the integration and messaging fabric, and the knowledge and data fabric.

Section 4 introduces the first version of the CyberAId reference architecture through three complementary views. The system context view defines the platform boundary, principal actors and external interaction points. The functional view introduces the main structural organisation of the platform and positions its components and shared services. The data and knowledge view introduces the principal information flows and the use of telemetry, security knowledge, threat intelligence, findings and reporting information across the platform. The section also presents initial integration and deployment considerations, including the principal component exchanges and their architectural constraints.

Finally, Section 5 presents the initial CyberAId blueprint instances organised around the four project pilots. Each blueprint identifies its cybersecurity objective, relevant requirements, selected architecture elements and components, principal interactions and controls, expected outcomes and initial validation considerations.

2. METHODOLOGY

This section presents the methodology followed for defining the first version of the CyberAId reference architecture and establishing the approach for the subsequent development of the CyberAId blueprints. The resulting architecture then provides a structural basis for translating these inputs into scenario-specific blueprints.

The requirements baseline already incorporates the relevant technical, operational, regulatory, standards-related and pilot-derived needs identified for the CyberAId framework. These considerations are therefore not treated as separate or parallel inputs to the architecture and blueprint processes. Instead, they enter the methodology through the requirements and remain traceable through the requirements catalogue.

Although the methodology follows a defined sequence, it is iterative. The architecture may be refined as the component landscape evolves and as relationships, interfaces and information flows become more clearly defined. Similarly, applying the blueprint methodology may identify gaps or changes that need to be reflected in later revisions of the reference architecture.

2.1 Architecture Modelling Approach

This section presents the approach used to develop the first version of the CyberAId reference architecture. The approach combines the requirements baseline, user stories and initial component landscape to define the platform boundary, organise the identified capabilities, establish component responsibilities and describe their principal interactions and information flows. The resulting architecture is represented through three complementary views and provides the common architectural basis for the blueprint methodology presented in Section 2.2.

2.1.1 Modelling Principles

The architecture modelling approach was informed by the architecture description concepts defined in ISO/IEC/IEEE 42010:2022 [1]. The standard distinguishes the architecture of a system from the description used to communicate that architecture and introduces concepts such as stakeholders, concerns, viewpoints, views and architecture models. These concepts were used as a general reference for organising the CyberAId architecture description. However, D2.2 does not claim formal conformance with the complete standard.

The CyberAId reference architecture is treated as a reusable architectural baseline that remains independent of detailed implementation, configuration and deployment choices. It establishes a common vocabulary, structural organisation and set of relationships that can guide the positioning and integration of CyberAId components across different financial sector environments. This understanding is consistent with the concept of a reference architecture as an abstract framework that captures common architectural knowledge and supports the development of more specific system architectures [2].

The primary structural representation of CyberAId adopts a layered architectural style. Layered architectures organise system elements into logical groups with distinct responsibilities and defined dependency relationships [3]. In CyberAId, the layers represent logical areas of responsibility rather than fixed physical deployment tiers. Components belonging to the same logical layer may therefore operate across different physical environments while preserving their architectural roles and interaction boundaries.

The modelling approach aims to remain technology neutral where this supports reuse and architectural stability. Generic infrastructure and integration functions are therefore described primarily according to their capabilities and responsibilities. Named project assets may be retained

where their identity is necessary to understand a specialised CyberAId function. This distinction allows the architecture to preserve a connection to concrete project components without unnecessarily coupling the reference architecture to implementation choices that may evolve.

2.1.2 Architecture Development Workflow

The development of the initial CyberAId reference architecture followed the requirements led workflow illustrated in Figure 1. The requirements baseline constitutes the primary architectural driver, supported by the user stories established in D2.1 and the initial CyberAId component landscape. The user stories provide the relevant actors, interactions and expected outcomes, while the initial component landscape identifies the available and planned capabilities that need to be positioned and connected within the architecture.

The workflow brings these inputs together through three connected modelling activities. The first aligns the architecture inputs and relates the requirements and user stories to architectural responsibilities. The second defines the logical structure, interactions, information flows and architecture views. The third assesses architectural consistency, cross cutting concerns, gaps and overlaps. The resulting findings may require the architecture or the initial component landscape to be revised.

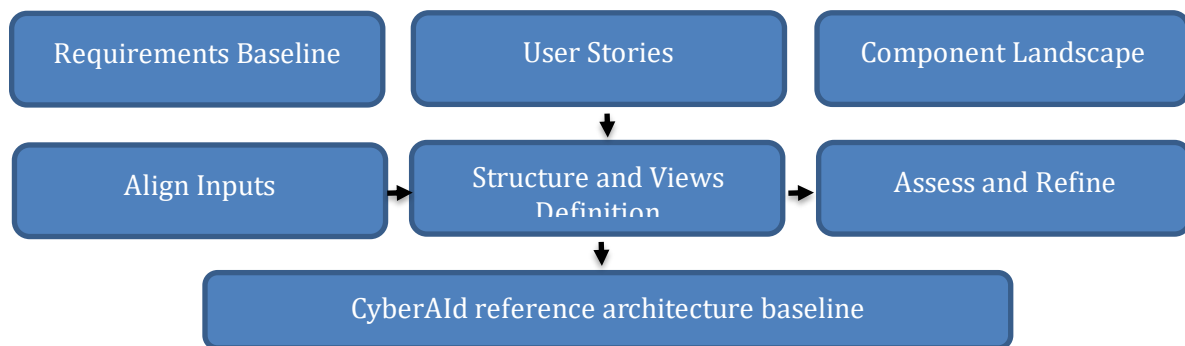


Figure 1: Modelling workflow for the CyberAId reference architecture

The architecture inputs were initially consolidated and aligned to establish a consistent basis for the modelling work. The requirements and user stories established in D2.1 were reviewed together with the initial CyberAId component landscape. This alignment focused on establishing consistent terminology, resolving duplicated or overlapping information and identifying missing or unclear relationships between the available inputs.

The aligned inputs were then used to allocate architectural responsibilities. The requirements were examined to determine which architectural functions, components or shared services are expected to address them. The user stories preserved the relationship between these responsibilities and the actors, interactions and expected outcomes identified in D2.1. This activity supported the identification of requirements that were not adequately covered, responsibilities shared by multiple components, overlapping functionality and component boundaries requiring further clarification.

The logical structure of the platform was subsequently defined according to the responsibilities, dependencies and interaction patterns of the CyberAId components and shared services. Their principal interfaces and information flows were represented through three complementary views. These are the system context view, the functional view and the data and knowledge view.

The system context view represents the CyberAid platform boundary, actors, monitored financial infrastructures and external interaction points established through the requirements and user stories. The functional view acts as the principal structural representation of the platform by organising components and shared services according to their architectural responsibilities and illustrating the principal dependencies and interaction paths between the layers. The data and knowledge view complements these representations by describing how telemetry, threat intelligence, contextual information, findings and reporting information are collected, processed, exchanged and used across the platform.

A separate deployment view is not included in this first version of the reference architecture. Common deployment considerations are outlined in Section 4.4, while scenario-specific assumptions are addressed within the blueprint instances.

Cross cutting concerns apply across the complete architecture rather than to an individual layer or view. Security, identity and access control, privacy, guardrails, observability, auditability and interoperability were therefore assessed across the architecture views and component interactions. The resulting architecture was also reviewed to identify missing elements, unclear responsibilities, duplicated functions, inconsistent information flows and unresolved integration points.

Regulatory, standards related, technical, operational and pilot derived considerations enter the architecture modelling process through the requirements baseline established in D2.1. Their original sources and mappings remain traceable through the requirements catalogue and are not treated as separate architecture inputs. The architecture modelling process translates these consolidated requirements into architectural responsibilities, constraints, interfaces and cross cutting concerns.

The feedback path included in Figure 1 represents the iterative character of the architecture development process. The architecture assessment may reveal missing capabilities, overlapping functions, unclear responsibilities or unsuitable component placements that require the component landscape or earlier modelling decisions to be revised. Likewise, changes to the requirements, user stories or components may require the architecture to be updated. The workflow therefore provides a consistent approach to architecture development without implying a strictly linear or one time process.

The resulting reference architecture should be understood as an initial logical and integration-level baseline. Its purpose is to provide a stable structure for positioning components, defining responsibilities and interaction boundaries, and supporting the progressive specification of interfaces and information flows. This baseline subsequently informs the blueprint modelling approach described in Section 2.2.

2.2 Blueprint Modelling Approach

The CyberAid blueprint modelling approach defines how the requirements baseline and relevant parts of the reference architecture are translated into reusable and scenario specific implementation guidance. While the reference architecture describes the overall CyberAid platform, each blueprint focuses on a defined financial sector cybersecurity objective and selects the requirements, assets, threats, controls, architecture elements, components, interfaces and deployment considerations relevant to that objective.

The approach is requirements driven and architecture aligned. The requirements baseline provides the needs, constraints, priorities and validation expectations that each blueprint must address. Regulatory, standards related, technical, operational and pilot derived considerations are incorporated through the requirements catalogue and are not treated as separate blueprint inputs. The reference architecture provides the common structure from which the relevant architectural elements, components, responsibilities and interaction paths are selected.

Selected concepts from the ENISA Cybersecurity Market Analysis Framework are used to support the scoping of the target environment and the identification of relevant infrastructure, assets, threats, security requirements and stakeholders [4]. The framework is used only as a supporting reference for these activities and is not adopted as the CyberAid blueprint methodology. Threat scenarios may also be structured using recognised sources such as MITRE ATT&CK, where applicable, to identify relevant adversary tactics and techniques [5].

The validation perspective is informed by the ENISA Cybersecurity Exercise Methodology [6]. Its treatment of objectives, scenarios, evaluation criteria and evidence supports the definition of a simple and repeatable validation approach. The CyberAid blueprints are not cybersecurity exercises. These concepts are used only where they support the assessment of blueprint outcomes.

The methodology covers the development of a blueprint, from the definition of its scope and requirements basis to the specification of its architecture selection, implementation guidance and validation approach. The blueprint may be revised when validation findings, architectural changes or updates to the requirements identify missing information, unsuitable selections or unresolved dependencies.

2.2.1 Blueprint Purpose and Structure

A CyberAid blueprint is a structured design artefact that explains how selected CyberAid capabilities can be applied to a specific financial sector cybersecurity objective. It translates the common reference architecture into guidance that can be adapted to a defined institutional, operational and deployment context.

The CyberAid blueprints collectively support the complete cybersecurity lifecycle, including proactive defence, real time monitoring, threat detection, incident response, recovery, forensics and reporting. Each individual blueprint addresses the parts of this lifecycle relevant to its cybersecurity objective rather than reproducing every CyberAid capability.

Each blueprint defines its objective, scope, applicable requirements, protected assets and relevant threats. It selects the CyberAid components needed for the scenario and describes their principal interactions, security controls and implementation considerations. Initial validation identifies a test scenario, expected result and required evidence. Detailed configurations, complete interface specifications and physical deployment choices will be refined during integration and pilot planning.

2.2.2 Blueprint Methodology and Template

The CyberAid blueprint methodology consists of four connected modelling activities. These activities define the blueprint, select the relevant parts of the architecture, describe how the selected elements work together and establish how the expected outcome will be assessed. Table 1 presents the modelling activities and distinguishes between the minimum information required for every blueprint and additional information that may be included when relevant.

Modelling activity	Minimum required content	Additional content when relevant
Define the blueprint	Identifier, title, version, status, cybersecurity objective, scope, target environment and applicable requirement identifiers	Main actors, assumptions, dependencies and exclusions
Select architecture and capabilities	Relevant reference architecture elements, CyberAid components, external services and principal responsibilities	Protected assets, relevant threats, MITRE

		ATT&CK techniques and trust boundaries
Describe operation and controls	Main interactions, information flows, interfaces, security controls and high-level deployment assumptions	Protocols, authentication needs, data handling constraints, guardrails and configuration guidance
Define validation and refinement	Expected outcome, acceptance criteria, required evidence, open issues and planned updates	Test procedures, metrics, validation responsibilities and detailed test scenarios

Table 1: CyberAid blueprint development methodology

The template is intended to be applied consistently across all CyberAid blueprints, while still allowing for scenario-specific adaptations where necessary. Where information is unavailable, the blueprint should record the relevant item as an open issue rather than requiring speculative detail. Additional content should be included only where it materially supports implementation, assessment or later refinement.

2.2.3 Mapping from Requirements to Blueprint Elements

Requirements provide the primary traceability basis for the CyberAid blueprints. Each blueprint selects the requirements relevant to its cybersecurity objective and records their identifiers within the blueprint definition.

The selected requirements are mapped to the relevant reference architecture elements, CyberAid components, responsibilities, interactions and controls. The expected outcome and validation evidence are then defined to indicate how the blueprint addresses those requirements.

Figure 2 presents the resulting concise traceability mapping, which links the requirements baseline to the blueprint objective and scope, the relevant architecture elements and components, the applicable controls and interactions, and the expected outcome and validation evidence. A separate mapping entry is not required for every sentence or design decision in the blueprint. Requirements may be grouped where they are addressed by the same components, controls or information flows, provided that the relevant requirement identifiers remain visible.

Where a requirement is only partially addressed, the blueprint should record the limitation and the required follow up action. Where the blueprint identifies a requirement gap, the gap should be returned to the requirements management process rather than introduced as an undocumented assumption.

Regulatory and standards related traceability is inherited from the mappings maintained in the requirements catalogue. The blueprint therefore records the relevant requirement identifiers without repeating the complete regulatory analysis. Additional compliance explanation is included only where it is necessary to understand the selected controls or expected validation evidence.

The mapping should be reviewed when the requirements baseline, reference architecture, component landscape or blueprint content changes. This ensures that the blueprints remain aligned with the evolving CyberAid framework without creating an excessive traceability burden.

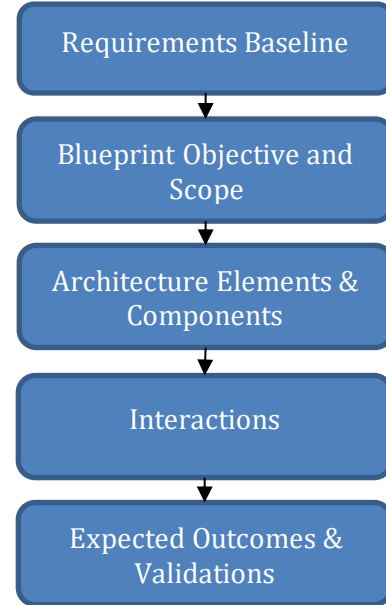


Figure 2: Requirements to blueprint traceability mapping

3. CYBERAID TOOL AND COMPONENT LANDSCAPE

This section presents the CyberAid tool and component landscape, organised into five architectural groups. Each component is described in terms of its purpose, its technical realisation and its position within the platform, at the level of detail this first architectural baseline requires.

Two of these groups, LLM Orchestration and Agentic Analysis, span two functional layers, whose internal design is specified in depth in D2.5. That deliverable defines the canonical agent architecture shared by every agent, the workflow state and context policy, the Security Context and guardrail model that govern what an agent may reach, the structured finding contract through which analysis leaves the platform, the workflow type catalogue and dispatch logic, and the human in the loop review tiers. The component descriptions in this section therefore state what each component does and how it relates to the rest of the architecture, and refer to the relevant sections of D2.5 for the mechanisms they share rather than restating them.

The division of responsibility between the two deliverables is that D2.2 holds the architectural placement and the boundaries between components, while D2.5 holds the internal design of the orchestration and agentic layers.

3.1 Presentation and HITL

3.1.1 Security Dashboards - GUI

The dashboards are where the platform and its users meet, in both directions. They are where the output of the platform reaches a person, and they are equally the point at which work enters it: a question put to an agent, material offered for analysis, a workflow raised by hand, a review decision, an operator grant. This section states the small set of views the orchestration layer needs in order to be operable, and the contract those views hold with it. It does not attempt a complete interface specification, and layout, visual design and interaction detail belong to the task that builds them.

The conversational interface is the primary way a user interacts with CyberAid. An analyst asks a question in natural language, such as why a particular email was flagged or whether anything in the available logs is suspicious, and the coordinator answers from the finding corpus with the evidence attached. The interface is built under T5.3 on the interaction pattern developed in the email threat detection work of T4.3, where an explanation traces a verdict back to the specific agent findings behind it and is followed by a prioritised set of recommended actions. Recommendations are accepted, rejected or corrected in the conversation itself, and those outcomes are recorded against the finding rather than spent once, feeding the scoped memory.

Being the primary interaction with CyberAid does not make it the only surface a security team works in. Wazuh is used as the main SIEM in CyberAid, and for some security teams its interface will remain the surface at the centre of daily work, with detection, alert triage and host telemetry continuing to be handled there. What CyberAid adds is what a SIEM console does not hold: the finding an agent produced from an alert, the evidence and model references behind it, and the review decision taken on it. Because a finding records the alert that triggered its workflow among its evidence references, an analyst can move between the two rather than having to choose one.

Three views are proposed on that basis, corresponding to the operator interfaces already identified, as shown in Table 2.

View	What it shows	Actions available
Analyst conversation	Natural-language querying across the finding corpus, with streaming responses,	Ask; supply material for analysis; raise a workflow; accept, reject

	the explanation behind a verdict and the recommended actions that follow from it	or correct a recommendation
Finding review	Findings awaiting Tier 2 countersignature or Tier 3 approval, and the full record of a selected finding including its evidence, retrieval references, tool calls and model references	Approve, reject or correct; open related findings
Administration	System status, the data sources within reach of a context, operator grants, extension pack manifests, connector health and the provider registry, together with audit export	Grant and revoke; enable and disable packs; export the audit record

Table 2: Baseline dashboard views

Figure 3 shows the interface as it currently stands under T5.3. The conversation carries the finding identifier, the severity and the confidence alongside the explanation, the workflow that produced it is shown as a completed step, and a Tier 3 finding is held for a decision that cannot be recorded without a written rationale. The navigation exposes the three views of Table 2 together with direct entry to the workflow types.

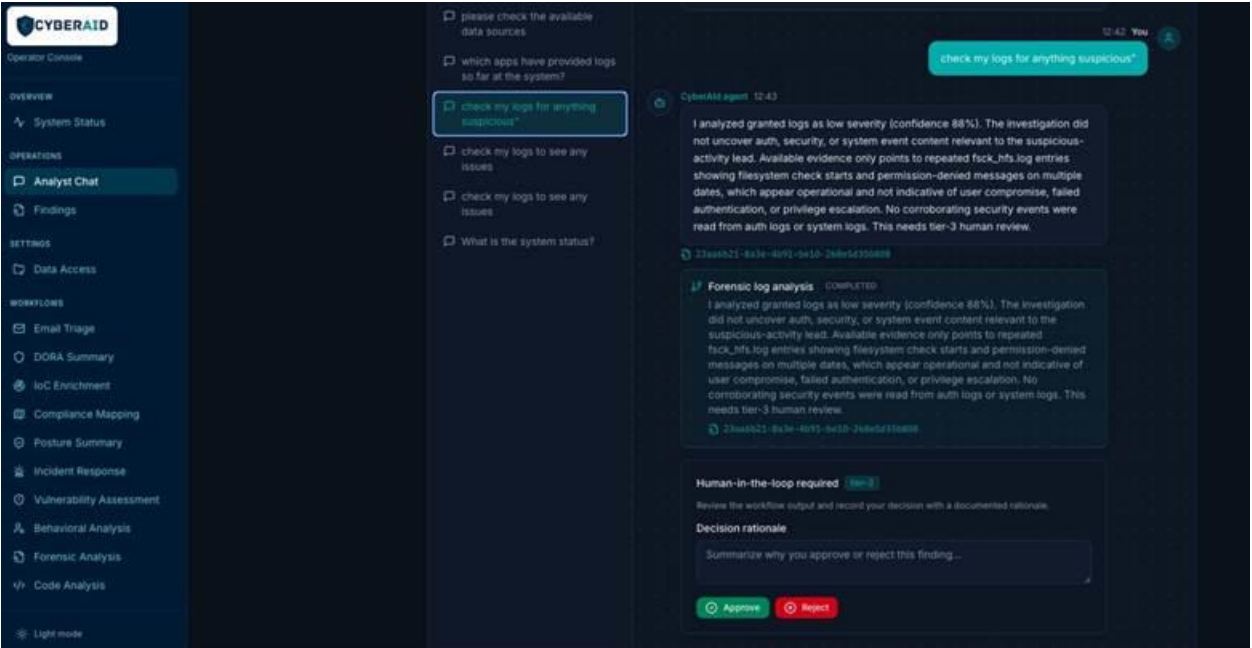


Figure 3: The analyst conversation, with a Tier 3 finding held for review

Figure 4 shows the same interface refusing a request that falls outside what it can reach. Rather than producing an answer it cannot ground, it states what it does not have access to, names the input required, and offers the route by which that input can be supplied. This is the grounding rule as an analyst experiences it: an assertion with no evidence behind it does not become a reported conclusion, and an absent capability is stated rather than approximated.

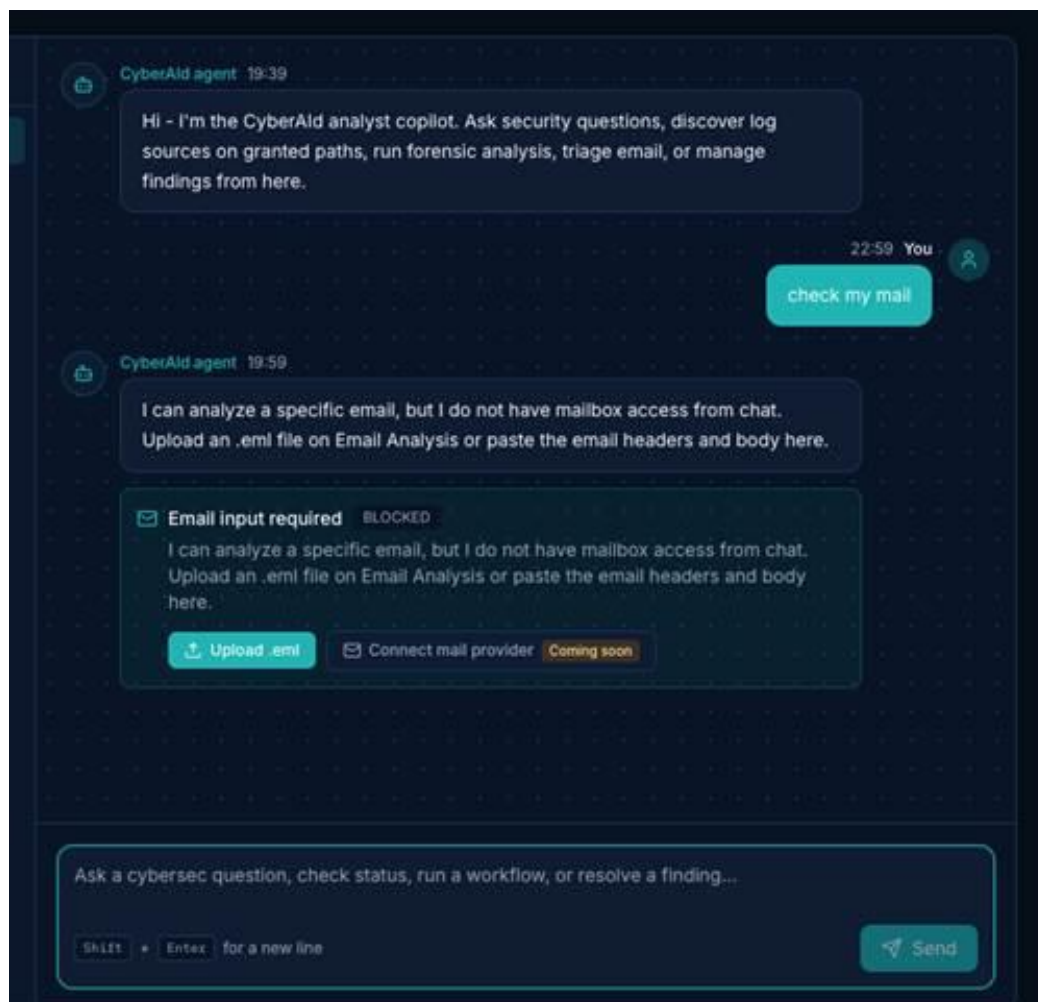


Figure 4: A request declined for want of access, with the required input named

What enters through a view is held to the same discipline as what enters through any other entry point. A request raised from a conversation is a workflow like any other, carrying the Security Context of the analyst who raised it, recorded against its entry class and written to the audit log; it is not a privileged path because a person initiated it, and it is subject to the same enforcement points as an alert arriving over the bus. Material an analyst supplies for analysis, such as an email offered for triage or a pasted header block, becomes evidence attached to that workflow and is referenced by any finding resting on it, so a conclusion drawn from analyst-supplied material is as traceable as one drawn from a detector's alert. Review decisions are inputs in the same sense and are treated below.

This set is a floor rather than a catalogue. Individual agents are expected to need views that cannot usefully be specified in advance, and those belong with the task that builds the agent rather than with this deliverable. The architecture accommodates them by extension rather than modification, on the condition that any added view observes the same three rules as the views above.

First, a dashboard displays findings, reports and audit events, and computes no assessment of its own. Counting, grouping, ordering and filtering for display are permitted; deriving a severity, recomputing a confidence or asserting a relationship the findings do not carry is not. The reasoning is the one applied to the Reporting agent, namely that a view reaching its own conclusion becomes an analysis path that never passed finding validation, evidence grounding or a review tier. Second, every read is an API call, carrying a Security Context, and there is no direct access to the finding store, the retrieval

indices or the graph, because a second read path would bypass the enforcement points along with the API. Third, no conclusion is displayed apart from its attribution: wherever a summary, severity or recommended action appears, the evidence behind it is reachable without leaving the view. The third rule binds the conversational interface as much as the others, since a generated answer is the point at which a claim is easiest to detach from what supports it.

Scope follows the context rather than the interface. Data scope, classification ceiling and permitted actions are decided by the Security Context travelling with the request, so two analysts in different organisations opening the same view see different content. Material outside a context is absent from the response rather than shown and disabled, because a visibly withheld row is itself a disclosure.

The review view is where the tiers are discharged. Tier 2 records an analyst countersignature and Tier 3 an explicit named approval with recorded justification, and both are written to the finding and to the audit log together with the identity of the approver. Neither the review view nor the conversation can lower a tier or approve past one, because the tier travels on the finding rather than being decided at the point of display. Rejections feed the evaluation criteria. The views hold no workflow state of their own, since long-running work is checkpointed by the runtime and rendered from that state, which is what allows a review to be resumed in a later session or by a different analyst.

Two decisions are left open at design stage. The task and partner responsible for building the dashboards are not fixed in this deliverable. Also, whether these views are delivered as a single console with role-dependent content or as separate surfaces is an implementation choice, provided that each reaches the platform through the same API surface and carries the same Security Context.

3.1.2 Operations & Administration CLI

The Operations & Administration CLI provides the text-based interface through which engineers deploy, configure and inspect a CyberAId installation. Where the Security Dashboards serve the analyst working an incident, the CLI serves the operator standing the platform up and keeping it running. Its scope covers initialisation of the data directory, configuration of the runtime, starting the backend service, indexing of knowledge corpora, management of operator grants over filesystem paths, execution of evaluation runs and export of audit records.

The component is implemented with Typer over the same typed models used by the backend service, so that a command and the equivalent API call validate against one contract rather than two. Commands are organised into families covering platform lifecycle, knowledge indexing, filesystem grants and evaluation. Every command that reads or writes platform data runs under a Security Context created for the command line entry point, which means an operator action carries the same authority object and leaves the same audit record as a request arriving from the operator console or the API. Filesystem grants are the case where this matters most, since a grant over a directory root is the second and independent authorisation an agent needs before a path scoped tool will resolve a file at all.

Within CyberAId, the component sits in the Presentation and HITL layer and reaches the orchestration layer through the published API surface rather than through a privileged internal path. Its position therefore mirrors that of the Security Dashboards, differing in audience and in the operations it exposes rather than in the authority it holds. The operator interfaces it realises and the grant mechanism it administers are specified in Sections 7.5 and 4.3 of D2.5.

3.1.3 Learning Centre

The Learning Centre is the platform component responsible for capacity building: it will host the training material produced by the project, make it discoverable, consumable, and certifiable, and provide a playground in which tools and methods can be tried out. The Learning Centre defines the

domain model of its entities, their relationships, and the interactions available to each actor. Two characteristics of the cybersecurity domain shape the concept: competence is demonstrated through practice, so hands-on material is a first-class content type; and the subject matter becomes outdated quickly, so content validity and review are part of the model to ensure that the dates of content creation and revision are always up-to-date. Every entity is described uniformly by the information its author must provide, what a user can see, what a user can do, and the criteria by which it can be filtered.

Regarding content organisation, three independent types of learning offer are foreseen: Learning Paths, Courses and Modules, forming a composition hierarchy down to the individual content items actually consumed, with duration and completion aggregating upwards and sequence made explicit; Tool Guides, giving practice-oriented instruction on a specific tool; and Skills, self-contained competence training units targeting providing one or more competences in a short period of time. All material is mapped onto a two-level taxonomy of thematics and subthematics with associated keywords, which enables browsing, filtering and recommendation. Assessments attach to courses and modules, and certifications of participation and of training depend on assessment results and completion criteria.

The Learning Centre is designed to support a broad range of content: expository material (video, slides, structured text, diagrams, glossaries); hands-on material (guided labs, command walkthroughs, challenge-based exercises, vulnerable-by-design applications); data-based material (captures, log and telemetry sets, forensic artefacts, analysis datasets); operational material (playbooks, checklists, detection rules, reference architectures); scenario material (tabletop and crisis simulations, incident walkthroughs, awareness exercises); assessment material; and reference material (advisories, vulnerability notes, standards extracts, case studies).

The Learning Centre will offer a playground as an isolated, disposable environment in which users can experiment with the tools and methods addressed by the training material, outside the sequence of any particular course. Its purpose is exploratory rather than assessed, such as trying a tool, reproducing a technique, testing a configuration or a detection rule against sample data. A learning offer may launch the playground with a specific baseline, which makes practical work available directly from a course, module, tool guide or skill without the material having to carry its own infrastructure. Practical outcomes that count towards certification are recognised through author-defined evidence and review, with the option of automated verification from the playground where the exercise permits it.

Actors are generalised person entities specialised into roles linked to an institution. These roles are modelled as relationships allowing a single person to hold multiple positions. Students have profiles, saved lists, interaction history and learning records. They can browse, enrol, access associated resources, use the playground, track progress, take assessments, submit evidence of practical work and obtain certifications. Authors provide metadata, structure resources and assessments including playground baselines external environment declarations and evidence criteria. Coordinators manage learning offers at curriculum level overseeing their review and revalidation cycles. Institutions offer affiliation metadata and aggregate authors and material supporting exploration by providers and assessing credibility.

This personalised recommendation system condenses user activity and profile data into a preference representation. It then scores thematic interests, identifies preferred content types and learning gaps and ranks candidate material based on thematic match, relevance to current progress, popularity and novelty. This ranking is continuously refreshed and informed by the current threat landscape.

3.2 LLM Orchestration and Agentic Analysis

The LLM Orchestration and Agentic Analysis group holds the components that reason over security information rather than produce it. The orchestration layer receives requests from the Presentation and HITL layer and activation events from the Integration and Messaging Fabric, classifies the work, dispatches it to one or more specialist components, reconciles their results, and applies a human checkpoint before a result reaches an analyst or an action reaches a monitored system. The Agentic Analysis layer holds the agents themselves.

All agents share one execution pattern. They differ in their instructions, in the tools they are permitted to call and in the knowledge partition they read, not in how they are built, secured or observed. Enforcement, evidence handling, finding validation and audit are properties of a shared runtime that an agent cannot opt out of, so adding a specialist adds domain capability without adding a new trust boundary. Agents act on the world only through the Tool Adapter Layer and reach stored knowledge only through the Context Broker, and none calls a security capability service or a model provider directly.

The components listed in this group are therefore described here in terms of the security domain each one covers and its relationship to the rest of the platform. The canonical agent architecture, the shared workflow state and context policy, the Security Context and guardrail model, the finding contract, the workflow type catalogue and the human in the loop tiers are specified in D2.5 and are not repeated in this deliverable.

3.2.1 Email Threat Detection

Wealth management and private banking present an unusually favourable target profile for socially engineered email attacks. Transaction values are high, payment instructions routinely arrive by email, client relationships are long-lived enough that an attacker can study them, and the operational tempo creates genuine urgency that a fraudulent request can imitate. Among the most frequently reported attacks in this sector are payment redirection, executive impersonation, vendor impersonation and capital-call fraud.

In recent years, because these attacks are highly frequent, the perpetrators are aiming to make their organised attacks more realistic by imitating the writing style of the author of an original email. To this end, the mechanisms that detect those attacks need to adapt to large language models to detect such attacks more quickly, in order to be proactive in terms of the actions that need to be taken to eliminate the threats. Based on the above, EMTD can tackle this problem by delivering an ontology-driven multi-agent framework for detecting socially engineered email threats in financial services.

Figure 5 Concerning its main architecture, EMTD decomposes the email analysis task into 66 atomic sub-agents grouped under 8 parent agents, which a Router Agent triggers selectively based on attack pattern classification. Figure 5 presents the holistic architecture of the EMTD framework.

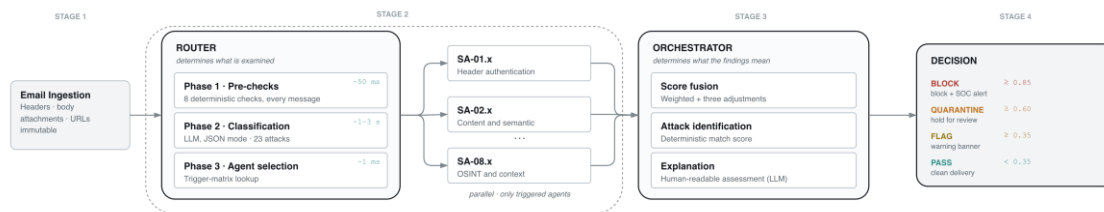


Figure 5: EMTD pipeline architecture

As shown in Figure 5, the framework is realised as a directed graph of processing nodes, implemented using the LangGraph¹ framework. The flow comprises four stages, corresponding to five nodes within the graph.

The first stage is email ingestion, where the message is provided as input to the system, including its headers, body, any attachments, and the URLs contained within it. This information is passed to the second stage, to the component termed the Router.

The second stage comprises the Router together with the analysis it initiates. The Router is a principal component that determines the scope of the examination, and its operation consists of three phases. In the first phase, eight deterministic checks are executed on every message without exception. Three of these validate the authentication of the sending domain, establishing whether the sending server is authorised to send on its behalf, whether the message carries a valid cryptographic signature, and whether the domain's own alignment policy is satisfied. Another two examine the message header for inconsistencies characteristic of impersonation: a mismatch between the sender address and the address to which a reply would be directed, and a discrepancy between the displayed sender name and the underlying address. Two further checks detect deliberate obfuscation of the message text, namely the substitution of visually similar characters and the insertion of zero-width characters that alter a string without altering its appearance. The eighth establishes whether any prior correspondence exists with the sender, since a first-time contact carrying a payment instruction warrants closer examination than an established one. The complete set of pre-checks is given in Table 3.

Code	Check	Purpose
SA-01.1	SPF Validator	Verifies the sending server is authorised for the domain
SA-01.2	DKIM Validator	Verifies the cryptographic signature of the message
SA-01.3	DMARC Evaluator	Evaluates domain alignment policy across SPF and DKIM
SA-02.8	Reply-To Mismatch Detector	Detects replies redirected to a different address
SA-02.10	Homoglyph Detector	Detects visually similar character substitution
SA-02.11	Zero-Width Character Detector	Detects invisible characters used to evade matching
SA-03.1	Display Name vs Address Checker	Detects a displayed name inconsistent with the address
SA-03.6	First-Time Contact Flagger	Flags absence of prior correspondence with the sender

Table 3: Deterministic pre-checks executed on every message.

In the second phase, a large language model analyses the content of the message together with the results of the preceding checks and identifies candidate attack patterns from a catalogue of 23 known attack types. The catalogue is organised into four categories: eleven types of impersonation and social engineering, six process-specific attacks that target the institution's operational procedures rather than the identity of a correspondent, five advanced technical attacks exploiting the mechanics of mail and web infrastructure, and one pre-attack intelligence type covering reconnaissance conducted in

¹ <https://reference.langchain.com/python/langgraph/overview>

advance of an attack. Most of these attack types therefore depend on semantic and contextual judgment rather than on technical indicators, which is what motivates the use of a language model at this stage. The classifier is over-inclusive by design: a false negative at the routing stage is unrecoverable, since a threat that is not routed for examination is never examined, whereas a false positive results only in additional agents being executed at negligible cost.

In the third phase, each candidate attack identifier is resolved into the set of checks required to examine it. A trigger matrix, maintained within the knowledge layer, associates every attack type with the sub-agents that provide evidence for it, and the union of the agents triggered across all candidates constitutes the execution plan. This phase performs no inference, it is a lookup operation, so that an identical set of candidates always yields an identical execution plan. The plan is then executed within the same stage: the selected agents are invoked concurrently, while those not selected are never invoked, and each returns a finding that carries its outcome, a confidence value, a severity level, and the specific evidence that produced it.

The third stage is orchestration, where the Orchestrator component fuses these findings into a weighted threat score, identifies the attack type by matching them against the expected profile of each candidate pattern, and generates a human-readable assessment for the analyst. Specifically, the Orchestrator combines the findings returned by the executed agents into a single threat score, computed as a weighted ratio:

$$\text{threat_score} = \sum (\text{confidence}_i \times \text{priority_weight}_i) / \sum (\text{weight}_i)$$

where the numerator is summed over suspicious findings only, and the denominator over all findings actually produced. Each agent is assigned a priority level reflecting the evidential strength of the check it performs, and this level determines its weight in the computation. A suspicious finding therefore contributes in proportion both to the confidence with which the agent reports it and to the significance of the check itself, so that a moderately confident authentication failure outweighs a highly confident but contextually weak signal.

The resulting value lies in the interval [0, 1] and is interpretable directly as the proportion of available evidential weight that points toward a threat. Because the computation is a deterministic function of the findings, and because each finding carries the artefact that produced it, any score can be decomposed into its constituent contributions and traced back to the specific checks and evidence that generated it.

Finally, the fourth stage maps the resulting score onto one of four decision levels: (a) BLOCK, (b) QUARANTINE, (c) FLAG, (d) PASS, each accompanied by the complete set of findings that produced it, which is declared as the output of the system.

3.2.2 Anomaly Detection

Anomaly detection is a challenging task in the financial services industry. As technology evolves, perpetrators continually seek new means of engaging in illegal activity such as fraud, identity theft, network intrusion, account takeover and money laundering, which may result in outcomes as severe as system disruption and service breakdown. Given the dynamic nature of the financial environment, combined with the rate at which data now moves, detecting anomalies in real time and at scale becomes increasingly difficult.

Three properties of this setting make the problem harder than anomaly detection in general. Normal behaviour cannot be specified in advance, since every institution operates differently and legitimate activity drifts continuously as processes, systems and personnel change. Labels are scarce and rarely transferable, because confirmed incidents are Infrequent, commercially sensitive, and shaped by the institution in which they occur, the same sensitivity extends to the underlying telemetry, which

typically cannot leave the environment that holds it. And the alert budget is small, since at the volumes involved even a very low false-positive rate produces more alerts than a security team can triage. Compounding these, attacks against financial institutions rarely consist of a single decisive event: they unfold over hours or days, across many observations, each individually defensible. Detection must therefore reason over sequences and over deviations from an entity's own history, rather than over isolated records.

CyberAid can extend its ability to accommodate the following forms of input, which differ in structure and in the temporal reasoning they permit. The first is tabular event records, such as firewall and Netskope logs together with abnormal events raised within the system. These are high in volume, semi-structured and the most readily available form of telemetry in this context. They constitute point observations and become the principal subject of analysis once aggregated per entity and time window. The second is time series records, including data derived from the activity of institutional users within a shared cloud environment, where observations are dependent on time and the informative signal lies in periodicity, level shifts and trend breaks rather than in any single value.

The approach follows from the problem's properties rather than any single technique. Because the informative signal lies in deviation from an entity's own history, observations are aggregated per entity and time window, so that the unit of analysis carries both the behaviour of that window and the baseline against which it is compared. This aggregation also means that analysis operates on derived behavioural summaries rather than on raw records, so that sensitive content remains within the environment in which it was produced and only scores and indicators are exposed to other components. Detections are then ranked by severity and by the confidence of the deviation, so that the volume surfaced to analysts remains within what a security team can realistically triage.

The aim is to tackle anomalous events by exploiting the aforementioned data types, and to recommend appropriate mitigation actions once a deviation is confirmed. Given that abnormal events are rare compared to normal ones, LLMs are employed to generate synthetic abnormal samples, offering a cost-effective means of addressing label scarcity**, without further exposure of real incident data,** and of improving the robustness and generalization of the anomaly detection system.

3.2.3 Automated Log Analysis & Event Correlation

The Automated Log Analysis & Event Correlation component provides CyberAid with an automated capability for processing, analyzing and correlating large volumes of heterogeneous log data originating from financial systems and Critical Infrastructures. Its primary purpose is to transform raw and potentially fragmented log events into contextualized security intelligence, reducing the dependence on manual log inspection and isolated analysis. The component identifies security-relevant patterns and anomalous activities, while correlating related events across different sources to provide a more comprehensive view of ongoing or emerging security incidents.

A distinguishing characteristic of the component is its ability to move beyond the analysis of individual log entries and exploit the relationships between entities, events and security-relevant information. Rather than treating logs as independent records, the component leverages a knowledge graph as a structured source of contextual information, allowing events and their relationships to be retrieved and analyzed in conjunction with the knowledge represented in the graph. This enables the identification of connections between seemingly unrelated events, the enrichment of log-derived evidence with contextual information, and the correlation of observations across different systems and time periods. Such an approach is particularly relevant for financial environments, where security incidents can involve multiple interconnected systems and where individual log entries may provide limited evidence when considered in isolation.

The component utilizes LightRAG[7] as the Retrieval-Augmented Generation (RAG) framework operating on top of the available knowledge graph. LightRAG combines retrieval capabilities with graph-based knowledge representation to support the analysis of security events and the discovery of relevant relationships within the available security knowledge. The knowledge graph provides the underlying structured representation of entities and their interconnections, while LightRAG enables the retrieval of relevant contextual information to support event analysis and correlation. The resulting architecture allows log information to be analyzed together with existing cybersecurity knowledge, facilitating contextual event interpretation, identification of related activities and generation of higher-level security insights. The component is therefore designed to support the transition from volume-oriented log processing towards context-aware analysis, where the significance of an event is determined not only by its individual characteristics but also by its relationships with other events, entities and known security information.

Within CyberAid, the component serves as the automated analysis and correlation layer between raw operational security data and the project's higher-level threat detection, investigation and response capabilities. It processes log information from financial systems and contributes correlated events, detected patterns and contextualized security findings to the CyberAid security analysis workflow. These outputs can subsequently be combined with information from other project components, including threat intelligence, vulnerability assessment, monitoring and risk assessment, to provide a more complete representation of the security situation. By transforming heterogeneous and high-volume logs into structured and contextual security intelligence, the component supports the LLM-driven orchestration of CyberAid and enables downstream capabilities to reason over correlated evidence rather than isolated log records. Ultimately, its role is to facilitate proactive threat detection and accelerate incident investigation and response in financial Critical Infrastructures, while reducing the operational burden associated with manual log review.

3.2.4 AI-Driven Pentesting & Attack Generator

This component aims to utilize LLMs developments and provide support for AI assisted penetration testing, tackling both test planning and test execution phases. During the planning stage, LLMs can analyze system documentation and adapt the type of pen-tests based on system characteristics, available resources, and the intended test goals or objectives. Based on this information, the models can generate testing scenarios and describe the sequence of actions required to assess the system's security posture. Additionally, test execution and tool management can be handled through tool-calling functionalities available in most LLM framework environments, allowing the model to initiate testing steps and orchestrate the penetration-testing process.

Applying LLMs to penetration testing can improve penetration-testing activities by reducing the effort required during test preparation and planning. By adapting specialized models to the context of each target environment, it becomes possible to define testing objectives and procedures more efficiently while maintaining flexibility across different scenarios. Beyond the planning phase, LLMs can support the automation of tasks such as environment assessment and the coordination of tools commonly employed in cyber-attack simulations. Recent developments in agent orchestration systems further extend these capabilities by allowing autonomous testing processes to adjust their execution steps according to information gathered throughout the assessment. However, introducing LLMs into the testing workflow also raises important concerns regarding the reliability, accuracy, and transparency of their decisions. As these systems operate with limited visibility into their internal reasoning processes, incorrect conclusions or inappropriate tool usage may introduce additional security risks. To address these concerns, autonomous penetration-testing solutions should incorporate mechanisms that promote accountability and traceability, including human-in-the-loop validation for critical operations, comprehensive reporting of collected data, and detailed records of the actions performed throughout the assessment, together with the reasoning and evidence that support each decision.

This AI-driven pentesting and attack generator component will be integrated into CyberAid by registering as a LangGraph agent that can be called and used by the agent orchestration system as a service. It is part of a service workflow composed of other components in the project, such as the Digital Twin environment where the attacks can be executed and simulated without harming real resources. The results from penetration testing are then redirected to the Analysis and Reporting component as a list of identified vulnerabilities, along with the data and information that supported the LLM decision system and the methodology generated in the processes.

3.2.5 Code Analysis

The Code Analysis component examines source code and continuous integration and delivery configuration for malicious patterns, insecure practice and supply chain exposure, and produces structured findings traceable to established control frameworks. Financial institutions carry a mixture of internally developed applications, vendor components and pipeline configuration, and many of the security-relevant defects in that mixture are ones a reviewer would recognise at a glance but that nobody has time to read. The component addresses that gap by examining repositories and pipeline events continuously rather than at review milestones.

The component will be realised as a native agent on the shared agentic runtime. Its workflow proceeds from a deterministic scan across the repository, through model reasoning over the regions that scan identified, to the emission of a validated finding. Evidence is carried as file and line references rather than as copied source, which matters where the code being examined is itself confidential.

Within CyberAid the component belongs to the Agentic Analysis layer and is delivered under T3.3. It is triggered by pipeline events and by repository scan requests, and its findings can be chained into a compliance mapping workflow where a defect engages a regulatory obligation. Its realisation pattern, the workflow type it owns and the schema of the findings it emits are specified in Sections 8.2, 8.3, 9.2 and 10.1 of D2.5.

3.2.6 Pentesting Outcomes Analysis & Reporting

This component aggregates the results of penetration testing and code analysis to consolidate identified vulnerabilities into a unified view of the system's security posture. By correlating findings from multiple tools and evaluation methods, it can identify duplicate results, highlight relationships between vulnerabilities, and provide additional context regarding their potential impact. This centralized analysis provides a more comprehensive understanding of the identified security issues while reducing the effort required to review findings from different assessment sources independently.

The analysis process begins by mapping identified vulnerabilities to established standards, such as Common Vulnerabilities and Exposures (CVEs), allowing the findings to be contextualized and quantified according to recognized security references. These vulnerability repositories provide detailed information regarding the affected components, the nature of the exposure, and the recommended remediation actions. The identified threats are then evaluated based on factors such as the likelihood of exploitation, their impact on the system, and the criticality of the affected assets within the architecture. This assessment should also consider the types of resources affected and their role within the overall system. By combining this information, the component provides a structured view of the identified risks and supports evidence-based decision-making by the orchestrator when prioritizing mitigation and remediation activities.

The integration of this component is directly dependent on the tools developed in WP3, including penetration testing, code analysis, and digital twin technologies. Together, these components support the complete lifecycle of an agent capable of providing vulnerability assessment services, from the identification of security issues to the collection of supporting evidence. The resulting findings and

associated context are then made available to the orchestrator, enabling evidence-based decision-making when selecting and prioritizing response actions. In addition, the component is connected to users through OLISTIC, providing access to vulnerability information, supporting evidence, and risk assessments. This information not only improves the visibility of identified security issues but also facilitates the auditing and validation processes required throughout the system's operation.

3.2.7 Risk & Compliance Assessment

The Risk & Compliance Assessment component scores risk and exploitability in the context of the assets a deployment actually holds, and identifies regulatory control gaps against DORA and NIS2. Two questions recur in a financial security function and neither is answered by a detection on its own. The first is what a given vulnerability is worth in this institution, given which assets depend on it and what in turn depends on them. The second is which obligation a finding engages and whether the control that obligation expects is present. The component answers both over the same asset model.

The component is an agent rather than an analytical engine. The assessment itself is performed by the Risk Assessment Engine positioned in the Security Capability Services layer, which the agent reaches as a tool through the Tool Adapter Layer. The agent supplies what the engine does not, namely scope enforcement, evidence referencing, severity and confidence expressed in platform terms, and a validated finding. Regulatory corpora and the DORA and NIS2 reporting templates are read from the knowledge layer through the Context Broker under the same scope enforcement. The component owns two workflow types. One covers vulnerability assessment, triggered by scanner output, by a published CVE or by a periodic reassessment, and calls the engine for exploitability and asset dependency analysis. The other covers compliance mapping, triggered by an analyst request or chained from an existing finding, and reasons over the control set the applicable regime expects against the controls the deployment records.

3.2.8 Reporting

The Reporting Agent composes findings into standardised documents for readers who will not read findings. A supervisor receiving an incident notification, an executive reading a posture summary and an auditor reconstructing a decision each need the same underlying material presented differently, and each expects a format the platform must produce rather than improvise. Five report types are anticipated, covering regulatory notifications drafted to a required format and timeline, incident narratives, posture summaries over a defined period or scope, mitigation recommendation sets, and ad hoc analyst queries answered across the finding corpus.

The component is realised as a native agent on the shared runtime and runs dedicated report composition workflows, one per report type, over the findings the specialist agents have emitted. A workflow proceeds in three steps. The requested report type determines which findings are in scope, that scope is checked against the requesting Security Context, and the selected findings are passed to the language model with the template for the type, which produces the narrative while the structure, ordering and citations remain fixed by the template. The scope check matters because a report is the artefact through which material from one scope could otherwise reach a reader in another, which makes report generation an enforcement point rather than a formatting step.

3.3 Security Capability Services

3.3.1 SIEM / XDR Platform

The SIEM / XDR Platform provides CyberAid with centralized monitoring, detection and analysis of security-relevant activity observed across the monitored financial environment. It is realized through Wazuh, an open-source SIEM/XDR engine, configured to ingest structured events produced by CyberAid's kernel-level monitoring and log-forwarding components. Incoming events are normalized

through custom decoders and evaluated against a rule set extended with financial-sector-specific detection logic, allowing the platform to identify and classify patterns such as suspicious network connections, connection bursts, unexpected process execution and potential privilege-escalation or data-exfiltration behavior.

Technically, the platform is built around the Wazuh² manager and its decoder and rule-processing pipeline, complemented by OpenSearch for indexing, search and visualization. Events originating from the eBPF-based monitoring components are parsed by purpose-built decoders that expose contextual information, including timestamps, process information and network communication details, which can then be evaluated by detection rules and assigned corresponding severity levels. Detection outputs are indexed in the Security Event & Log Store and can be visualized through dedicated dashboards summarizing event severity, activity over time, detected attack categories and critical security events.

Within CyberAId, the SIEM / XDR Platform provides the link between low-level system and network telemetry and higher-level operational security monitoring. Its structured monitoring and detection outputs can be made available to CyberAId's broader alert-generation, analysis and investigation capabilities through the platform's integration mechanisms. The resulting security information can also support digital-forensics and post-incident-analysis activities by providing contextual evidence for incident reconstruction and root-cause analysis. In this way, the component contributes to the monitoring, detection and reporting capabilities that underpin financial-sector cybersecurity and operational resilience, including relevant regulatory expectations under DORA and NIS2.

3.3.2 Kernel-Level Monitor (eBPF) + Connection Tracer

The Kernel-Level Monitor provides CyberAId with visibility into network, host and process-level activity below the application layer, observed directly from the operating-system kernel. It is based on extended Berkeley Packet Filter (eBPF)³ technology, enabling the monitoring of network connections and process activity without requiring modifications to the applications being protected. This is particularly relevant in financial infrastructures, where payment, banking, authentication and trading services need to be monitored with minimal interference. The component combines network connection tracing with host-level system and process-activity monitoring to provide low-level visibility into host and network behavior.

Technically, the component uses eBPF-based monitoring mechanisms and associated user-space tooling that interact with relevant kernel-level hooks to capture security-relevant host, process and network activity. Captured activity is transformed into structured security events containing contextual information required for downstream analysis, including timestamps, process information and network communication details. These structured events can subsequently be evaluated by the platform's detection capabilities to identify security-relevant behavioral patterns. The component is designed to support extensible monitoring of system activity and to operate with low runtime overhead, allowing its monitoring capabilities to be adapted to the requirements of financial-sector environments.

Within CyberAId, the Kernel-Level Monitor provides kernel-level telemetry that can be made available to the platform's monitoring, detection and investigation capabilities through the platform's integration mechanisms. Its structured monitoring events can support SIEM/XDR-based analysis and indexing, while also providing contextual information that can support digital-forensics and post-incident-analysis capabilities for incident reconstruction and root-cause analysis. In this way, the component provides a common source of low-level security telemetry supporting both real-time

² <https://wazuh.com>

³ <https://ebpf.io>

security monitoring and subsequent investigation while remaining non-intrusive to the applications and services being observed.

3.3.3 Graph Analytics Engine

The Graph Analytics Engine provides CyberAId with graph-based analytical capabilities for detecting sophisticated fraud patterns and suspicious transaction networks in financial environments. The component analyses the relationships among financial entities, transactions and observed behaviours to identify connections and coordinated activity that may not be visible through transaction-level or rule-based analysis. It is intended to support the detection of fraud typologies such as transaction rings, rapid fund movement, structuring and layering, while enabling temporal analysis of how financial relationships and transaction flows evolve.

The engine operates on the information maintained in the Fraud Property Graph and is specified to combine Neo4j and Cypher-based pattern matching with specialised graph-processing algorithms implemented through Python graph-analysis libraries. These mechanisms support relationship exploration, path analysis, community and network-structure analysis, and the identification of anomalous or coordinated behaviour across interconnected financial entities. The analytical capability is designed to support both predefined fraud-pattern searches and broader graph exploration, while scalability mechanisms such as partitioning and parallel processing may be applied as needed to maintain near-real-time analysis under high transaction volumes. Its findings can also support analyst-facing visualisations through which suspicious networks may be filtered and investigated according to time windows, entity types, paths and available risk-related information.

Within the CyberAId reference architecture, the Graph Analytics Engine is positioned in the Security Capability Services layer as a deterministic analytical service. It exchanges graph entities and relationships with the Fraud Property Graph and produces graph-derived findings that can be exposed to the LLM orchestration layer for contextual interpretation and explanation. Relevant findings may also be provided to the Risk Assessment Engine, allowing graph evidence to contribute to the wider assessment of cyber and fraud risk. In this way, the component connects financial transaction-network analysis with the platform's broader monitoring, knowledge-driven reasoning and decision-support capabilities, while its detailed graph algorithms, implementation and validation are addressed through the dedicated WP4 activities.

3.3.4 Digital Twin Service

The Digital Twin Service provides CyberAId with a controlled testing environment for evaluating cybersecurity defence capabilities in representative financial infrastructure scenarios. The component builds upon the NITRO project, whose cyber range environment is adapted and enhanced within CyberAId to address the specific requirements of financial infrastructures. In this environment, separate attacker and defender virtual machines can be created, configured and operated without exposing operational financial systems to the activities performed during testing. Through a central platform dashboard, authorised users can access the environment, manage the lifecycle of the virtual machines, monitor their operational status and establish secure remote SSH connections. The isolated virtual environments allow attack and defence activities to be executed, observed and compared under controlled and repeatable conditions.

Technically, the Digital Twin Service builds upon a Proxmox⁴-based environment, with the CyberAId implementation initially focusing on the infrastructure and observability capabilities required to deploy services, monitor resources and capture cybersecurity incidents. The environment is intended

⁴ <https://proxmox.com/en/>

to support monitoring, asset configuration management, threat detection, vulnerability assessment, penetration testing and AI-assisted analysis tools. Integration with Wazuh and Tetragon is being investigated to support security monitoring and the collection of kernel-level telemetry through eBPF. Within CyberAId, the enhanced Digital Twin Service provides the execution environment for the AI-Driven Pentesting and Attack Generator. Penetration testing scenarios generated according to a specified test type and objective can be deployed against the defender environment, while the resulting activities, defensive responses and observations can be monitored, analysed and exported for subsequent reporting. Further integration with the Risk Assessment Engine is also foreseen so that testing outcomes can contribute to the assessment of vulnerabilities, attack paths and defensive measures. In this way, the Digital Twin Service supports the realistic validation of security controls and cybersecurity tools while maintaining separation between experimental activities and operational financial infrastructures.

3.3.5 Risk Assessment Engine

The Risk Assessment Engine provides CyberAId with continuous, asset-centric risk and exploitability assessment. The component is realised through OLISTIC, the cyber-physical security risk assessment and management platform, which supports dynamic and near-real-time risk assessment aligned with ISO/IEC 27001. OLISTIC allows an organisation to define, graphically represent and document the assets within the scope of assessment, to declare the dependencies among them, and to associate each asset with applicable threats and vulnerabilities together with their likelihood, resulting impact and confidentiality classification. On this basis it maintains a continuous assessment process that reflects changes in the asset structure and in threat models as they occur, and proposes countermeasures derived from established security policies and standards.

A distinguishing characteristic of the platform is its treatment of dependencies and cascading effects, which matters in financial infrastructures where the consequences of an incident rarely remain confined to the asset first affected. OLISTIC models multi-order dependencies between interacting entities and supply chain actors, identifies the critical paths along which impact propagates and visualises interdependencies, criticality levels and probabilities. Its exploitability assessment uses pathfinding and network analysis techniques over a graph-based representation of asset interconnections to identify attack paths and to produce exploitability measures that quantify the significance of each vulnerability, supporting the systematic prioritisation of remediation. The platform is further complemented by the OLISTIC Threat Intelligence Network, a honeypot-based capability oriented towards the detection of advanced persistent threats. Technically it is built on a graph-based asset representation and integrates pathfinding and network analysis algorithms and uses the Drools business rule management system to encode assessment rules and compute attack paths. It is deployed as a containerized application and exposes its capabilities through APIs.

Within CyberAId, the Risk Assessment Engine consolidates relevant asset, dependency, vulnerability, monitoring and penetration-testing information to calculate risk and exploitability measures and identify mitigation candidates. It exposes these results through governed tool interfaces. Risk & Compliance Assessment provides contextual interpretation and regulatory mapping, while Reporting composes the resulting validated information for technical, executive or compliance audiences.

3.3.6 Federated Analytics & Compliant Data Sharing

The Federated Analytics mechanism in CyberAId aims to provide a means of information sharing between components or entities within the platform, as well as between platforms of the CyberAId ecosystem, in order to avoid data silos and enable more combinatorial capabilities of the various tools.

The mechanism aims to exploit the annotation (or decoration) mechanisms of Python, in order to provide a set of capabilities related to pushing relevant data between components, in an effort to enrich their awareness and information sources. Annotations are a powerful and common way of injecting functionalities within existing code, while being a good candidate for LLM agents incorporation due to their structured nature (i.e. they can be included as a library and enrich existing code).

The four levels of information that are expected at this stage to be available for sharing include:

- Baseline data that can be shared as raw data, after being properly sanitized, so that similar components across organizations can enrich e.g. training sets
- Context data, such as guidelines, rules, mark up files, skills etc., which is considered also very important in the CyberAid LLM agent scope and framework, as a mean to enrich setup and operation of agents
- Function logic (e.g. analytics functions, agentic tool functions etc) that may be shared between CyberAid developers in a form of a common repo of agentic functions or orchestration logic. This involves sharing of the function code developed for a given purpose.
- The output of a function (e.g. a processed event), that can be forwarded to other components (e.g. monitoring or messaging components) as a mean to be distributed to entities or components that need to enrich their own input considerations for the detection of more complicated events. This will create the ability to share more abstract information that can aid a tool to understand the wider context of a cyberthreat

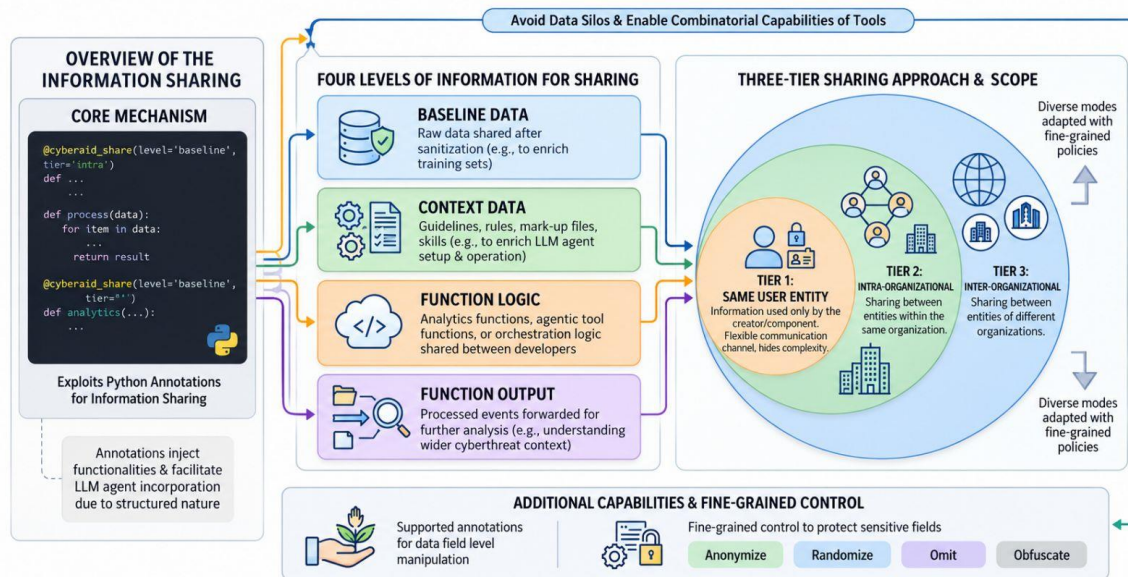


Figure 6: Overview of Annotation Mechanisms Approach for the CyberAid Platform

Sharing of the information (of whatever level) is expected to follow a three-tier approach. The basic approach is sharing between the same user entity, i.e. enabling the information to be used only by that user/component. The aim of this mode is to enable a much more flexible communication channel that is more abstract and hides the complexity of pushing data within the CyberAid platform and the need

to create specialized clients for different interactions. This is based on the incorporation within the annotation logic of needed communication clients for a variety of protocols, with the ability to parameterize them within the annotation configuration. Following, the second tier includes sharing between the entities of the same organizational entity (intraorganizational sharing) while the final tier includes sharing between entities of different organizations. This will enable diverse sharing modes that can be adapted at the base level of information at a fine-grained manner, based on the policies that need to be in place for each use case.

Other cases of supported annotations are expected to be developed, in order to enable a fine-grained manipulation at the data field level, including abilities to anonymise, randomise, omit or obfuscate specific sensitive fields that should not be included in the sharing process.

3.3.7 Quantum Token 2FA Service

Within the CyberAid component landscape, the QUBO quantum token-based two-factor authentication system (Q2FA) is positioned as a Security Capability Service that provides an additional, cryptographically unforgeable authentication layer for financial transaction flows. Rather than replacing existing authentication mechanisms, Q2FA is designed to operate as a complementary verification step, invoked at the point where a transaction or data exchange requires an elevated level of assurance. The Q2FA capability integrates into the Integration and Messaging Fabric through a defined REST interface, allowing it to be invoked consistently across CyberAid use cases without requiring changes to its underlying logic.

The security guarantee provided by Q2FA rests on the no-cloning theorem: because an unknown quantum state cannot be copied without disturbing it, any attempt to intercept, duplicate, or tamper with a quantum token is inherently detectable. This gives the framework a tamper-evidence property that classical token-based two-factor authentication cannot offer, making it particularly suited to scenarios where non-repudiation and single-use guarantees are critical, such as inter-institutional transaction confirmation.

The Q2FA setup consists of two QUBO-provided modules, the Quantum Trusted Token Provider module (QTTP) and the Quantum Token User module (QTU), which interact with two user-implemented roles, the initiator and the verifier. The initiator originates the data or transaction to be authenticated, while the verifier acts as the counterpart that confirms its authenticity through the Q2FA process. Because the QTTP relies only on data derived from the transaction (a hash value and a data reference) rather than on the transaction content itself, the same interface can be reused across different transaction types by adapting naming conventions only, without altering the underlying data flow or module logic. This separation between the authentication mechanism and the transaction content is what allows Q2FA to be treated as a generic, reusable Security Capability Service.

Standard Input Data Definition:

To support reuse across use cases, a standard input data structure is defined for any process that invokes Q2FA. This structure is independent of the specific business context and consists of the following elements:

Authentication Input

- **User ID:** a shared secret unique identifier for the initiator, known to both initiator and verifier prior to any transaction.
- **Data Reference:** a unique identifier agreed between initiator and verifier for the specific transaction or data object being authenticated.
- **Hash:** a hash value computed over the transaction data combined with the User ID, generated

independently by both the initiator and the verifier for comparison during authentication.

Security Configuration Input

- **Hash Length:** selectable in bits (256 or 512), trading off transaction speed against cryptographic strength.
- **Security Level:** a numeric parameter (1–10) defining the number of quantum states used per logical bit of the hash, trading off transaction speed against security assurance.

Process Control Input (queried, not submitted)

- **Transaction Status Request:** allows the verifier to query progression, estimated completion time, and success status at any point during authentication.
- **Error Status Request:** allows the verifier to retrieve error names and logs associated with the transaction, returned as an empty list where no errors occurred. This gives implementing systems a consistent way to detect and diagnose failed authentication attempts.

All interface exchanges use JSON payloads for both GET and POST endpoints, following QUBO's standard OpenAPI specification. The interface additionally supports an optional PQC-encrypted transport mode, implemented as a wrapper around the base OpenAPI definition; in this mode, the encrypted payload's underlying structure must still conform to the standard input data definition above.

Standardization Across Use Cases:

Because the Q2TP module operates solely on the hash and data reference, the same input structure applies unchanged regardless of the transaction context. Use-case-specific adaptations are limited to endpoint and parameter naming, while the underlying data contract remains identical, as illustrated below for the generic (basic) and an example (trading) use case:

Standard Field	Basic Use Case Naming	Trading Use Case Naming
Data Reference	Data reference	Data reference on trading data
Hash	Hash	Hash of trading data and User ID
Authentication trigger (Get)	Get Authentication Request	Get Trading Data Authentication Request
Authentication submission (Post)	Post Authentication Request	Upload Trading Data
Status query (Get)	Get Transaction Status	Get Transaction Status of Trading Data Verification
Error query (Get)	Get Error Status	Get Error Status of Trading Data Verification

Table 4: Q2FA input and operation naming across use cases

This mapping confirms that Q2FA can be adopted as a standard, blueprint-referenceable component: any use case invoking it need only map its transaction data to the four standard input fields (User ID, Data Reference, Hash, Security Configuration) to obtain quantum-token-based authentication. Because the mapping is limited to naming rather than structure, onboarding a new use case is largely a configuration exercise rather than an integration effort, reducing both implementation cost and the risk of inconsistent authentication behavior across the platform. This is consistent with the blueprint methodology described in Section 2.2, which connects a defined objective to applicable requirements, architecture elements, components, interactions, and expected validation evidence.

3.3.8 Digital Forensics & Post-Incident Analysis

The Digital Forensics & Post-Incident Analysis capability provides CyberAId with post-detection forensic processing and analysis functions supporting the investigation of cybersecurity incidents. It builds on the outputs of the platform's monitoring and incident-identification capabilities, consuming relevant security events, detection findings, alerts, logs and other available evidence originating from the monitored financial environment and CyberAId monitoring components. The capability supports evidence preservation and integrity, incident timeline reconstruction, root-cause analysis, MITRE ATT&CK contextualization and the generation of structured forensic findings and reports, enabling investigators to establish a coherent representation of the activities and evidence associated with an incident.

Technically, the capability processes collected forensic evidence together with its associated contextual and temporal information to identify relationships between relevant events, reconstruct the sequence of activities associated with an incident and determine potential causes and affected elements. Historical security information retained in the Security Event & Log Store, together with findings and telemetry generated by monitoring and detection components such as the SIEM / XDR Platform and Kernel-Level Monitor, can provide inputs to this analysis. The resulting forensic case can contain evidence references, integrity and provenance information, reconstructed timelines, root-cause findings, MITRE ATT&CK mappings and structured reporting information for subsequent investigation and incident-response activities.

Within CyberAId, the Digital Forensics & Post-Incident Analysis capability is positioned in the Security Capability Services layer and provides the forensic-analysis stage following monitoring and incident identification. It can obtain relevant evidence from the Security Event & Log Store and other monitoring and detection components. Its structured forensic outputs can be made available through the platform's integration mechanisms to the LLM Orchestration, incident-response, reporting, and human-in-the-loop capabilities for contextual analysis, coordination and decision support.

3.4 Integration and Messaging Fabric

3.4.1 Event & Alert Bus

Security events are generated on the platform in one of two ways. One is through pre-defined rules applied when a specific event occurs, the other is non-deterministic and relies on AI-based logic. When produced, these events are published to a security event topic. Afterwards, these messages are consumed in an attempt to detect suspicious or potentially malicious patterns. Pattern detection takes into consideration the sequence and type of events in order to flag a specific pattern as an alert. When detected, the sequence of events is assigned a severity level following the CVSS framework guidelines and generates an alert message. This message is published to another topic, an alert topic to be displayed to the user on the SIEM and consumed by incident response agents.

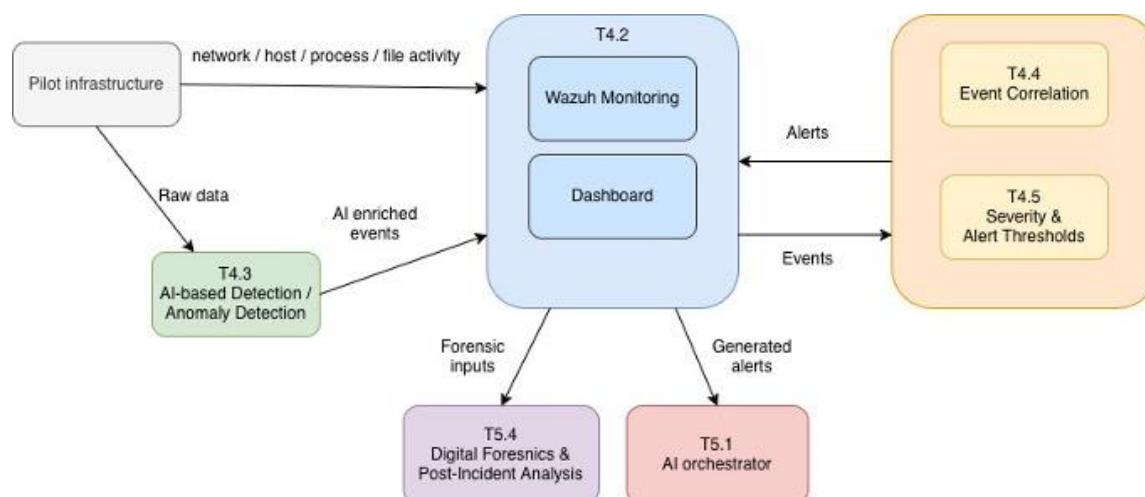


Figure 7: Event and alert interactions

Pattern detection is done with the help of an AI agent performing CEP. The agent looks at a sliding window of incoming events, flagging suspicious activity based on historical user behaviour and known system vulnerabilities. The alert's assigned score reflects the severity of the threat. The consumers of the alerts use this to highlight and prioritize which incidents to tackle first. Furthermore, the alert contains information related to the events that triggered it and provides additional contextual information with mitigation strategies or other actions that can be taken.

3.4.2 Tool Adapter Layer & Registry

The Tool Adapter Layer & Registry provides the single governed path through which the agentic components reach anything outside the orchestration layer. Security capability services, external security tooling and the filesystem are reached through it and are never called directly by an agent. The registry holds, for every tool identifier the platform exposes, the declared argument schema against which a call is validated and the handler that executes it, which is what makes the set of things an agent can do an inspectable list rather than a property of its prompt.

Three adapter types are supported. Native typed adapters are the preferred form for capability internal to the platform, since a declared schema and a typed result give the runtime something it can validate. Command line wrappers cover established security tooling that exposes no usable API. MCP adapters cover external tools and services, where a shared protocol removes the need for a bespoke integration per product. Whichever type is used, the invocation sequence is identical.

Within CyberAid the component belongs to the Integration and Messaging Fabric and consumes the security context established by Authentication & Access Control. It also carries the return path of the platform. The Event & Alert Bus transports observations upward from the detectors to the orchestration layer, while an approved action travels downward through the adapter layer, so that anything changing the state of a monitored system passes through a single accountable path.

3.4.3 Log Collector & Forwarding

The Log Collector & Forwarding component supports the collection and forwarding of logs and security events from the monitored environment and the platform's security capability services into CyberAid's downstream processing, analysis and storage services. It provides a common mechanism for collecting security-relevant information and forwarding it towards downstream components responsible for processing, analysis and storage.

Technically, the component uses lightweight log collection and forwarding mechanisms such as Filebeat⁵ to collect structured security events together with relevant host and application logs and make them available to downstream CyberAId services for subsequent processing and indexing. The component is intended to preserve the structure and contextual information associated with the collected events so that they can be consistently interpreted by the receiving components.

Within CyberAId, the Log Collector & Forwarding component forms part of the Integration and Messaging Fabric and supports the movement of logs and security events between the monitored financial environment, the security capability services and the Knowledge and Data Fabric. In this way, it provides a common ingestion and forwarding mechanism supporting monitoring, detection, investigation and audit activities across the platform.

3.4.4 Authentication & Access Control

The Authentication & Access Control component provides centralised identity and access management for CyberAId users and services. It supports user and service authentication, single sign-on, role-based authorisation, session management and security token issuance. Following authentication, it creates a security context containing the identity, role and access scope associated with each request.

The component may be realised using an identity and access management solution such as Keycloak⁶, supporting standards such as OpenID Connect and OAuth 2.0. The security context is propagated across the platform to ensure that components operate within the appropriate access boundaries. For cybersecurity tool invocations, the Tool Adapter Layer uses this context to enforce authorisation for the requested tool and operation and to record the interaction for auditing purposes.

3.5 Knowledge and Data Fabric

3.5.1 Security Knowledge Model & CTI Platform

The Security Knowledge Model and CTI Platform provide the semantic foundation through which CyberAId organises, contextualises and exchanges cybersecurity knowledge. The component establishes a common conceptual representation connecting threats, threat actors, attack techniques, vulnerabilities, assets, controls, observations, incidents, financial processes, risks and regulatory obligations. By maintaining explicit relationships among these concepts, together with provenance, confidence, temporal validity and governance information, it enables independently developed CyberAId services to interpret cybersecurity information according to a shared semantic context rather than maintaining separate and potentially inconsistent representations.

The CTI integration capability supports the acquisition and logical processing of intelligence from heterogeneous external and internal sources, including STIX/TAXII-compatible feeds, financial-sector information-sharing communities, CVE and NVD vulnerability information, MITRE ATT&CK, security advisories, threat reports and operational knowledge generated by CyberAId components. Intelligence is progressively validated, normalised, correlated, contextually enriched and mapped to the concepts of the Security Knowledge Model while preserving source traceability, confidence, timestamps and applicable handling restrictions. OpenCTI has been used as a representative reference environment for examining STIX-compatible intelligence and validating conceptual mappings; however, the architectural component remains technology-independent and does not prescribe a single CTI product, storage technology or implementation platform.

⁵ <https://www.elastic.co/beats/filebeat>

⁶ <https://www.keycloak.org>

Within the Knowledge and Data Fabric, the component acts as a shared semantic reference for authorised monitoring, risk-assessment, orchestration, reporting and decision-support services. It enables these consumers to retrieve contextual knowledge and supporting evidence, relate operational findings to recognised threats and vulnerabilities, and ground AI-assisted analysis in traceable cybersecurity information. Access to the knowledge layer is expected to respect appropriate confidentiality, privacy, access-control and auditability requirements, particularly where financial-sector or organisation-specific information is involved. The component therefore provides the architectural bridge between external intelligence sources and CyberAid's analytical capabilities, while the detailed semantic domains, relationship model, governance attributes and CTI integration workflow are specified in D2.4.

3.5.2 Semantic Retrieval & Vector Store

The Semantic Retrieval & Vector Store provides the retrieval capability through which agents ground their reasoning in cybersecurity and organisational material rather than in model parameters alone. It holds the document corpora an agent consults during analysis, including threat reporting, security advisories, internal playbooks, control descriptions and regulatory text, and returns the passages relevant to a query together with the references needed to cite them.

Agents do not query the store directly. Access runs through the Context Broker, which enforces the requesting Security Context before a query reaches the store, confining retrieval to the namespaces that context names and to results at or below the permitted classification. Knowledge is partitioned rather than pooled. Each agent reads a domain partition holding the corpora relevant to its role alongside a shared partition of cross domain material, both filtered by the requesting context. Partitioning also improves relevance, since a query against a corpus sized to one domain returns better results than the same query against an undifferentiated one. The storage technology is deliberately held behind this interface. Embedded SQLite vectors, Qdrant, Chroma, pgvector and LightRAG are the options under consideration, with corpus size, filtered query performance and whether the deployment can host an additional service as the deciding factors.

Within CyberAid the component belongs to the Knowledge and Data Fabric and operates alongside the Security Knowledge Model and CTI Platform, which supplies the semantic reference against which retrieved material is interpreted, and alongside the graph stores, which serve structured traversal rather than passage retrieval. Retrieved material enters a finding as a reference to a document and section rather than as copied content, which bounds where regulated material comes to rest and gives a reviewer a path back to the source.

3.5.3 Fraud Property Graph

The Fraud Property Graph provides the graph-oriented data structure used to represent interconnected financial entities, transactions and behavioural information for fraud analysis. It captures financial relationships as explicit nodes, properties and relationships, allowing the platform to preserve not only individual transaction records but also the wider network context in which those transactions occur. This representation supports the identification of hidden connections, multi-entity patterns and temporal transaction structures that are difficult to detect when financial activity is analysed as a collection of isolated records.

The component is specified around a Neo4j property-graph environment, providing a persistent graph store over which Cypher-based queries and specialised graph algorithms can operate. Depending on the deployment scenario and available data, represented information may include financial entities, accounts, transactions, counterparties, observed behaviours and the relationships connecting them, together with temporal and contextual attributes required for investigation. The graph can support the representation of patterns associated with rings, rapid fund movement, layering, structuring and

other coordinated behaviours. Its structure also enables analysts and analytical services to traverse paths, examine neighbourhoods and investigate how apparently separate activities are connected across a financial network.

Within the CyberAid Knowledge and Data Fabric, the Fraud Property Graph is distinct from the Graph Analytics Engine: the graph maintains and exposes the graph-organised information, while the engine applies queries and analytical methods to identify suspicious structures and generate findings. Graph entities and relationships are exchanged between these two components through a defined integration interface, and the resulting evidence can subsequently support orchestration, explanation, investigation and risk assessment. Where appropriate, graph elements may also be associated with contextual cybersecurity knowledge while preserving the separation between the finance-specific fraud graph and the broader Security Knowledge Model. The detailed property schema, analytical techniques, performance evaluation and validation scenarios will be refined through the corresponding WP4 activities.

3.5.4 Security Event & Log Store

The Security Event & Log Store provides CyberAid with a central repository for security events, logs and platform records, supporting detection, investigation and audit activities across the platform. It offers a consistent location in which security-relevant information can be retained, searched and made available to the components and analysts that require it.

Technically, the component is realized on OpenSearch⁷, which provides scalable indexing, search and retrieval over the stored events and logs. It ingests security-relevant events and logs made available through the platform, retaining them together with their contextual information, including timestamps and associated process and network details, so that stored records remain meaningful for subsequent querying and analysis. The indexed content can also be presented through dashboards and made accessible to the platform's analysis and investigation capabilities.

Within CyberAid, the Security Event & Log Store forms part of the Knowledge and Data Fabric and serves as the shared repository in which security-relevant records are retained and made available for detection, investigation and audit. Its retained and searchable records can support digital forensics and post-incident analysis capabilities by providing historical context for incident reconstruction and root cause analysis, and can support the traceability and record keeping expectations relevant to financial-sector operational resilience. In this way, the component provides a common, queryable basis of security information underpinning both operational monitoring and subsequent investigation.

⁷ <https://opensearch.org>

4. CYBERAID REFERENCE ARCHITECTURE

This section presents the first version of the CyberAid reference architecture. It defines the logical organisation of the platform, the boundaries between its main functional areas and the principal interactions through which its capabilities are combined. The components positioned within the architecture have already been introduced in Section 3. Therefore, this section focuses on their architectural placement, responsibilities and relationships without repeating their individual technical descriptions.

The architecture follows a layered organisation in which components are grouped according to their functional responsibilities. The layers represent logical areas of responsibility rather than fixed physical deployment locations. Components belonging to the same layer may consequently operate in different environments while maintaining the same architectural role and interaction boundaries. Common integration and deployment considerations are outlined in Section 4.4. Scenario-specific assumptions are addressed through the CyberAid blueprints, while concrete deployment decisions are refined during implementation and integration.

The architecture is presented through three complementary views. The System Context View defines the CyberAid platform boundary and its relationships with users, monitored financial infrastructures and external entities. The Functional View provides the principal structural representation of the platform and shows how its components and shared services are organised and connected. The Data and Knowledge View describes how telemetry, security knowledge, findings, evidence and risk information move between the main architectural areas. These views represent different aspects of the same architecture and should be considered together.

4.1 System Context View

The System Context View, presented in Figure 8, positions CyberAid within its operational environment. CyberAid is represented as a unified platform consisting of six internal functional layers that collectively support its interactions with users, monitored financial infrastructures and external

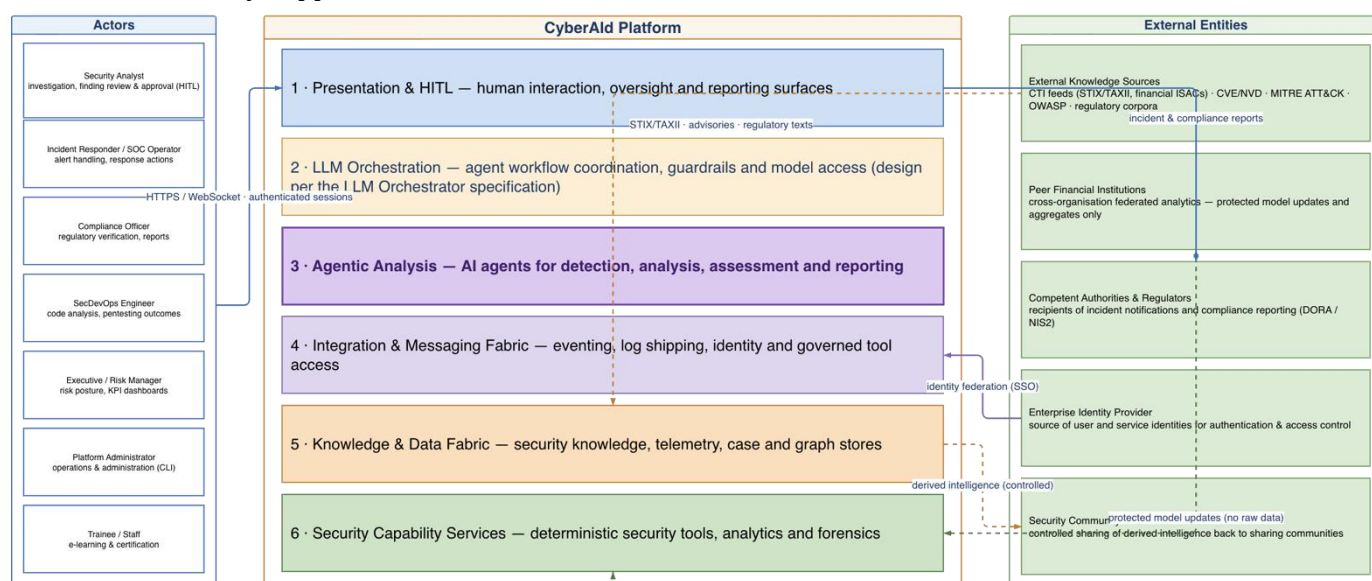


Figure 8: CyberAid System Context View

entities. At this level, the layers are shown only to communicate the overall organisation and boundary

of the platform. Their components, responsibilities and internal interaction paths are described in greater detail in the Functional View.

CyberAid supports several user roles with different responsibilities. Security analysts investigate alerts and review findings generated by the platform. Incident responders and SOC operators coordinate the handling of identified threats and review proposed response actions. Compliance officers use the platform to examine regulatory information, evidence and reports. SecDevOps engineers interact with the code analysis and security testing capabilities. Executives and risk managers consume aggregated information concerning organisational risk and security posture. Platform administrators manage operational and configuration activities, while staff and trainees access the learning capabilities of the platform.

These users interact with CyberAid through authenticated graphical or command line interfaces. Where human oversight is required, interactions are subject to human in the loop checkpoints before findings are accepted or security actions are approved. The access control mechanisms establish the identity, role and security context associated with each interaction.

The financial infrastructure represents the environment monitored, analysed and protected by CyberAid, but it is not part of the CyberAid platform boundary. Depending on the adopting organisation, this environment may contain endpoints, servers, databases, cloud and network infrastructure, file systems, payment systems, core banking platforms and trading environments. These systems provide telemetry, logs, events, transaction information and other operational data to the CyberAid platform. They may also receive monitoring configurations, security recommendations or approved response actions.

CyberAid interacts with external knowledge sources that provide information for security analysis and contextual enrichment. These include cybersecurity threat intelligence feeds, financial information sharing communities, vulnerability databases, attack technique repositories, security guidance and regulatory corpora. The information obtained from these sources is processed through the Knowledge and Data Fabric before it becomes available to the analytical capabilities of the platform.

Other external entities include enterprise identity providers, peer financial institutions, competent authorities and security communities. Enterprise identity providers support the authentication of users and services. Peer financial institutions may participate in controlled federated analytics without exchanging raw organisational data. Competent authorities may receive approved incident notifications or compliance reports. Derived threat intelligence may also be shared with authorised security communities according to the applicable access, confidentiality and information handling controls.

The exchanges shown in the context view remain technology neutral. They include authenticated user interaction, telemetry and event collection, threat intelligence ingestion, identity federation, controlled information exchange and the communication of approved security or compliance information. The specific interfaces, protocols, data structures and deployment controls are progressively refined through the integration activities and blueprint instances.

Although the six layers are displayed within the CyberAid platform boundary, they do not represent separate systems or physical deployment tiers. They are logical areas of responsibility that operate together as part of the unified CyberAid platform.

4.2 Functional View

The Functional View, presented in Figure 9, is the principal structural representation of the CyberAid platform. It positions the components introduced in Section 3 within six internal functional layers and relates them to the external financial infrastructure that the platform monitors and protects. The view

separates human interaction, AI supported coordination, agentic analysis, integration, shared knowledge and deterministic security capabilities while preserving the relationships required for integrated cybersecurity workflows.

The Presentation and HITL layer provides the human interaction and oversight capabilities of the platform. It contains the Security Dashboards and GUI, the Operations and Administration CLI and the Learning Centre. The dashboards and CLI provide access to operational information and initiate requests that are processed through the orchestration layer. The Learning Centre provides access to training material and related educational functions.

The LLM Orchestration layer (detailed at D2.5) coordinates requests, analytical activities and responses across the platform. It receives user initiated requests from the presentation layer and event driven activations from the Event and Alert Bus. It identifies the relevant analytical capabilities, coordinates their execution, combines their outputs and applies the required human checkpoints before results are returned or actions are proposed. This layer represents an architectural coordination function and does not expose the detailed internal logic of the orchestration mechanism.

The Agentic Analysis layer contains the AI based capabilities that perform reasoning, interpretation and controlled tool use. These capabilities cover email threat detection, anomaly detection, automated log analysis and event correlation, AI-driven penetration testing and attack generation, code analysis, penetration-testing outcomes analysis and reporting, risk and compliance assessment, and general report composition. The agents obtain contextual knowledge and evidence from the Knowledge and Data Fabric and invoke deterministic security services through the Tool Adapter Layer and Registry.

The Integration and Messaging Fabric provides the common communication mechanisms that connect the platform components. The Event and Alert Bus transports events, alerts and activation messages between producers and consumers. The Log Collector and Forwarding component transfers logs and telemetry towards the shared data stores. The Authentication and Access Control component establishes the authenticated identity and security context associated with platform interactions. The Tool Adapter Layer and Registry provides governed access from the agentic capabilities to the available security services.

The Knowledge and Data Fabric provides the shared knowledge, retrieval and evidence infrastructure of the platform. It contains the Security Knowledge Model and CTI Platform, the Semantic Retrieval and Vector Store, the Fraud Property Graph and the Security Event and Log Store. These components support the organisation, retention, retrieval and contextualisation of information required by the analytical and reporting capabilities. External knowledge sources remain outside the CyberAid platform and provide controlled inputs to this fabric.

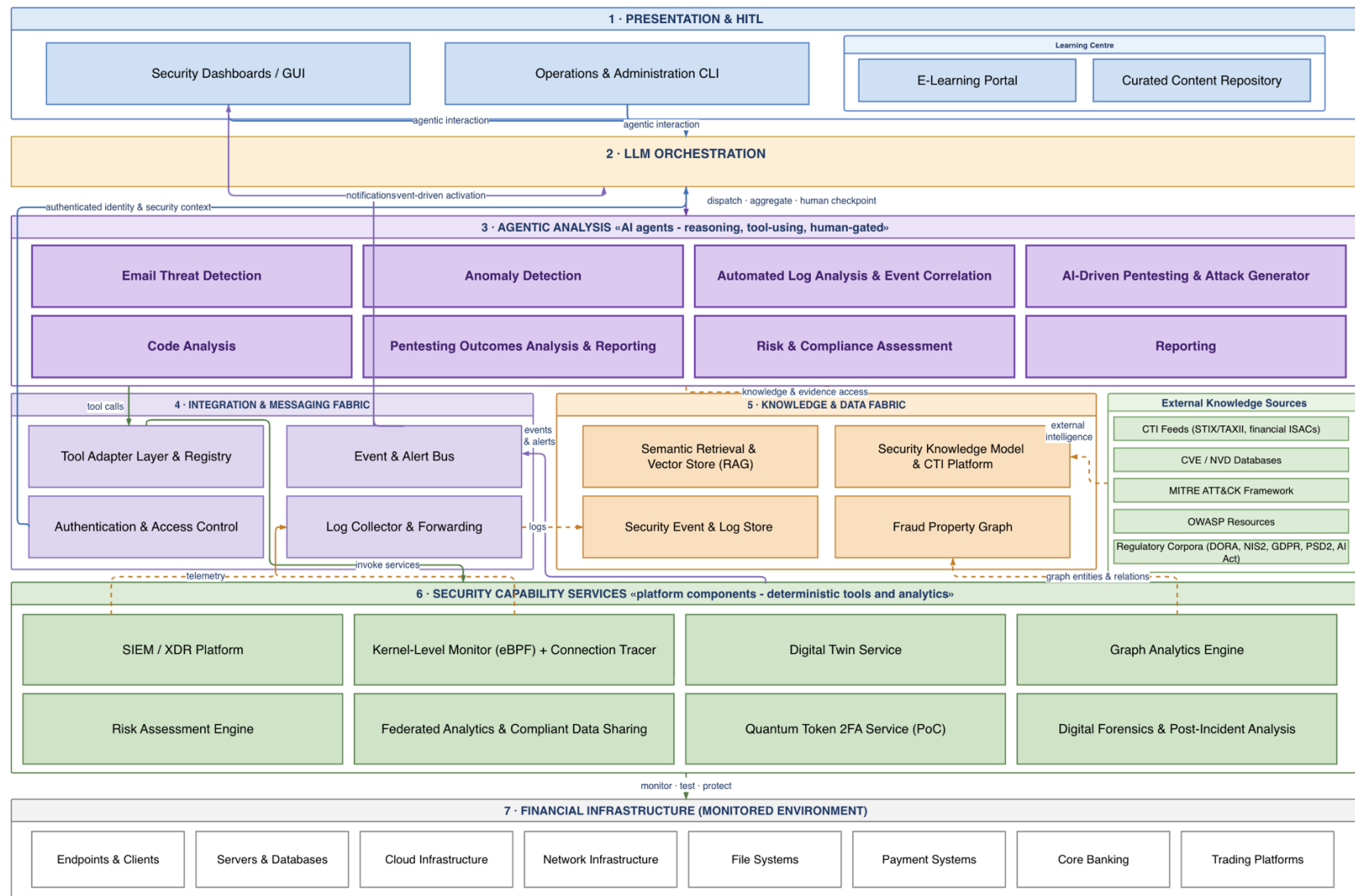


Figure 9: CyberAId Functional View

The Security Capability Services layer contains deterministic tools and analytical services that monitor, assess, test and protect the financial infrastructure. It includes the SIEM and XDR Platform, the Kernel-Level Monitor and Connection Tracer, the Graph Analytics Engine, the Digital Twin Service, the Risk Assessment Engine, Federated Analytics and Compliant Data Sharing, the Quantum Token 2FA Service and the Digital Forensics & Post-Incident Analysis. These services may generate telemetry, findings and alerts or perform specific actions when invoked through authorised platform interactions.

The Financial Infrastructure band represents the monitored environment and is not an internal CyberAid platform layer. It includes the endpoints, servers, databases, cloud resources, network infrastructure, file systems, payment services, banking platforms and trading systems of the adopting organisation. Its inclusion in the view shows the relationship between the CyberAid security capabilities and the systems they monitor, test and protect.

The principal interaction path begins when a user request or security event reaches the orchestration layer. The orchestration layer coordinates the relevant agentic capabilities, which obtain contextual information from the Knowledge and Data Fabric and invoke security services through the Tool Adapter Layer and Registry. Security services produce telemetry, findings and alerts that are transferred through the integration mechanisms and retained in the appropriate stores. Results are subsequently returned to the orchestration and presentation layers for review, reporting or approved action.

Security, identity and access control, privacy, guardrails, observability, provenance and auditability apply across these interactions. They are not limited to a single architectural layer. Each interaction is expected to preserve the relevant security context and provide sufficient evidence for review and audit.

4.3 Data and Knowledge View

The Data and Knowledge View, presented in Figure 10, complements the functional representation by showing how information originates, moves and is consumed across the CyberAid platform. The view organises these information movements into telemetry and detection, knowledge, correlation and case management, graph and risk analysis, and data governance.

The telemetry and detection flow begins in the monitored financial infrastructure. Logs, system calls, network connections, transactions, emails and other security events are collected by the monitoring capabilities. The SIEM and XDR Platform generates detection findings, while the Kernel-Level Monitor and Connection Tracer provide system and network telemetry. The Log Collector and Forwarding component transfers this information to the Security Event and Log Store. Prioritised alerts are published through the Event and Alert Bus and may activate the LLM Orchestration layer or notify users through the Security Dashboards.

The knowledge flow begins with external cybersecurity and regulatory sources. The Security Knowledge Model and CTI Platform ingests and enriches information from threat intelligence feeds, vulnerability databases, MITRE ATT&CK, OWASP resources and regulatory corpora. Curated information is made available through the Semantic Retrieval and Vector Store to support grounded analysis. Derived intelligence may be shared with authorised security communities subject to applicable handling restrictions.

The correlation and case flow combines stored events with analytical findings. Event and log data are retrieved from the Security Event and Log Store, while Email Threat Detection and Anomaly Detection provide scored findings. Automated Log Analysis and Event Correlation combines these inputs with security knowledge to maintain incident relationships and support case analysis. Correlated cases may be used by orchestration, dashboards, reporting and forensic analysis.

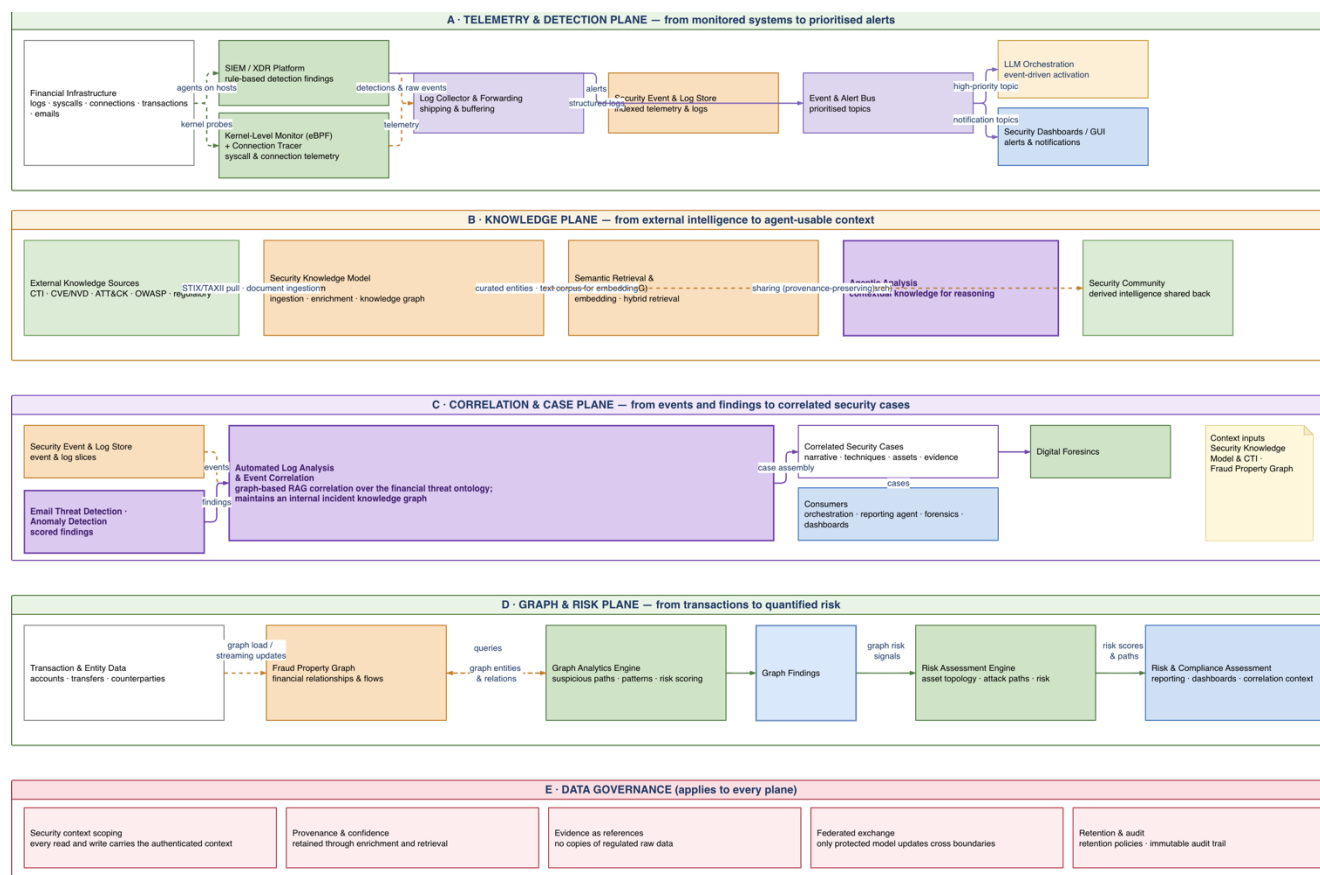


Figure 10: CyberAid Data and Knowledge View

Transaction and entity information is represented in the Fraud Property Graph. The Graph Analytics Engine analyses these entities and relationships to identify suspicious paths, patterns and coordinated activity. The resulting graph findings may be combined by the Risk Assessment Engine with asset, threat and dependency information to produce risk scores and attack paths for contextual assessment and reporting.

Data governance applies across all information flows. Authorised operations should retain identity, security context, provenance and confidence information. Sensitive data should be referenced rather than unnecessarily copied, and federated exchanges should transfer only the protected updates or aggregated information required by the scenario.

Retention and audit controls apply throughout the data lifecycle. Security events, findings, decisions, approvals and outputs should be retained according to applicable policies, with audit records sufficient to reconstruct significant processing and decision points.

Together, the three architecture views provide the initial structural and integration baseline for CyberAid. The System Context View defines the platform boundary, the Functional View positions its components, and the Data and Knowledge View explains how information supports monitoring, analysis, decision making and reporting. This baseline will be refined as the components, interfaces and blueprint instances mature.

4.4 Integration and Deployment Considerations

The CyberAid reference architecture establishes the component boundaries, principal information exchanges and security constraints that guide subsequent integration and deployment. The six

functional layers represent logical areas of responsibility, allowing their components to operate across different environments while preserving the relationships defined in the architecture views.

Integration follows the interaction paths established in the Functional View and the Data and Knowledge View. Events and alerts are exchanged through the Event & Alert Bus, agentic capabilities invoke security services through the Tool Adapter Layer & Registry, and shared knowledge and evidence are accessed through governed retrieval interfaces. These exchanges must preserve the applicable Security Context, access restrictions and information provenance.

Table 5 summarises the principal component exchanges and their architectural constraints. It provides an initial interface baseline; specific protocols, endpoint definitions, message schemas and configurations will be refined during technical integration.

Source and destination	Information exchanged	Architectural constraints
Presentation interfaces ↔ LLM Orchestration	User requests, workflow status, findings, reports and review decisions	Authenticated access, Security Context propagation, access-scope enforcement and traceable review decisions
Detection and correlation components → Event & Alert Bus → orchestration and presentation capabilities	Security events, alerts and workflow triggers	Authorised publication and consumption, identifiable sources, timestamps and correlation information
Agentic capabilities ↔ security services through the Tool Adapter Layer & Registry	Tool requests, execution status, results and error information	Input validation, per-call authorisation, invocation auditing and controlled execution of approved actions
Log Collector & Forwarding → Security Event & Log Store	Logs, structured events and telemetry	Secure transfer, source identification and preservation of timestamps and contextual information
Agentic capabilities ↔ Knowledge and Data Fabric through governed retrieval interfaces	Queries, contextual knowledge and evidence references	Enforcement of access scope and classification restrictions, with preservation of provenance
Fraud Property Graph ↔ Graph Analytics Engine; graph findings → risk-assessment and orchestration capabilities	Graph entities, relationships and analytical findings	Consistent entity references, controlled graph access and traceability of findings to supporting evidence

Table 5: Principal CyberAId component interactions and interface requirements

Deployment may follow on-premises, cloud or hybrid arrangements, subject to institutional requirements and the characteristics of each pilot environment. Component placement must account for data locality, access control, performance and operational continuity. Monitoring and collection functions require access to the relevant financial infrastructure, while Digital Twin and experimental environments remain isolated from production systems. The blueprint instances provide the scenario-specific context for refining these deployment choices.

The technical integration of CyberAId components is undertaken within T2.5, building on the architectural specifications established in T2.2. This work develops the concrete interface configurations, communication mechanisms, data transformations and integration tests required to connect the components into operational workflows. The first integrated components will be delivered

through D2.6. Feedback from integration and deployment activities will inform subsequent architectural refinements, which will be consolidated in D2.3.

5. CYBERAID BLUEPRINTS

The CyberAid blueprints translate the common reference architecture into scenario-specific selections of requirements, architectural elements, components, information flows and controls. Whereas the reference architecture describes the overall organisation of the CyberAid platform, each blueprint selects the parts of that architecture required to address a defined cybersecurity objective within a financial-sector environment.

The individual use cases are treated as connected operational threads of the corresponding blueprint. The use-case narratives determine the scenario and operational sequence, while the requirements baseline remains the primary traceability basis, in accordance with the methodology defined in Section 2.2.

The blueprints presented in this first version are initial design artefacts rather than final implementation or deployment specifications. They identify the required capabilities, responsibilities, interactions, controls and expected outcomes while recording information that remains subject to refinement. Detailed protocols, component configurations, physical deployment topologies and complete test procedures will be defined as the corresponding components and integrations mature.

Although the principal component descriptions are available, some interfaces, deployment assumptions and pilot-specific responsibilities remain subject to refinement. These limitations are recorded as open issues in the corresponding blueprint.

The initial blueprints are presented in Table 6.

Blueprint	Pilot and use-case coverage	Cybersecurity objective
BP-01	Pilot 1, UC#1.1 to UC#1.3	Detect client impersonation attempts by combining email analysis, privacy-preserving federated analytics, adaptive monitoring and contextual reasoning.
BP-02	Pilot 2, UC#2.1 to UC#2.3	Detect and investigate sophisticated money-laundering patterns through transaction graphs, correlated monitoring information and contextual analysis.
BP-03	Pilot 3, UC#3.1 to UC#3.3	Improve multi-channel anomaly detection, incident response and enterprise-wide cybersecurity risk assessment.
BP-04	Pilot 4, UC#4.1 to UC#4.3	Protect algorithmic trading systems through integrity monitoring, market-data resilience and regulatory-compliant incident handling.

Table 6: Initial CyberAid blueprints

5.1 BP-01: Tiered Client Impersonation Detection for Private Banking and Trading Systems

BP-01 addresses impersonation attempts involving financial instructions exchanged between clients, asset managers and custodian institutions. The blueprint combines the EMTD email-analysis framework with privacy-preserving federated analytics, adaptive monitoring configuration and human review. Its three operational threads support the common objective of identifying suspicious communications that may appear legitimate and remain below conventional transaction-verification thresholds.

Modelling activity	BP-01 definition
Define the blueprint	Purpose: Detect potential impersonation attacks in communications containing trading or transfer instructions while protecting confidential client information. The blueprint covers email content, headers, attachments, URLs, sender information, historical communication context, federated analytical updates, monitoring-rule adaptation and analyst-facing findings. Execution or blocking of the corresponding financial transaction remains outside the blueprint. Target environment: Private banks, asset managers and custodian institutions operating through distributed and potentially on-premises environments. Applicable requirements: R-BLP-11, R-BLP-14 to R-BLP-16, R-MON-08 to R-MON-10, R-MON-15, R-MON-17, R-MON-18, R-PRO-13, R-PRO-14, R-PRO-16, R-PRO-22, R-REP-06 and R-REP-10 to R-REP-14 together with R-REP-17.
Select architecture and capabilities	Email Threat Detection Federated Analytics & Compliant Data Sharing SIEM / XDR Platform Event & Alert Bus Tool Adapter Layer & Registry Authentication & Access Control Security Event & Log Store Security Dashboards – GUI Quantum Token 2FA Service
Describe operation and controls	An incoming email is ingested together with its headers, body, attachments and URLs. EMTD performs its deterministic pre-checks and uses its Router to identify candidate attack patterns. The corresponding specialised agents are selected through the configured trigger matrix and executed concurrently. Their findings, confidence values, severity levels and supporting evidence are consolidated into a weighted threat score. The result is mapped to the applicable decision level and presented to an authorised analyst. In parallel, participating institutions may produce protected analytical or model updates without transferring their raw communications. Federated insights may inform controlled updates to monitoring rules, correlation parameters or alert thresholds. Any automatic configuration change must remain within predefined and approved boundaries. Controls include local or on-premises processing where required, data minimisation, encryption, authentication of participating institutions, role-based access, security-context propagation, output data-leakage checks, validation of federated updates, evidence retention and audit logging.

Define validation and refinement	The expected outcome is an evidence-backed impersonation finding containing the detected attack pattern, supporting checks, threat score, decision level and appropriate recommendation. Validation should use anonymised historical communications and synthetic impersonation scenarios. Required evidence includes EMTD findings, score calculations, decision outputs, false-positive measurements, federated-processing records, privacy-control evidence, monitoring-configuration changes and analyst decisions. Pilot targets include more than 90% accuracy for simulated impersonation attempts, federated performance at smaller institutions within 5% of larger institutions, no exposure of sensitive client data during federated processing, a 35% improvement in monitoring precision, more than 40% reduction in false alerts and fewer than 5% of legitimate transactions requiring enhanced verification. Open issues include completion of the Federated Analytics, Quantum Token 2FA, Tool Adapter and dashboard descriptions; specification of the federated model lifecycle; definition of approved monitoring-configuration changes; and confirmation that the pilot KPI targets are represented in the requirements baseline.
----------------------------------	---

Table 7: BP-01 Tiered Client Impersonation Detection

5.2 BP-02: Advanced Anti-Money Laundering Detection for Payment Service Providers

BP-02 addresses the detection and investigation of money-laundering patterns within e-wallet, card-payment and IBAN-connected services. Its operational flow follows the pilot sequence from transaction-network analysis to correlated log analysis and contextual investigation.

Modelling activity	BP-02 definition
Define the blueprint	Purpose: Detect coordinated money-laundering activities and support efficient and explainable investigation of high-risk alerts. The blueprint covers transaction-graph construction, graph analytics, transaction and security-log correlation, system-level monitoring, alert investigation, risk assessment and preparation of supporting investigation information. Submission of suspicious activity reports remains under the control of the adopting institution. Target environment: Payment service providers operating e-wallet, card-payment and IBAN-connected account services. Applicable requirements: R-MON-03, R-MON-22 to R-MON-28, R-PRO-17 to R-PRO-21, R-REP-06, R-REP-07, R-REP-10 to R-REP-14, R-REP-16 and R-REP-18.

Select architecture and capabilities	Fraud Property Graph Graph Analytics Engine SIEM / XDR Platform Kernel-Level Monitor (eBPF) + Connection Tracer Log Collector & Forwarding Automated Log Analysis & Event Correlation Anomaly Detection Risk Assessment Engine Security Event & Log Store Security Knowledge Model & CTI Platform Semantic Retrieval & Vector Store LLM Orchestration Event & Alert Bus Tool Adapter Layer & Registry Authentication & Access Control Security Dashboards – GUI Digital Twin Service
Describe operation and controls	Transaction records are represented in the Fraud Property Graph as accounts, customers, merchants, counterparties, transfers and their relationships. The Graph Analytics Engine examines transaction velocity, temporal patterns, relationship depth and repeated fund movements to identify structures associated with smurfing, structuring, layering and coordinated account activity. Graph findings are correlated with authentication events, API activity, transaction logs and relevant system-level telemetry. A prioritised alert is published through the Event & Alert Bus and made available for contextual analysis. The orchestration and analytical capabilities retrieve relevant graph relationships, historical information, monitoring evidence and risk information and produce a consolidated case for analyst review. Where technically supported, selected scenarios may be reproduced within the Digital Twin Service using synthetic or appropriately sanitised data. Controls include restricted access to transaction and KYC information, data minimisation, encryption, role separation, security-context propagation, audited tool invocation, sanitisation of test data, evidence grounding and human review before operational or regulatory action.
Define validation and refinement	The expected outcome is a prioritised AML investigation case containing the suspicious transaction network, related monitoring information, contextual explanation, risk assessment and supporting evidence. Initial validation should use synthetic AML scenarios representing structuring, smurfing, layering, shell-company activity and coordinated account behaviour. Required evidence includes graph-query results, correlation records, monitoring outputs, alert-prioritisation results, investigation timelines and reviewed reporting material. Indicative pilot targets include more than 85% detection of simulated money-laundering scenarios, more than 50% reduction in false positives, more than 60% reduction in investigation time, more than 90% of highest-risk alerts confirmed as requiring investigation and identification of more than 75% of complex laundering networks. The digital-twin fidelity target remains provisional until the supported representation of payment and transaction flows is established. All pilot targets remain subject to confirmation through the

	requirements baseline and pilot planning. Open issues include the final transaction-graph schema, integration with existing transaction-monitoring systems, the role of the Digital Twin Service and assignment of responsibility for AML regulatory-report generation.
--	---

Table 8: BP-02 Advanced Anti-Money Laundering Detection

5.3 BP-03: AI-Enhanced Anomaly Detection and Incident Response for Banking Cybersecurity

BP-03 combines multi-channel anomaly detection, AI-supported incident response and enterprise-wide risk assessment. It defines a continuous path in which banking telemetry is analysed, detected anomalies are enriched and investigated, and validated findings contribute to the organisational risk posture.

Modelling activity	BP-03 definition
Define the blueprint	Purpose: Improve the detection, investigation and handling of cyber incidents across banking channels while maintaining a dynamic view of organisational cybersecurity risk. The blueprint covers network, host, authentication, transaction and user-behaviour information; anomaly detection; contextual enrichment; incident triage; analyst decision support; response recommendations; risk aggregation; and controlled scenario analysis. Target environment: Large banking environments containing digital channels, payment systems, servers, workstations, mobile devices, ATMs and integrated enterprise systems. Applicable requirements: R-BLP-03, R-BLP-04, R-BLP-12, R-BLP-13, R-MON-01, R-MON-03 to R-MON-07, R-MON-18, R-MON-22, R-MON-30, R-MON-32, R-PRO-17, R-PRO-18, R-PRO-20, R-PRO-21, R-REP-06, R-REP-08 to R-REP-14, R-REP-16 and R-REP-18 to R-REP-23.
Select architecture and capabilities	SIEM / XDR Platform Kernel-Level Monitor (eBPF) + Connection Tracer Anomaly Detection Automated Log Analysis & Event Correlation Log Collector & Forwarding Event & Alert Bus Security Event & Log Store Security Knowledge Model & CTI Platform Semantic Retrieval & Vector Store LLM Orchestration Risk Assessment Engine Digital Twin Service Digital Forensics & Post-Incident Analysis Authentication & Access Control Tool Adapter Layer & Registry Security Dashboards - GUI

Describe operation and controls	Monitoring components collect network, process, authentication, application and transaction information from the participating banking channels. The SIEM / XDR Platform, Anomaly Detection and Automated Log Analysis & Event Correlation identify deviations from operational and behavioural baselines. Prioritised anomalies are published through the Event & Alert Bus and receive an initial severity and incident classification. LLM Orchestration coordinates the relevant analytical components and retrieves supporting logs, threat intelligence, historical patterns, asset information and business context. Findings are corroborated and organised into a correlated security case containing the affected assets, evidence, confidence and proposed containment or mitigation actions. The analyst reviews the result through the dashboard and may request further analysis, approve an action, reject a recommendation or escalate the case. Validated findings are provided to the Risk Assessment Engine to update the corresponding risk information. The Digital Twin Service may be used to execute representative attack or defence scenarios in an isolated environment, but risk impact analysis remains the responsibility of the Risk Assessment Engine. Controls include authenticated and authorised access, least-privilege tool use, security-context propagation, evidence-grounded output, prompt-injection and data-leakage protection, privacy controls, tamper-evident audit records and human approval for actions exceeding predefined automation thresholds.
Define validation and refinement	The expected outcome is an integrated anomaly-to-decision workflow producing an evidence-backed security case, response recommendations and updated organisational risk information. Initial validation should introduce representative anomalies across multiple banking channels and verify detection, routing, contextual enrichment, analyst interaction, auditability and risk updating. Required evidence includes monitoring records, analytical findings, correlated case information, explanations, confidence information, analyst decisions, response measurements and risk-score changes. Indicative pilot targets include more than 90% detection of simulated attacks, more than 60% reduction in false positives, more than 40% reduction in mean time to detect, more than 50% reduction in mean time to respond, more than 30% increase in incidents handled per analyst, more than 85% correlation between predicted and actual risk levels, more than 25% reduction in security-operations workload and identification of at least three previously undetected attack patterns. These targets remain subject to confirmation through the requirements baseline and pilot planning. Open issues include confirmation of the interfaces and information exchanges among the dashboard, orchestration, tool adapters and Digital Forensics; definition of automation thresholds; pilot-specific scalability assumptions; and confirmation of the interfaces between correlated cases, the Digital Twin Service and the Risk Assessment Engine.

Table 9: BP-03 AI-Enhanced Anomaly Detection and Incident Response

5.4 BP-04: Cyber Resilience for High-Frequency Trading

BP-04 addresses the protection of algorithmic trading operations, proprietary trading code and time-sensitive market-data services. It combines low-overhead integrity monitoring, anomaly-triggered

contingency actions and evidence-supported incident reporting while preserving the performance requirements of high-frequency trading.

Modelling activity	BP-04 definition
Define the blueprint	Purpose: Protect algorithmic and high-frequency trading operations by detecting unauthorised access to or manipulation of trading algorithms and market-data services. The blueprint covers algorithm-repository monitoring, API and access monitoring, market-data and execution-latency observation, feed-integrity analysis, threat scoring, support for approved failover activation, evidence preparation and regulatory-reporting support. Changes to trading algorithms, execution of contingency measures and communication with supervisory authorities remain under the control of the adopting institution. Target environment: Algorithmic and high-frequency trading operations using proprietary trading models, market-data providers, broker interfaces, execution platforms and contingency data sources. Applicable requirements: R-MON-11 to R-MON-14, R-MON-19 to R-MON-21, R-MON-31, R-PRO-10, R-PRO-12, R-REP-01 to R-REP-04, R-REP-06 to R-REP-16 and R-REP-18.
Select architecture and capabilities	Kernel-Level Monitor (eBPF) + Connection Tracer SIEM / XDR Platform Code Analysis Anomaly Detection Automated Log Analysis & Event Correlation Log Collector & Forwarding Security Event & Log Store Security Knowledge Model & CTI Platform LLM Orchestration Risk Assessment Engine Event & Alert Bus Tool Adapter Layer & Registry Authentication & Access Control Security Dashboards – GUI Digital Forensics & Post-Incident Analysis
Describe operation and controls	The Kernel-Level Monitor (eBPF) + Connection Tracer and SIEM / XDR Platform observe access to algorithm repositories, relevant process activity, network connections and API communication. Code Analysis may provide supporting information concerning vulnerabilities or unauthorised changes where the required repository integration is available. Market-data and execution telemetry is assessed for latency changes, missing values, unexpected patterns and other integrity deviations. When an anomaly or threshold breach is identified, an alert is published and correlated with the affected service, source, process and recent activity. LLM Orchestration gathers the available monitoring, code, threat and risk information and produces a consolidated finding. For market-data integrity events, CyberAid may issue an approved failover trigger or recommendation. The actual switch to an alternative provider, broker feed or local cache is executed within the external trading environment through a separately authorised integration.

	Incident evidence, timelines, decisions and actions are retained for investigation and reporting. Controls include low-overhead collection, authenticated and encrypted API communication, least-privilege repository access, integrity protection for monitoring records, approved failover thresholds, prevention of unauthorised response actions, evidence grounding and complete audit logging.
Define validation and refinement	The expected outcome is a low-latency monitoring and decision-support workflow capable of detecting algorithm manipulation and market-data anomalies, supporting approved contingency measures and producing a reconstructable incident record. Initial validation should include simulated unauthorised repository access, controlled algorithm modification, API anomalies, delayed or manipulated market-data feeds and failure of a primary data provider. Required evidence includes access and integrity alerts, code-analysis findings where applicable, latency measurements, failover-trigger and external-execution records, incident timelines, risk assessments and reviewed regulatory documentation. Indicative pilot targets include more than 95% accuracy in detecting unauthorised algorithm access or manipulation, more than 98% detection of market-data anomalies with a response time below 50 ms, more than 99% successful activation of approved contingency measures, complete conformity of incident documentation with the applicable reporting requirements and less than 5 ms additional latency in trading operations. These targets remain subject to confirmation through the requirements baseline, integration design and pilot planning. Open issues include completion of the Code Analysis, Digital Forensics and regulatory-reporting component descriptions; assignment of responsibility for the failover integration; definition of supported trading and broker interfaces; and confirmation of the feasibility of the latency targets.

Table 10: BP-04 Cyber Resilience for High-Frequency Trading

6. CONCLUSIONS AND NEXT STEPS

This deliverable presented the first version of the CyberAId reference architecture and established the methodology for developing the CyberAId blueprints. Building on the requirements baseline from D2.1, it translated the identified cybersecurity needs and constraints into an initial structural and integration-level representation of the CyberAId framework.

The tool and component landscape provided a consolidated view of the capabilities considered within the first architectural baseline. Components were organised according to their main architectural responsibilities across presentation and human interaction, orchestration and agentic analysis, deterministic security services, integration and messaging, and shared knowledge and data management. This organisation provides a common vocabulary for positioning the components and understanding their relationships without prescribing their detailed internal implementation.

The reference architecture represented the CyberAId platform through the System Context View, Functional View and Data and Knowledge View. These views collectively defined the platform boundary, external actors and systems, logical layers, component responsibilities, principal interaction paths and information flows. Security, identity and access control, privacy, guardrails, observability, provenance and auditability were treated as concerns that apply across the architecture rather than as responsibilities of a single component or layer.

The blueprint methodology established how relevant parts of the reference architecture can be selected and adapted to defined financial sector cybersecurity objectives. The methodology uses requirements as its primary traceability basis and connects them with architecture elements, CyberAId components, interactions, controls, expected outcomes and validation evidence. The initial blueprint portfolio is organised around the four pilot scenarios and provides the basis for progressively developing practical applications of the common architecture.

The next steps will focus on validating component descriptions and boundaries, resolving outstanding responsibility overlaps, and refining the interfaces and information flows connecting the principal architectural areas. The blueprint instances will be further developed as the technical components, integration mechanisms and pilot environments mature. Feedback from technical partners and pilot stakeholders will be used to resolve open issues and maintain alignment between the requirements baseline, component landscape, reference architecture and blueprints. These activities will support the consolidation and extension of the final CyberAId reference architecture and blueprint portfolio.

REFERENCES

- [1] ISO/IEC/IEEE, ISO/IEC/IEEE 42010:2022, Software, systems and enterprise: Architecture description, second edition. Geneva: International Organization for Standardization, 2022. Available at: <https://www.iso.org/standard/74393.html>
- [2] R. Cloutier, G. Muller, D. Verma, R. Nilchiani, E. Hole and M. Bone, "The Concept of Reference Architectures," Systems Engineering, vol. 13, no. 1, pp. 14-27, 2010. doi: 10.1002/sys.20129.
- [3] M. Richards and N. Ford, Fundamentals of Software Architecture: An Engineering Approach. Sebastopol, CA: O'Reilly Media, 2020. Available at: <https://www.oreilly.com/library/view/fundamentals-of-software/9781492043447/>
- [4] European Union Agency for Cybersecurity, ENISA Cybersecurity Market Analysis Framework, ECSMAF Version 3.0. Athens: ENISA, March 2026. doi: 10.2824/4565747.
- [5] The MITRE Corporation, MITRE ATT&CK Enterprise Matrix. Available at: <https://attack.mitre.org/matrices/enterprise/>
- [6] European Union Agency for Cybersecurity, The ENISA Cybersecurity Exercise Methodology, Version 1.0. Athens: ENISA, February 2026. doi: 10.2824/4949728.
- [7] Guo, Z., Xia, L., Yu, Y., Ao, T., & Huang, C. (2025, November). LightRAG: Simple and Fast Retrieval-Augmented Generation. In EMNLP (Findings) (pp. 10746-10761)