Harmonizing People, Processes, and Technology for Robust Cybersecurity

D4.1 Integrated Cybersecurity Monitoring Suite v1

Document Summary Information

Grant Agreement No	101158555	**Acronym**	CYberSynchrony
Full Title	Harmonizing People, Processes, and Technology for Robust Cybersecurity		
Start Date	01/06/2024	**Duration**	36 months
Project URL	https://www.cybersynchrony.eu/		
Deliverable	D4.1 Integrated Cybersecurity Monitoring Suite v1		
Work Package	WP4 – Advanced Proactive Cyber Risk Monitoring and Rapid Response(CYBRITE)		
Contractual due date	31/5/2025	**Actual submission date**	3/6/2025
Type	Other	**Dissemination Level**	Public
Lead Beneficiary	Maggioli (MAG)		
Responsible Author	Alexandros Psychas(MAG)		
Contributions from	Alexandros Psychas (MAG), E. Michos (IANUS), Leandros Maglaras (TRUST)		

Funded by the European Union

ECCC

Funded by the European Union under Grant Agreement no. 101158555 and supported by the European Cybersecurity Competence Centre (ECCC). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Cybersecurity Competence Centre. Neither the European Union nor the granting authority can be held responsible for them.

Revision history

Version	Issue Date	Changes	Contributor(s)
0.1	15.03.2025	Initial deliverable structure	Alexandros Psychas (MAG)
0.2	25.03.2025	Integration of deliverable ToC and task scoping, Architecture sections implementation	Alexandros Psychas (MAG)
0.3	09.04.2025	Sections 5, 6, Annex III, Annex IV	Leandros Maglaras (TRUST)
0.4	20.04.2025	Technical contributions – Wazuh, Suricata, Anomaly Detector IDS	Evangelos Michos (IANUS), Alexandros Psychas (MAG)
0.5	05.05.2025	Final draft ready for internal review	Alexandros Psychas (MAG), Leandros Maglaras (TRUST)
0.6	08.05.2025	Peer review	Evangelos Michos (IANUS),
0.7	13.05.2025	Peer review	George karavokiros (BYTE)
0.8	23.05.2025	Final draft addressing comments from internal review	Alexandros Psychas (MAG), E. Michos (IANUS), Leandros Maglaras (TRUST)
0.9	02.06.2025	Final quality check	Panayiotis Klitou (EBOS)
1.0	03.06.2025	Final version ready for submission	Alexandros Psychas (MAG)

Disclaimer

The content of the document herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

While the information contained in the documents is believed to be accurate, the authors(s) or any other participant in the CYberSynchrony consortium make no warranty of any kind with regard to this material including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Neither the CYberSynchrony consortium nor any of its members, their officers, employees or agents shall be responsible or liable in negligence or otherwise howsoever in respect of any inaccuracy or omission herein.

Without derogating from the generality of the foregoing neither the CYberSynchrony consortium nor any of its members, their officers, employees or agents shall be liable for any direct or indirect or consequential loss or damage caused by or arising from any information advice or inaccuracy or omission herein.

Copyright message

© CYberSynchrony Consortium This Deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.

According to CYberSynchrony's Quality Assurance Process for reports:

- 3 months before Deliverable's Due Date: Table of Contents ready to be reviewed by WP leader/ Technical Manager/ Quality Manager
- 2 months before Deliverable's Due Date: 50% should be complete. Review by the Quality Manager (& Technical Manager for software).
- 1 month before Deliverable's Due Date:100% should be complete. Review by the Quality Manager and Peer Reviewers.
- M: Sent to PC for Submission to the EC after addressing all comments by Quality Manager and Peer Reviewers

Table of Contents

Executive Summary 8
1 Introduction 9
1.1 Mapping CYberSynchrony Outputs 9
1.2 Linkage to other project outputs 10
1.3 Deliverable Structure 10
2 Position in CYberSynchrony Architecture 12
3 Wazuh 14
3.1 Component Description 14
3.2 Architecture 15
3.3 Technologies Used 16
4 Suricata IDS 18
4.1 Component Description 18
4.2 Architecture 19
4.3 Technologies Used 20
5 Anomaly detector IDS 22
5.1 Component Description 22
5.2 Architecture 22
5.3 Technologies Used 24
6 WEKA 25
6.1 Component Description 25
6.2 Architecture 25
6.3 Technologies Used 27
7 Human RM (HRM) 28
7.1 Component Description 28
7.2 Architecture 28
8 Conclusions and next steps 30
Annex I: *Preliminary PoC and Prototypes* (Wazuh) 31
Annex II: *Preliminary PoC and Prototypes (Suricata)* 35
Annex III: *Preliminary PoC and Prototypes (Anomaly Detector IDS)* 38
Step-by-Step Walkthrough 38
Usage Scenarios 39
System Requirements and Deployment 40
Annex IV: *Preliminary PoC and Prototypes (WEKA)* 41
Annex V: *Preliminary PoC and Prototypes (HUMAN RM (HRM))* 43

List of Figures

Figure 1: CYberSynchrony Modular Architecture – Updated Overview 12
Figure 2: Wazuh Internal Architecture 15
Figure 3: Suricata Internal Architecture 20
Figure 4: Detection workflow diagram showing data flow from capture to alerting 23
Figure 5: Anomaly detector data flow and operation 24
Figure 6: Architecture of WEKA in the CyberSychrony ecosystem 26
Figure 7: HRM phases 29

© CYberSynchorny, 2024

Figure 8: Architecture of HRM in the CyberSychrony ecosystem 29
Figure 9: Wazuh-Suricata Placement in Architecture (internal connections) 31
Figure 10: Wazuh Overview Page 32
Figure 11: Wazuh Alerts 32
Figure 12: Wazuh Saved objects 33
Figure 13: Wazuh events (with custom data from last month) 33
Figure 14: Verifying Suricata is up and running 35
Figure 15: Docker container statistics for Suricata 36
Figure 16: Container logs for Suricata 36
Figure 17: AI Anomaly detector IDS alert dashboard 38
Figure 18: Weka log data analysis pipeline 41
Figure 19: Cyber-physical process testbed 42
Figure 20: HSMEs users'/assets model 43

List of Tables

Table 1: Adherence to CYberSynchrony's GA 9
Table 2: Links to other Project Outputs/Work 10
Table 3: Wazuh alignment with CYberSynchrony objectives. 14
Table 4: Wazuh technology stack. 16
Table 5: Suricata alignment with CYberSynchrony objectives. 18
Table 6: Suricata technology stack. 21
Table 7: Anomaly detector IDS alignment with CYberSynchrony objectives. 22
Table 8: Anomaly detector IDS technology stack. 24
Table 9: WEKA alignment with CYberSynchrony objectives. 25
Table 10: WEKA technology stack. 27
Table 11: HRM's alignment with CYberSynchrony objectives. 28
Table 12: Wazuh API dacumentation 34
Table 13: Suricata API documentation 37

© CYberSynchorny, 2024

Acronyms and abbreviations

Abbreviation / Term	Description
AE	Autoencoder
API	Application Programming Interface
AP	Adversary Profile
ASIC	Application-Specific Integrated Circuit
CPU	Central Processing Unit
CPS	Cyber-Physical System
CSPA	CyberSecurity Posture Assessment
CSV	Comma-Separated Values
CVE	Common Vulnerabilities and Exposures
DPDK	Data Plane Development Kit
EC	European Commission
EICAR	European Institute for Computer Antivirus Research
ELK	Elasticsearch, Logstash, Kibana
ENISA	European Union Agency for Cybersecurity
ET	Emerging Threats
EVE	Event and Alert Output Format in Suricata
GA	Grant Agreement
GNU	GNU's Not Unix
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HMI	Human-Machine Interface
HRM	Human Risk Management
HSME	Healthcare Small/Medium Enterprise
IA	Information Assurance
ICT	Information and Communication Technology
IDS	Intrusion Detection System
IEC	International Electrotechnical Commission
III	Industrial Internet of Intelligence
IP	Internet Protocol
IPS	Intrusion Prevention System

© CYberSynchorny, 2024

ISO	International Organization for Standardization
MAT	Malware Analysis Tool
MHSA	Multi-Head Self-Attention
MISP	Malware Information Sharing Platform
MOA	Memorandum of Agreement
MODBUS	Modular Digital Bus (Industrial Protocol)
NIC	Network Interface Card
NSM	Network Security Monitoring
OISF	Open Information Security Foundation
OWASP	Open Worldwide Application Security Project
PCAP	Packet Capture
PLC	Programmable Logic Controller
REST	Representational State Transfer
RM	Risk Management
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SIEM	Security Information and Event Management
SMB	Server Message Block
SME	Small and Medium-Sized Enterprise
SOC	Security Operations Center
SOAR	Security Orchestration, Automation, and Response
SPAN	Switched Port Analyzer
SSH	Secure Shell
SVM	Support Vector Machine
TAP	Test Access Point
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UEBA	User and Entity Behavior Analytics
UP	User Profile
VFD	Variable Frequency Drive
VM	Virtual Machine

© CYberSynchorny, 2024

WEKA	Waikato Environment for Knowledge Analysis
WP	Work Package
XDR	Extended Detection and Response
YAML	YAML Ain't Markup Language

© CYberSynchorny, 2024

Executive Summary

This deliverable, D4.1 “Integrated Cybersecurity Monitoring Suite v1,” presents the design, architectural positioning, and initial implementation of the key monitoring components developed within CYberSynchrony's Work Package 4 (WP4). It reflects the progress made in the realization of Tasks T4.1 (Unveiling SIEM & Traffic Analysis Deployment) and T4.2 (Entity Behaviour Analytics Systems for Behaviour Insight), focusing on the deployment, integration, and operational roles of the platform’s intrusion detection and behavioural monitoring tools.

The suite includes a combination of on-premises and cloud-based components—namely, Wazuh, Suricata IDS, AI-based Anomaly Detector, WEKA, and the Human Risk Management (HRM) module. Suricata and Wazuh, supported by Sysmon, form the foundational telemetry and log analysis layer, capturing both system and network-level data. These components ensure real-time monitoring and detection through rule-based correlation of security events. The AI Anomaly Detector extends this capability through advanced behavioural analytics, using attention mechanisms and autoencoders to identify subtle anomalies in stored network logs while ensuring data integrity through interaction with the Unalterable Forensic Techniques pipeline.

Positioned as standalone components, WEKA and HRM support post-forensic analysis and behaviour-driven risk assessment. WEKA enables machine learning-based analysis of historical incidents, while HRM brings a socio-technical layer to risk management, modelling user profiles and adversary behaviour to support human-in-the-loop decision making.

For each of these components, these deliverable details the architecture, core functionality, and design rationale, illustrating how they contribute to the broader monitoring ecosystem of CYberSynchrony. Given that D4.1 is a type “Other” deliverable, the corresponding demonstrators and preliminary prototypes are also included in the annexes, showcasing the current implementation status and usage workflows of each system.

This deliverable builds upon the architectural principles defined in D2.1 and is closely linked to outputs from WP2 and WP3, particularly regarding data integrity, modular integration, and threat intelligence correlation. The content herein offers a comprehensive foundation for subsequent deployments and integrations, paving the way toward a unified and intelligent cybersecurity monitoring environment under the CYberSynchrony platform. An updated and expanded version of this work will be presented in Deliverable D4.3, which will document the full implementation and all integration developments of the monitoring components.

© CYberSynchorny, 2024

1 Introduction

1.1 Mapping CYberSynchrony Outputs

This section presents the CYberSynchrony's Grant Agreement (GA) commitments, as extracted from the formal deliverable and task description, in respect to their outputs and work to be performed.

Table 1: Adherence to CYberSynchrony's GA

Component no	Component title	Respective Document Chapter(s)	Justification
DELIVERABLE			
D4.1	D4.1 Integrated Cybersecurity Monitoring Suite v1	Chapters 1–7, Annex I–V	This deliverable is designed to showcase the general architecture and technical design of key components developed under Tasks T4.1 and T4.2. It serves as a foundational reference for the implementation and integration of core monitoring and detection capabilities in the CYberSynchrony platform.
TASKS			
T4.1	T4.1 Unveiling SIEM & Traffic Analysis Deployment	Chapters 3–6, Annex I–IV	T4.1 focuses on the deployment and design of SIEM systems and network traffic analysis tools. The described components—such as Wazuh, Suricata, and anomaly-based IDS—are aligned with this task's objective of enabling real-time monitoring and early threat detection.
T4.2	T4.2 Entity Behaviour Analytics (UEBA) Systems for Behaviour Insight	Chapters 5–7, Annex IV–V	T4.2 is represented through the inclusion of the anomaly detector for entity anomalies, the WEKA module for post-forensic analysis, and the Human Resource Management (HRM) component. These tools support entity behaviour modelling, risk detection, and human-in-the-loop intelligence workflows.

© CYberSynchorny, 2024

1.2 Linkage to other project outputs

The present deliverable builds upon and aligns with earlier project outcomes to ensure consistency, interoperability, and seamless integration of CYberSynchrony components. The following table outlines the key work packages, tasks, and deliverables that have influenced the design decisions, data flows, and implementation strategies described in D4.1.

Table 2: Links to other Project Outputs/Work

WP Number	Task Number	Deliverable Number related	Content
WP2	T2.4 – CYberSynchrony Ecosystem Design	D2.1	The component design, interaction flows, and integration strategies presented in D4.1 are directly influenced by the overarching system architecture and end-user requirements developed in Task 2.4. These ensure interoperability, modularity, and consistency across all monitoring components.
WP3	T3.2 – Adopting Unalterable Forensic Techniques	D3.1	The SIEM and IDS components described in this deliverable are integrated into a data handling pipeline that aligns with the data integrity principles of T3.2. This ensures all inputs to the IDS components remain unaltered creating more robust and trustworthy IDS.

1.3 Deliverable Structure

This deliverable is structured into eight main chapters and four annexes. Each chapter serves a specific role in presenting the technical development and integration of CYberSynchrony's monitoring components:

Chapter 1: Introduction

Outlines the deliverable's objectives and how it aligns with the CYberSynchrony Grant Agreement. Includes:

- Mapping CYberSynchrony Outputs – alignment with GA components and project tasks
- Linkage to other project outputs – dependencies on previous and parallel work
- Deliverable Structure – document organization

Chapter 2: Position in CYberSynchrony Architecture

Describes how the components documented here fit within the overall CYberSynchrony platform, including interactions and dependencies.

Chapter 3: Wazuh

Component description, architecture, and technology stack for the Wazuh Security Information and Event Management.

Chapter 4: Suricata IDS

Component description, architecture, and technology stack for the Suricata intrusion detection systems.

Chapter 5: Anomaly Detector IDS

Details the anomaly-based intrusion detection module—its detection logic, architecture, and implementation technologies.

© CYberSynchorny, 2024

Chapter 6: WEKA

Describes the integration of WEKA for post-forensic analysis: component function, architecture, and supporting technologies.

Chapter 7: Human RM (HRM)

Explains the Human Resource Management module's role in human-in-the-loop decision making and behavioural analysis.

Chapter 8: Conclusions

Summarizes the deliverable's contributions and how the described work supports the project's broader objectives.

Annexes

- Annex I: Preliminary PoC and Prototypes (Wazuh): PoC demonstrations and walkthroughs for Wazuh.
- Annex II: Preliminary PoC and Prototypes (Surcata): PoC demonstrations and walkthroughs for Suricata.
- Annex III: Preliminary PoC and Prototypes (Anomaly Detector IDS): Demonstrations of the anomaly-based IDS functionality.
- Annex IV: Preliminary PoC and Prototypes (WEKA): Prototype outputs and interaction scenarios using WEKA.
- Annex V: Preliminary PoC and Prototypes (HUMAN RM – HRM): Illustrations of HRM's support for user-oriented threat evaluation and decision support.

© CYberSynchorny, 2024

2 Position in CYberSynchrony Architecture

This section outlines how the components described in the current deliverable are positioned within the overall CYberSynchrony architecture. The architectural view provided here builds on and updates the initial modular design presented in Deliverable D2.1, reflecting the current implementation plans and integrations as the project progresses. The figure below presents the full modular architecture of the CYberSynchrony platform. The components relevant to this deliverable are highlighted in red, signifying their direct contribution to the integrated cybersecurity monitoring suite.

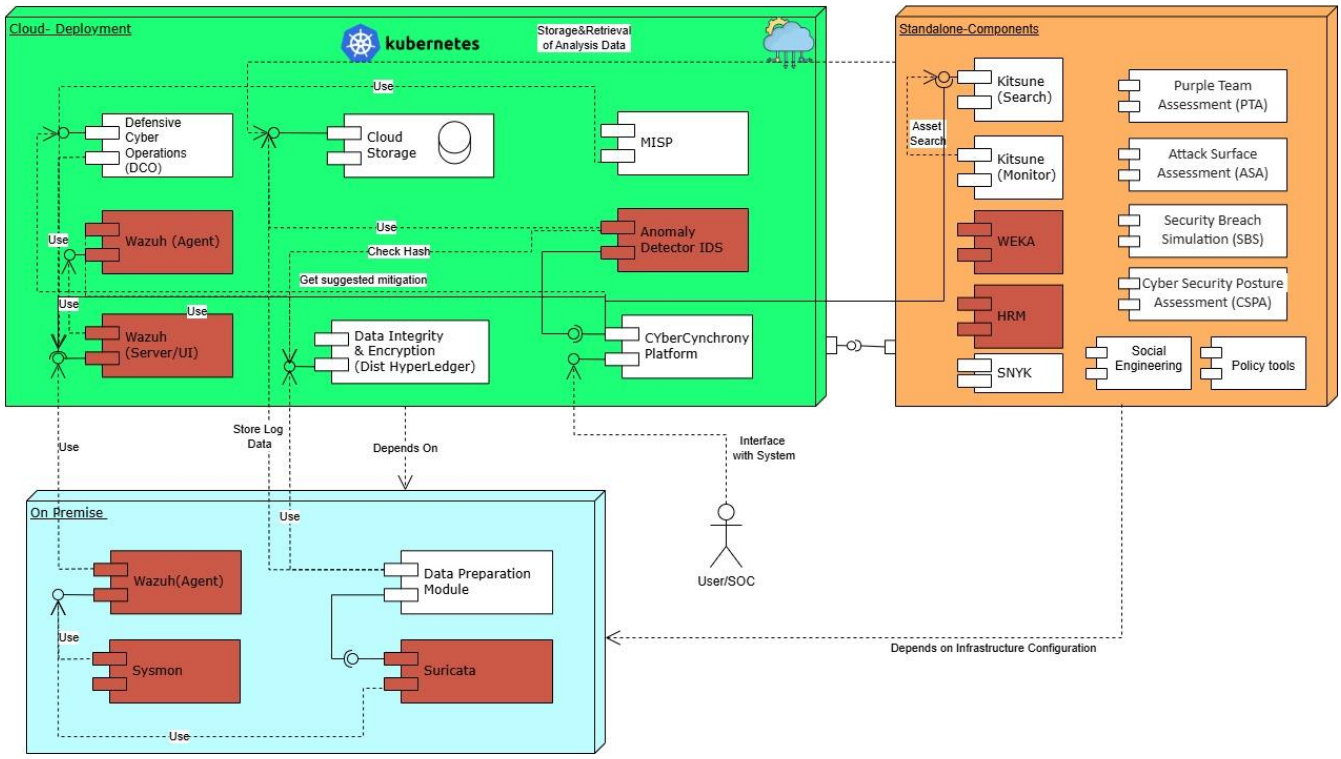

Figure 1: CYberSynchrony Modular Architecture – Updated Overview

Suricata and Wazuh, in combination with Sysmon, form the core of CYberSynchrony's on-premises monitoring capability. Suricata operates as the primary network intrusion detection system (IDS), capturing real-time packet streams and generating protocol-rich logs. Sysmon collects low-level system events such as process creation and file modifications. Wazuh functions as the central agent for collecting this telemetry. Deployed locally, the Wazuh agent aggregates data from both Suricata and Sysmon, normalizes it, and forwards it to the cloud-hosted Wazuh server stack. This setup allows for comprehensive log analysis and threat detection across both host and network layers. The cloud Wazuh agent, meanwhile, assists in integrity verification and can fetch additional data from long-term cloud storage if needed. This triad ensures robust, layered monitoring, enabling cross-correlation between network-level anomalies (from Suricata) and host-level behaviours (from Sysmon), all processed and visualized via Wazuh Server/UI.

The AI-based Anomaly Detector complements the rule-based detection capabilities of Wazuh and Suricata with advanced behavioural analysis. It leverages a combination of Multi-Head Self-Attention (MHSA) and Autoencoders (AE) to learn normal network behaviours and detect deviations indicative of potential threats. Rather than relying on direct telemetry collection, this component operates on log batches stored in the Cloud Storage. Before performing any analysis, the system initiates a check with the Integrity Module to ensure the logs have not been altered. This guarantees that the data is trustworthy for ML-based threat detection. Its position in the architecture reflects a decoupled, scalable analytical layer that enhances the platform's ability to detect subtle and previously unknown threats using historical and near-real-time traffic patterns.

© CYberSynchorny, 2024

The WEKA and Human Risk Management (HRM) components are designed for post-forensic analysis and user behaviour modelling, respectively. As such, they are positioned as standalone components that operate outside the continuous real-time detection pipeline. WEKA supports machine learning-driven post-event analysis. It processes historical data retrieved from cloud storage and enables scenario-based evaluation and validation of cybersecurity events, with a focus on anomaly detection through ensemble learning techniques. HRM contributes a human-centric risk analysis layer to the monitoring suite by profiling ICT users and adversaries through psychological and behavioural indicators. It feeds these insights into the broader risk management process within the CYberSynchrony ecosystem. While they are not involved in real-time monitoring, both components are tightly integrated with shared access to cloud storage and the platform's analytical workflows.

© CYberSynchorny, 2024

3 Wazuh

3.1 Component Description

Wazuh is a powerful open-source SIEM (Security Information and Event Management) and XDR (Extended Detection and Response) platform that serves inside T4.1. Its primary purpose is to provide centralized security monitoring, log analysis, threat detection, and compliance auditing across a wide array of IT assets. Within CYberSynchrony, Wazuh is deployed in the on-premises environment, where it plays a critical role in ingesting and analysing telemetry data from endpoints, servers, and network infrastructure. By continuously collecting logs and correlating them against known threat signatures and behavioural patterns, Wazuh enables the rapid identification of suspicious activities, unauthorized changes, and potential breaches.

Wazuh is a free and open-source security platform designed for threat prevention, detection, and response across various environments. The GitHub repository of this component is available online[1]. More information can be found in the official webpage[2] of the component.

In CYberSynchrony, Wazuh serves as the central analytical engine for endpoint and log-based detection, working closely with complementary tools like Suricata and Sysmon. Wazuh agents, deployed across systems, gather detailed system telemetry including process creation, network connections, registry changes, file modifications, and more. This data is forwarded securely to the Wazuh Server for processing and correlation. Alerts are indexed using Elasticsearch/OpenSearch via the Wazuh Indexer and visualized through the Wazuh Dashboard (a Kibana-based interface). This architecture ensures that system administrators and SOC teams can monitor hundreds of endpoints simultaneously, correlate incidents, and launch rapid response actions using playbooks or external SOAR platforms. IANUS is responsible for the deployment, configuration, and optimization of the Wazuh stack within the CYberSynchrony architecture. This includes fine-tuning detection rules, integrating with Sysmon for deep telemetry capture, connecting Wazuh to Suricata for unified endpoint and network threat analysis, and aligning its event models with the CYBRITE platform's overall incident response workflows.

Within the CYberSynchrony ecosystem (see Figure 1), **Wazuh** serves as the core endpoint detection and centralized log correlation engine, operating across both the on-premises and cloud environments (see **Error! Reference source not found.**). On the on-premises layer, the **Wazuh Agent** is deployed directly on monitored endpoints (e.g., servers, workstations), where it collects system-level telemetry such as process executions, file changes, registry modifications, and network activity. This data is enriched with **Sysmon** logs and passed through the **Data Preparation Module** for formatting and consistency. The structured logs are then securely stored in the **Cloud Storage** component. From there, the cloud-based **Wazuh Agent** retrieves these validated logs and forwards them to the **Wazuh Server/UI** for parsing, correlation, and rule-based analysis. Suspicious patterns are identified, indexed via **OpenSearch/Elasticsearch**, and made accessible through the **Wazuh Dashboard**, where security analysts perform alert triage and threat investigation. The **Wazuh stack** is integrated with **MISP** and the **CYberSynchrony Platform**, allowing endpoint data to be contextualized with threat intelligence and visualized through a unified SOC interface. This architecture enables **Wazuh** to function both as a local intrusion detection engine and a cloud-native SIEM, ensuring layered scalability, regulatory compliance, and data assurance through **Distributed Ledger Technology (HyperLedger)**.

Table 3: Wazuh alignment with CYberSynchrony objectives.

Project Objective	Justification
Objective 1: Implement state-of-the-art	Wazuh delivers modern SIEM and XDR functionalities, integrating log collection, file integrity monitoring, threat intelligence correlation, and compliance checks into a unified

[1] https://github.com/wazuh/wazuh
[2] https://wazuh.com/

© CYberSynchorny, 2024

cybersecurity monitoring tools	monitoring framework. Its open-source nature and modular design ensure it remains at the cutting edge of cybersecurity monitoring practices.
Objective 2: Achieve timely detection of potential cyber incidents	Wazuh continuously collects and analyses logs from endpoints, servers, and security appliances. Using predefined rules, threat intelligence feeds, and heuristics, it enables the rapid detection of unauthorized access, suspicious behaviour, and other cyber threats in real time.
Objective 3: Identify anomalies using advanced cybersecurity analytics	By using behavioural monitoring and anomaly detection rules, Wazuh can spot deviations from expected activity (e.g., unauthorized file changes, abnormal process executions). It correlates logs from various sources to surface hidden threats that may evade traditional signature-based detection.
Objective 4: Enable rapid and efficient response to detected cyber threats	Wazuh can automate incident response actions such as blocking IPs, isolating compromised hosts, or triggering alerts to external SOAR platforms. This supports rapid containment and mitigation of cyber threats, reducing the window of exposure and impact.

3.2 Architecture

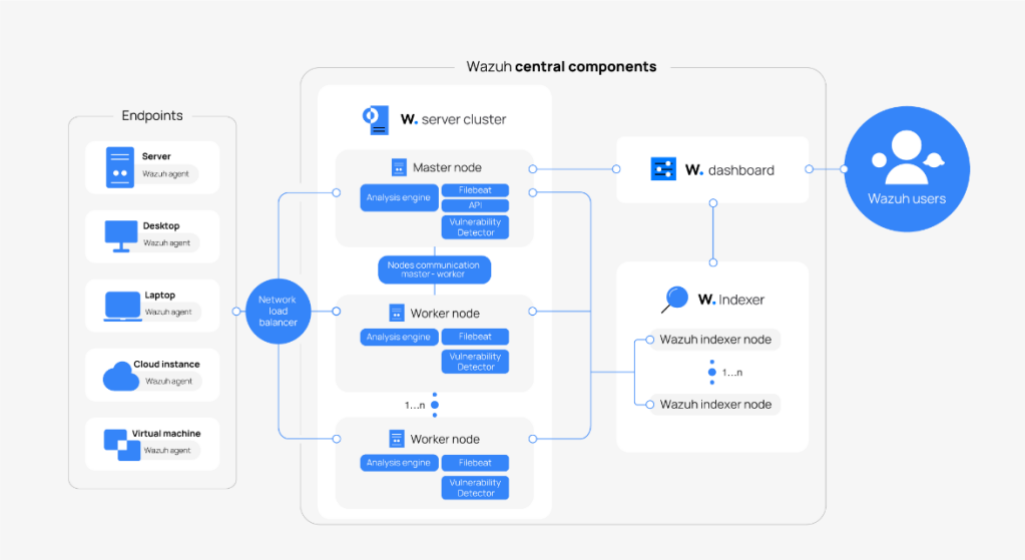

Figure 2: Wazuh Internal Architecture

Wazuh's internal architecture[3] is modular and horizontally scalable, designed to support high-throughput environments and real-time cybersecurity operations. It is built around four main component layers: Wazuh Agents, Wazuh Server Cluster (Master/Worker nodes), Wazuh Indexer, and the Wazuh Dashboard (see Figure 2). Deployed on a variety of endpoints such as servers, desktops, laptops, virtual machines, and cloud instances, the Wazuh Agent collects logs, performs file integrity checks, detects rootkits, and monitors system behaviour. These agents forward telemetry to a central Wazuh Server Cluster, which includes a Master node and multiple Worker

[3] https://documentation.wazuh.com/current/getting-started/architecture.html

© CYberSynchorny, 2024

nodes. The Master node handles the Filebeat-based data flow and provides RESTful API access, while Worker nodes execute detection logic using Wazuh's analysis engine and are responsible for processing events received from the agents.

The architecture includes a Network Load Balancer to ensure even distribution of agent communications across multiple Worker nodes, enhancing performance and availability. After processing, the analysed data is forwarded to the Wazuh Indexer, a dedicated cluster built on OpenSearch or Elasticsearch, where security events are indexed for fast search and long-term storage. Users interact with this data via the Wazuh Dashboard, a user-friendly web interface that supports alert visualization, rule management, agent monitoring, and threat investigation. The dashboard pulls indexed data from the Wazuh Indexer and displays it in a structured, intuitive format, giving SOC analysts real-time visibility into the security posture of monitored assets.

Functionally, Wazuh provides comprehensive coverage for key cybersecurity operations. It supports log normalization, correlation, anomaly detection, vulnerability detection (via CVE feeds), and policy compliance auditing. Within CYberSynchrony, IANUS will integrate Wazuh with Sysmon for endpoint-level process monitoring and Suricata for deep network packet inspection, forming a full-spectrum detection ecosystem. Alerts from Wazuh are further enriched via integration with the MISP threat intelligence platform, and then made actionable through the central CYberSynchrony UI. This cohesive architecture ensures seamless coordination between local monitoring, central processing, and human oversight.

Wazuh is a state-of-the-art open-source solution due to its extensibility, scalability, and robust security capabilities. Unlike many proprietary SIEM platforms, Wazuh offers flexibility in deployment (bare-metal, Docker, Kubernetes), customizable rule sets, and full transparency through its open-source codebase. It is supported by a vibrant global community, frequent updates, and enterprise-grade features without licensing barriers. Its architecture enables organizations in CYberSynchrony to deploy secure, interoperable, and automated threat detection workflows tailored to their operational and regulatory environments.

3.3 Technologies Used

Wazuh is developed using a combination of high-performance languages and modern frameworks to ensure scalability, extensibility, and cross-platform compatibility. The core backend of Wazuh is primarily written in C for system-level operations within the Wazuh Agent and Python for log parsing, rule evaluation, and event correlation in the server-side components. The frontend, namely the Wazuh Dashboard, is built using JavaScript, leveraging the Vue.js framework and Kibana plugins for real-time, interactive visualizations. Communication between agents and the server is secured through TLS encryption, and internal data pipelines are managed with Filebeat, an Elastic Stack component used for log shipping.

Wazuh's storage and indexing are powered by OpenSearch or Elasticsearch, enabling fast and scalable query capabilities across large volumes of security data. For API interactions and third-party integrations, Wazuh provides a RESTful API, facilitating automation, integration with SOAR tools, and export to other platforms like MISP or external SIEMs. Its deployment supports a wide range of environments — from Docker containers and Kubernetes clusters to bare-metal installations, making it highly adaptable to different infrastructure setups in the CYberSynchrony project.

Table 4: Wazuh technology stack.

Technology	Purpose
C	Used for performance-critical operations within Wazuh Agents (e.g., file monitoring, log collection).
Python	Backend scripting for event parsing, rule application, and core server logic.

© CYberSynchorny, 2024

JavaScript (Vue.js)	Frontend development of the Wazuh Dashboard UI.
Kibana plugins	Custom visualization layers and dashboards for alert and log analysis.
Filebeat	Log shipping component used to forward processed logs from agents and worker nodes to the Wazuh Indexer.
OpenSearch / Elasticsearch	Scalable storage and full-text search for security event data.
TLS Encryption	Ensures secure communication between Wazuh components.
REST API	Enables automation, external integrations, and remote management.
Docker / Kubernetes	Containerized deployment and orchestration in hybrid and cloud-native environments.
Ubuntu, CentOS, Windows	Supported operating systems for agent and server deployment.

© CYberSynchorny, 2024

4 Suricata IDS

4.1 Component Description

Suricata is an open-source, high-performance network threat detection engine that functions as an Intrusion Detection System (IDS), Intrusion Prevention System (IPS), and Network Security Monitoring (NSM) tool. Developed and maintained by the Open Information Security Foundation (OISF), Suricata plays a vital role in T 4.1 by monitoring real-time network traffic and detecting both known and novel cyber threats at the packet and flow level. IANUS is responsible for the deployment, configuration, and integration of Suricata, ensuring it works in tandem with endpoint and telemetry-based tools like Wazuh and Sysmon to create a comprehensive, multi-layered detection capability.

Suricata is licensed under the GNU General Public License (GPLv2), ensuring transparency and community-driven development. The source code is publicly available on GitHub[4], and comprehensive information, including documentation and downloads, can be found on the official website[5]

Operating in IDS mode within the on-premises layer of the CYberSynchrony architecture, Suricata receives mirrored traffic or live packet streams from monitored networks and generates JSON-formatted event logs that capture critical insights into network behaviour. These logs are processed through the Data Preparation Module and sent to the Wazuh Server for correlation, visualization, and alerting within the centralized SIEM environment.

By combining protocol-level visibility (HTTP, DNS, TLS, SSH, SMB, etc.) with support for community and custom rule sets (e.g., ET Open, Snort-compatible rules), Suricata greatly enhances the ability of Member States and critical infrastructure operators to detect cyber incidents that may not be visible through endpoint telemetry alone. This allows for detection of insider threats, lateral movement, zero-day exploits, and command-and-control communications that are common in modern attacks. Additionally, Suricata's compatibility with MISP allows for the direct ingestion of threat intelligence feeds, ensuring that detection capabilities evolve in tandem with the threat landscape.

Suricata functions as the primary IDS in CYberSynchrony, deployed exclusively in the on-premises segment to analyse raw network traffic in real time (*see Error! Reference source not found.*). It is co-located with the **Wazuh Agent** on the same endpoint, forming a dual telemetry point for both host and network perspectives. **Suricata** inspects packet streams using signature-based rules (e.g., ET Open ruleset) and performs deep packet inspection across protocols such as HTTP, DNS, TLS, and SMB to identify indicators of attack, such as command-and-control (C2) traffic, port scanning behaviour, or exploitation attempts. Upon detection, **Suricata** generates event logs in EVE JSON format, which are immediately routed through the **Data Preparation Module**. This module ensures that **Suricata**'s network events are normalized and compatible with the **Wazuh** ingestion pipeline. Once processed, these alerts are sent alongside host-based telemetry to the cloud-deployed **Wazuh Server** for SIEM correlation. The integration of **Suricata** with **Wazuh** allows for cross-layer threat detection, where a single malicious behaviour can be simultaneously validated through both endpoint signals and network activity.

Table 5: Suricata alignment with CYberSynchrony objectives.

Project Objective	Justification
Objective 1: Implement state-of-the-art cybersecurity monitoring tools	Suricata brings advanced network-layer inspection and protocol analysis to the project. It is widely recognized as one of the most efficient and accurate open-source IDS/IPS systems, with capabilities that match or exceed those of commercial alternatives.

[4] https://github.com/OISF/suricata
[5] https://suricata.io

© CYberSynchorny, 2024

Objective 2 : Achieve timely detection of potential cyber incidents	Suricata analyzes real-time packet data and identifies threats using rule sets and anomaly detection logic. It generates actionable alerts within milliseconds of malicious activity, allowing for swift incident awareness.
Objective 3 : Identify anomalies using advanced cybersecurity analytics	Suricata detects protocol misuse, abnormal traffic flows, and out-of-pattern behaviours across multiple protocols. It provides deep packet inspection and flow data for use in behavioural analysis and threat hunting.
Objective 4 : Enable rapid and efficient response to detected cyber threats	When deployed in IPS mode or combined with firewalls and SOAR tools, Suricata can actively block malicious traffic. In IDS mode (used within CYberSynchrony), it feeds detailed alerts into the SIEM (Wazuh), where they can trigger automated or manual response playbooks.

4.2 Architecture

Suricata's architecture follows a modular and high-performance pipeline that enables real-time traffic inspection, deep packet analysis, and threat detection at scale. The process begins with Packet Capture, where Suricata intercepts traffic directly from the network interface card (NIC) using libraries such as libpcap or AF_PACKET, or it can process offline data using PCAP files. Once captured, traffic flows into the Packet Decoding stage. Here, Suricata uses dedicated protocol decoders (e.g., for Ethernet, IPv4/IPv6, TCP/UDP, HTTP, TLS, DNS, SSH, SMB, and more), which unpack each layer of the packet and extract relevant metadata. This stage is essential for enabling context-aware detection, allowing the system to understand the structure and intent of each packet.

The decoded traffic is then processed by the Detection Engine, which lies at the heart of Suricata. It uses Snort-compatible rules and supports multi-threaded parallelism, allowing for simultaneous evaluation of thousands of rules against live traffic without degradation in performance. These rules can be sourced from public feeds such as Emerging Threats (ET Open), custom rule sets, or directly generated from MISP threat intelligence indicators. The detection logic includes signature matching, anomaly detection, flow tracking, and content inspection. Additionally, Suricata supports stream reassembly and protocol detection on non-standard ports, increasing its detection accuracy in evasion-prone environments.

After detection, events are passed to the Output Module, which formats alerts and metadata into EVE JSON (Event/Alert Output), Suricata's native logging format. These outputs can be sent in real-time to external systems such as Wazuh, Logstash, or Kafka, or stored locally for forensic analysis. Suricata's output layer is highly configurable, supporting alert severity levels, payload logging, protocol metadata, flow records, and more. This stage enables the full integration of Suricata into the broader CYberSynchrony architecture, particularly through

© CYberSynchorny, 2024

the Data Preparation Module, which ensures that network alerts are normalized before reaching the cloud-based Wazuh server.

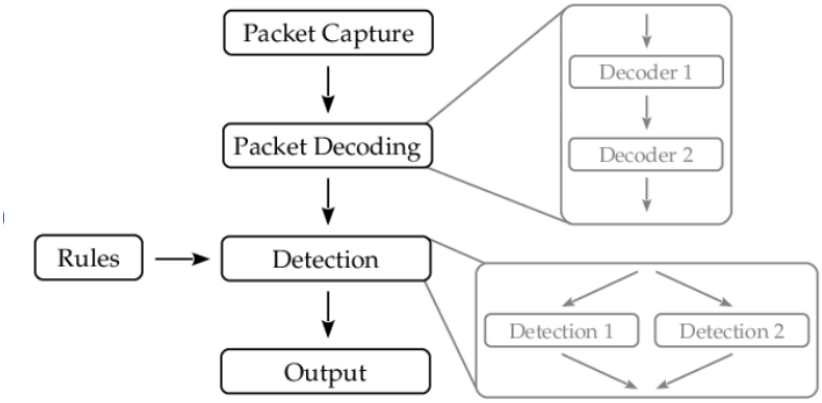

Figure 3: Suricata Internal Architecture [6]

Suricata's design is considered state-of-the-art for several reasons. First, it combines IDS, IPS, and NSM functionalities in a single engine, reducing the need for multiple tools. Its multi-threaded detection engine, support for offloading packet processing to GPUs or DPDK, and inline operation mode make it suitable for both high-speed networks and embedded deployments. Furthermore, Suricata adheres to open standards, supports deep protocol awareness, and is actively maintained by a global community through the Open Information Security Foundation (OISF). Compared to traditional IDS/IPS tools, it offers superior performance, flexibility, rule compatibility, and integration with modern cybersecurity platforms like SIEMs, SOARs, and threat intelligence frameworks (e.g., MISP). This makes Suricata an ideal tool for real-time network threat detection in mission-critical environments such as those addressed by CYberSynchrony.

4.3 Technologies Used

Suricata is developed in C, a low-level systems programming language that ensures high performance and memory efficiency, which is critical for real-time packet processing and deep inspection tasks. Its core engine handles packet capture, protocol decoding, detection rule processing, and output generation, all built for maximum speed and concurrency. While Suricata does not have a frontend per se, it offers configuration via YAML, and integrates seamlessly with log shippers and visualization tools such as EVE JSON, ELK Stack (Elasticsearch, Logstash, Kibana), or Wazuh Dashboard for frontend-style interaction. The engine uses libraries like libpcap, PF_RING, and AF_PACKET for packet capture; Hyperscan for fast pattern matching across large rule sets; and Lua scripting support to enable custom detection logic or alert formatting. It is highly extensible and can operate on a variety of platforms including Linux, FreeBSD, and Windows (in limited form). For deployment, Suricata supports both standalone and clustered installations, making it ideal for integration within the CYberSynchrony architecture, where it is deployed at the on-premises layer, feeding data into the broader monitoring and analysis pipeline.

[6] Ghafir, I., Svoboda, J., Hammoudeh, M., & Prenosil, V. (2016, August). *A survey on network security monitoring implementations*. In 2016 International Conference on Future Internet of Things and Cloud Workshops (FiCloudW) (pp. 77–82). IEEE. https://doi.org/10.1109/W-FiCloud.2016.30

© CYberSynchorny, 2024

Table 6: Suricata technology stack.

Technology	Purpose
C	Core development language for high-performance packet processing
YAML	Configuration management (e.g., detection rules, outputs, protocols)
libpcap / AF_PACKET / PF_RING	Libraries for packet capture on different OS-level interfaces
Hyperscan	High-speed regular expression matching for deep content inspection
Lua	Scripting engine for extending detection logic or formatting output
EVE JSON	Structured output format for alerting and metadata export
Elasticsearch / Logstash / Kibana (ELK Stack)	Visualization and indexing of Suricata logs via third-party integration
Wazuh	SIEM integration for alert ingestion, correlation, and visualization
Linux / FreeBSD	Supported operating systems for production deployment
Docker	Optional deployment via containerization (community-supported)

© CYberSynchorny, 2024

5 Anomaly detector IDS

5.1 Component Description

Anomaly detector functions as network monitoring tool in CYberSynchrony. Its purpose is to identify potential security threats through real-time examination of network traffic patterns.

The system implements a dual-model architecture combining a classifier for known attack type identification such as Distributed Denial of Service attacks (DDoS) and an autoencoder for anomaly detection trained on normal network traffic, enabling it to recognize both established and zero-day attack signatures using transformer-based feature encoding and automated threshold-based alerting. The anomaly detection capabilities are powered by real-time processing via Kafka streaming, making it suitable for monitoring high-throughput network environments while maintaining low latency response times.

Within the CYberSynchrony ecosystem, the Anomaly Detector integrates with several key components, particularly the Cloud Storage for reading network logs and the Data Integrity Module for ensuring the trustworthiness of the analysed data. Before performing any analysis, the system initiates integrity verification to confirm logs have not been altered, guaranteeing that anomaly detection is based on authentic network traffic data.

Table 7: Anomaly detector IDS alignment with CYberSynchrony objectives.

Project Objective	Justification
Objective 1: Implement state-of-the-art cybersecurity monitoring tools	The AI Anomaly Detector leverages deep learning techniques, including Multi-Head Self-Attention (MHSA) and Autoencoders, to perform dynamic traffic profiling and detect behavioural deviations beyond static rule-based systems.
Objective 2 : Achieve timely detection of potential cyber incidents	The component enables near real-time processing of validated network logs and flags suspicious patterns as they emerge, supporting early-stage awareness of security threats.
Objective 3 : Identify anomalies using advanced cybersecurity analytics	The model identifies outliers in traffic behaviour by learning normal patterns over time, supporting the detection of complex and previously unknown attack vectors.

5.2 Architecture

The anomaly detector is modular with several connected modules that work together as a coherent system.

The setup uses three main containerized services:

1. Zeek monitoring - captures and formats network traffic data
2. Kafka broker - handles our data streams
3. Anomaly detector engine - where the actual analysis happens

The actual detection process follows multiple steps to analyse traffic effectively:

First, the system takes in data from Zeek through Kafka streams. Zeek is configured to log various network characteristics, both categorical (like protocol types) and numerical (like packet sizes), and transform them into a standard format our system can process. Next, we use self-attention mechanism as feature extractor to find relationships between different traffic features. These features are fed to the actual Detector components that work through two parallel paths:

© CYberSynchorny, 2024

1. The autoencoder component compresses traffic data to a smaller representation and then tries to reconstruct it. When reconstruction errors are high, it suggests the traffic pattern is unusual.
2. The classifier component directly identifies known attack types, giving context to the anomalies found.

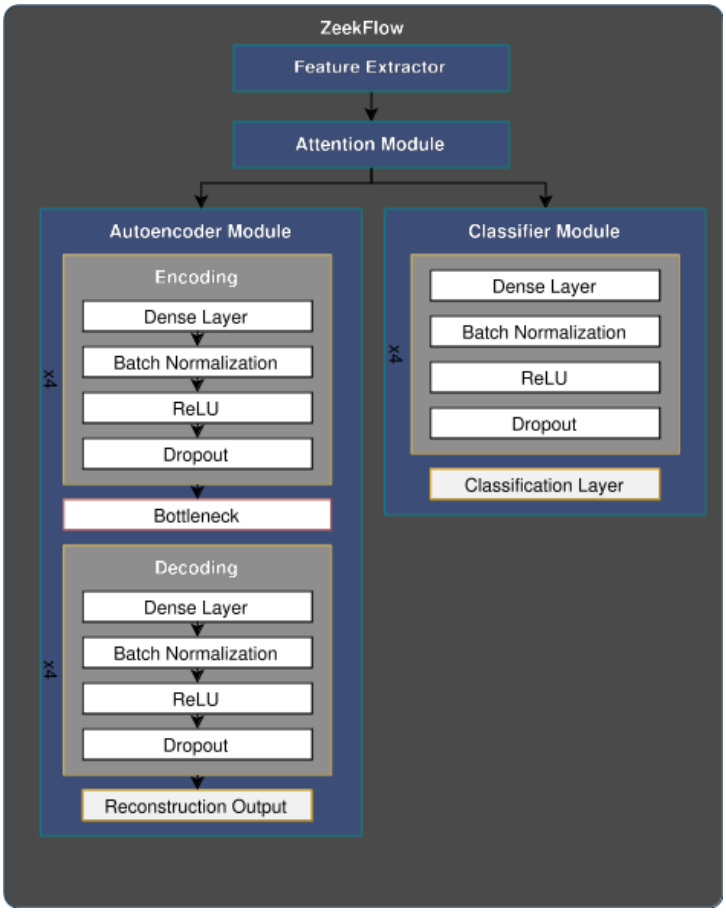

Figure 4: Detection workflow diagram showing data flow from capture to alerting

For each network flow, we calculate how different it is from normal patterns. If this difference exceeds our threshold, we mark it as suspicious and include details about the likely attack type.

After processing, alerts about potential threats are sent to configured endpoints where security teams can take appropriate action.

© CYberSynchorny, 2024

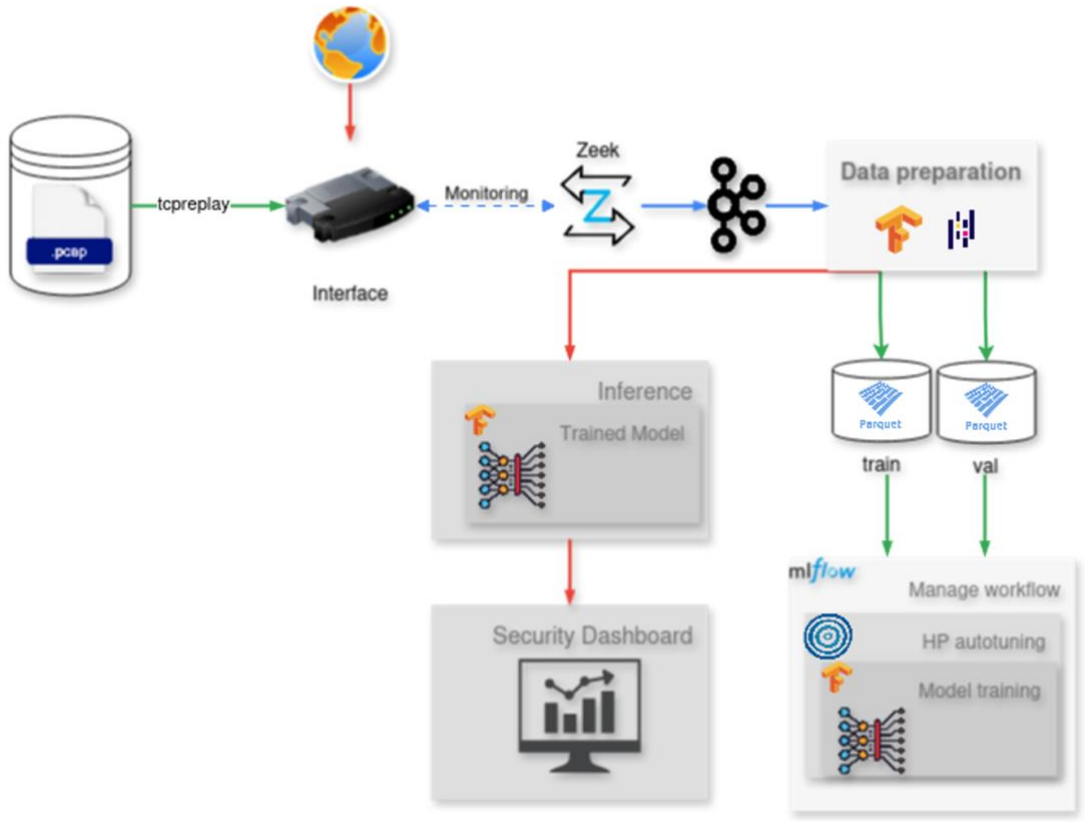

Figure 5: Anomaly detector data flow and operation

5.3 Technologies Used

Table 8: Anomaly detector IDS technology stack.

Technology	Purpose
Python 3.12	Powers our entire codebase including data handling, model creation, and system integration
TensorFlow & Keras	Build and train our neural network
Apache Kafka	Provide reliable message handling for streaming network data between system components
Pandas & NumPy	Handle data manipulation and numerical operations needed for feature engineering
Docker	Package our system components for consistent deployment across environments
Zeek	Generate structured network logs that feed our detection system
MLflow	Manage our machine learning development lifecycle
Scikit-learn	Additional data processing and model evaluation
Optuna	Automatically tune our model parameters for optimal performance

© CYberSynchorny, 2024

6 WEKA

6.1 Component Description

WEKA (Waikato Environment for Knowledge Analysis) is a powerful open-source software suite for machine learning and data mining tasks. Designed with a user-friendly graphical interface, WEKA allows users to easily apply a wide range of machine learning algorithms to datasets for tasks such as classification, regression, clustering, and association rule mining. It includes tools for data pre-processing, visualization, and evaluation, and supports scripting and integration with other platforms. WEKA is written in Java and is widely used in academia and industry for research, teaching, and prototyping machine learning models.

WEKA is a valuable tool in the context of state-of-the-art machine learning solutions due to its accessibility, flexibility, and strong educational value. While modern deep learning frameworks like TensorFlow or PyTorch dominate cutting-edge AI applications, WEKA excels in rapid prototyping, algorithm comparison, and applied data science—especially for structured data problems. Its graphical interface eliminates the need for extensive coding, making it ideal for students, researchers, and domain experts who want to experiment with machine learning without a steep learning curve. WEKA includes a comprehensive collection of classic and widely used algorithms that serve as reliable benchmarks or baselines for more advanced solutions. Additionally, its integration with scripting (e.g., through Java or Python via WekaDeeplearning4j) and compatibility with popular data formats allow it to fit within larger, more complex pipelines. Overall, WEKA offers a solid, interpretable foundation for understanding machine learning principles and evaluating new methods in comparison with proven techniques.

Weka aligns closely with the objectives of WP4, and particularly with Task 4.2, titled *Entity Behaviour Analytics (UEBA)*. UEBA focuses on monitoring the behaviour of users and entities within a network to detect anomalies that may indicate potential security threats. By establishing and analysing baseline behavioural patterns, UEBA is able to flag deviations that could signify malicious activity or breaches. WEKA supports this process through its robust machine learning capabilities, enabling the development and testing of models that can effectively identify such anomalies. Its flexibility in handling structured data, combined with its range of classification and clustering algorithms, makes Weka an ideal tool for implementing UEBA strategies. Continuous behavioural analysis through WEKA helps in early risk detection, promoting swift response and strengthening internal cybersecurity defences.

Table 9: WEKA alignment with CYberSynchrony objectives.

Project Objective	Justification
Objective 2: Achieve timely detection of potential cyber incidents.	WEKA can achieve timely detection of potential cyber incidents through its powerful suite of machine learning algorithms and pre-processing capabilities.
Objective 3: Identify anomalies using advanced cybersecurity analytics.	WEKA can effectively identify anomalies in the cybersecurity context by using advanced machine learning algorithms and data analytics techniques

6.2 Architecture

WEKA is well-suited for building an Intrusion Detection System (IDS) due to its extensive suite of machine learning algorithms, intuitive interface, and powerful data pre-processing tools. Intrusion detection often involves analysing large volumes of structured network traffic data to identify patterns indicative of malicious activity. WEKA simplifies this process by enabling users to quickly load, clean, and transform datasets, experiment with different classifiers such as decision trees, support vector machines, or ensemble methods, and evaluate model performance using built-in cross-validation and visualization tools. Its ease of use allows security analysts and

© CYberSynchorny, 2024

researchers to focus on feature selection and model tuning without needing to write extensive code. Additionally, WEKA's support for real-time classification and external integrations makes it a practical choice for developing and testing IDS prototypes before deploying more complex solutions.

WEKA can achieve timely detection of potential cyber incidents through its powerful machine learning capabilities and comprehensive suite of tools for data pre-processing, classification, and anomaly detection. Here's how Weka can contribute to early detection:

1. **Data Pre-processing**: Weka enables the effective transformation of raw network traffic and system logs into structured datasets. Through feature extraction, normalization, and scaling, Weka ensures that data is in an optimal format for analysis. This helps in reducing noise and focusing on critical patterns indicative of cyber incidents.
2. **Anomaly Detection**: Weka offers several algorithms that are well-suited for detecting anomalies in network traffic, system behaviours, or user activities. Methods like clustering (e.g., k-means), density-based algorithms, and decision trees (e.g., J48) can identify unusual patterns that deviate from normal system behaviour, which could signify potential security breaches or attacks.
3. **Real-Time Processing**: While Weka is traditionally used for batch processing, it can be integrated with real-time data streams for more immediate analysis. By leveraging tools like **MOA (Massive Online Analysis)**, which integrates with Weka, it is possible to build models that continuously monitor incoming data, detect anomalous behaviour, and trigger alerts for immediate response.
4. **Classification Models**: Weka supports a wide array of classification algorithms (e.g., Random Forest, SVM, Naive Bayes) that can be trained to detect specific types of cyber incidents, such as malware activity, unauthorized access, or DoS attacks. Once trained on historical incident data, these models can classify real-time inputs to quickly identify threats.
5. **Ensemble Methods**: Weka's ensemble methods, like **bagging** or **boosting**, help combine the output of multiple models to improve accuracy and robustness in detecting cyber incidents. These methods are beneficial in scenarios where no single model can detect all types of attacks, but a combination of models can achieve higher accuracy.
6. **Continuous Evaluation and Tuning**: Weka's ability to continuously evaluate and adjust models through techniques like cross-validation or performance evaluation helps ensure that detection systems remain accurate and adaptive to evolving cyber threats. This ongoing refinement contributes to maintaining a low false positive rate while ensuring timely incident detection.

In CyberSychrony WEKA will be an external IDS tool. WEKA will collect data from the cloud storage of CyberSychrony and will use them in order to identify several cybersecurity attacks. Through the use of ensemble methods WEKA will be able to correctly identify malfunctions of the systems in inspection and produce alerts to the users. WEKA achieves timely detection of potential cyber incidents by enabling fast, accurate analysis through data pre-processing, anomaly detection, and real-time classification, all of which can be further enhanced by integrating ensemble methods and continuous evaluation strategies.

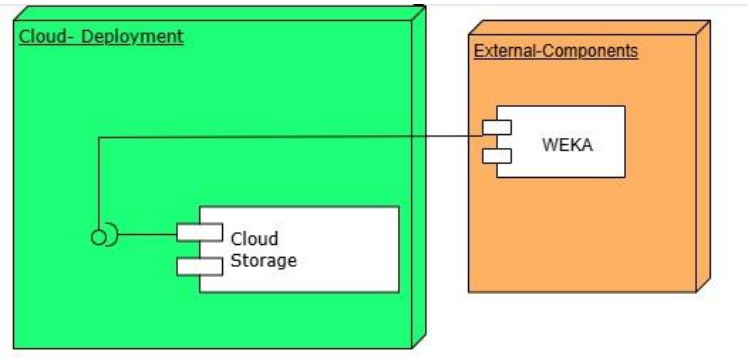

Figure 6: Architecture of WEKA in the CyberSychrony ecosystem

6.3 Technologies Used

Table 10: WEKA technology stack.

Technology	Purpose
Java Standard Edition (SE) libraries	for GUI components (Swing), I/O operations, networking, and multithreading
Apache Commons Math	Used for mathematical operations and statistical functions
XStream	For serializing objects to XML, useful for saving and loading models and workflows
LibSVM / LibLINEAR	Integrated for support vector machine implementations and linear classification
JFreeChart	For generating visualizations and charts within WEKA
MOA (Massive Online Analysis)	Integrates with WEKA for stream data mining
Deeplearning4j (via WekaDeeplearning4j package)	Adds deep learning support through integration with this Java-based deep learning library
Python (via WekaPython or PyWeka)	Allows integration with Python scripts and libraries
R (via RWeka)	Provides access to Weka from R, enabling advanced statistical analysis and visualization

© CYberSynchorny, 2024

7 Human RM (HRM)

7.1 Component Description

Human-Centric Risk Management (HRM) is an approach that extends traditional technical risk management methods by incorporating cyber socio-psychology techniques to identify human-related threats and assess associated vulnerabilities and risks. It is straightforward to implement, leveraging open-source tools like ENISA and OWASP for technical risk assessment, and using co-creation workshops to evaluate and manage human-centric risks. HRM aligns with the ISO 27001 framework and is specifically designed to help SMEs integrate human factors into their security strategies. By taking a proactive stance toward identifying human threats and applying best practices in security management, HRM enables companies to strengthen their overall security posture—transforming employees into key defenders against evolving cyber threats.

Table 11: HRM's alignment with CYberSynchrony objectives.

Project Objective	Justification
Objective 3: Identify anomalies using advanced cybersecurity analytics.	HRM supports Objective 3 by detecting behavioural anomalies in ICT users through socio-psychological profiling and risk indicators from the CyberSecurity Posture Assessment (CSPA). By correlating user behaviour with technical risk metrics, it identifies threats that traditional analytics might miss, enhancing anomaly detection beyond system-level monitoring.

7.2 Architecture

Socio-psychological tools are essential for managing human threats in risk management, as they assess and mitigate the impact of human factors on cybersecurity and organizational safety. These tools evaluate psychological and social behaviours that shape security practices. For example, the Security Behaviour Intentions scale measures attitudes toward behaviours such as password management and software updates, which are critical for maintaining strong cybersecurity protocols.

The Human-Centric Risk Management (HRM) methodology incorporates human factor considerations into the ISO 27001 framework, helping companies manage security risks more effectively by taking into account the profiles of their ICT users (such as administrators, defenders, operators, employees, and third parties). HRM focuses on proactively identifying and mitigating human-related threats, applying best practices for security management to enhance a company's overall security posture and safeguard valuable assets against evolving cyber threats.

HRM utilizes the Behaviour Model (B=MAT) to determine the type of cue needed to prompt the desired action, based on an individual's motivation and ability. According to Fogg, the likelihood of a behaviour (B) occurring is the result of the interaction between Motivation (M), Ability (A), and the appropriate Trigger (T), which is why it is known as the B=MAT model.

In HRM, extended psychological profiles are used to analyse not only motivations, abilities, and triggers (based on Fogg's model) but also personality traits and social characteristics. Cyber profiling is the tool employed to identify human threats and vulnerabilities of ICT users, serving as a proactive approach to select targeted social controls that mitigate employees' susceptibility to human threats. The HRM methodology uses a multidimensional cyber psychological profile to assess the factors that influence secure behaviours.

The HRM methodology comprises the following three main phases according to standards (Figure 7)

© CYberSynchorny, 2024

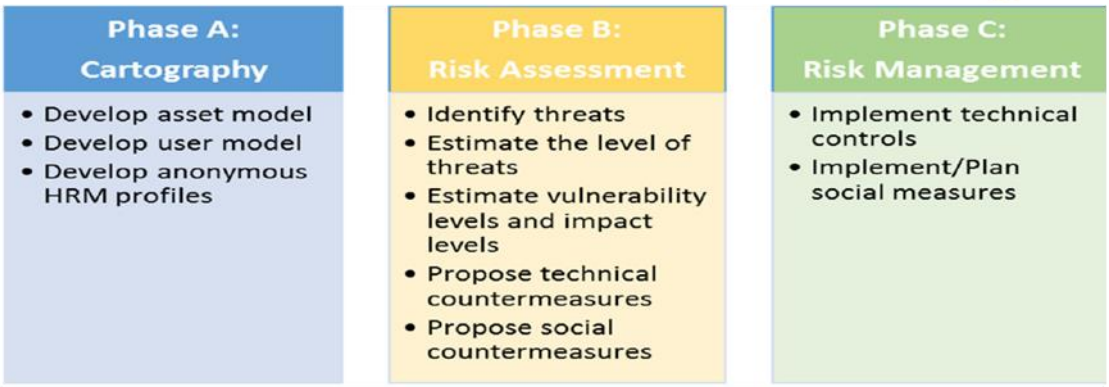

Figure 7: HRM phases

HRM in CyberSychrony is an external tool that will offer an enhanced risk assessment. The Human Risk Management (HRM) methodology builds on ISO 27001 frameworks and utilizes available tools to assess technical threats and estimate associated risks. For threats related to the human element, HRM applies socio-psychological techniques to evaluate the maturity of ICT users in adopting security practices, as well as the strength of potential adversaries. It develops and assesses profiles of both ICT users and adversaries, integrating these evaluations into comprehensive risk assessments. HRM will collect Risk Indicators from the CSPA, which functions as the reporting and analysis layer for operational cybersecurity activities. The CSPA consolidates outputs from the Endpoint Discovery and Vulnerability Assessment (EDVA) and Purple Team Assessment (PTA) services, mapping them against the SANS Top 20 Critical Security Controls to deliver a data-driven overview of an organisation's cyber maturity. The resulting control-by-control assessments and recommendations are stored in the platform's storage and later accessed by HRM to inform the creation of accurate and evidence-based user and adversary profiles.

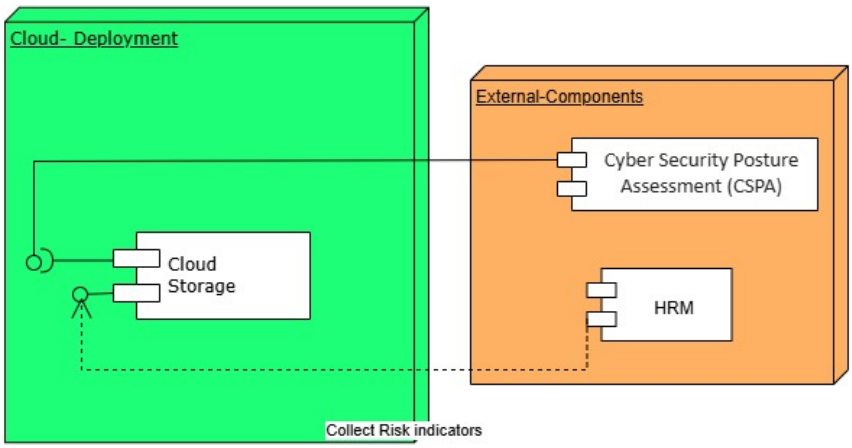

Figure 8: Architecture of HRM in the CyberSychrony ecosystem

© CYberSynchorny, 2024

8 Conclusions and next steps

Deliverable D4.1 presents the first integrated release of the CYberSynchrony Monitoring Suite, which implements key components of the CYBRITE module under WP 4. These components establish a hybrid, modular architecture for advanced cyber threat monitoring, leveraging both signature-based and behaviour-based techniques. The system integrates on-premises data collection (e.g., Suricata, Sysmon), cloud-based correlation and visualization (Wazuh Server/UI), and advanced analytics (AI Anomaly Detector, UEBA), forming the foundation for real-time detection and contextual threat awareness.

The deliverable outlines the individual capabilities of each tool and, more importantly, describes how they interoperate through a shared cloud storage layer, verified data lifecycle (via WP5), and unified user access via the CYberSynchrony Platform. This layered design ensures that data is continuously collected, validated, analysed, and visualized, enabling faster detection of cyber threats and supporting timely incident response actions. Tools such as WEKA and HRM extend this pipeline with standalone forensic and human risk analysis capabilities, enriching the monitoring process with post-incident insight.

The developments in this deliverable directly address key project objectives of WP4, including the deployment of state-of-the-art monitoring tools (Objective 1), enabling timely detection (Objective 2), and leveraging advanced analytics for anomaly identification (Objective 3).

Looking ahead, the next steps will focus on the deployment and validation of the monitoring components within the fully integrated CYberSynchrony platform, ensuring seamless interoperability with threat intelligence systems, response orchestration modules, and the distributed ledger infrastructure. Particular emphasis will be placed on ensuring compatibility, data exchange fidelity, and lifecycle traceability, so that the Monitoring Suite can operate as a trusted, scalable component of the end-to-end cybersecurity ecosystem foreseen by the project.

© CYberSynchorny, 2024

Annex I: *Preliminary PoC and Prototypes* (Wazuh)

As part of the initial prototype phase in T4.1, a default deployment of the open-source Wazuh security platform has been successfully installed (downloaded from the official Github repo[7]) and is fully operational within the IANUS development environment. This deployment includes the Wazuh Server, Indexer (OpenSearch-based), and the Wazuh Dashboard, running inside a virtual machine (VM) hosted on IANUS infrastructure. This instance is accessible locally within the IANUS network and provides a functional base for real-time log collection, threat detection, and dashboard-based alerting. The VM is configured to simulate endpoint activity using Wazuh agents and Sysmon logs to demonstrate core monitoring and analysis workflows without any custom modifications at this stage. The current Wazuh deployment reflects a standard operational configuration, offering insight into its default capabilities, interface structure, and interaction flow. This version enables security analysts to visualize alerts, monitor agent status, and examine incoming telemetry through the integrated dashboard UI. Access to the test environment is restricted to IANUS technical personnel, and no production data is involved. This PoC serves as a foundation for future customization and integration with Suricata, MISP, and the broader CYberSynchrony platform. Subsequent iterations will involve rule tuning, dashboard customization, API integrations, and enriched cross-layer analytics to support the advanced detection and response workflows planned in the project.

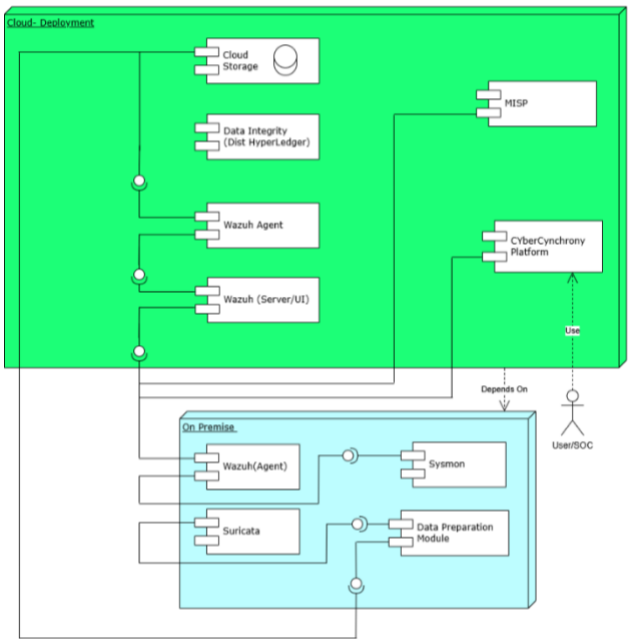

Figure 9: Wazuh-Suricata Placement in Architecture (internal connections)

[7] https://github.com/OISF/suricata

© CYberSynchorny, 2024

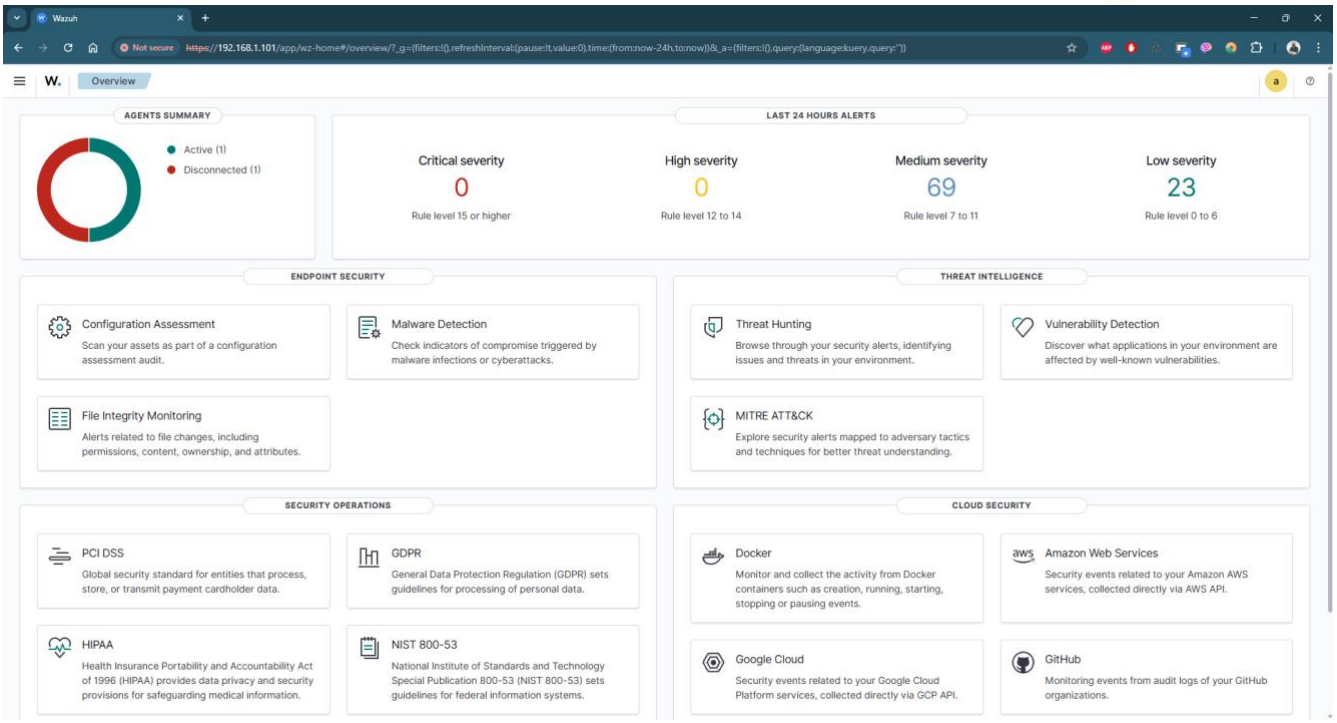

Figure 10: Wazuh Overview Page

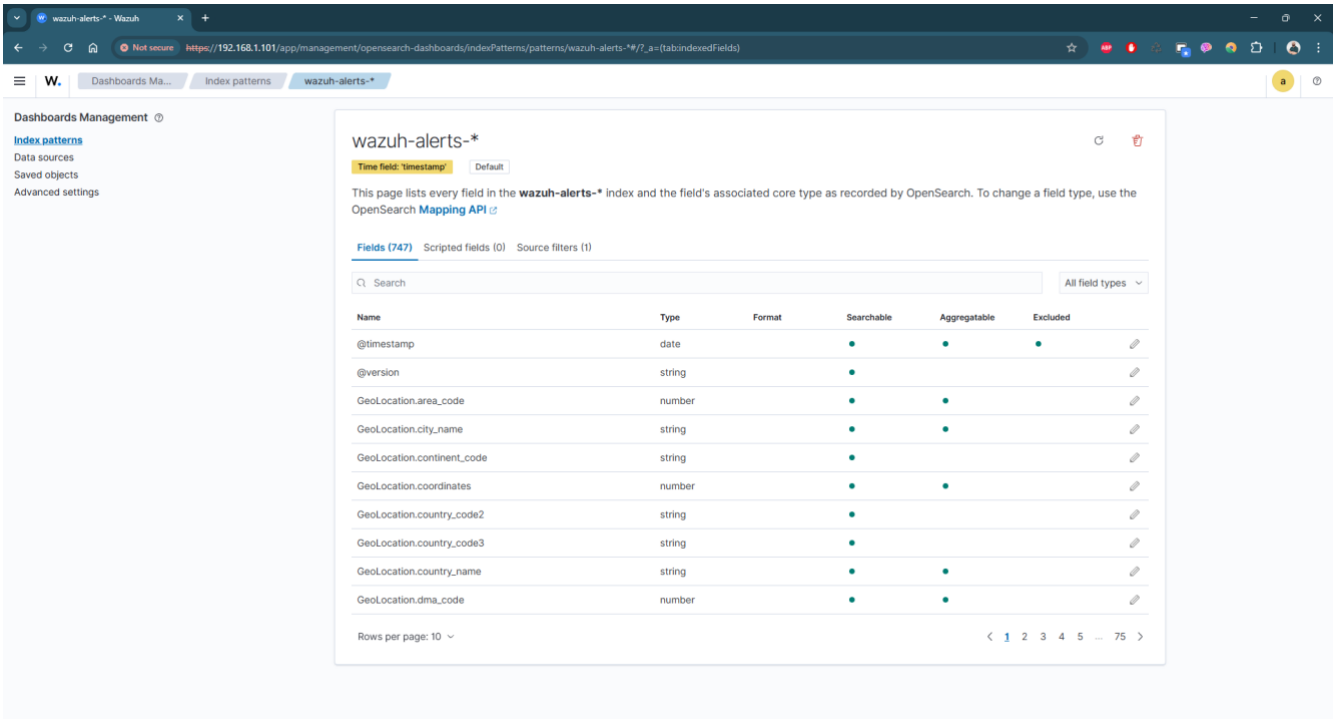

Figure 11: Wazuh Alerts

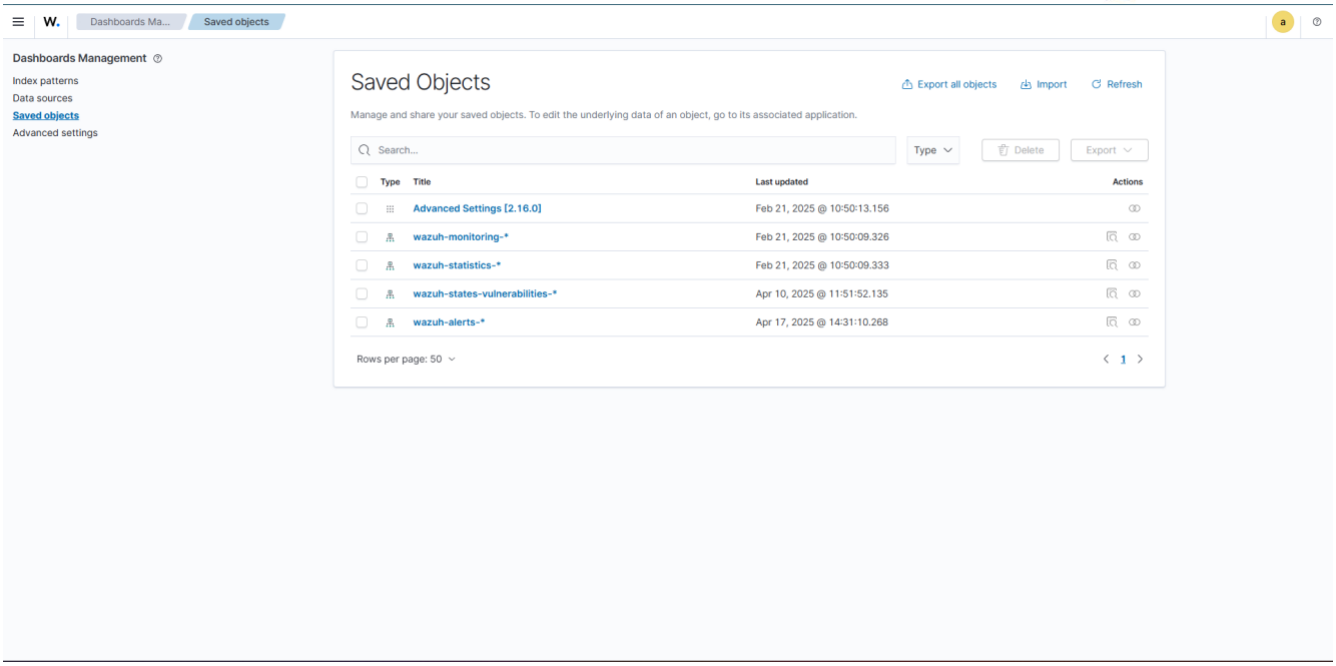

Figure 12: Wazuh Saved objects

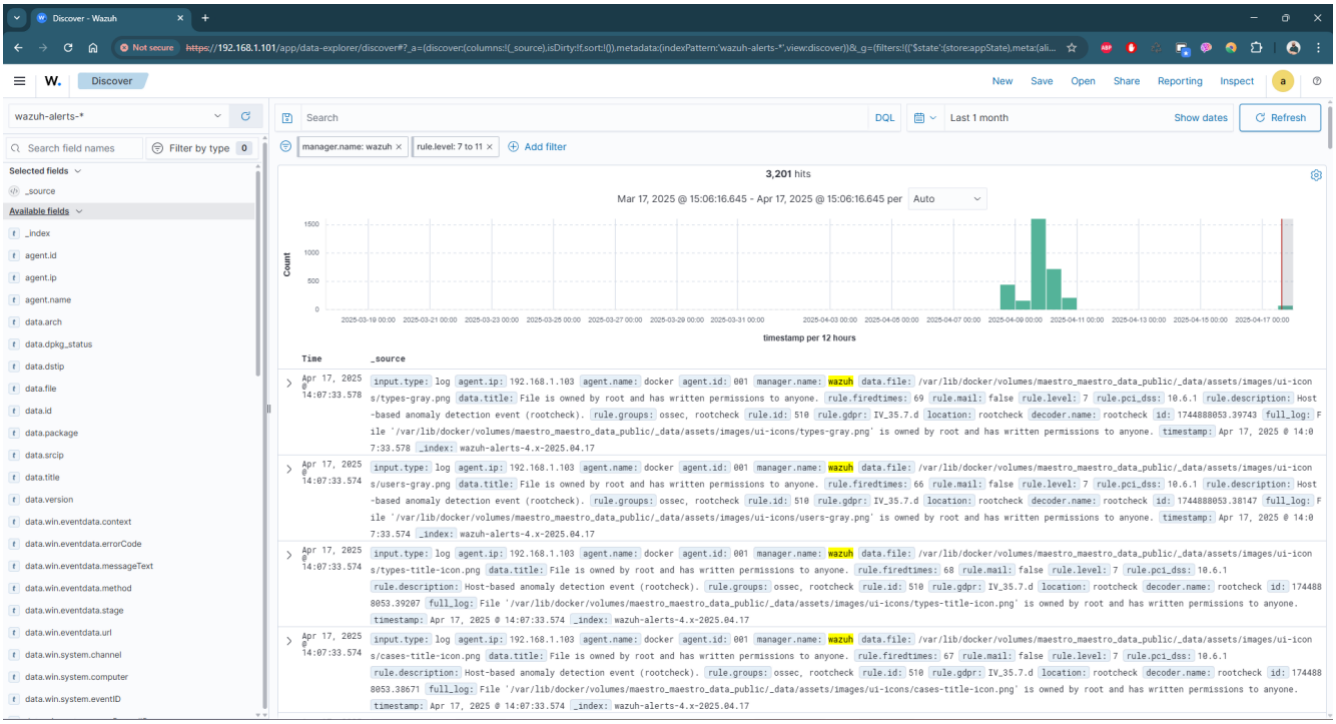

Figure 13: Wazuh events (with custom data from last month)

Table 12: Wazuh API dacumentation

Endpoint	Description
`GET /agents`	Retrieve a list of all registered agents.
`GET /agents/{agent_id}`	Fetch detailed information about a specific agent.
`POST /agents`	Register a new agent with the Wazuh manager.
`DELETE /agents/{agent_id}`	Remove an agent from the manager.
`GET /rules`	List all active detection rules.
`GET /rules/groups`	Retrieve rule groups and associated rules.
`PUT /rules/{rule_id}`	Update a specific rule.
`GET /decoders`	List all decoders used for log analysis.
`GET /manager/status`	Check the status of the Wazuh manager.
`GET /security/user/authenticate`	Obtain a JSON Web Token (JWT) for API authentication.

In regards to rule management and configuration, Suricata abides with the following:

- Custom Rules:
  - Wazuh allows the creation of custom rules to detect specific threats or behaviors. These rules can be added to the local_rules.xml file and are prioritized over default rules.
- Decoders:
  - Custom decoders can be defined to parse logs from new or uncommon sources, enhancing Wazuh's ability to interpret diverse log formats.
- Rule Testing:
  - Utilize the API to test new rules and decoders before deployment, ensuring they function as intended.
- Integration with MITRE ATT&CK:
  - Wazuh maps rules to MITRE ATT&CK techniques, aiding in threat classification and response planning.

© CYberSynchorny, 2024

Annex II: *Preliminary PoC and Prototypes (Suricata)*

As part of the early prototype phase of T4.1, a default installation of Suricata has been deployed within a local virtual machine hosted by IANUS. This installation represents an out-of-the-box configuration using Suricata, running in IDS mode on a Linux-based Docker container. The current setup is designed to monitor live traffic from a simulated interface and generate real-time alerts based on the Emerging Threats (ET Open) rule set. The VM is accessible from within the IANUS local network, and no external connectivity is required for its operation. The containerized instance was selected to ensure portability, isolation, and easier integration.

Unlike other tools that offer web-based dashboards, Suricata is entirely command-line driven, and interaction is done via terminal commands and log file monitoring. Users can verify that Suricata is running by checking active processes or by tailing logs such as /var/log/suricata/suricata.log or viewing real-time alerts in eve.json. The tool is already generating alerts from test traffic, including known malicious patterns like EICAR test signatures. Logs are formatted in JSON and can be piped into visualization or SIEM tools such as Wazuh or Logstash. This PoC installation serves as a baseline to demonstrate Suricata's packet inspection, rule matching, and alerting functionalities, and it will be expanded later to include custom rules, enriched MISP feeds, and tighter integration with the CYberSynchrony data processing pipeline. The Suricata prototype was deployed using Docker.

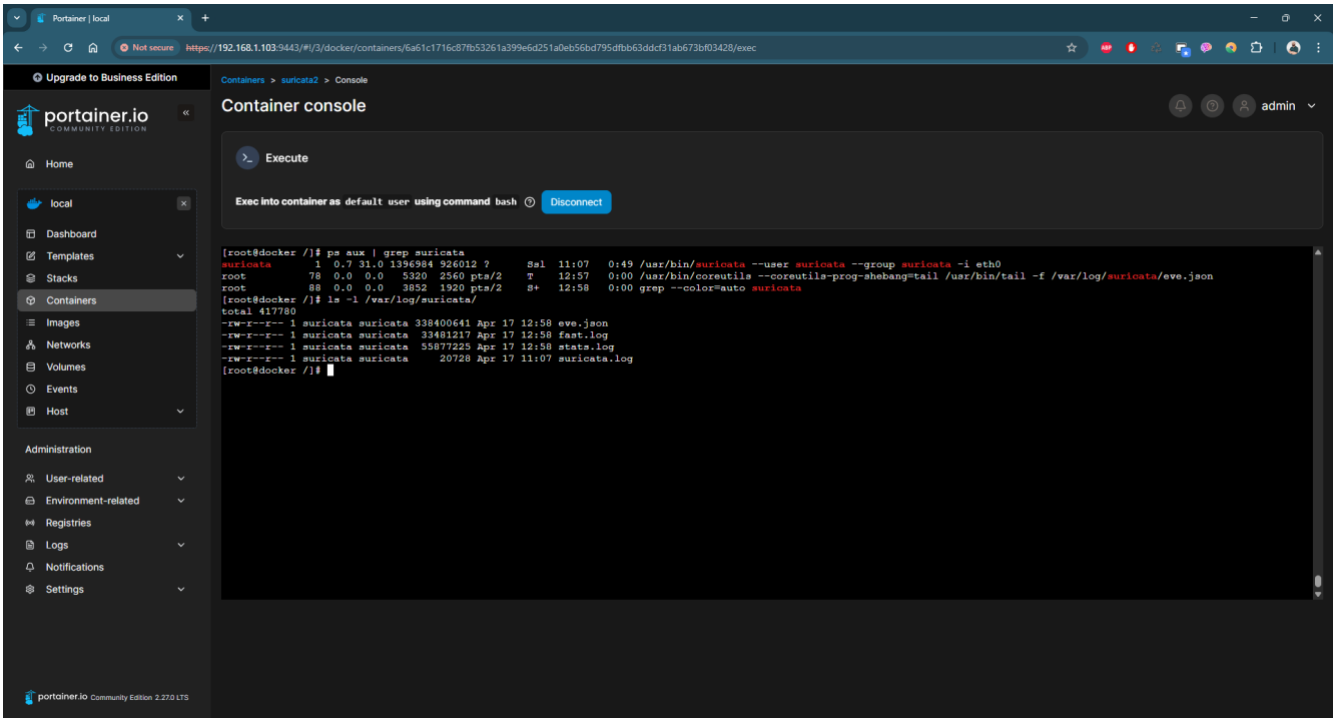

Figure 14: Verifying Suricata is up and running

© CYberSynchorny, 2024

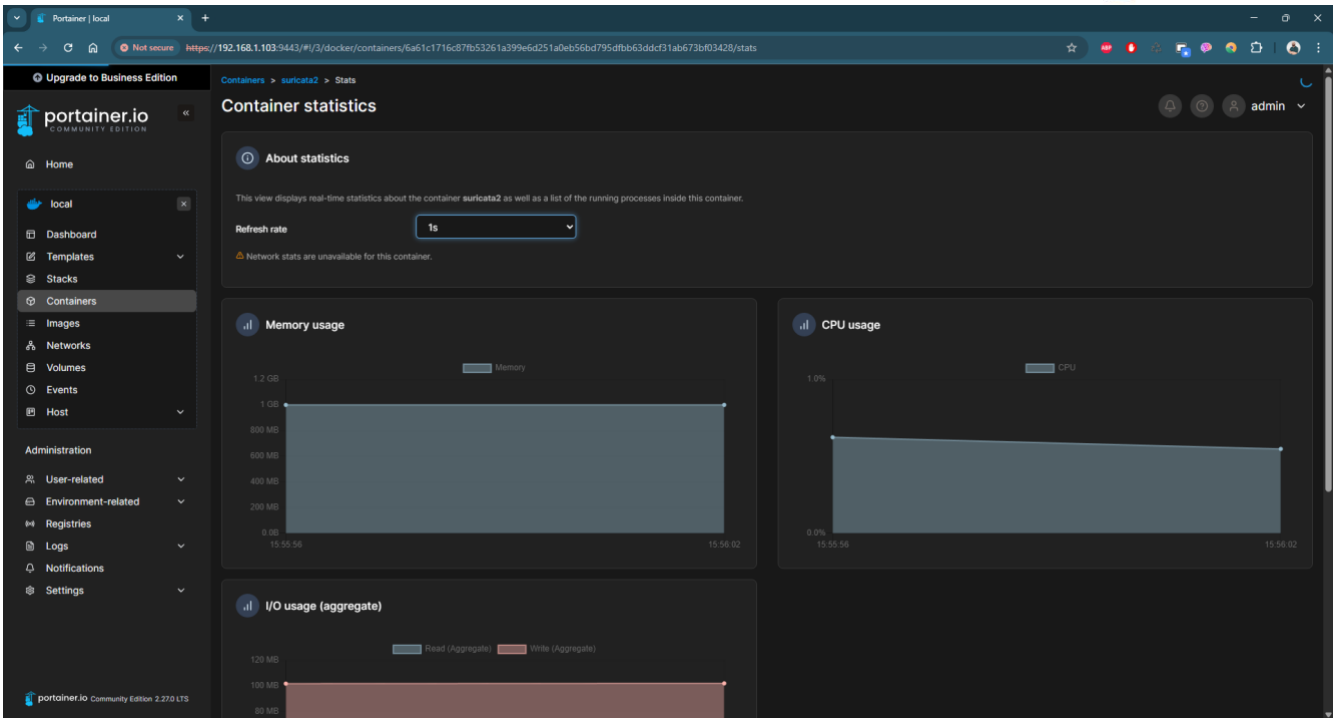

Figure 15: Docker container statistics for Suricata

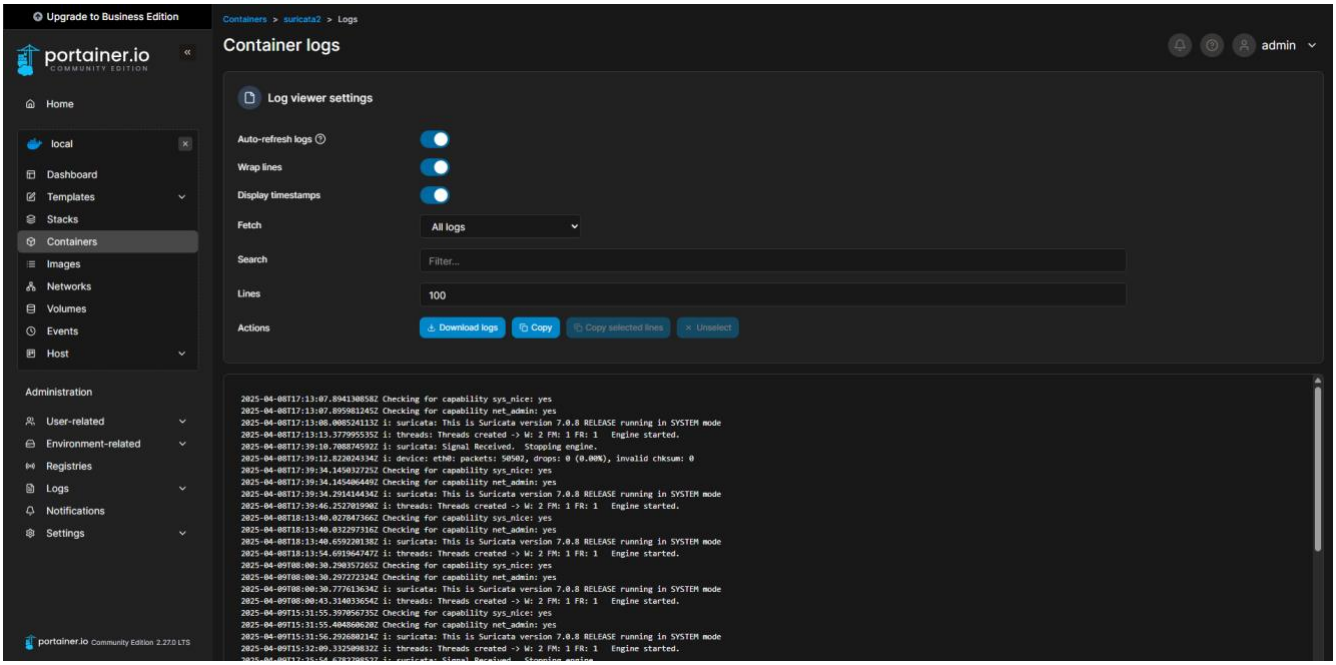

Figure 16: Container logs for Suricata

Table 13: Suricata API documentation

Endpoint	Description
`POST /metadata`	Upload a PCAP file to extract metadata such as alerts and protocol information.
`POST /full`	Upload a PCAP file to extract metadata and files, returning a comprehensive analysis.
`POST /test`	Test custom Suricata rules against a PCAP file to verify their effectiveness.
`POST /validate`	Validate the syntax and structure of custom Suricata rules before deployment.

Regarding rule configuration and management, Suricata offers the following:

- Rule Files:
  - Suricata uses .rules files containing signatures to detect various threats. These files can be customized to tailor detection capabilities.
- Rule Testing:
  - Before deploying new rules, use the /test endpoint to ensure they trigger as expected on sample traffic.
- Rule Validation:
  - The /validate endpoint checks the syntax of rules, preventing errors that could disrupt Suricata's operation.
- Integration with External Feeds:
  - Suricata can be configured to automatically update its rules from external threat intelligence feeds, keeping detection capabilities current.

© CYberSynchorny, 2024

Annex III: *Preliminary PoC and Prototypes (Anomaly Detector IDS)*

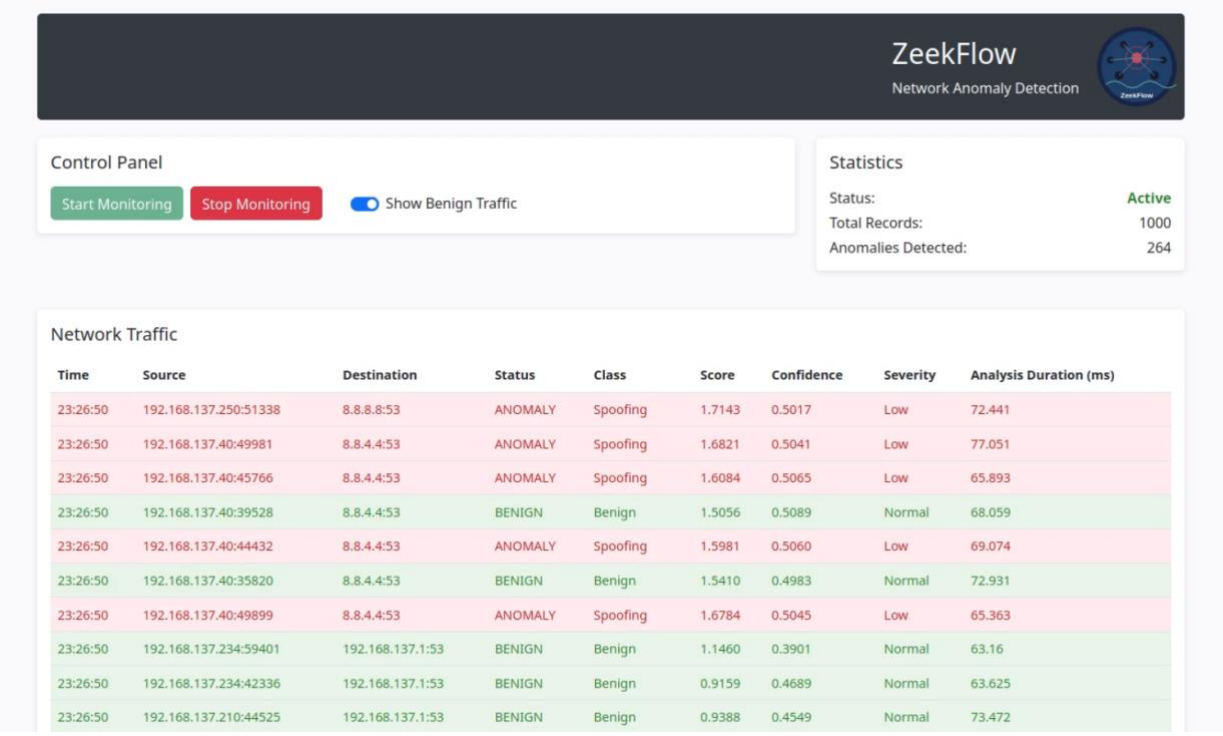

Figure 17: AI Anomaly detector IDS alert dashboard

The preliminary proof-of-concept of our Anomaly Detector IDS demonstrates its ability to monitor network traffic in real-time and identify suspicious patterns using our transformer-based detection approach. This working prototype shows how the system processes streaming network data and alerts security teams to potential threats.

Our dashboard provides a comprehensive view of network activity, with clear visual separation between normal traffic patterns and detected anomalies. The current implementation effectively highlights both benign connections and potential threats within a unified interface.

Step-by-Step Walkthrough

Starting the Detection System

To begin monitoring your network:

1. Open the ZeekFlow dashboard through your web browser
2. Click the "Start Monitoring" button in the control panel
3. The system activates Zeek for traffic monitoring on the configured interface
4. Statistics begin appearing on the dashboard as network traffic flows through

Reviewing Network Flow Data

The dashboard presents network traffic analysis in several ways:

1. A detailed table of network connections with color-coding based on risk level
2. Key statistics panels showing total flows, anomaly counts, and detection rates

Each connection entry includes:

- Source and destination IP addresses and ports
- Protocol information

© CYberSynchorny, 2024

- Assigned traffic class (e.g., Benign, DoS, Recon)
- Confidence score for the classification
- Anomaly score showing the degree of deviation from normal behavior
- Severity rating based on anomaly threshold

Investigating Detected Anomalies

When suspicious activity is detected:

1. Examine the severity rating and confidence score to prioritize investigation
2. Review all related flows from the same source to identify attack patterns

The detail view provides extensive information about each connection, including raw feature values, comparative metrics, and analysis timestamps, helping security teams understand why the activity was flagged.

Testing Detection Capabilities

The system's detection capabilities can be tested in a controlled environment:

1. Prepare test PCAP files containing various attack types like DDoS or reconnaissance
2. Replay these files through the system using the provided replay tools
3. Observe how the system identifies and classifies each attack type
4. Adjust detection threshold if needed to balance sensitivity versus false positives

Usage Scenarios

Scenario 1: SOC Analyst Daily Monitoring

Security Operations Center (SOC) analysts use the Anomaly Detector during routine monitoring to identify potential threats in real-time. The system runs continuously, helping analysts by:

- Flagging unusual network behavior that might indicate attacks in progress
- Identifying specific attack types like DDoS, port scanning, or data exfiltration
- Providing confidence scores to help prioritize investigation efforts
- Showing traffic trends to identify coordinated attacks affecting multiple systems

The real-time nature of the detector allows for immediate response to suspicious activity. Analysts can quickly investigate high-severity alerts while the system continues monitoring for additional threats.

Scenario 2: Incident Response Investigation

After a security incident has been reported, incident response teams can use the Anomaly Detector to:

1. Upload PCAP files from the affected time period for analysis
2. Identify when suspicious activity first began
3. Discover lateral movement or other related attacks that might have been missed
4. Generate timeline reports showing the progression of the attack

The system's ability to detect subtle anomalies often reveals attack components that traditional signature-based tools miss, providing a more complete picture of security incidents.

Scenario 3: Network Security Assessment

Security teams can leverage the Anomaly Detector to evaluate their organization's security posture by:

1. Running the system during controlled penetration testing
2. Assessing detection rates of simulated attacks
3. Fine-tuning detection thresholds based on performance
4. Identifying network segments with unusual baseline behavior that might indicate pre-existing compromise

© CYberSynchorny, 2024

This approach helps organizations understand their security blind spots and improve overall monitoring capabilities.

System Requirements and Deployment

The current prototype requires:

- Docker Engine 28.0.1 or newer for containerized deployment
- Network interface with access to traffic (via TAP, SPAN port, or in-line)
- Minimum 8GB RAM and 4 cores CPU (16GB RAM and 8+ cores recommended)
- Optional GPU support for faster inference on high-volume networks

The system is deployed as a set of Docker containers using docker-compose, making it easily portable across environments. The dashboard is accessible via web browser, with APIs available for integration with existing security tools and SIEM platforms.

© CYberSynchorny, 2024

Annex IV: *Preliminary PoC and Prototypes (WEKA)*

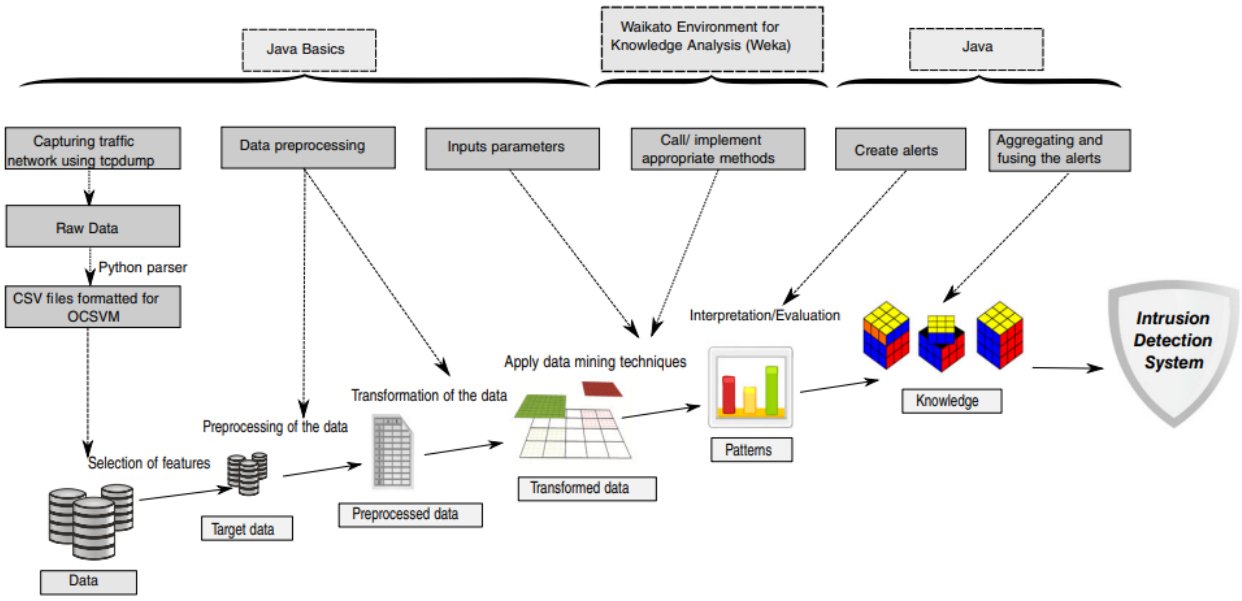

Figure 18: Weka log data analysis pipeline

Weka was utilized as part of an educational module designed to teach students how to build their own Intrusion Detection System (IDS)[8]. The module integrated tools such as **TCPdump**, **Java**, **Weka**, and **Python** to provide a comprehensive learning experience. As outlined in Figure 18, students began by capturing live network traffic using TCPdump. The generated traffic files were then converted into CSV format using Java, creating an initial dataset for feature selection.

- Feature extraction plays a critical role in classification problems. Attributes in raw datasets can be continuous, discrete, or symbolic, often varying greatly in scale and format. Since most classification algorithms cannot directly process such raw data, pre-processing becomes a crucial step. This typically involves two main stages: first, converting symbolic attributes to numeric values, and second, applying scaling techniques to ensure that attributes with larger numeric ranges do not dominate those with smaller ranges.
- During the pre-processing phase, students developed custom Java classes to either select relevant features or generate new ones. In subsequent labs, students were introduced to Weka's functionality by integrating its procedures directly within Java. They experimented with various classification algorithms, tuning parameters such as kernel type, gamma, number of outliers, and number of clusters depending on the specific method used.
- Further activities involved constructing multiple classifiers that could run in parallel on similar datasets, and combining their outputs using ensemble-based approaches such as majority voting. Additionally, students explored alternative fusion mechanisms based on clustering techniques like **K-means**, selecting methods that minimized communication overhead between nodes while retaining essential information for detecting attacks.

The reference testbed that was used to create the datasets simulated a Cyber-Physical System (CPS) that mimics real-world industrial operations. It integrates both operations and field networks, structured according to a layered architecture inspired by the IEC 62443 standard. At its core, the testbed models a liquid pump process, represented by an electric motor controlled through a Variable Frequency Drive (VFD), which enables multiple

[8] Maglaras, Leandros, et al. "Teaching the process of building an Intrusion Detection System using data from a small-scale SCADA testbed." Internet Technology Letters 3.1 (2020): e132.

© CYberSynchorny, 2024

rotor speeds. The VFD receives control signals from a Modicon M340 Programmable Logic Controller (PLC), which adjusts the motor speed based on predefined liquid temperature thresholds.

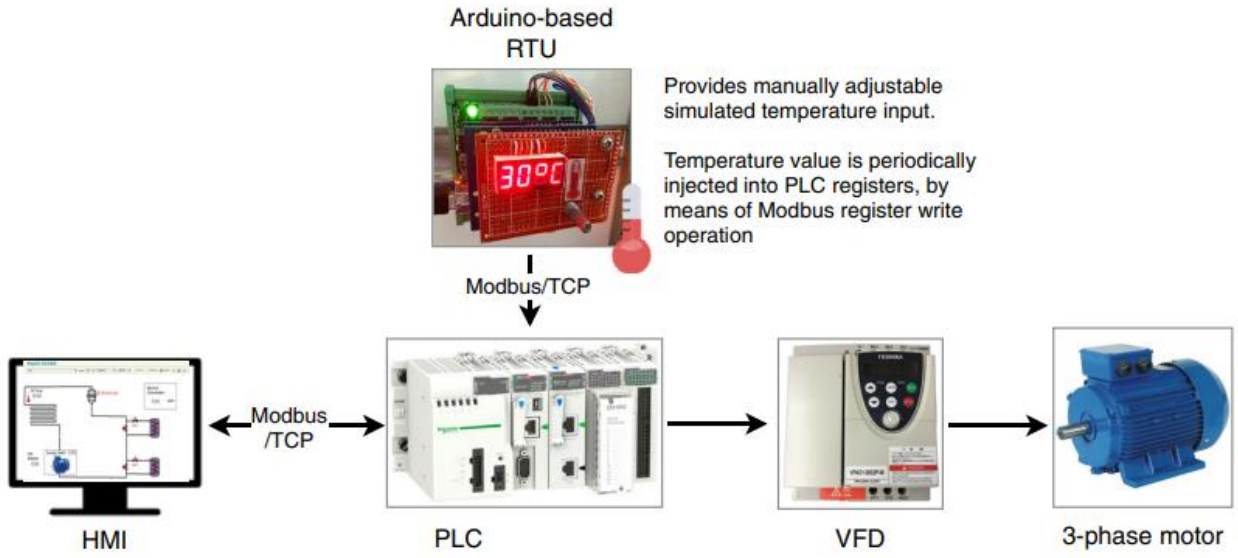

Figure 19: Cyber-physical process testbed

Temperature readings were simulated via a MODBUS Remote Terminal Unit (RTU) device, which is represented in the setup by a potentiometer connected to an Arduino microcontroller. This Arduino was connected to the network using an Ethernet expansion board with a Wiznet 5100 ASIC, simulating a typical field device setup. Horizontal communication occurs between the PLC and the RTU, offering students insight into the impact of device-to-device communication on system performance and behavior.

In addition, the PLC communicated vertically with a Human-Machine Interface (HMI), which acts as the supervisory control element of the system. Through this architecture, students gain hands-on experience with SCADA systems using the Modbus/TCP protocol and develop a practical understanding of how industrial communication layers interact within a security-focused framework.

To evaluate the efficiency of their Intrusion Detection Systems (IDS) in terms of both accuracy and communication overhead, each student is provided with labeled datasets containing a variety of attack scenarios extracted from the testbed. In the final week of the module, students are given the opportunity to test their trained IDS models against real-time attacks in a controlled, isolated lab environment. During this session, lab tutors simulate a range of network intrusions—specifically Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks—targeting random machines within the network.

These attacks are designed to escalate gradually, beginning with a low packet rate and intensifying to thousands of packets per second, closely mimicking real-world attack behavior. Using their IDS implementations, students must detect and respond to these live threats, refining their detection strategies in real time. The session is gamified, with students competing based on how quickly and accurately they can identify attacks, as well as the quality and clarity of the alerts generated by their systems. This practical, competitive exercise reinforces core concepts of intrusion detection and prepares students for handling dynamic cybersecurity threats.

© CYberSynchorny, 2024

Annex V: *Preliminary PoC and Prototypes (HUMAN RM (HRM))*

HRM was applied in a case study of an SME. An SME healthcare enterprise (HSME), operating across two separate facilities, and offering e-health services to its personnel and patients was the case. These services encompass, amongst others, e-diagnosis, e-prescriptions, and the handling of patients' sensitive data. Through this use case, all phases of the above HRM methodology were implemented[9]. The first step was asset cartography as illustrated in Figure 20 .

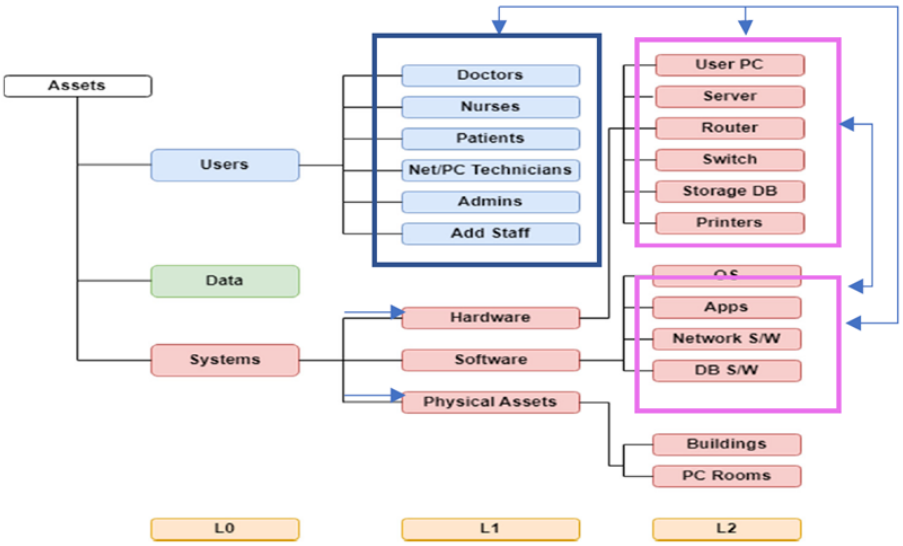

Figure 20: HSMEs users'/assets model

In the second phase, Risk Assessment strategies were implemented using the ENISA RM Toolbox. The toolbox was utilized to execute the strategies outlined in Phase B. As part of the initial steps, attack/risk scenarios were defined, and assets were identified from a technical perspective, which had been addressed in the previous phase. This phase also included threat mapping, utilizing the third library within the ENISA RM Toolbox.

To calculate the HRM score for the unauthorized-access-to-sensitive-patient-data scenario, we applied the following formula from the HRM methodology:

$$\mathbf{Risk = T \times V \times I \times AP \times 1/UP}$$

Where:

- **T** is the Threat (unauthorized access to patient data)
- **V** is the Vulnerability (potential for unauthorized access due to weak access control)
- **I** is the Impact (severity of unauthorized data access)
- **AP** is the Adversary Profile (e.g., motivated, skilled attackers with resources)
- **UP** is the ICT User Profile (doctors' compliance with security protocols)

For this scenario:

- The **AP** score takes into account the adversary's skills (7), motive (8), opportunity (6), and size (7).
- The **UP** score is based on doctors' compliance with security protocols, assumed to be 8 for this case.

[9] Kioskli, Kitty, Eleni Seralidou, and Nineta Polemi. "A Practical Human-Centric Risk Management (HRM) Methodology." *Electronics* 14.3 (2025): 486.

© CYberSynchorny, 2024

By multiplying these values, we calculate the HRM score, which reflects a high-risk level for unauthorized access. This score emphasizes the need for stronger access controls and more comprehensive security measures to protect sensitive patient data.

Finally, HSMEs must address risks by implementing the following measures:

1. **Technical Controls**: These include advanced access control systems, data encryption, and robust network and endpoint security.
2. **Administrative Controls**: This involves developing policies, managing access, providing employee training, and conducting regular security audits.
3. **Physical Controls**: These include access control systems, surveillance, alarms, and restricted-access storage.
4. **Social Controls**: This focuses on enhancing software and IT skills based on the personality traits, social factors, and technical skills identified earlier.

© CYberSynchorny, 2024