CISSAN

Collective intelligence supported by security aware nodes

D5.4 Collective intelligence algorithms for secure device communication and enhancing overall network security in runtime

Editor: Ilgin Safak, University of Jyväskylä and Jari Partanen, Bittium

Abstract

The report presents an in-depth analysis of collective intelligence solutions developed and implemented in the CISSAN, examining the potential of collective intelligence solutions in enhancing the detection, analysis, and response to cyber threats. It presents the collective intelligence mechanisms developed and used in CISSAN, including the optimal distribution of security functionalities for enabling collective intelligence, and cyber threat intelligence sharing using results from local data quality verification, distributed anomaly detection, and blockchain-based trust scoring using an extended version of the Structured Threat Information eXchange (STIX) threat report format. The report describes how the collective intelligence mechanisms are implemented and validated on the CISSAN platform and its use cases for facilitating the sharing of cyber threat intelligence, performing collective decision-making and enabling collaborative threat mitigation using automated disaster recovery and generative adversarial network based adaptive defence against complex and dynamic threats in a zero-trust environment. The report describes technical processes, the flow of data and information, the integration of system components, and the role of the technical drivers in enabling collective intelligence. The report covers security, privacy and compliance considerations and draws lessons from the validation of the collective intelligence methods in the CISSAN platform and its use cases, evaluating the effectiveness, advantages, and challenges of collective intelligence solutions in practical scenarios.

Project CISSAN

Public Report

January 2026

Participants in project CISSAN are (in alphabetic order with the project coordinator first):

- University of Jyväskylä (coordinator)
- Affärsverken Karlskrona AB
- Arctos Labs Scandinavia AB
- Blekinge Tekniska Högskolan
- Blue Science Park
- Bittium Biosignals Ltd
- Bittium Wireless Ltd
- Clavister AB
- Councilbox Ltd
- Geodata ZT GmbH
- Mattersoft
- Mint Security Ltd
- Netox Ltd
- Nodeon Ltd
- Scopesensor Ltd
- Savantic AB
- Techinova AB
- Wirepas Ltd

CISSAN-Collective intelligence supported by security aware nodes

D5.4 Collective intelligence algorithms for secure device communication and enhancing overall network security in runtime

Editor: Ilgin Safak, University of Jyväskylä and Jari Partanen, Bittium

Project coordinator: Ilgin Safak, University of Jyväskylä.

2026 CELTIC published project result

CELTIC-NEXT participants in project CISSAN

Executive Summary

The digital transformation of critical infrastructures has introduced unprecedented efficiencies while simultaneously exposing systems to new cybersecurity threats. As the Internet of Things (IoT) and Operation Technology (OT) devices, and digital communication platforms proliferate across networks, the attack surface grows, increasing the need for effective detection and response capabilities. However, limitations such as resource constraints make traditional security mechanisms a poor match for many IoT / OT deployments. One approach to addressing this challenge is the use of collective intelligence (CI), where users or devices combine their individual resources and observations to perform security tasks collaboratively rather than in isolation.

This report brings together interrelated research efforts on CI that aim to advance the security of IoT / OT networks at runtime and together propose methods for enhancing overall network security during operation by applying Structured Threat Information eXchange (STIX) for Cyber Threat Intelligence (CTI) sharing in CISSAN and CI based cyber threat hunting in industrial IoT (IIoT) networks. The objective is to enable standardized, machine-readable CTI sharing in CISSAN by representing locally detected anomalies, and the framework's trust-based decisions, in a format interoperable with external tools and stakeholders, and to demonstrate how CI can enhance runtime security in resource-constrained IIoT/OT networks by combining data quality verification, distributed anomaly detection, local threat hunting with peer validation and trust-based mitigation. The approach includes the deployment of a lightweight, cooperative, and distributed framework where each device performs local data quality verification, anomaly detection, trust scoring, exchanges anomaly reports with peers for consistency scoring, and feeds peer observations into centralized aggregation to compute global trust and support isolation decisions, with outputs designed for Security Information and Event Management (SIEM)/CTI integration. Threats are mitigated collaboratively using automated disaster recovery, blockchain-based device management, access control and generative adversarial network (GAN) based adaptive defence mechanisms in a zero-trust environment.

CISSAN adopts STIX 2.1 as the common CTI representation and prototype automated STIX bundle generation on the management server, triggered specifically by trust-score threshold violations (blacklisting events). CISSAN contributes to CI standardization efforts in WP5 by extending STIX, which aligns well with CISSAN's existing Java Script Object Notation (JSON)-based telemetry and supports graph-style linking of context and evidence. Native STIX objects do not model trust scores, motivating a dedicated Trust Score STIX domain objects (SDO) with properties for target, value, threshold status, method, and evidence references. Generating STIX per blacklisting event yields higher-signal CTI than per-anomaly or per-cycle exports, reducing noise while preserving actionable context. Incorporating the STIX threat report format delivers interoperable CTI packages that externalize CISSAN's collective-intelligence outcomes (especially trust-based compromise signals), improving cross-tool ingestion (SIEM/CTI platforms), auditability of trust decisions, and the ability to share high-value intelligence beyond consortium boundaries without overwhelming recipients with low-level telemetry. It provides a practical, scalable pathway for CI enabled threat hunting in IIoT/OT, where devices become both sensors and sentinels, detections propagate quickly across the network, and trust-aware aggregation enables coordinated response while keeping edge-side overhead low and preserving compatibility with SIEM/CTI workflows.

Key findings include the following. Effective monitoring can be achieved on production-grade industrial hardware without heavyweight analytics or centralized detection alone. Peer comparison enables cooperative threat hunting: convergence increases confidence, while divergence can indicate compromise, misconfiguration, or manipulation. Trust-based isolation (driven by aggregated peer evidence) serves as the primary mitigation when a device exhibits anomalous traits.

Together this interrelated research demonstrates an overall enhancement and option for enhancing overall network security during operations, where runtime defence can shift from centralized oversight toward distributed resilience, where field devices contribute directly to network-wide security awareness. Trust can function as a governance mechanism for device communication, enabling the network to degrade or revoke participation rights based on aggregated peer evidence. Additionally, security outcomes become portable and shareable when expressed as STIX, supporting cross-organization collaboration and reducing dependency on vendor-specific formats. Through the promotion of trusted collaboration, the reduction of dependence on third-party sources of cybersecurity intelligence, and the facilitation of a united, EU-led threat detection and response, the extended STIX threat report format proposed by CISSAN plays a direct role in enhancing the digital sovereignty of Europe.

List of Authors

In alphabetic order by partner name:

- Jari Partanen, Bittium
- Rodrigo Martinez, Councilbox
- Klaus Chmelina, Geodata
- Veikko Markkanen, University of Jyväskylä
- Mikko Lehkonen, University of Jyväskylä
- Xiaobang Sun, University of Jyväskylä
- Ilgin Safak, University of Jyväskylä
- Oliver Bölin, Techinova

Table of Contents

List of Figures and Tables

Abbreviations

AI	Artificial Intelligence
BKVS	Blockchain Key Value Store
CER	Critical Entities Resilience
CI	Collective Intelligence
CPU	Central Processing Unit
CTI	Cyber Threat Intelligence
DDoS	Distributed Denial of Service
DTW	Dynamic Time Warping
ECDSA	Elliptic Curve Digital Signature Algorithm
EMA	Exponential Moving Average
EU	European Union
FQDN	Fully Qualified Domain Name
GAN	Generative Adversarial Network
GDPR	General Data Protection Regulation
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICS	Industrial Control System
ID	Identifier
IIoT	Industrial Internet of Things
IoT	Internet of Things
IP	Internet Protocol
JSON	Java Script Object Notation
LLM	Large Language Model
ML	Machine Learning
MQTT	Message Queueing Telemetry Transport
MSE	Mean Square Error
NIS	Network and Information Systems
OT	Operational Technology
PoC	Proof of Concept
QoS	Quality of Service
REST	Representation State Transfer
RTU	Remote Terminal Units
SCADA	Supervisory Control and Data Acquisition
SCO	STIX Cyber-observable Objects
SDO	STIX Domain Object
SIEM	Security Information and Event Management
SRO	STIX Relationship Object

SSH	Secure Shell
STIX	Structured Threat Information eXpression
TAXII	Trusted Automated eXchange of Intelligence Information
TLS	Transport Layer Security
TTL	Time To Live
UC	Use Case
VLAN	Virtual Local Area Network
VPN	Virtual Private Network
WASM	WebAssembly Module

1 Introduction

This report offers the design, specifications, and analysis of algorithms that leverage collective intelligence (CI) to provide enhanced levels of security and resilience to interconnected Internet of Things (IoT) / Operational Technology (OT) devices and networks in run-time in the CISSAN platform, its use cases and solutions. This report outlines the conceptual framework and system assumptions in which these algorithms are applied, describing the operation and interactions among interconnected devices, monitoring elements, and intelligent agents that work together in a run-time environment to detect potential threats and make corresponding adjustments to network security policies in a timely and autonomous manner. This report also describes the application and utilization of the Structured Threat Information eXpression (STIX) threat intelligence exchange format in this work, explaining the extensions to the STIX threat report format that have been applied in this project and their implementation in this threat intelligence exchange platform to provide enhanced machine-readable and standardized threat exchange in this work. This report, therefore, offers a comprehensive framework that outlines the application and utilization of run-time collective intelligence and standardized threat intelligence in CISSAN.

1.1 Objective of this document

This report describes the algorithms and run-time mechanisms for CI with the goal of providing a secure communication mechanism for devices and improving the security of the network in the CISSAN platform and its use cases. This report describes the algorithms, run-time specifications, and information exchange procedures, including the utilization and extension of the STIX threat report format, in order to provide a technical guide for developers, integrators, and researchers on how CI is achieved in the CISSAN platform, how threats are expressed and transferred, and how security decisions are made at run-time, and at the same time, it forms the basis for testing, future improvements, and repurpose in other security domains in the cybersecurity domain.

1.2 Background and Motivation

Modern distributed systems, cyber-physical infrastructures, and IoT / OT networks are expanding rapidly in both scale and complexity. Traditional centralized anomaly detection approaches, which rely on a single point of control, are increasingly inadequate for these environments. High communication overhead, single points of failure, and limited scalability are significant drawbacks of centralized models.

Modern cyber threats tend to be organized, dynamic, and extremely rapid, requiring real-time detection and response capabilities that cannot be provided by an individual device or organization. Therefore, a pressing demand has arisen for CI methods that can facilitate shared threat intelligence, observation correlation, and dynamic response among distributed entities during runtime. However, issues related to interoperability, standardized threat expressions, and runtime enforcement infrastructure have hindered collective efforts. This report is driven by this context, addressing it by specifying CI algorithms and runtime security mechanisms for secure communication among devices, utilizing standardized and extended STIX-based threat intelligence, and facilitating dynamic network defense for complex cybersecurity scenarios.

2 Collective Intelligence Mechanisms in CISSAN

2.1 Definition and Principles of Collective Intelligence

In the context of IoT cybersecurity, CI refers to a security paradigm where multiple networked devices cooperate to produce better security awareness and stronger defensive decisions than any single device could achieve alone. Rather than treating endpoints as isolated sensors, CI enables nodes to share security-relevant signals, validate each other's observations, and combine partial perspectives into a network-level understanding that supports timely response during operations.

A practical way to introduce CI is through its core components, which function both as building blocks and as a roadmap for implementation. At the foundation there are three prerequisites. First, CI requires a source of intelligence. Devices must be able to observe or derive security-relevant signals. Cooperative defense cannot emerge if nodes remain entirely oblivious to security responsibilities. Second, CI depends on a communication overlay that allows security information to be exchanged among peers. In many industrial settings this may need to be a dedicated channel, as operational protocols can be constrained in message structure or tightly controlled in ways that limit the exchange of security metadata. Third, CI mechanisms should explicitly address feasibility under resource constraints, since Central Processing Unit (CPU), memory, storage, and bandwidth limitations often determine whether a security function can be sustained during normal operation.

Building on these prerequisites, CI requires mid-level coordination and communication logic that connects local observations to cooperative tasks. Nodes must be able to initiate CI interactions in a predictable format, receive and interpret CI-related messages, and return results in a form that can be aggregated. In practice, this commonly implies the use of structured, machine-readable representations and consistent conventions for describing observations and responses, while recognising that the exact message flow varies depending on the CI method and the operational environment.

Once these foundations are in place, CI can support several classes of runtime security functionality. It may enable distributed computing, where security tasks that are too heavy for a single device are partitioned across multiple participants and their outputs combined. It can provide load balancing, where security responsibilities shift between nodes when one device cannot perform them reliably due to operational load or temporary constraints. CI can also support information enrichment, where partial observations from multiple devices are combined into a richer and more actionable picture before being forwarded to centralized monitoring or higher-level analytics. Across these variants, the overarching objective is autonomous defence: enabling the network to share relevant security signals, develop shared awareness of threats, and support timely, evidence-based actions during runtime with minimal reliance on manual intervention and third parties.

In summary, CI for runtime security is best understood as a layered construct: local security awareness, a communication overlay, and coordination logic, enabling distributed validation, enrichment, and response that improves network security during operations.

2.2 CISSAN Collective Intelligence Mechanisms

The collective intelligence mechanisms used in the CISSAN platform, including those that could potentially be used post-project are explained this section.

2.2.1 Distribution of Security Sub-functions

The CISSAN project uses an optimization-based approach to optimize the distribution of security tasks in the network to primary and failover devices. The optimization process takes into consideration various factors, such as devices, dependencies, quality of service, and resilience, to name a few. The optimization solver uses this information to optimize the distribution of tasks, which is then executed by the CISSAN Orchestrator to achieve distributed security operations and collective intelligence.

The mathematical formulation for this optimization process is not included in this report due to current research and publication activities, although this has been validated through joint use case demonstration in the CISSAN platform. The security tasks are optimally distributed across IoT networks in the CISSAN platform to enable collective intelligence. The following technical details, such as optimization algorithms used, the optimization problem objective, how the optimization problem is used in distributing the security tasks on the CISSAN platform and how the data is coordinated and routed is explained.

The architectural principles, the optimization model, and the logic behind the distribution of Security Sub-functions across the CISSAN IoT network are described in detail in CISSAN deliverable report D2.3 (Section 3.6.5).

2.2.2 Data Quality Verification

The purpose of the Data Quality Verification module is to calculate the local believability scores from the geotechnical sensor data of nodes locally on IoT devices in the CISSAN platform, which are aggregated at the CISSAN Management server. The module also calculates an overall believability score to validate the results which are aggregated at the Security Information and Event Management (SIEM) server and the CISSAN Management server. The local believability scores generated are used as input for the *TrustScoring* module that calculates local trust scores for the devices. These trust scores are aggregated at the management server and used for calculating a global trust score for the local network as well as blacklisting of network devices.

To ensure that data quality verification metrics are meaningful and comparable within the context of the specific dataset, domain-specific preprocessing (e.g., normalization) and custom thresholding or model integration will be performed to account for differences in scale and semantics. Anomaly detection may be performed after a data quality issue is detected.

In the tunnel construction use case (use case 3 (UC3)), geotechnical sensor readings are manipulated (data tampering attack) to either generate false safety readings or critical readings that aren't true. A tunnel instability either goes unnoticed or unnecessary actions are provoked. Geotechnical sensor data is transmitted to the CISSAN platform, where data quality verification is performed locally on IoT devices (see Figure 1). Believability scores are transmitted to the CISSAN Management Server and are also aggregated at the SIEM server to identify security incidents related to anomalous data, i.e., data with low believability scores. Alerts related to such incidents from geotechnical sensors will be triggered by the SIEM server.

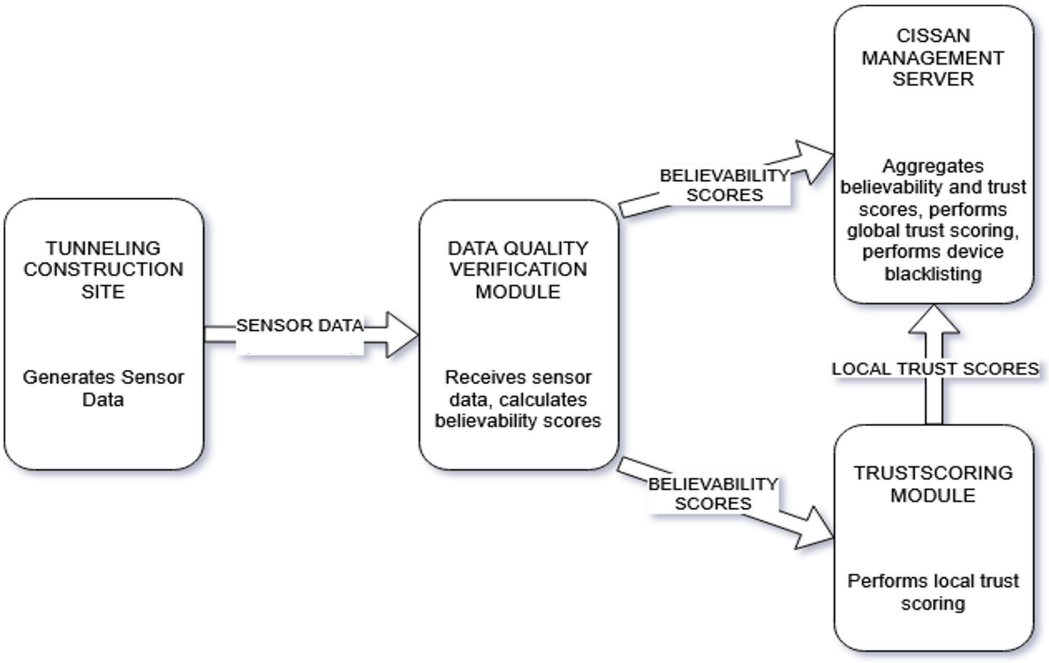

Figure 1. Incident detection at the CISSAN platform for UC3

For believability score calculation we use variance-based and similarity-based methods. The variance-based method is able to detect anomalies using local linear regression within a sliding window. The similarity-based method uses dynamic time warping (DTW) to measure similarity between two time series data from neighbouring sensors. A more detailed workflow of the methods is described below.

Variance-based method workflow:

The process involves moving a fixed-size window across the time series and performing a calculation at every step. By sliding a window across the data, we perform a "Local Linear Regression." This allows the model to adapt to changing trends.

Step 1: Parameters setting

- Window size
- Variance threshold

Step 2: Local Model Fitting

For each window ending at time t, fit a linear model using the time indices as the independent variable x and the observations of sensor as the dependent variable y:

$$\hat{y}_i = \beta_0 + \beta_1 x_i$$

Step 3: Calculate the Residuals

$$e_i = y_i - \hat{y}_i$$

Step 4: Variance-Based Scoring

To identify an anomaly, we look at the Standard Deviation of the Residuals σ_e. If a new data point falls significantly far from the predicted line—beyond the variance threshold—it is flagged.

Similarity-based method workflow:

To perform DTW on long time series, we compare a "current" window of data with a "reference" window of known normal behaviour. Unlike variance-based method, which focuses on trends and values, DTW focuses on whether the pattern of the data looks "wrong," even if it is stretched or delayed.

Step 1: Parameters setting

- Window size
- DTW distance threshold

Step 2: Compute DTW Distance

For each pair of neighbouring sensors:

- Extract their time series T_1 and T_2
- Compute the **DTW distance** between T_1 and T_2. This involves:
  - Creating an $n \times m$ cost matrix of pairwise distances.
  - Finding the optimal warping path with the minimum cumulative cost.

Step 3: Similarity Thresholding

- Set a threshold for DTW distance to consider two sensors “similar”.
- Sensors with small DTW distances are responding similarly (i.e., their signals follow similar patterns, even if not time-aligned).
- If DTW distances are large, anomaly detection will be performed on the data using the relevant anomaly detection model to identify device(s) displaying anomalous behaviour and impacted zones/segments in the network.

2.2.3 Machine Learning-Based Distributed Anomaly Detection

In the CISSAN platform, machine learning (ML)–based distributed anomaly detection is applied to identify suspicious deviations in Global Navigation Satellite System (GNSS)-derived positioning behaviour within the transportation use case. The primary goal is to detect patterns consistent with GNSS spoofing, jamming, or other interference by analysing fleet-wide positioning data streams.

The anomaly detection process is typically triggered after upstream data quality verification indicates a discrepancy or reduced believability, after which the Anomaly Detection module performs statistical and ML analysis on aggregated time series observations.

Observed GNSS time series are compared against reference series representing normal behaviour, and deviations are quantified as anomaly evidence. These anomaly detection outputs are converted into an anomaly risk score representing the severity and confidence of suspected anomalous behaviour. In this way, distributed anomaly detection provides a technical mechanism for transforming collective GNSS observations into actionable risk indicators that can drive trust-aware governance and collective intelligence decision-making in the CISSAN platform.

While this anomaly risk scoring pipeline has been validated in the CISSAN laboratory environment using GNSS data received via MQTT from the Mattersoft transportation system, it has not yet been integrated into operational public transport production systems. The primary rationale is that deployment into critical infrastructure requires additional validation of false positive behaviour, operational response procedures, and stakeholder acceptance, which are beyond the scope of the current project. However, the results indicate strong potential for post-project adoption as an external security intelligence layer for GNSS anomaly monitoring.

Anomaly Risk Generation and Sharing Flow (Transportation Use Case)

Public Transport Vehicles (GNSS)

|

| GNSSLocation (~1 Hz) + Runmonitoring (event-driven)

v

Mattersoft GNSS Data Integration

(Read-only, authenticated MQTT interface)

|

v

CISSAN Platform Ingestion Layer

|

v

Data Quality Verification

(Believability scoring, plausibility checks)

|

| if low believability / discrepancy detected

v

Distributed Anomaly Detection Module

- Statistics-based (sliding window regression, residual variance)

- Similarity-based (DTW time series comparison)

- ML-based modelling (platform-supported anomaly scoring)

|

v

Anomaly Risk Score

(severity + confidence)

|

v

Trust Scoring Module

(uses anomaly risk score as trust input)

|

v

Collective Intelligence Actions

(e.g., trust degradation, alerts, governance decisions)

2.2.4 Trust Scoring

Trust scoring is a cross-cutting mechanism used to support securing IoT networks across all CISSAN use cases. It provides a continuously updated indicator of whether a device should be considered reliable enough to participate in communication and collective security tasks. In practice, trust acts as a runtime eligibility metric: it turns distributed observations into a defensible rationale for a state change within the protected network (trusted --> untrusted) and enables coordinated action to isolate units when risk becomes unacceptable.

Trust scoring in CISSAN is performed both centrally and per-segment. Centrally, the responsibility lies in aggregating individual trust scores calculated by security aware nodes within the use-case segments of the CISSAN platform, into a comprehensive and actionable view of the security status of the network. Per-segment, nodes host applicable trust calculation methods, based on the available telemetry, typically from CISSAN solutions within their operational environment. An overview of the local-to-central trust-scoring flow is shown in Figure 2.

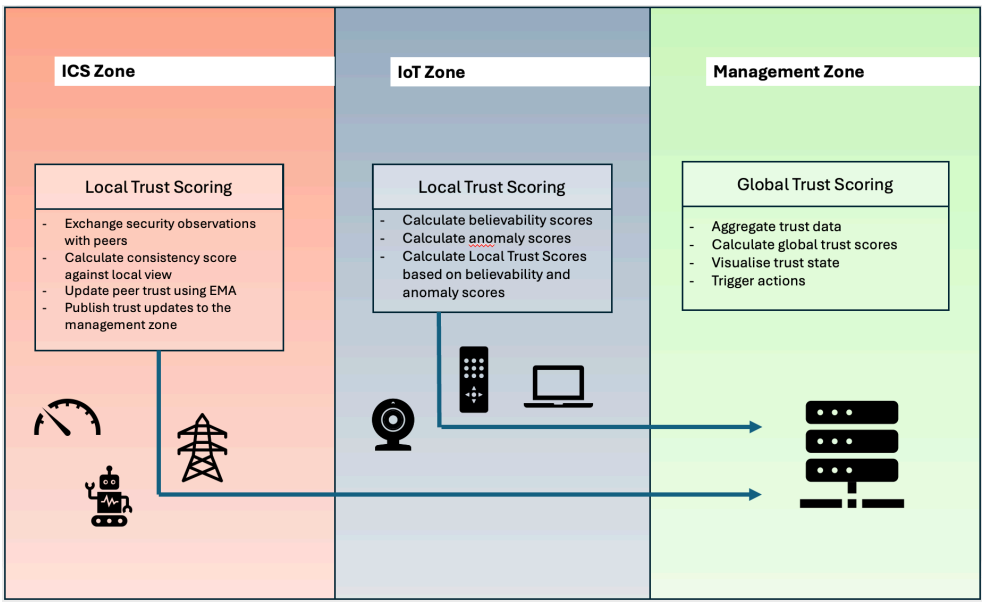

Figure 2. Trust Scoring Flow

Local trust scoring at IoT device level

Calculating local trust scores is handled by IoT devices of the CISSAN platform environment running the Liquid AI software solution. This solution consists of an orchestrator running on the orchestrator server and supervisors running on individual devices (Raspberry Pis) which execute WebAssembly modules (WASMs) distributed by the CISSAN Orchestrator. The distribution of the modules is based on the results of the optimization tool. The *TrustScoring* WASM was first implemented in Python and then translated to a WASM. WebAssembly was chosen since it allows the same module to be run on heterogenous hardware without modification due to its capability of executing code compiled from multiple programming languages in a sandboxed environment. The trust scoring module calculates a trust score based on existing trust data and observed responsiveness of peer devices.

The *TrustScoring* WASM takes as input a list of peer devices and existing trust related data, namely the anomaly and believability scores generated by the *AnomalyDetection* and the Data Quality Verification modules related to use cases one (transportation) and three (tunnelling and construction). Several formulas are employed to calculate the final trust score. For the score derived from ping data, the trust model developed in [1] is employed and adapted for CISSAN use where trust score η is defined by:

$$\eta(\beta) = \frac{\sum_i^n \phi(f, e)}{n}$$

where β is the feedback trace, a continuous consistency function ϕ is used to check the consistency of feedback f against evidence e with n denoting the total number of feedback. During the first stage of calculating local trust scores, the *TrustScoring* WASM initiates pings on all peer devices to gather information on their responsiveness. The ping data (min/max/average/standard deviation/number of received packages) is used to calculate the initial trust score of the target peer device with weight favouring packet loss over overall latency and standard deviation of the executed pings. In the second phase, the initial trust score generated from ping results is merged with the score calculated from the existing trust data.

This score is calculated with the formula presented in [2]:

$$EV_{S_i} = \frac{\sum_{j=1}^{SessionLength_i} w_j EV_{P_j}}{\sum_{j=1}^{SessionLength_i} w_j}, \ \forall i \in 1,2,\dots,M$$

which has been adapted for trust scoring use.

$$TS = \frac{\sum_{j=1}^{DataPoint_i} w_j IS}{\sum_{j=1}^{DataPoint_i} w_j}$$

where,

$$w_j = \frac{1}{Recency_j + 1}$$

and $Recency_j$ represents the recency value of the data point.

Each of the anomaly/believability scores has a weight applied to it, with more recent scores having a larger weight. The scores are summed together and divided by the sum of all the weights giving the final score derived from the trust data. By giving more weight to more recent scores, the current state of the devices is emphasized, ensuring that while historical data indicating good trust is not disregarded, recent scores will have a greater effect on the overall trust score of the device allowing anomalous behaviour to be detected faster. During the merging of the two scores (score derived from pings and score derived from trust data) more weight is placed on the second score (trust data score) emphasizing the importance of the core functions of the device as related to the use case. The result is a score between zero (0) and one (1), with 0 indicating the lowest (not trustworthy) and 1 indicating the highest level of trust (trustworthy).

The final trust scores for each peer device are then stored locally in JavaScript Object Notation (JSON) format. Information contained includes the device that executed the scoring, target device, local trust score, a flag for successful pings, a flag for successful calculation of the score derived

from trust data and time stamps in two different formats, both in seconds since the Unix epoch and in date format for human readability (see Figure 3).

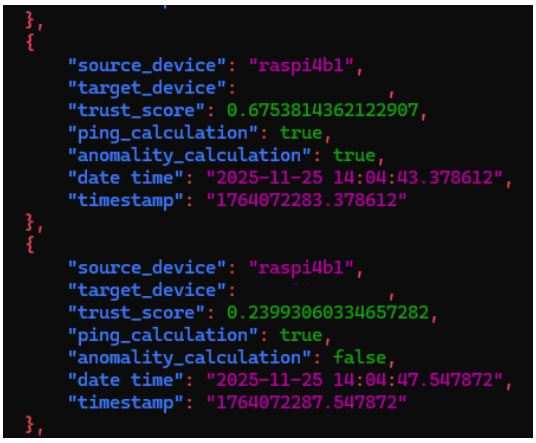

Figure 3. Trust Scoring Result

The CISSAN Management Server fetches the local trust scores from the devices via a Hypertext Transfer Protocol (HTTP) request. If the trust score is below an acceptable threshold, the device is blacklisted and potential disaster recovery for redistributing tasks among the peer devices is initiated. In addition, the CISSAN Management Server uses the local trust scores to calculate a global trust score for the network. This is done by using the trust model presented in [1]:

$$\mu(\alpha) = \frac{\sum_i^I \frac{\phi(f,e)}{n} \cdot w_i}{\sum_i^I w_i}$$

Furthermore, trust scores are stored in the Councilbox Eventchain System for traceability and consistency allowing for comparison of historical data and in the case of security events as well as auditing.

Centralized trust scoring and the global trust score

Trust within Use Case 2 (UC2) segment

The applied trust model within the UC2 segment is based on peer-derived consistency combined with central aggregation. Each participating device maintains trust assessments of its peers and updates them over time based on how well a peer's reported security state aligns with the local device's own observation snapshot. The core trust update follows an exponential moving average (EMA), which intentionally introduces asymmetric dynamics: trust is accumulated gradually under sustained consistency, while meaningful deviations can reduce trust more rapidly. This provides stability against transient noise while remaining responsive to persistent anomalies or manipulation. Thresholds applied to the aggregated trust view then define when a device is treated as untrusted and subject to containment actions such as blacklisting or isolation.

From a technical perspective, the trust update process follows a consistent runtime flow. First, an incoming peer report is received over the MQTT security communication overlay and parsed into a normalized structure containing a list of anomaly entries. Messages originating from the local unit are ignored to avoid self-reinforcement. After ingesting, the peer's report is evaluated against the recipients local understanding of security status, by executing the local monitoring cycle and extracting the locally observed anomaly set for the same moment in time (detailed explanation of the various anomaly classes can be found in D5.2).

Consistency is then computed as a ϕ (phi) score based on the overlap between the peer's reported anomalies and the locally observed anomalies. In the current model, anomalies are compared using their categorical identifiers (event type and subtype), and ϕ is calculated as the proportion of matching categories over the union of all categories observed by either side:

$$\phi = \frac{|\text{matches}|}{|\text{matches}| + \left|\text{only}_{\text{in}_{\text{incoming}}}\right| + \left|\text{only}_{\text{in}_{\text{local}}}\right|}$$

If neither side reports anomalies, or if the peer and local anomaly category sets are identical, ϕ evaluates to 1.0, reflecting full agreement. This comparison is intentionally lightweight and interpretable, which also means that repeated occurrences of the same anomaly category do not increase ϕ, since the calculation is based on category presence. A known limitation of any pairwise consistency measure is that two compromised nodes could, in principle, appear consistent with each other. In such a case, the situation is expected to be detected at the network level through collective evaluation: benign nodes would observe inconsistency relative to the compromised units, while a compromised peer would reveal itself by diverging from the broader peer consensus as trust updates accumulate.

The trust score, $TrustScore_{D_i,n}$, of a peer device, D_i, is updated using an EMA, where the latest value, $TrustScore_{D_i,n-1}$, contributes a controlled fraction of the new trust level:

$$TrustScore_{D_i,n} = (1 - \alpha)\, TrustScore_{D_i,n-1} + \alpha\, \phi$$

Here, α was taken as a fixed value of 0.2 in the simulations.

This gives the most recent ϕ an immediate 20% influence while retaining 80% of the accumulated history, ensuring that trust evolves smoothly and is less sensitive to transient deviations. Because the update is an EMA, older ϕ values decay exponentially over time, meaning recent behaviour always contributes more than distant behaviour, while the chosen α keeps the response relatively stable. The updated trust score, together with the latest ϕ and a timestamp, is stored in the local trust table and shared upstream as a trust update for aggregation at the management server. In the current demo configuration, a trust update is published whenever the trust value changes (i.e., any non-zero delta), ensuring the management layer receives a continuous stream of peer assessments for global trust computation. Figure 4 displays a local trust table structure from a participating Remote Terminal Unit (RTU):

```
{
  "rtuA": {
    "trust_score": 0.84,
    "last_phi": 0.8,
    "last_update": "2025-11-25T11:39:36Z"
  },
  "rtuB": {
    "trust_score": 0.42,
    "last_phi": 0.2,
    "last_update": "2025-11-26T07:52:18Z"
  }
}
```


Figure 4. Local trust table

Within the Industrial Control System (ICS) Zone, trust scoring is a core CI component because it translates peer consistency into runtime containment decisions. Agreement increases trust gradually, repeated disagreement decreases it, and aggregated trust provides a clear eligibility signal for blacklisting and isolation when risk becomes unacceptable. The mechanism is benefitting from additional benign participants but requires clustering and coordination to remain feasible under resource constraints.

2.2.5 Fully Decentralized Anomaly Detection Using Distributed Autoencoder Agents

CISSAN introduces a fully decentralized, agent-based anomaly detection framework, where each agent operates autonomously, detecting anomalies in its local environment. Collaboration between

agents occurs only, when necessary, such as when a local anomaly is suspected. This approach minimizes network load and avoids the need for central coordination, making the system more robust and scalable.

The system's primary goal is to enable self-organizing, peer-to-peer coordination for anomaly diagnosis. Agents communicate directly via HTTP, exchanging latent representations derived from local encoders. This design allows the network to collectively assess whether an anomaly is local (e.g., sensor noise or drift) or systemic (e.g., a global fault or coordinated attack).

The decentralized anomaly detection framework is built around autonomous agents that perform three core tasks: local anomaly monitoring, coordination signalling, and joint reconstruction. Each agent node continuously analyses streaming data using statistical and learned representations. Upon detecting a local outlier, an agent requests latent data from its peer agents. The network then combines these latent vectors to evaluate the anomaly in a shared latent space using an autoencoder-like model.

This architecture operates as a decentralized federation of intelligent nodes, each capable of both local inference and network-level reasoning. The system is designed to be robust, scalable, and capable of handling dynamic environments where anomalies can be either local or systemic.

Each agent in the system comprises several key components:

- **Agent Service**
  The agent service is a Flask-based microservice that manages local model inference, handles peer communication, and exposes endpoints for coordination and health checks. This service is responsible for running the local autoencoder, managing the agent's state, and coordinating with other agents when necessary.
- **Local Autoencoder**
  Each agent runs a lightweight autoencoder that transform its local data into a latent vector. A shared decoder, or a locally replicated copy, reconstructs the original input from all agents' latent vectors. This allows the system to evaluate anomalies in a shared latent space, providing a common ground for comparison and diagnosis.
- **Coordinator Routine**
  When an agent detects a local anomaly via a Z-score trigger, it enters a coordination phase. During this phase, the agent sends latent requests to all other agents, collects their encoded representations, and runs a joint decoding step to reconstruct its input. The agent then computes a reconstruction error (e.g., using χ^2 or mean square error (MSE)) to evaluate global consistency. A high reconstruction error indicates that the anomaly cannot be explained locally, implying a true systemic fault.
- **State Management**
  Each agent can be in one of three states: NORMAL, RESPONDING, or COORDINATING. In the NORMAL state, the agent runs local monitoring. In the RESPONDING state, the agent replies to peer requests. In the COORDINATING state, the agent executes a coordination round and pauses local checks. This state management ensures that agents can handle both local and network-level tasks efficiently.

Agents communicate using Representational State Transfer (REST) JSON messages via Flask. The communication protocol includes the following endpoints:

- POST /coordination: Used to request latent vectors from peer agents or signal the completion of a coordination round.
- POST /sync_time: This endpoint supports logical time synchronization across the agent network
- POST /shutdown: This endpoint triggers a graceful shutdown of the agent.
- GET /health: Returns the agent's status.
- GET /ready: The ready endpoint is used during startup and deployment to verify that an agent has completed initialization.

The protocol ensures minimal coupling and easy scalability. A new agent can join the network simply by running its agent service with a unique port and ID.

Methodology

Local Anomaly detection

Each agent periodically collects local samples. The samples consist of 10 metrics (frequency, voltages, currents) and extended to 27 features (statistical descriptions of the metrices). A Z-score-based detector computes the anomaly score for each sample. If the calculated value is larger than a given threshold a local anomaly is suspected, triggering the coordination phase.

Latent Space Reconstruction

Agents share their encoded latent vectors. The initiating agent collects all latent vectors, concatenates them, and reconstructs the input. The reconstruction vector is computed as the chi^2 of the difference between the initial vectors and the reconstructed ones.

A high reconstruction error indicates that even with information from other agents, the anomaly cannot be explained, implying a true systemic fault.

Synchronization and Locking

To avoid coordination loops, agents lock local monitoring when responding to peer requests. Once the coordination phase ends, signaled by "done" messages, agents resume normal operation after

a cooldown period. This synchronization mechanism ensures that the system remains stable and avoids unnecessary coordination overhead.

Model Architecture

The core of the anomaly detection system is a modular autoencoder design that enables efficient distributed operation. The architecture consists of three main components: a multi-input encoder with per-agent submodules, a shared latent fusion layer, and a decoder that reconstructs per-agent outputs.

Each agent in the network runs only its own sub-encoder and maintains a local copy of the global decoder. This design choice minimizes the computational overhead on individual agents while still enabling collective anomaly assessment. The encoders are implemented as small multi-layer perceptrons (MLPs), making them suitable for deployment on edge devices with limited computational resources.

The shared latent fusion layer combines the latent representations from all agents, creating a unified feature space for anomaly detection. This approach allows the system to leverage information from multiple agents while maintaining the decentralized nature of the architecture, see Figure 5.

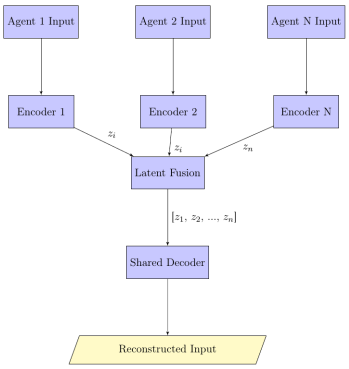

Figure 5. Modular autoencoder architecture with latent fusion

Training and Deployment

The training process for the encoder-decoder model can be conducted either centrally on aggregated data or in a distributed manner using federated learning techniques. While the current implementation uses centralized training for simplification, future work will explore federated learning extensions to enhance privacy and scalability.

Deployment is straightforward and scalable. Each agent is launched as a standalone process with a small configuration that specifies its identifier, listening port, and the expected number of peers. No centralized service discovery or orchestration is further required. Agents discover one another through preconfigured network addresses and immediately begin normal operation.

Because coordination is initiated only when anomalies are suspected, network traffic remains low during normal conditions. If an agent fails or is temporarily unreachable, coordination rounds proceed with the available peers, and the system continues operating without global interruption.

2.2.6 Cyber threat hunting in IIoT networks

Industrial Internet of Things (IIoT) environments increasingly face sophisticated cyber threats that bypass traditional perimeter-based security through compromised credentials and misuse of legitimate access. At the same time, the deployment of advanced security mechanisms on industrial controllers remains constrained by limited computational resources, strict real-time requirements, and operational safety considerations. These challenges create a gap between academic security research and deployable protection mechanisms for operational technology environments.

In CISSAN, effective and coordinated security monitoring is achieved in resource-constrained IIoT environments without relying on heavyweight analytics or centralized detection alone. Experimental

validation on production-grade industrial hardware demonstrates that distributed threat hunting and CI can provide timely, network-wide situational awareness while remaining operationally feasible. By combining local detection, peer-based validation, and trust-aware aggregation, this effort offers a practical and scalable approach to maintaining security posture and enabling proactive defence in industrial networks.

This section overviews Remote Terminal Monitor (Rotor), a lightweight and distributed security framework designed for Linux-based IIoT and OT networks. Within the scope of CISSAN deliverable D5.4, it highlights the frameworks components and functionality related to collective intelligence. The complete framework proposal and the Proof of Concept (PoC) work is part of a pending research paper.

Rotor

Rotor is a distributed security framework designed for Linux-based industrial IoT and OT devices. Its purpose is to enable resource-constrained controllers and embedded gateways to participate and cooperate directly in threat detection and collective defence. By shifting security responsibility closer to the operational process, Rotor reduces the reliance on centralized, outdated or third-party security solutions, and enables efficient detection and reaction to potential anomalies lurking within the system. This aligns with contemporary research that emphasize device-level visibility and cooperative security in industrial networks, while also overcoming the main limitations introduced by the typical resource constrained conditions within the embedded domain.

At a system level, Rotor integrates three complementary security capabilities that together form a distributed, CI based threat-hunting architecture:

Local Threat Hunting and Anomaly Detection: Each device runs an embedded Rotor monitor that inspects local system activity including logs, process, file integrity, network connections, and shell behaviour. This local perspective captures malicious traces that perimeter defences and access control mechanisms cannot detect. Rotor monitor produces the local intelligence, which serves as the primary input to the frameworks CI mechanisms. This section is further detailed in D5.2.

Peer-To-Peer Security Intelligence Exchange: CI is achieved when, participating devices are not treated as isolated sensors. Instead, each node publishes its anomaly reports and receives the reports of its peers. When a device receives a peer's report, it compares the anomalies against its local state and produces a consistency score that reflects how similarly the two devices behave under comparable operational conditions. This mechanism introduces a cooperative threat-hunting model:

- Convergence between nodes strengthens confidence in observed events
- Divergence may indicate compromise, misconfiguration, or manipulation

  The model enables emergency actions based on peer-shared and validated observations, to react to threats before they realize on the validating unit. It serves as the main CI component to provide input for enhancing security during runtime.

Network-level Trust Evaluation and Central Aggregation: When the peer-level consistency scoring indicates a device exhibiting anomalous traits, the primary mitigation is trusting based isolation. A management server aggregates anomaly reports and peer-generated trust deltas to compute global trust scores for each device. These scores determine whether a unit remains eligible to participate in communications or must be isolated due to suspected compromise. The server also provides a unified network-level view of ongoing security events and enables integration with SIEM platforms, CTI tooling, and intelligence sharing efforts. At this level, Rotor transforms local detections into a coordinated network-wide response capability.

Together these mechanisms form a distributed, cooperative threat-hunting ecosystem in which each device acts both as a local sensor and an intelligent security sentinel. Detections made by one unit reinforce the awareness of all others, and trust-based reasoning reduces dependency on any single devices view of the system. This architecture ensures that low-resource industrial devices can meaningfully contribute to situational awareness without requiring intrusive agents or heavyweight analytics.

Rotors design addresses several gaps in current research and practise. While existing literature recognises the importance of decentralized detection, collaborative defence, and device level threat

hunting, real world adoption is limited by hardware constraints and the absence of practical frameworks. Rotor contributes to the field by:

- Deploying sustainable hunt support anomaly detection directly on embedded industrial devices
- Implementing a peer-comparison model to leverage collective resources instead of limiting analysis to a single node
- Applying trust-scoring to identify compromised or deceptive nodes
- Producing structured outputs that support future CTI integration
- Advance a more system-oriented research direction where not only threats are detected, but also proactively shared among nodes, and action based on the consensus taken.

In doing so Rotor framework bridges the gap between conceptual models of collective intelligence in cyber defence and viable industrial deployment, offering a scalable pathway toward distributed threat hunting in IIoT and OT networks.

The framework has been applied to industry standard RTUs and completely integrated into the CISSAN platform. A PoC demo has also been conducted and validated within the environment. A more detailed description of the CI components along with the experiment are available in the CISSAN deliverable D6.3 report. A research paper related to the PoC work is pending.

2.2.7 GAN-Based Synthetic Data Augmentation for Collective Intelligence

Generative Adversarial Networks (GANs) offer a complementary mechanism for enhancing CI on the CISSAN platform by addressing the long-standing challenge of data scarcity in cybersecurity environments. In IoT and IIoT networks, labelled attack samples are inherently scarce and unevenly distributed across different devices and use cases, limiting the effectiveness of ML-based detection models. GANs alleviate this problem by generating realistic synthetic network traffic and sensor data, thereby expanding the training set available to the platform's distributed anomaly detection components.

In the CISSAN environment, GANs are trained on operational data collected at the platform level, learning the statistical distributions of normal and anomalous network behaviour. The generator component of the GAN generates synthetic samples indistinguishable from real-world observations, while the discriminator component continuously evaluates and improves the quality of these samples. This adversarial training process generates high-fidelity synthetic datasets that can be shared among participating nodes without exposing the original data, thus enriching the collective training corpus while adhering to the principle of data minimization.

GAN-based approaches contribute to collective intelligence in several ways. First, it supports model-level collaboration: nodes do not need to share raw data; instead, they can exchange synthetic datasets generated by GANs that capture key features of their local threat landscape. This allows peer nodes to train more powerful detection models, enabling generalization in heterogeneous environments. Second, GANs support proactive defence scenario generation. By conditioning the generator on specific attack types or anomaly patterns, the platform can generate diverse threat scenarios, stress-testing distributed detection mechanisms before real-world events occur. Third, adversarial examples generated by GANs can be used to evaluate the robustness of deployed anomaly detection models in the network, identifying weaknesses that cannot be found from historical data alone.

The Generative Adversarial Network (GAN) architecture is based on Conditional Generative Adversarial Networks (cGANs), in which the generator takes noise vectors and class labels (normal or anomalous) as input, enabling it to selectively generate specific traffic categories. The discriminator receives real samples and generated samples along with their labels, learning to distinguish between real and synthetic data. Training is performed on a centralized platform using aggregated, anonymized data, and the resulting generator model can be distributed to edge nodes for local synthetic data generation, thereby reducing the need to share duplicate data.

2.3 Threat Intelligence Sharing in CISSAN

In modern systems, cyber defence prevention means not only to withstand opportunistic attacks but also to anticipate and adapt to the evolving tactics of highly resourced, persistent adversaries. While traditional security controls aim to reduce the likelihood of compromise, they still primarily respond to incidents after they occur. To shift towards proactive defence, organizations increasingly rely on Cyber Threat Intelligence (CTI), the systematic collection and analysis of information about potential or ongoing attacks.

A key goal of CTI exchange is to ensure that once an attack technique or indicator is identified, it cannot be reused effectively. Sharing threat intelligence increases the cost of developing new attack vectors and acts as a deterrent to future intrusions, especially outside active conflict contexts. This collaborative defence model depends on the ability to share, receive, and interpret CTI efficiently and consistently across organizational and technological boundaries.

STIX is an open standard, developed by the MITRE Corporation, and maintained by OASIS. It provides a standardized and machine-readable format for representing and exchanging CTI. STIX defines a common data model and serialization syntax, typically expressed in JSON that allows diverse systems to describe cyber threats in a structured way. By supporting concepts such as threat actors, indicators, attack patterns, and relationships between them, STIX enables interoperability among security platforms and organizations. It is open source and freely available, making it particularly valuable for entities with limited resources seeking to participate in collective intelligence networks alongside larger institutions.

Within the CISSAN project, the focus lies in distributing security responsibilities across network participants to enhance overall resilience. CTI sharing represents a natural extension of this approach: it allows nodes to exchange local knowledge, transforming isolated observations into shared situational awareness. In this context, adopting STIX as a representation format aligns with CISSAN's vision of resource-efficient, proactive cybersecurity, empowering even constrained devices or organizations to contribute to, and benefit from, a broader intelligence ecosystem.

2.3.1 STIX

STIX works by organizing CTI into three complementary categories of objects. These are SDO, STIX cyber-observable objects (SCO), and STIX Relationship objects.

STIX Domain Object (SDO): SDOs represent the core concepts used to describe threats and adversary behaviour. STIX defines 18 SDO types derived from commonly represented concepts in CTI. The complete set of SDOs is displayed in Figure 6. Each object models a high-level aspect of a threat: for example, an Attack Pattern may describe a tactic associated with a MITRE Att&ck technique, while an Indicator may contain a pattern for detecting malicious activity in a network or host environment. SDOs provide the semantic backbone of a threat intelligence report, capturing the who, what and why of an observed event (see Figure 6).

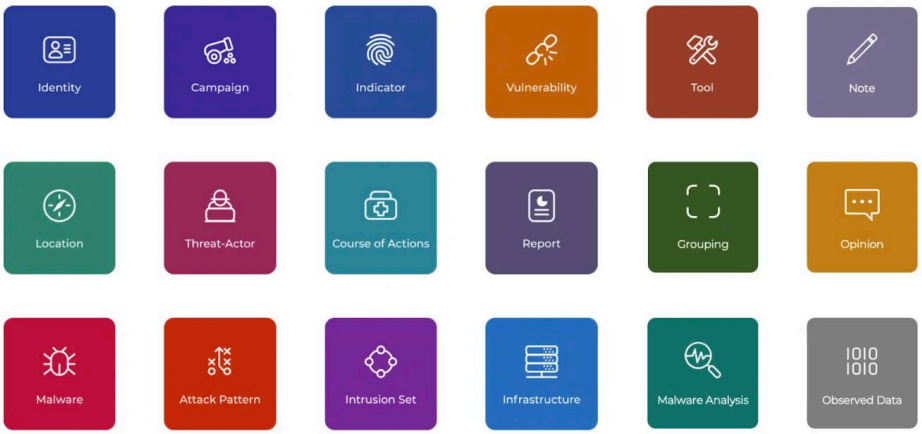

Figure 6. SDOs (retrieved from https://www.sekoia.io/en/glossary/stix/)

STIX Cyber-observable Objects (SCO): SCOs complement the high-level representation of the SDOs by representing the technical facts observed on systems or networks. SCO model elements such as files, process, network traffic, to answer the question of what precisely was observed at a given time. SCOs can be used independently or referenced by an SDO to link technical artefacts with the context in which they were seen. This separation between high-level intelligence objects and observables ensures the distinguishing between raw evidence and analyst interpretation. STIX defined SCOs are visible in Figure 7.

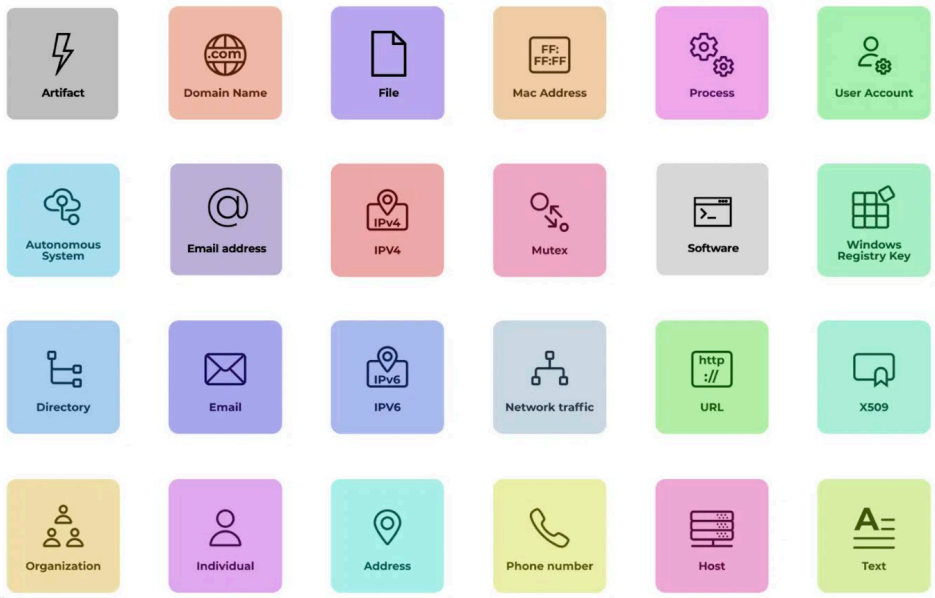

Figure 7. STIX SCOs (retrieved from https://www.sekoia.io/en/glossary/stix/)

STIX Relationship Objects (SROs): To provide meaningful CTI, the different instances of information must be connected. SROs serve this purpose, by defining how different SDOs and SCOs relate to one another. They allow STIX to express threat intelligence as a graph, where nodes correspond to objects and edges represent the semantic relationships between them. This graph-based structure enables more seamless automation, information enrichment and clearer analytics. STIX provides two types of SROs: Relationships - the generic SRO, used for most connections, and Sighting - which reports when a particular SDO has been observed in an environment.

Format: STIX content, as described above, is represented using a structured JSON serialization that encodes each object and preserves the graph-based relationships between them. This standardized serialization ensures interoperability across tools and platforms, enabling CTI to be exchanged consistently between heterogeneous systems. Within the CISSAN project, JSON is already used extensively in anomaly reporting and inter-node communication, making the adoption of the STIX serialization model a natural extension for supporting future CTI sharing.

2.3.2 Extending STIX

Within the CISSAN project, a major goal is to develop anomaly detection capabilities that leverage the collective intelligence of distributed network nodes. A central element of this approach is the sharing of locally detected anomalies so they can contribute to a wider, collaborative defence posture. While this information is exchanged on the network level, a natural extension is to share it with appropriate external channels to support broader defence efforts. To enable this, we prototype automatic generation of STIX-formatted threat intelligence packages within the platform. This ensures that information produced by different CISSAN components can be shared in a standardized, interoperable, and machine-readable way beyond the consortium boundaries if necessary.

However, CISSAN produce a type of information that is not natively represented in the current STIX object model: trust scores. Trust scores quantify the reliability of reporting nodes based on the consistency of their contribution to Collective Defence. In the CISSAN design, trust scores determine

whether a device's output should be considered credible, and whether the device should eventually be blacklisted from communication if its trust value falls below a defined threshold.

To support this functionality in a standardized way, we propose extending the STIX data model with a custom STIX Domain Object (SDO) to represent trust scores. Modelling it as a dedicated SDO allows trust information to be expressed explicitly, timestamped, linked to the device that is being assessed, and connected via STIX Relationship Objects to the anomalies that influenced the score.

In this model, a trust score becoming critically low is a meaningful security event: it signals that a node may be compromised, misconfigured, or intentionally deceptive. Representing this as a first-class SDO provides analysts with clear, standardized evidence that a device should no longer be trusted, while also enabling machine-level reasoning about trust dynamics across the network. This extension ensures that the collective-intelligence mechanisms integrate cleanly into the broader cybersecurity ecosystem supported by STIX. The proposed Trust Score SDO properties are displayed in Table 1.

Table 1. Trust Score SDO properties

Property Category	**Properties**
Required Common Properties	type, spec_version, id, created, modified
Optional Common Properties	created_by_ref, revoked, labels, confidence, external_references, object_marking_refs, granular_markings, lang
Not Applicable Common Properties	hashes, extensions, first_observed, last_observed, count, defanged
Trust Score Specific Properties	target_ref, score_value, score_threshold, is_below_threshold, calculation_method, basis_refs, explanation

Table 1 outlines the related Common Properties specified by STIX alongside the custom properties introduced for the Trust Score SDO. The custom properties capture how global trust assessments are generated within the CISSAN framework and provide the context needed to interpret the resulting trust value. Each custom property is briefly explained below:

target_ref: A reference to the STIX *Identity* object representing the device whose trust score is being evaluated. This establishes the association between the trust assessment and the specific operational unit within the network. The Trust Score SDO therefore functions as an event-level statement concerning the security reliability of the referenced device.

score_value: A numerical value representing the global trust score assigned to the device. The score is derived from the aggregation of peer-reported trust evaluations and reflects the network's collective assessment of the device's behaviour and reporting consistency. A lower value indicates reduced confidence in the device's security posture or integrity.

score_threshold: The minimum acceptable trust score defined for operational safety. If the calculated score falls below this threshold, the device is considered untrustworthy or potentially compromised. Including this value allows automated systems and analysts to contextualise the trust score relative to predefined operational policies.

is_below_threshold: A Boolean indicator signifying whether the current trust score violates the configured threshold. This provides a machine-actionable representation of trust degradation,

enabling automated response mechanisms (e.g., quarantining a device, halting communication, or alerting operators) without further interpretation of the numeric score.

calculation_method: A descriptor identifying the algorithm or procedure used to compute the global trust score on the management server. While the present implementation employs a consistent aggregation method (e.g., a mean of peer evaluations), explicitly including this field supports transparency, auditability, and future extensibility should alternative calculation models be adopted.

explanation: A human-readable rationale describing the circumstances that led to the trust score falling below threshold. This may include references to declining peer evaluations, inconsistencies detected in anomaly reporting, or other behavioural indicators that contributed to the score reduction. The explanation assists analysts in understanding why the device's trustworthiness changed, complementing the numerical assessment with contextual insight.

basis_refs: An optional list of STIX object references identifying the specific peer evaluations, anomaly reports, or other contributing objects that served as inputs to the global trust score calculation during the defined correlation window. This property enhances traceability by linking the trust decision to its underlying evidence, enabling detailed auditability and supporting forensic analysis when trust degradation occurs.

In addition to the custom Trust Score SDO, the generated report must include the underlying evidence that led to the trust decision. This is represented using STIX Observed Data objects, which are designed to convey raw observations without asserting their meaning. In the applied solution, each alert relevant to the blacklisting decision is parsed into an Observed Data SDO, and the original event content is embedded as a cyber-observable object. Because the considered CISSAN events are not covered by the standard STIX SCO vocabulary, they are expressed as a project-specific custom SCO using the x-rotor-event type (with the x- prefix indicating an extension). The Trust Score SDO then references the supporting observations via basis_refs, which enables traceability from the high-level trust decision to the specific observed evidence.

The resulting payload therefore follows a two-layer structure: a trust-decision object (x-cissan-trust-score) linked to a set of Observed Data objects containing the raw detections. This provides a standards-aligned way to package and exchange trust-related security events, while preserving the original evidence needed for auditability and downstream correlation in CTI workflows.

The following structure represents the complete STIX payload structure generated in UC2 context, including the bundle, Trust Score SDO, Observed Data SDO, and an example evidence SCO:

```
{
  "type": "bundle",
  "id": "bundle--<UUID>",
  "spec_version": "2.1",
  "objects": [
    {
      "type": "identity",
      "spec_version": "2.1",
      "id": "identity--<RTU_UUID>",
      "created": "<ISO 8601 UTC timestamp>",
      "modified": "<ISO 8601 UTC timestamp>",
      "name": "<rtu_id>",
      "identity_class": "device"
    },
    {
      "type": "observed-data",
      "spec_version": "2.1",
      "id": "observed-data--<UUID>",
      "created": "<ISO 8601 UTC timestamp>",
      "modified": "<ISO 8601 UTC timestamp>",
      "first_observed": "<ISO 8601 UTC timestamp>",
      "last_observed": "<ISO 8601 UTC timestamp>",
      "number_observed": 1,
      "objects": {
        "0": {
          "type": "x-rotor-event",
          "event_type": "<string>",
          "subtype": "<string>",
          "source": "<string>",
          "source_rtu": "<string>",
          "severity": <integer>,
          "timestamp_unix": <integer>,
          "message": "<string>",
          "metadata": { "<key>": "<value>" }
        }
      }
    },
    {
      "type": "x-cissan-trust-score",
      "spec_version": "2.1",
      "id": "x-cissan-trust-score--<UUID>",
      "created": "<ISO 8601 UTC timestamp>",
      "modified": "<ISO 8601 UTC timestamp>",
      "target_ref": "identity--<RTU_UUID>",
      "score_value": <float>,
      "score_threshold": <float>,
      "is_below_threshold": <boolean>,
      "calculation_method": "<string>",
      "basis_refs": ["observed-data--<UUID>"],
      "explanation": "<string>"
    }
  ]
}
```

This structure provides a standards-aligned representation of trust decisions within the CISSAN framework, enabling seamless integration with existing CTI workflows and ensuring that trust-related security events can be shared, analysed, and correlated alongside other STIX objects.

Although the previously described approach was applied in the CISSAN platform, another solution was also considered for UC1 context. This effort is outlined in the deliverable CISSAN deliverable D7.2 report.

2.4 Collaborative Decision-Making and Consensus Mechanisms in CISSAN

2.4.1 Blockchain-based device and trust management

Eventchain, the blockchain system developed by Councilbox for the CISSAN platform, manages device status through transactions validated by Byzantine Fault Tolerant (BFT) consensus. Every device registration, trust score change, and blacklisting decision must be independently validated by a majority of network nodes before it becomes part of the permanent record. This produces a tamper-evident, auditable history of the complete device lifecycle.

Consensus mechanism for device status

Eventchain employs a 51% BFT consensus protocol in which a designated master node creates blocks from collected transactions, and all hub nodes independently validate them before the block is accepted. The process operates as follows:

1. The master node assembles pending transactions into Merkle trees, constructs a block containing one or more trees, and signs the block with its Elliptic Curve Digital Signature Algorithm (ECDSA) secp256k1 private key.
2. The signed block is broadcast to all hub nodes via a RabbitMQ fanout exchange through isolated per-hub queues.
3. Each hub node independently validates the block by verifying: (a) the master's ECDSA signature, (b) the hash chain continuity with the previous block, (c) the correctness of all Merkle roots, and (d) the consistency of anomaly detection scores embedded in the transactions.
4. Upon successful validation, each hub sends a signed validation vote back to the master node.
5. The master determines the identity of each validating hub cryptographically from the vote signature rather than from any claimed identifier. This zero-trust approach ensures that a compromised node cannot impersonate another hub.
6. Consensus is achieved when the master receives majority of unique validations. For example, in a three-node network, at least two validations are required.
7. If the quorum is not reached, the block is rejected. There is no degraded mode: the system stalls until quorum is restored and then resumes automatically, ensuring that no unvalidated data enters the chain.

Device registration

The CISSAN Management Server registers devices by sending a registered event through the Metadata API with metadata such as manufacturer, model, firmware version, hardware identifier, and capabilities. The API signs this as a Blockchain Key Value Store (BKVS) transaction and submits it to the consensus pipeline. Once validated by hub nodes, all nodes hold an identical record of the registration.

Trust score management

The CISSAN Management Server sends updated_trust_score events whenever a device's trust score changes. Each event includes attributes such as numerical score, reason, confidence value, and contributing factors (communication_pattern, data_integrity, timing_consistency, etc.). Each recorded change receives a unique transaction hash that enables independent verification. The complete trust score history for any device is queryable from the blockchain.

Blacklisting

When a device's trust score drops below a defined threshold, the CISSAN Management Server sends a blacklisted event with the reason, final trust score, breached threshold, triggering anomalies, and isolation duration. The same consensus process applies, so all nodes hold an identical record of when and why each device was blacklisted. Because the blockchain record is independent of the blacklister service state, the governance history persists regardless of service restarts.

In addition to recording device governance events from the CISSAN Management Server, the Councilbox Eventchain system performs internal anomaly detection on IoT events ingested via HTTP / MQTT. This mechanism operates as a consensus-level operation: hub nodes independently compute anomaly scores and verify them during block validation. It is not connected to the CISSAN Management Server's trust scoring pipeline. The anomaly detection mechanism is described in detail in CISSAN deliverable D5.3 report.

2.4.2 Distributed Autoencoder Agents

Overview: Decentralized Autoencoder Agents

The system uses a multi-agent architecture where each agent runs a local autoencoder and collaborates with peers to detect anomalies in distributed time-series data (e.g., power grid measurements). The key innovation is the consensus protocol that allows agents to:

- Locally detect anomalies using autoencoders.
- Collaboratively validate anomalies by sharing latent representations.
- Achieve consensus on global anomalies via distributed coordination.

Agent Architecture and Local Anomaly Detection

Local Autoencoder

- Each agent loads a pre-trained autoencoder (encoder + decoder) and a subset of the data.
- The autoencoder compresses input data into a latent space, then reconstructs it. High reconstruction error indicates a local anomaly.
- Local anomaly detection:
  - Agents monitor their data stream, compute reconstruction errors, and calculate a Z-score (standard deviations from the mean).
  - If the Z-score exceeds Z_THRESHOLD, the agent triggers a coordination phase.

State Management

- Agents operate in states: NORMAL, RESPONDING, COORDINATING, COOLDOWN.
- State transitions ensure agents don’t overload the network or conflict during coordination.

Collaborative Decision-Making: The Coordination Protocol

When an agent detects a local anomaly, it initiates a **collaborative consensus process**:

Latent Space Sharing

- The initiating agent (coordinator) broadcasts its current data index to peers.
- All agents (including the coordinator) compute their latent representations for the same timestamp.
- Agents share their latents with the coordinator, which aggregates them into a global latent vector.

Global Anomaly Detection

- The coordinator uses the shared decoder to reconstruct the global data from the aggregated latents.
- It computes metric-wise reconstruction errors (e.g., for frequency, current, voltage) and normalizes them using pre-fitted RobustScalers.
- If any normalized error exceeds GLOBAL_ERROR_THRESHOLD, a global anomaly is flagged.

Consensus Mechanism

- Consecutive Confirmation: To avoid false positives, the system requires N_CONSECUTIVE (e.g., 3) consecutive coordination events to confirm a global anomaly.
- Cooldown Period: After coordination, agents enter a COOLDOWN_STATE to prevent network flooding.

2.4.3 GAN-Based Collaborative Decision Making

GANs contribute to collaborative decision-making on the CISSAN platform by enabling distributed verification and enhanced threat intelligence. In the context of CI, the challenge lies not only in local anomaly detection but also in achieving reliable consensus among heterogeneous network participants regarding the nature and severity of threats. GANs support this process by enabling nodes to generate and share synthetic threat representations, thereby facilitating consensus without exchanging sensitive operational data.

The discriminator component of a GAN plays a particularly crucial role in collaborative decision-making. Trained on a corpus of representative legitimate network behaviours and known attack patterns, the discriminator serves as a quality gate for incoming threat reports. When a node receives anomaly reports or trust updates from peers, the discriminator can evaluate whether the reported patterns are consistent with the learned threat distribution, providing an independent, data-driven second opinion that complements the consensus-based trust scoring mechanism already deployed in the CISSAN platform. This mechanism helps identify false or tampered reports, thereby enhancing the integrity of the collaborative decision-making process.

Furthermore, GANs support hypothesis analysis for collective decision-making. By generating synthetic attack scenarios based on the current network state, GANs can simulate potential threat evolution paths. These simulated scenarios can be shared among decision-making nodes for evaluation before implementing mitigation strategies, enabling the network to collectively assess the expected impact of isolation, redirection, or reconfiguration decisions. This forward-looking capability complements existing trust-based passive mechanisms by adding a predictive dimension to collaborative governance.

2.5 Run-Time Security and Adaptive Defence in CISSAN

2.5.1 Rotor

In Rotor, CI is treated as a practical means to strengthen runtime security in IIoT and OT networks by combining local device observations with peer-to-peer exchange and collaborative decision-making. As reported in Section 2.2.2, this is put to practice via the Network level trust evaluation and updating mechanism. When the peer-level consistency scoring indicates a device exhibiting anomalous traits, the primary runtime mitigation is trust based isolation. The CISSAN Management Server aggregates anomaly reports and peer-generated trust deltas to compute global trust scores for each device. These scores determine whether a unit remains eligible to participate in communications or must be isolated due to suspected compromise. The server also provides a unified network-level view of ongoing security events and enables integration with SIEM platforms, CTI tooling, and intelligence sharing efforts. At this level, Rotor transforms local detections into a coordinated network-wide agreement on node status, and network level response capability. Full details are not included in this report due to ongoing publication activities, but the approach has been validated within the CISSAN platform. Please refer to Section 2.2.6. and CISSAN deliverable D6.3 report for details.

2.5.2 Fully Decentralized Anomaly Detection Using Distributed Autoencoder Agents

The proposed decentralized, agent-based anomaly detection mechanism can be directly leveraged as a network security defence system. Each agent continuously monitors its local data stream and performs lightweight anomaly detection, functioning as a host-based intrusion detector. When a local anomaly is suspected, the agent initiates a coordination phase in which peers exchange compact latent representations rather than raw data, preserving privacy and minimizing information leakage. A shared decoder enables each agent to independently reconstruct the global system state and assess whether the anomaly is local or indicative of a coordinated attack. This collective verification step may allow the system to detect false data injection and coordinated multi-node attacks, without relying on a central authority or explicit consensus protocols. The architecture eliminates single

points of failure, degrades gracefully under partial outages, and scales naturally with network size. Overall, the system behaves as a distributed immune system, transforming local suspicions into collective intelligence for resilient, privacy-preserving cyber defence.

2.5.3 Automated Disaster Recovery

Automated disaster recovery in CISSAN builds on the task-allocation and orchestration logic defined in the platform architecture. The CISSAN Orchestrator continuously monitors device health, task execution status and communication availability. Because the system aggregates device state, trust scores and anomaly reports from multiple peers, disaster recovery actions are informed by network-wide collective intelligence rather than isolated observations. When a device becomes unreachable or fails to meet its operational thresholds, the CISSAN Orchestrator triggers failover based on the pre-computed assignments produced by the Arctos Labs Optimization Solver.

The failover devices are selected during the optimization phase to satisfy capacity, latency and disaster-recovery constraints. At run-time, the CISSAN Orchestrator redeploys the affected security tasks to these designated devices and restores task execution without requiring manual intervention. Once the previously failed device returns to a stable state, tasks can be moved back if this improves overall performance or risk posture. The architectural workflow and failover logic are described in Section 3.6.5 of the CISSAN deliverable D2.3 report, and the disaster recovery flow is described in Section 6 of the CISSAN deliverable D5.5 report.

2.5.4 Distributed Blacklisting and Device Isolation

The blacklister service runs in the CISSAN lab as a small Flask application. It keeps no persistent database; it applies isolation by running Secure Shell (SSH) commands against a MikroTik router, which manages bridge ports and Virtual Local Area Network (VLAN) membership for a fixed set of allowed switch ports. The service exposes a REST API used by the CISSAN Management Server: GET /status/<port> to read a port's current VLAN, POST /block/<port> to move the port to a quarantine VLAN, and POST /unblock/<port> or POST /unblock/<port>/<vlan> to restore the port to its previous or a specified VLAN. Only ports in the configured allowlist are accepted; blocking and unblocking are implemented by invoking RouterOS scripts on the router over SSH.

The CISSAN Management Server records the blacklisting events on the Eventchain blockchain via the Metadata API, capturing the final trust score, triggering anomalies, and isolation duration. Because the blockchain record is independent of the blacklister service state, the governance history persists even after service restarts. This combination of real-time network enforcement and permanent record-keeping provides both operational effectiveness and long-term auditability.

2.5.5 GAN-Based Adaptive Defence

In terms of runtime security and adaptive defence, GANs contribute to the CISSAN platform by enabling continuous model reinforcement and proactive threat prediction. Traditional anomaly detection models, trained on historical data, may become less effective as attackers' techniques evolve. GANs overcome this limitation by continuously generating new attack patterns as adversarial training partners, forcing detection models to remain robust to evolving threats.

The adaptive defence mechanism operates internally within the platform in a red-blue team cyclical manner. The GAN generator's task is to generate increasingly sophisticated synthetic attack samples that attempt to bypass existing anomaly detection models deployed across CISSAN nodes. When the generator successfully generates samples that bypass detection, they are incorporated into a retrained corpus, thereby improving the detection margin. This iterative process ensures that the detection model can co-evolve with potential attack strategies, providing an automated adversarial training method that enhances the network's overall defence capabilities.

GANs support adaptive defence through three complementary mechanisms. First, GAN supports attack simulation: by generating realistic attack traffic based on the current network topology and device characteristics, GAN provides a testing environment for evaluating the readiness of distributed detection components. Second, GAN supports anomaly threshold calibration: the critical samples generated by the GAN (i.e., those approaching the decision boundary between normal and

anomalous behaviour) can help operators and automated systems fine-tune detection thresholds, achieving a balance between sensitivity and the false-positive rate. Third, GAN facilitates resilient testing of the trust scoring pipeline by generating synthetic peer reports with varying levels of consistency, enabling the platform to verify whether the trust aggregation mechanism correctly identifies and responds to tampered or unreliable inputs.

This approach aligns with CISSAN's overall design philosophy, which shifts from purely passive defence to proactive defence, ensuring the network can predict and respond to threats before they appear in the operational environment.

3 Implementation and Validation of CI Mechanisms on the CISSAN Platform and its Use Cases

In this section, the implementation and validation of CI mechanisms, including data quality verification, anomaly detection and trust scoring are explained.

3.1 Implementation and validation of the CISSAN orchestration server for optimally distributing security subfunctions and automated disaster recovery

In CISSAN, the optimal distribution of security subfunctions and the automated disaster recovery mechanism were implemented by extending and customizing the Liquid AI Orchestrator into the CISSAN Orchestrator, as described in Section 3.7 of the CISSAN deliverable D2.3 report. As part of this work, the WASM security modules (Data Quality Verification, *AnomalyDetection*, and *TrustScoring*) were developed to be distributed to devices according to the task allocation produced by the Arctos Labs Optimization Solver. The Liquid AI Orchestrator was extended with automated failover and recovery capabilities, enabling continuous execution of the distributed security tasks by dynamically replacing inactive nodes with predefined alternatives and reintegrating recovered devices. The resulting orchestration workflow, validated in the joint scenario presented in the CISSAN deliverable D6.3 report, demonstrates how optimized task distribution combined with automated disaster recovery enables CI and supports collaborative defence across the CISSAN platform.

3.2 Implementation of the CISSAN Management Server, Rotor tool and trust management module for trust and device management

The CISSAN Management Server aggregates information from the Rotor tool executed on the RTUs in the ICS zone and the WASMs executed on the Raspberry Pis in the IoT zone. More detailed descriptions of the Rotor tool and the WASMs can be found in Section 2 of this report. Aggregated information includes anomaly scores, believability scores and trust scores. Trust scores are used for device blacklisting which is based on pre-determined thresholds. Scores and device blacklist status are updated on the blockchain for traceability and can be viewed on the CISSAN Management Server's Graphical User Interface (GUI). A more detailed description of the CISSAN Management Server can be found in CISSAN report D2.3.

WASM provisioning is handled in collaboration with the Arctos Labs Optimization Solver, the CISSAN Orchestrator and the CISSAN Management Server. The CISSAN Management Server requests an optimized deployment from the optimizer tool which calculates the optimal deployment for the WASM-modules based on the resources of the IoT devices in the network. The Arctos Labs Optimizer Solver sends the deployment template to the CISSAN Management Server which forwards the information to the CISSAN Orchestrator which deploys the WASMs on the IoT devices.

3.3 Implementation and validation of the Rotor framework

Implementation of the Rotor framework is described and validated in CISSAN deliverable reports D2.3 and D6.3 and is also part of a pending research paper. The framework was applied in the ICS Zone of the CISSAN Lab. This limits testing and targets solutions to UC2 scope, however, considering the nature of the UC2 networks and devices, there is strong potential to extend beyond specific use case borders.

3.4 Implementation and validation of distributed anomaly detection

The *AnomalyDetection* module was built with WebAssembly as it allows the same module to be run on heterogenous hardware without modification due to its capability of executing code compiled from

multiple programming languages in a sandboxed environment. The WASMs are managed by the management server and the CISSAN Orchestrator consisting of the orchestrator software running in a Docker container on a server and supervisors running on the IoT devices as services. Running the supervisors as services allows for the supervisor to be restarted automatically in cases where the device itself is restarted or in cases where the supervisor experiences a failure which causes it to shut down.

Provisioning of the module is handled by the CISSAN Management Server, the Arctos Labs Optimization Solver and the CISSAN Orchestrator. The CISSAN Management Server requests an optimized deployment from the optimizer tool which calculates the best device for the module based on the available resources of the IoT devices in the network. The optimized deployment is sent to the CISSAN Management Server which forwards it to the CISSAN Orchestrator. The CISSAN Orchestrator then deploys the module to the device indicated in the optimized deployment. The module itself calculates anomaly scores for the devices which are fetched by the CISSAN Management Server via HTTP for storage. The anomaly scores are used later by the *TrustScoring* module for calculating local trust scores. The local trust scores are aggregated at the management server and are used for calculating global trust scores for the network and for blacklisting devices with low trust scores.

3.5 Distributed Autoencoder Agents

The solution is a PoC implementation for the smart grids use case. The technology stack chosen balances performance, ease of use, and compatibility with edge deployment constraints. Python 3.12 serves as the primary programming language due to its extensive ecosystem for ML and web services. TensorFlow and Keras are employed for implementing the encoder-decoder model, providing a flexible and efficient framework for neural network training and inference.

For inter-agent communication, Flask is utilized to create lightweight RESTful services. This choice enables easy integration with HTTP-based protocols and simplifies the implementation of endpoints for coordination and health checks. The Requests library is used for making HTTP client calls between agents, ensuring reliable and efficient communication.

Concurrency safety is addressed through Python's threading and locks mechanisms. These tools allow agents to handle multiple requests simultaneously while maintaining data consistency during coordination phases. This is particularly important for preventing race conditions when agents are in the COORDINATING or RESPONDING states.

3.6 Implementation and validation of data quality verification

Two Data Quality Verification modules were implemented based on Python code from Geodata of the DTW and variance-based data quality verification methods. The modules were translated to WebAssembly for its capability of being run on heterogenous devices without modification. The management of the module is handled by the CISSAN Management Server and the Liquid AI software solution consisting of an orchestrator running in a docker on the CISSAN Orchestration server and supervisors running on the Raspberry Pis as services. Running the supervisors as services allows for the supervisor to be restarted automatically in cases where the device itself is restarted or in cases where the supervisor experiences a failure which causes it to shut down.

Provisioning of one of the modules to the devices is initiated by the CISSAN Management Server requesting an optimized deployment from the optimizer tool. The optimizer calculates the most suitable device for the deployment based on the available resources of the IoT devices in the network. The optimized deployment is returned to the CISSAN Management Server, which forwards it to the CISSAN Orchestrator, which then deploys the module on the chosen device. The module execution is then handled by the management server. The results of the Data Quality Verification modules aggregated at the CISSAN management Server and are later used by the *TrustScoring* module for calculating local trust scores for the devices. These trust scores are aggregated at the CISSAN Management Server and are used for calculating global trust scores of the network and for blacklisting devices with low trust scores.

Due to data coming in different forms from different use cases, we performed data preprocessing to unify all datasets from different cases into Geodata format. However, we found that experiment

performance results from other use cases are not as high as that obtained using the Geodata dataset. By further investigation, we think the data pattern is the main cause of poor generalization results. Dataset (voltage and Global Positioning System (GPS)) from smart grids and public transportation are not linearly distributed. Linear regression-based data quality verification is found to not be suitable for these datasets. As for the similarity-based methods, unlike neighbouring sensor readings from Geodata, we do not have similar features in other use cases. A potential solution is to extract some normal time series data from historical datasets as a reference to compare with.

3.7 Technical implementation of STIX generation

The integration of STIX within the CISSAN platform requires a practical mechanism for transforming aggregated anomaly information and global trust evaluations into structured, shareable threat intelligence. Although most components already produce standardized JSON output, additional logic is needed to extract relevant evidence, correlate it across devices and time intervals, and serialize it into valid STIX objects. This section outlines the architectural design and technical workflow for automated STIX generation within CISSAN, following the model in which STIX packages are produced specifically during significant trust-related security events.

A central asset in this process is the CISSAN Management Server, which serves as an aggregation and relay point for most of the CISSAN solutions leveraging relevant threat information. Because the server already receives the necessary telemetry in JSON Lines format it is the most appropriate location for STIX report generation. Implementing automated STIX export therefore requires the development of an experimental STIX generator service capable of parsing locally stored logs and transforming selected entries into STIX-compliant structures, when a blacklisting event occurs.

The key design choices concern the source data to be used and the trigger conditions for generating a STIX bundle. Parsing all possible CISSAN telemetry would be unnecessarily complex; instead, we demonstrate the concept using Rotor anomaly reports, which are fully controlled within CISSAN, independent of third-party software, and structurally predictable. This implementation focus also bounds the prototype to UC2, since Rotor is deployed in the ICS zone and produces the evidence relevant to trust-score degradation and blacklisting in that operational context. The second design decision involves determining when a STIX report should be generated. Three options were evaluated:

1. Per anomaly event – technically simple, but results in extreme verbosity and little analytical value.
2. Per reporting cycle – similarly noisy, dependent on periodic execution models, and inconsistent with CISSAN's scalability goals.
3. Per trust-related security event (blacklisting) – generates fewer, high-value reports at meaningful moments in the system.

Option 3 provides the best balance: it avoids excessive noise, ensures that only significant security events trigger CTI generation, and guarantees that the resulting STIX package contains enough contextual information to be operationally useful.

Even so, this approach introduces challenges. The most significant relates to temporal correlation: in CISSAN, a blacklist event is triggered when a device's global trust score falls below a configured threshold. This decline can occur gradually due to small inconsistencies piling up or rapidly due to severe anomalies. Therefore, the STIX generator must apply a correlation window (e.g., "include anomalies from the last *N* minutes") to collect the evidence most relevant to the decline. The length of this window must be tuned for each deployment. However, Rotor influences trust scores in a way that trust must be slowly build up over long consistency, while inconsistencies, depending on their severity could cause rapid drop in this level.

Another limitation is that, in this model, Trust Score SDOs become the primary entry point for STIX generation. Other SDO types are not automatically produced. While broader support could be implemented, such an expansion would dramatically increase report volume and complexity, which is unnecessary for demonstrating the trust-centric CTI extension developed in CISSAN.

With these design choices, the proposed implementation follows a five-step workflow:

1. **Monitoring for trust-score threshold violations:** The CISSAN Management Server continuously tracks global trust scores. When any device's trust score falls below the defined threshold, a blacklist event is triggered.
2. **Evidence collection:** Upon triggering, the STIX generator retrieves relevant anomaly evidence and peer trust-delta messages falling within the predefined correlation window.
3. **STIX bundle construction:** The generator constructs: one Identity SDO representing the affected unit; one Trust Score SDO (x-cissan-trust-score) representing the blacklist event; and N Observed Data SDOs, one per relevant Rotor anomaly in the latest evidence window for the target RTU. Each Observed Data object contains a custom cyber-observable (x-rotor-event) in its objects field to preserve the raw detection content. The Trust Score SDO then links to these supporting observations using basis_refs, providing traceability from the trust decision to the collected evidence. Optional enrichment objects may be added in future work but are not required for the core bundle generation.
4. **Serialization:** The generated objects are serialized into a STIX 2.1 bundle in JSON format and written to a local export directory and/or transmitted to CTI-sharing channels.
5. **Optional enrichment phase:** Each anomaly type may be mapped to its corresponding MITRE ATT&CK technique, enabling the generator to add Attack-Pattern SDOs and used-by relationships. This enhances analytical depth without increasing data collection overhead.

This workflow provides a structured and efficient technical pathway for embedding CTI sharing into the CISSAN framework, ensuring that trust-related security events and their supporting evidence can be transmitted in a standardized, interoperable format. Validation in this work is therefore defined at two levels. First, the implementation is validated operationally by repeatedly generating syntactically correct STIX 2.1 bundles whenever a blacklisting event occurs and by confirming that each bundle contains the expected objects and evidence links within the correlation window. Second, semantic validity of the proposed extension requires alignment with STIX modelling conventions and is ultimately confirmed through external review and standardization processes. In this report, the contribution is the working, automated generation pipeline and a consistent representation that can be assessed by the relevant bodies.

3.7.1 Proposed Implementation of the GAN-Based Approach

The GAN-based research is proposed as a PoC process and is planned for implementation and validation in the CISSAN lab environment. The proposed implementation targets the smart grid use case (UC2), where structured data from grid monitoring (including frequency, voltage, and current measurements) provides a well-defined feature space for training and evaluating the generative model. This feature set is consistent with that of the distributed autoencoder agent described in Section 2.2.5, ensuring that the GAN output can be directly used by existing detection components.

A conditional GAN architecture is envisioned, comprising a generator with three hidden layers (256, 512, 256 neurons, Leaky ReLU activation function, batch normalization) and a discriminator with a mirror structure. Training will be conditionally designed based on aggregated anonymized data collected from the CISSAN platform to selectively generate normal and anomalous traffic categories. This is crucial for enhancing the handling of imbalanced datasets commonly found in CISSAN's three use cases: Transportation (UC1), Smart Grid (UC2), and Tunnel Construction (UC3).

The validation plan will proceed on two levels. First, standard metrics such as Kullback-Leibler divergence and maximum mean difference will be used to compare the distributions of real and synthetic samples to assess statistical fidelity. Second, samples generated by GANs will be used to augment the anomaly detection model's training set, and the resulting performance improvement will be measured to evaluate the model's practical utility. Data scarcity is a common characteristic across all three CISSAN use cases, especially for rare attack types, where it limits model effectiveness.

Full implementation and validation on the CISSAN platform will be a later-stage project activity. Production deployment will require work on model update scheduling, dataset version control, and integration with WASM-based distribution pipelines used by other CISSAN modules.

4 Security, Privacy, and Compliance Considerations

This section describes how the collective intelligence algorithms presented in this report address security, privacy, and regulatory compliance requirements on the CISSAN platform and use case systems.

4.1 Data Protection and GDPR Considerations

A thorough evaluation of how the CISSAN platform conforms to applicable standards and regulations can be found in the CISSAN deliverable D7.2 report as per the standardization action plan described in CISSAN D7.1 report, and this section will focus only on the implications of data protection and General Data Protection Regulation (GDPR) considerations of CI algorithms used in the CISSAN platform.

4.1.1 Data quality verification

In the Data Quality Verification module, the primary data in question is sensor data. They are generated, transferred, and analysed. No personal data is utilised; therefore, it does not fall under the GDPR.

However, in case an anomaly is detected by the Data Quality Verification module, an alarm is generated and sent to notification targets, i.e., users. This requires personal data to be managed in the system (name, email address, phone number), and the GDPR applies.

In the example of the tunnel construction use case, GDPR compliance is ensured through managing personal data in the existing data management system GeodataHub of Geodata that applies/has integrated service-oriented architecture data protection mechanisms and processes like data minimisation, purpose limitation, access control, data deletion on request, etc.

4.1.2 Distributed anomaly detection

The behaviour-based anomaly detection component processes raw OT process measurements collected in substations, possibly along with timestamps and technical identifiers (device/substation IDs) needed for baselining and correlation. The data is not expected to include personal data in normal operation; therefore, GDPR typically does not apply to the measurement stream itself. If anomaly events are forwarded to operator workflows that include user identifiers or contact details (e.g., alerting lists, ticketing systems), GDPR-relevant data is handled by the operator's existing tools and processes. The anomaly-detection service limits what it stores and shares to what is necessary for security monitoring and auditability.

4.1.3 Trust management

In the trust management workflow, trust scoring is derived from device-level anomaly reports and peer-to-peer trust updates that are purpose-built for cybersecurity monitoring and response. The exchanged data consists of limited device information (device hostname, Internet Protocol (IP) address within the network), technical security observations and derived trust values, and apart from hostnames and IP addresses does not intentionally include personal data such as names, user identifiers, or content data. As such, apart from the hostnames and IP addresses, the mechanism aligns with GDPR principles of data minimization and purpose limitation by restricting collection and sharing to what is necessary for runtime security assurance and device eligibility decisions. To better comply with GDPR principles, The *TrustScoring* module needs to be modified in the future so that the collection of hostnames and IP addresses is no longer necessary. An avenue suggested for this would be to employ the capabilities of the CISSAN Orchestrator which is able to identify devices without private information from the devices. Where operational logs could indirectly relate to human activity, the platform treats this as security telemetry: access is limited to authorized components and personnel, use is confined to cybersecurity purposes, and retention should be bounded to the minimum period required for incident investigation, auditing, and iterative improvement of detection and trust policies.

Eventchain processes no personal data. Device identifiers are technical (e.g., "sensor-42"), and trust scores, anomaly results, and blacklisting events contain only numerical scores, detection methods, timestamps, and thresholds. BKVS namespace isolation ensures each API instance's data is cryptographically separated via its public key. GDPR compliance is ensured by design.

A transaction content deletion mechanism allows administrators to erase transaction payloads while preserving hash chain integrity, supporting the right to erasure without breaking verification. In practice, since no personal data is stored, this serves as an additional safeguard rather than a routine requirement.

4.1.4 Security and disaster recovery orchestration

The security and disaster recovery orchestration involves multiple CISSAN components and is carried out during different operational phases. The data exchanged during task allocation consists of tasks, devices, and network characteristics. Most of the parameters describe general metrics on capacity, quality of service (QoS) estimates, QoS targets, task dependencies, and so forth. Aside from abstract identifiers such as hexadecimal encoded device and module identities (except for textual function names) there are no explicit information such as IP-addresses, GPS coordinates, Fully Qualified Domain Name (FQDN) which can be used to pinpoint single device instances.

The data exchanged across entities during task deployment and recovery operations consists of control-plane information such as deployment manifests, device identifiers, module references, execution parameters, and health status reports. During normal operation and disaster recovery, messages include device state updates, deployment updates, and execution status notifications required for coordination and monitoring. Devices are identified using hexadecimal identifiers that serve solely a technical purpose and do not encode or reference personal information. The system does not associate devices or their IP addresses with users or user-generated data. Device IP addresses are stored only to enable technical communication between the orchestrator and edge devices and are linked exclusively to hardware operating in a controlled laboratory environment. As a result, the exchanged and stored information does not constitute personal data and does not pose a privacy risk.

Data associated with security and disaster recovery orchestration has by design its persistent storage limited to as few CISSAN components as possible. The data used for Security and disaster recovery orchestration is by its nature not related to any personal data subject for regulation as specified in GDPR.

4.1.5 STIX for Cyber Threat Intelligence Sharing in CISSAN

In terms of STIX-based cyber threat information exchange, the CISSAN platform primarily operates on technical security information such as indicators, alerts, observed information, and device or system identification for purposes of runtime detection and response. It operates in conformity with data minimization and limitation of purposes by only including relevant information in STIX bundles that is relevant for security analysis and response. Personal information is not processed; however, where information could be inferred from operational information such as logs, access to STIX information is limited to authorized components and users for purposes of cybersecurity only. The retention period for STIX reports and relevant logs is limited to what is necessary for incident analysis and improvement of detection models.

4.1.6 Fully Decentralized Anomaly Detection Using Distributed Autoencoder Agents

Personal data is protected in this system by design: raw data never leaves the local agent, and only compressed latent representations are shared among peers. These latent representations do not contain direct identifiers, minimizing the risk of re-identification. Data exchanges are limited to what is strictly necessary for anomaly detection, following the principles of data minimization and purpose limitation. Combined with secure communication channels and local processing, this ensures that the system aligns with GDPR requirements for privacy by design and by default.

4.1.7 Data Protection and Privacy Based on GANs

The GAN-based approach implemented in the CISSAN platform is designed to process technical data, particularly network traffic characteristics, sensor measurements, and derived anomaly indicators. GAN training does not use any personal data as input, and the synthetic data generated by the generator does not encode or reconstruct personally identifiable information. The main data types processed include aggregated electrical measurements (voltage, frequency, current) from smart grid use cases and derived statistical features for anomaly scoring.

From a GDPR perspective, a key advantage of GAN-based approaches lies in their design's support for data minimization. GANs do not require sharing raw operational data between nodes; instead, they allow nodes to use synthetic datasets that capture the statistical characteristics of the raw data without copying individual records. This reduces the leakage of sensitive operational information and enables effective collaborative model training. Furthermore, since the GAN generator model itself does not memorize individual training samples after proper regularization, the risk of indirectly leaking personal data through model parameters is minimized. In scenarios where the underlying telemetry data may be indirectly related to human activities (e.g., GPS data for transportation), the GAN training process applies preprocessing steps before model training, including aggregation, feature extraction, and anonymization, to ensure that no personally identifiable information remains in the synthesized output. Access to training data and GAN models is limited to authorized CISSAN platform components and personnel, consistent with the platform-level access control policy described in the project report.

4.2 Secure Information Sharing

For the use case systems and CISSAN platform implementations the information exchanged between devices, nodes, and platform components is protected at run time. The use of secure communication protocols, cryptographic mechanisms, authentication and authorisation, integrity protection, and standardised formats are applied here are further explained. This includes how to secure information sharing enables collective intelligence while preventing unauthorised access, manipulation, or leakage of sensitive network and device data.

4.2.1 Data quality verification

The Data Quality Verification module receives sensor data from the CISSAN Management Server (GeodataHub) via secure communication protocols, calculates believability scores and sends them back via the same protocols (Hypertext Transfer Protocol Secure (HTTPS), MQTT). Using these protocols, the information exchange between these two components is considered secure by design.

The sensors, gateways, and other devices used for generating and transferring data do not directly belong to the data quality verification component. However, their data exchange is secured primarily by security chips, encryption, and multi routing mechanisms.

Access to data on the data management server (GeodataHub) is protected through multi-factor authentication and restrictive user authorization and rights management.

4.2.2 Distributed anomaly detection

The behaviour-based anomaly detection component is designed so that raw measurements remain local to the substation where possible, while only derived indicators (e.g., anomaly scores, deviation features, event summaries) are shared for cross-checking and CI. Integrity and traceability can be supported through message integrity protection and security logging (model/version, timestamps, sender identity), and security-relevant event exchanges can be mapped to standardized to reduce ambiguity and limit manipulation or leakage.

4.2.3 Trust management

In the ICS-segment trust workflow, information sharing is designed to exchange only security-relevant summaries, rather than raw process data. Reports and trust deltas are distributed over an MQTT-based overlay, which supports standard protections such as broker access control, topic scoping, and encryption using Transport Layer Security (TLS) to provide confidentiality and integrity in transit when deployed outside a controlled environment. In the CISSAN platform setup, these protections are not fully enabled because communication occurs within a closed, access-controlled segment; nonetheless, the mechanism is compatible with production hardening by applying authenticated publish/subscribe policies, least-privilege topic permissions, and encrypted transport. Overall, exposure is constrained by limiting shared content to security telemetry and trust indicators, restricting who can publish/subscribe to trust channels, and enabling cryptographic protection when operating beyond an isolated test network. In the CISSAN platform's IoT zone, information sharing is limited to communication necessary for enabling the developed security functions. Communication between the CISSAN Orchestrator, the Raspberry PIs and the CISSAN Management Server is handled with HTTP due to how the CISSAN Orchestrator and supervisors have been developed. While HTTP is limited in its security, the CISSAN platform setup itself has access controls in place making it a closed environment.

4.2.4 Security and disaster recovery orchestration

The security and disaster recovery orchestration involves multiple CISSAN components and is carried out during different operational phases. As a general design objective, information sharing is subject to data minimization in which no excessive data which may solely identify, for example, device instances are included when not explicitly required. For the components collocated in the CISSAN platform, information sharing occurs within a closed, access-controlled segment. The Optimization Solver is, for development efficiency during the project, deployed outside of the CISSAN lab, and information sharing is protected by cryptographic site-to-site Virtual Private Network (VPN) communication mechanisms. If required in a post project production environment, a variety of enhancements may be applied such as co-location of components, additional authentication and authorization mechanisms, and stronger cryptography.

4.2.5 STIX for Cyber Threat Intelligence Sharing in CISSAN

STIX provides a standard, machine-readable representation for cyber threat intelligence that improves interoperability and traceability between components and domains. Any secure distribution of STIX content depends on the surrounding sharing mechanisms (e.g., authenticated access control, encrypted transport, and governance in a CTI platform or Trusted Automated eXchange of Intelligence Information (TAXII) service). In this work, the contribution is the generation of extended STIX reports from selected CISSAN evidence to demonstrate how observations can be expressed in a shareable CTI format. Operational publishing to external parties is intentionally out of scope for the lab environment and would be handled using established industry CTI sharing infrastructure.

4.2.6 Fully Decentralized Anomaly Detection Using Distributed Autoencoder Agents

In this system, data is shared in a manner that explicitly prioritizes security, privacy, and regulatory compliance. Agents never transmit raw measurements or identifiable time-series data; instead, they exchange fixed-dimensional latent representations produced by locally executed encoders. These latent representations act as information-limited summaries that significantly reduce the risk of sensitive data reconstruction. Communication occurs only on demand—triggered by suspected anomalies—thereby minimizing data exposure and network attack surface. All exchanges are peer-to-peer and can be protected using standard transport-layer security mechanisms (e.g., TLS), ensuring confidentiality and integrity in transit. Since no centralized data aggregation exists, the architecture avoids single points of compromise and aligns naturally with data-minimization principles found in regulations such as GDPR. Local autonomy over raw data storage further simplifies compliance, as data residency and access controls remain entirely within each agent's administrative domain.

4.2.7 GAN-Based Secure Information Sharing

Information sharing in the GAN-based workflow is designed to minimise the exposure of raw operational data. Rather than exchanging raw telemetry between nodes, the GAN approach enables the sharing of synthetic datasets and trained model parameters. The generator model, once trained centrally on aggregated and anonymised data, can be distributed to edge nodes where it produces synthetic training samples locally. This means that only the model weights, not the original data, need to be transferred between platform components.

Within the CISSAN platform environment, the trained GAN models are distributed from the management server to participating nodes using the same HTTP-based communication channels employed by other CISSAN modules. In the current implementation, communication occurs within a closed, access-controlled network segment. For production deployments, the communication channels can be hardened using TLS encryption and authenticated endpoints, consistent with the security posture described for other CISSAN components. The synthetic datasets generated by the GAN are formatted in the standard JSON format used across the platform, ensuring seamless integration with downstream *AnomalyDetection* and *TrustScoring* modules without requiring additional data transformation or exposing raw measurements.

4.3 Ethical and Legal Aspects

For your use cases systems and CISSAN platform, the main ethical and legal considerations related to the use of collective intelligence algorithms and automated security mechanisms are explained in the following sections. These include issues such as automation of security decisions, accountability and responsibility, transparency and traceability of actions, potential impact on operators or organisations, and compliance with relevant legal and regulatory frameworks (e.g., Network and Information Systems (NIS) 2, Critical Entities Resilience (CER) Directive, GDPR, where applicable).

4.3.1 Data quality verification

Calculating believability scores for sensor data through the data quality verification component may raise ethical and legal concerns. Legally, the scores may directly affect the safety of projects (e.g., tunnel projects) and lead to costly actions. Thus, the question of who is legally responsible for their provision, quality control, and application must be clearly answered before any application.

Since it is a human operator’s decision to select and configure the data quality rules, individual preference and personal traits are human characteristics that may lead to disproportional and unequal processing results raising ethical concerns. Certain sensors and data producers could be preferred/treated more tolerant than others, etc. Therefore, the applied data quality rules must be transparent and explicable. Their parameters must be understandable, suitable for the project and quality controlled; erroneous rules or parameters may be catastrophic and must be avoided at all circumstances.

To counter such concerns, appropriate control processes must be established accompanying the use of the Data Quality Verification module before commercialization.

4.3.2 Distributed anomaly detection

The behaviour-based anomaly detection system employed in this project processes raw OT process measurements in substations (e.g., voltage, frequency, and related electricity-derived signals). It learns a baseline from historical measurements and then continuously monitors current measurements after deployment to detect deviations. Ethical and legal considerations primarily relate to automation of security decisions, accountability, and operational impact: anomaly outputs are treated as decision-support signals, and any action with safety or service-continuity implications remains under human oversight and established OT procedures. Clear responsibility is assigned for baseline selection, configuration, deployment and parameter changes, supported by change-management practices to avoid inappropriate tuning or drift-related misdetections. To ensure

transparency, traceability and auditability, the system must maintain logs that link alerts to the relevant measurement windows, model/version and configuration parameters

4.3.3 Trust management

The use of collective intelligence and automated security mechanisms raises ethical and legal considerations primarily around automation, accountability, and the operational impact of runtime decisions. In the smart grids use case, trust related CI mechanisms can influence device eligibility and connectivity, so the key risk is disproportionate or erroneous action that could affect safety, availability, or operator control. This is addressed by designing decisions to be evidence-driven and traceable: trust changes are derived from explicit peer comparison outcomes, aggregated rather than based on single-device assertions, and recorded through structured logs that support auditability and post-incident review. To prevent abrupt or opaque actions, trust evolution is intentionally bounded and gradual under normal conditions while major state changes such as blacklisting are tied to clearly defined thresholds and remain visible through operator-facing monitoring views (SIEM/management dashboards), supporting human oversight and accountability. A further ethical and security concern is the integrity of the trust mechanisms themselves, since device-level monitoring may require elevated privileges. This is mitigated operationally through controlled access, segmentation, and logging to detect misuse, and by treating trust outcomes as part of governance rather than as an unquestionable “black box” decision. From a compliance perspective, the approach aligns with the intent of critical-infrastructure cybersecurity frameworks (e.g., NIS2) by strengthening detection, response readiness, and incident traceability, while GDPR considerations are supported through data minimization and access control, since trust signals and shared reports are designed as technical security telemetry rather than personal data.

Similar to the smart grids use case, the ethical and legal considerations for the use of collective intelligence and automation of security mechanisms in the transportations use case and the tunnel construction use case are centred on how CI mechanisms may affect device eligibility and connectivity. To cope with this, the trust scores that are used for decision making are based derived from observed peer responsiveness and the results of anomaly detection methods with emphasis on recent results while considering historical anomaly data. Most recent scores and changes in blacklisting status are visible for human operators via monitoring views.

4.3.4 Security and disaster recovery orchestration

The ultimate purpose with automated security and disaster recovery orchestration is to assist the security administrators to make better deployment decisions and to prepare the network upfront for disaster recovery operations. Even for small networks the combinatorics of all possible deployment solutions are overwhelming and a subject well suited for automation. A human-in-the-loop approach ensures that automated decisions, both for initial task deployment and fallback at disaster recovery, may be subject to evaluation and approval before being applied to the system. Automated disaster recovery orchestration reduces the risk of imprecise human operations during situations with high stress. The concept of isomorphic and liquid software as developed in the Liquid AI project plays a vital role as it allows software modules to be developed independently of the underlying hardware and deployed uniformly across the system. This is achieved by utilizing WebAssembly as a common and controlled execution environment for software modules implementing the application logic.

The components involved in the security and disaster recovery orchestration functionality are generic and independent of the business purpose and domain in which the network is used. Therefore, it is primarily the network itself and the security tasks being deployed, which are subject to regulatory frameworks when it comes to, for example, operational impact the network may have on people, organizations, and properties.

The components responsible for security and disaster recovery orchestration operate on data which are not explicitly tied to personally identifiable data even if a given network should be designed to operate in such a domain.

The components responsible for security and disaster recovery orchestration contribute to aspects such as accountability, transparency and traceability within their respective areas, each according to their specific implementation.

The optimization mechanism used in CISSAN provides distinctive qualities. Unlike trending ML approaches, especially manifested by large language models, the CISSAN mechanism is fully deterministic which means full transparency to the decision engine behaviour. The system

administrator may also tune the behaviour using mechanisms such as setting weights on selected parameters, binding specific tasks to specific devices or types of devices.

The CISSAN Orchestrator ensures transparency and traceability by logging all device status changes and deployment updates to its database. Human oversight is maintained through the management server, which operates as the actor responsible for initiating actions and approving system-level decisions, while the CISSAN Orchestrator is limited to executing the requested tasks.

4.3.5 STIX for Cyber Threat Intelligence Sharing in CISSAN

The use of STIX in the automation of threat intelligence information sharing and correlation presents various ethical and legal issues with regards to accountability and transparency in the utilization of shared security information. To mitigate such issues, the CISSAN platform ensures the traceability of STIX reports and the origin of the information, as well as the relationships between the information and the decisions made based on the shared information. This ensures auditing and control of various organizational and regulatory activities with regards to the automation of security actions based on STIX-based inputs. The utilization of an open and accepted standard is essential in ensuring compliance with regulatory requirements and best practices, while avoiding issues with regards to a lack of transparency in decision-making processes in critical infrastructure cybersecurity operations.

4.3.6 Fully Decentralized Anomaly Detection Using Distributed Autoencoder Agents

From an ethical and legal perspective, the proposed system is comparatively low risk. Its design explicitly avoids centralized data collection and prevents the sharing of raw or personally identifiable data, which strongly aligns with principles of data minimization, purpose limitation, and privacy by design. By operating on locally generated latent representations, the system reduces the likelihood of unintended data leakage or secondary use of sensitive information.

With respect to the European Union (EU) AI Act, the framework would typically fall outside the “high-risk” category, as it performs monitoring and anomaly detection rather than automated decision-making that directly affects individuals’ rights or safety. The system is therefore closer to a limited-risk or minimal-risk AI application, provided it is not deployed in a safety-critical or regulated domain (e.g., medical diagnosis or credit scoring).

To be AI Act–ready, transparency obligations should still be addressed, including clear documentation of model behavior, limitations, and update procedures. Human oversight, auditability of anomaly decisions, and robust cybersecurity controls would further strengthen compliance.

4.3.7 GAN-Based Methods: Ethical, Legal, and AI Act Compliance

The use of GANs for synthetic data generation and adversarial training in the CISSAN platform raises specific ethical and legal considerations, particularly with respect to the EU AI Act and the responsible deployment of generative AI in cybersecurity contexts.

EU AI Act classification and compliance

Under the EU AI Act, the GAN-based methods developed in CISSAN would be classified as limited-risk or minimal-risk AI systems. The GANs are used for cybersecurity monitoring support, specifically for data augmentation, adversarial training, and model robustness testing, rather than for automated decision-making that directly affects individuals or safety-critical processes. The GAN outputs (synthetic training data, adversarial test samples) serve as inputs to human-supervised anomaly detection and trust scoring pipelines, and no autonomous actions affecting device eligibility or network access are taken solely based on GAN outputs. This design ensures that the GAN methods remain in a supporting role within the decision chain, preserving human oversight over consequential security decisions.

Transparency and documentation

To satisfy the transparency obligations of the AI Act, the GAN training procedure, model architecture, training data sources, and intended use are documented in the CISSAN technical deliverables. The model behaviour and limitations are explicitly described, including known failure modes such as mode collapse or distribution drift. The synthetic data generated by the GAN is clearly labelled as such within the platform, preventing confusion with real operational measurements. Model versioning and update logs are maintained to support auditability and reproducibility.

Ethical considerations

From an ethical perspective, the primary concern with GAN-based methods in cybersecurity is their dual-use potential: the same techniques that enable defenders to generate realistic attack scenarios could, in theory, be misused to craft adversarial attacks. This risk is mitigated in the CISSAN context by restricting access to the GAN models and their outputs to authorized platform components and personnel, by operating within a controlled laboratory environment, and by designing the GAN workflow specifically for defensive purposes. The synthetic data generation is governed by the same access control and audit policies that apply to all other CISSAN security components.

The deterministic and auditable nature of the GAN training process, combined with the clear documentation of model parameters and training data, supports the accountability requirements of both the AI Act and broader cybersecurity governance frameworks such as NIS2. By maintaining full traceability from training data through model outputs to downstream security decisions, the CISSAN platform ensures that GAN-based methods contribute rather than undermine the transparency and explainability of the CI system.

Conclusion

This report outlines the collective intelligence algorithms that were developed and implemented on the CISSAN platform to improve secure device communication and network security in complex IoT and OT environments. In particular, the report describes how distributed and collaborative mechanisms can be used to improve traditional security architectures so that these systems can more effectively detect, analyse, and respond to emerging threats in an adaptive and coordinated way. STIX is used to provide a standardized and machine-readable format for sharing cyber threat intelligence and can provide the basis for achieving greater interoperability between heterogeneous components and domains.

A proof-of-concept decentralized anomaly detection framework where each agent serves as both an analyser and a coordinator was developed. The system achieves collective intelligence and resilience through a simple yet effective communication protocol, laying the groundwork for self-organizing, cooperative fault detection across heterogeneous, large-scale systems. The key innovation of this approach lies in its fully decentralized architecture, which eliminates single points of failure and enables scalable deployment. This framework provides a solid foundation for further research and development in the field of decentralized anomaly detection. Future work includes integrating the framework into the CISSAN platform and addressing the remaining challenges, particularly in the areas of distributed model updates, robust handling of network issues, and consensus-building among agents. The goal is to create a fully adaptive, self-organizing system capable of operating in dynamic, large-scale environments with minimal human intervention.

Decentralized anomaly detection mechanisms are used for obtaining and sharing trust scores as a STIX object in an extended STIX report format in achieving collective intelligence. This provides a basis for achieving greater contextual awareness in IoT/OT systems so that these systems can more effectively detect, analyse, and respond to emerging threats in a dynamic way. Example extended STIX threat report formats are provided in this report for the transportation and smart grids use cases.

The potential benefits of collective intelligence for enabling secure device communication and network security solutions in run-time are provided. Although not all partners were able to adopt the STIX threat report format, this is planned to be addressed post-project. While there is still more work that needs to be done to fully industrialize CISSAN solutions, the approach that is presented in the report can provide the basis for achieving greater scalability, interoperability, and resiliency in these solutions so that they can more effectively support the European Union's strategic, operational, and regulatory objectives in cybersecurity.

References

[1] C. A. Baykara, I. Şafak and K. Kalkan, “ZETROS : A zero-trust IoT network security framework using distributed blacklisting, trust scoring and smart contracts,” *Computer Networks,* 2025.

[2] N. Taşgetiren, Ş. Ilgın and M. S. Aktaş, “Aging-Based Weighting for Session Classification in User Behavior Analysis,” in *Computational Science and Its Applications – ICCSA 2025 Workshops*, 2025.

[3] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. 2018. Truth Will Out: Departure-Based Process-Level Detection of Stealthy Attacks on Control Systems. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018.

[4] Wissam Aoudi, Magnus Almgren, A scalable specification-agnostic multi-sensor anomaly detection system for IIoT environments, International Journal of Critical Infrastructure Protection, Volume 30, 2020.

[5] Magnus Almgren, Wissam Aoudi, Robert Gustafsson, Robin Krahl, and Andreas Lindhé. 2018. The Nuts and Bolts of Deploying Process-Level IDS in Industrial Control Systems. In Proceedings of the 4th Annual Industrial Control System Security Workshop (ICSS), 2018.

[6] Wissam Aoudi and Magnus Almgren. 2021. A Framework for Determining Robust Context-Aware Attack-Detection Thresholds for Cyber-Physical Systems. In Proceedings of the 2021 Australasian Computer Science Week Multiconference (ACSW), 2021.

[7] Kemal, Mohammed S., Wissam Aoudi, Rasus L. Olsen, Magnus Almgren, and Hans-Peter Schwefel, Model-free detection of cyberattacks on voltage control in distribution grids, In *2019 15th European Dependable Computing Conference (EDCC), 2019.*

[8] N. Q. T. a. J. R. V. Jeffrey, “A hybrid methodology for anomaly detection in Cyber–Physical Systems,” *Neurocomputing,* vol. 568, no. 127068, 2024.

[9] N. Q. T. a. J. R. V. Jeffrey, “A hybrid methodology for anomaly detection in Cyber–Physical Systems,” *Neurocomputing,* vol. 568, no. 127068, 2024.