Topology Meets Security: ICS Intrusion Detection via Topological Deep Learning

***Abstract*—Industrial Control Systems (ICS) are organized around the Purdue model hierarchy of sensors/actuators, Programmable Logic Controllers (PLCs), and supervisory layers. Despite this, existing data-driven intrusion detectors treat ICS telemetry as either a multivariate sequence or a pairwise graph, collapsing the hierarchy and missing higher-order, cross-level dependencies that govern control propagation and sensing dynamics. Motivated by this critical modeling gap, we introduce `TMS-ICS`, a hierarchy-aware, process-level industrial intrusion detection framework that preserves this structure in both modeling and inference. At its core, a Topological Neural Network (TNN) encodes the Purdue hierarchy, where the physical processes and basic control layers are modeled using Combinatorial Complex (CC). CC cells carry benign telemetry, from which we compute features to train the TNN. As such `TMS-ICS` organically preserves Purdue semantics by aligning 0-cells (devices: sensors/actuators), 1-cells (functional dependencies), and 2-cells (PLC control groups) with the plant design, inherently improving its interpretability. The framework requires only benign traces and a device-to-PLC mapping, making integration practical in ICS environments. We evaluate `TMS-ICS` on three ICS benchmarks: `SWaT`, `WADI`, and `TEP`. `TMS-ICS` raises point-level detection on `SWaT` to 82.49% F1 and sets a new state-of-the-art 51.25% F1 on `WADI`. Our ablation study validates that hierarchical modeling matters. By incorporating 2-cells (PLC groups), `TMS-ICS` boosts precision by 6.7% and F1 by 7.5%. Such capability enables actionable intrusion localization across hierarchical levels, from field devices to PLC control zones. We make `TMS-ICS` available to the public.**[1]

1. Introduction

Industrial Control Systems (ICS) constitute the critical infrastructure foundation of modern society, powering essential services across oil and gas operations, electrical grids, and water management facilities. These systems face an escalating threat landscape, with the Department of Homeland Security (DHS) issuing increasingly urgent warnings about sophisticated cyber attacks targeting these vital assets from both domestic and international threat actors [49]. Among the most concerning attack vectors in this domain are ***(i)*** the malicious overwriting of process-variable tags[2] to falsify readings (`MITRE ATT&CK ICS T0856` [35]), and ***(ii)*** the injection of unauthorized commands that drive actuators into unsafe states (`MITRE ATT&CK ICS T0855` [36]). Both tactics disrupt physical processes while seeking to remain invisible to routine control checks. The increasing sophistication of cyber-physical attacks, coupled with the limitations of current data-driven detection techniques, underscores the need to revisit intrusion detection systems that better align with the ICS Purdue model [42], [48] and ICS dependencies within control loops.

Current Challenges and Limitations. The inherent complexity of ICS, combined with the inability of state-of-the-art detectors to accurately model and analyze its real-world hierarchical structure, poses dual challenges: they undermine accurate, practical system modeling, while ultimately hindering effective anomaly detection and attack localization. These limitations manifest in the following ways:

Lack of ICS hierarchical architecture modeling: ICS follow a hierarchical, multi-level structure captured by the Purdue model [42], [48]. This typically encompasses distinct layers for sensing (e.g., sensors), actuation (e.g., pumps), control (e.g., Programmable Logic Controllers-PLCs), and monitoring (e.g., Human Machine Interface-HMI). Across various interdependent and often geographically distributed subprocesses, this architecture gives rise to complex causal relationships that necessitate careful modeling. Nonetheless, current data-driven Industrial Intrusion Detection Systems (IIDS) neglect these hierarchical and structural intricacies, and tend to focus solely on observable I/O, overlooking cross-level dependencies [51].

Operational burden of manual, expert-led ICS detector design: The design and training of an IIDS demand significant manual *and* computational labor, from carefully crafting the system architecture to analytically encoding its physical and temporal constraints. Knowledge-based approaches still require experts to handcraft process invariants or state-space models for each control loop, which does not scale to large plants. At the same time, recent data-driven methods that automate parts of this process, such as `GeCo`'s automatic mining of state-space models from historical data, remain highly resource-intensive, reporting long offline training times on realistic installations (*e.g. more than 40 days on `WADI`*) [51]. These manual and computational costs slow down IIDS training iteration and delay practical deployment, ultimately weakening the ability to detect and respond effectively to evolving cyber threats in ICS environments.

1. https://anonymous.4open.science/r/TMS-ICS

2. A *tag* is the named variable (e.g. `LIT101.PV`) that a PLC exposes over its protocol; it carries the live value of a sensor, actuator, or internal register.

Deficiencies in ICS anomaly detection models: Existing state-of-the-art deep learning methods, including CNNs [31], LSTMs [19], [29], GNNs [15], [44], [46], and physics-aware schemes [9], [28], [47] model ICS dynamics at only a single level of abstraction (i.e., spatial, temporal, pairwise relational, or physics-constrained, respectively). As a result, current models struggle to capture higher-order ICS assets and cross-level interactions between PLCs and their subordinate sensors and actuators.

Absence of adequate evaluation metrics for hierarchical anomaly localization: Localizing anomalies at a specific hierarchy within interdependent cause-effect processes is essential as delays related to this task can result in cascading failures and collateral damage that become increasingly difficult to manage [6], [21]. Despite its significance, current deep learning models lack accurate anomaly localization capabilities and standardized metrics for evaluating their effectiveness.

Contributions. We introduce **T**opology **M**eets **S**ecurity for **ICS** (**TMS-ICS**), a novel security framework that leverages algebraic topology to model ICS for enhanced intrusion detection and localization. Our approach utilizes Combinatorial Complexes (CC), a mathematical structure that extends traditional graphs, hypergraphs, and simplicial complexes [26]. These complexes capture hierarchical relationships through order-preserving rank functions, enabling the representations of interactions across multiple levels of the ICS hierarchy [27]. The framework maps ICS components into a hierarchical structure where sensors and actuators constitute 0-cells, cause-effect relationships define 1-cells, and PLC-grouped entities form 2-cells. We encode time-series data as rank-specific features ("cochains") attached to cells at each level, so that sensors/actuators, their functional links, and PLC groups all carry their own learned representations. **TMS-ICS** integrates Topological Neural Networks (TNN) and, in doing so, casts anomaly detection as a self-supervised reconstruction of normal system behavior. The architecture employs multiple Higher-Order Attention (HOA) layers that compute attention weights via intra- and inter-rank message passing. Three independent rank-specific decoders reconstruct input (original) cochains enabling granular anomaly detection and localization. For precise fault localization, we introduce the Effective Operator Hierarchical Localization (EOHL) metric, which quantifies reconstruction errors across ranks at specific timestamps. This metric identifies compromised components and their relational adjacencies, crucial for understanding vertical cascading effects in ICS environments.

A comprehensive evaluation shows that **TMS-ICS**, trained only on benign traces with simple rank-wise thresholds, outperforms strong learning baselines on SWaT, achieves state-of-the-art performance on WADI, and remains competitive with invariant/state-space mining while avoiding long offline mining. Ablations confirm that the PLC-level (2-cell) layer boosts accuracy and multi-rank localization, while pinpointing root causes within a single control cycle. The detector runs in real-time on a commodity CPU and generalizes well across ICS domains, including TEP [11]. We compare manual, semi-automatic, and automatic edge construction regimes and observe predictable trade-offs; full results appear in Section 4.

In summary, **TMS-ICS** makes the following key contributions:

1. **Hierarchy-aware ICS Modeling:** We map the Purdue hierarchy to a three-rank CC (tags become 0-cells, cause-effect links 1-cells, and PLC zones 2-cells) offering a framework that models hierarchical dependencies, while capturing higher-order relationships, overcoming the modeling limitations of existing methods.
2. **TNN-based Intrusion Detection.** We recast industrial intrusion detection as a rank-wise reconstruction objective on the defined CCs. These CC cells carry benign ICS data readings while their respective calculated cochains (features) are used for TNN model learning [27]. Using three lightweight decoders (for 0-, 1-, and 2-cells), our intrusion detector raises an alert whenever any rank's residual surpasses its validation threshold. No attack data or manual rules are required during the learning stage.
3. ***Top-Down Hierarchical Anomaly Localization:*** We propose a top-down anomaly attribution technique that starts detection at higher-rank cells, descending to components for precise localization. Paired with our proposed EOHL metric, which measures reconstruction error across ranks, this approach identifies faulty components and their adjacencies, enabling faster, targeted remediation in practical ICS environments.
4. ***Comprehensive Evaluation in Operational Conditions:*** We validate **TMS-ICS** on the SWaT, WADI and TEP benchmarks. We provide three interchangeable edge-construction regimes: manual (Piping and Instrumentation Diagram (P&ID)), semi-automatic (state-space), and automatic (embedding), allowing operators to trade interpretability for onboarding speed. The obtained results demonstrate its ability to effectively model ICS assets, infer anomalies and localize faults, while successfully addressing the stringent demands of production environments.
5. **Open resources for reproducibility.** All code, artifacts, and evaluation scripts are released at https://anonymous.4open.science/r/TMS-ICS.

The remainder of the paper is organized as follows. Section 2 provides background information and discusses our threat model. Section 3 details the **TMS-ICS** architecture and design rationale. Section 4 presents the experimental setup and results, answering our research questions. Section 5 surveys related work. Finally, Section 6 offers concluding remarks and highlights future work.

2. Background

This section reviews the essentials, namely, the studied architecture and related threat model, and topological deep-

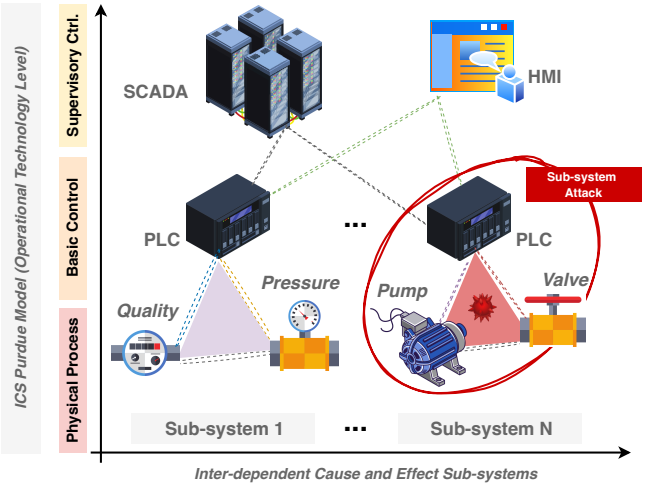


Figure 1: **Water-system Purdue-based ICS Example.** Two representative subsystems, each controlled by a PLC and supervised by SCADA/HMI. The sketch shows the main actuators (Pump, Valve); other sensors are omitted for clarity. The red circle illustrates a coordinated subsystem manipulation (attack).

learning principles that empower our approach.

2.1. Industrial Control Systems (ICS)

ICS automate physical processes such as water treatment, power generation, and manufacturing. At the field level, *sensors* report the state of the process (e.g., levels, pressures, flows) and *actuators* (e.g., valves, pumps) change it. PLCs implement control logic that maps sensor readings to actuator commands; HMIs and SCADA servers supervise multiple PLCs. This stack mirrors the def-facto Purdue model and induces a natural *hierarchy*; devices, their signals, interdependencies, and PLC-grouped subsystems.

A simple ICS example. As sketched in Figure 1, an industrial water network is organized into subsystems, each controlled by a PLC. Within a subsystem, a pump raises pressure and flow, while a motorized valve throttles them. The PLC closes the loop by reading local sensors (e.g., pressure, flow) while issuing discrete commands (start/stop the pump; open/close the valve) to keep operation within target ranges; SCADA/HMI stations supervise multiple PLCs and allow setpoint changes. An adversary who can issue commands could coordinate actuators within a subsystem, for example, periodically boosting the pump while partially closing the valve, or toggling several valves in the same zone. This produces pressure surges and brief flow dips that propagate downstream. Viewed per tag, each signal may remain within nominal bounds, reducing the sensitivity of univariate detectors. At the PLC-subsystem level, however, the correlation becomes evident as several field devices exhibit synchronized anomalous behavior that contradicts expected control logic patterns. As shown in Section 4.4, this subsystem view surfaces such attacks early and consistently, whereas flat or pairwise models often miss or fragment the alert.

ICS-specific Threat Model. We view an ICS from an *operator-defender* perspective, consistent with literature- and industry-driven models [24], [37], [45]. Field devices (sensors, actuators) form a Common Industrial Protocol (CIP) ring at *Level 0*; their PLCs exchange setpoints over a star-switched *Level 1*. As in many real deployments, these communication links often lack encryption or strong authentication mechanisms. Adopting the taxonomy by Cárdenas *et al.* [13], any actor who reaches the L1 network (e.g., via a misconfigured VPN, a rogue insider, or brief on-site access) can; ***(i)*** passively monitor PLC traffic; ***(ii)*** forge or replay EtherNet/IP/CIP packets; ***(iii)*** modify tag values to falsify readings (`MITRE ATT&CK ICS T0856` [35]); and ***(iv)*** inject unauthorized actuation commands (`MITRE ATT&CK ICS T0855` [36]). To this end, the attacker seeks to degrade safety or operational throughput by persistently or intermittently manipulating process tags; for example, in a water-treatment plant this might cause tank overflows, chemical misdosing, or flow restrictions. **`TMS-ICS`**, trained exclusively on *benign* historical data, continuously monitors a mirrored tag stream inaccessible to adversaries. Upon detecting anomalies, human operators can investigate and implement countermeasures, including PLC isolation or manual control activation. We consider a threat model to be out of scope of this study when the adversary alters hard-wired interlocks, causing immediate physical damage, or compromises the isolated instance of **`TMS-ICS`**. Purely network-side attacks that do not perturb the physical process are also out of scope.

This threat model accurately captures today's ICS security landscape: sophisticated network adversaries confronting defenders who rely primarily on data-driven monitoring rather than comprehensive cryptographic protection. Consequently, **`TMS-ICS`** must deliver: ***(i)*** high precision detection despite operational noise, and ***(ii)*** rapid, hierarchy-aware anomaly localization; requirements that we evaluate in Section 4.5.

Intrusion Detection for ICS. Modern ICS comprise distributed control loops that exchange setpoints and measurements across multiple PLCs using vendor-specific protocols. As plants adopt remote maintenance and cloud analytics, formerly isolated networks become reachable from corporate IT and, ultimately, the public Internet [14]. Incidents that tamper with control-loop inputs or override actuator commands have moved from theory to practice in water treatment, pipelines, and manufacturing [8], [12]. Industrial IDSs typically run without hardware changes and differ along two axes: *input data* and *detection technique* [33]. Inputs include network traffic, host logs, and physical-process traces (time-series tags for sensors and actuators). We use the latter, making our detector process-aware. Detection techniques split into *knowledge-based* and *behavior-based*. Knowledge-based methods rely on labeled attacks, whereas behavior-based methods learn normal operation during training and flag deviations during inference. Because attack samples are scarce and plant-specific [4], [16], [32], [50],

which risks overfitting in the context of supervised methods, we adopt a behavior-based, process-aware design trained solely on benign traces.

Within this behavior-based setting, prior work splits into *knowledge-centered* (process invariants, observers) and *data-centered* approaches (sequence models, graph models, invariant and state-space mining). We follow the data-centered route while injecting the Purdue hierarchy as an inductive bias (Sections 2.2 and 5). In practice, two properties matter: *generalizability* (transfer across ICS hardware, protocols, and domains with minimal re-tuning [38]) and *localization* (pinpointing the root cause so operators can respond appropriately [16], [20]). Given that ICS constitute stratified architectures integrating field devices, operational interactions, and PLC zones [45], anomaly detectors that operate on isolated devices or aggregate plant operations into singular latent embeddings fail to preserve cross-level structural relationships [2], [15], [17]. Accordingly, we leverage topological learning to explicitly model multi-level dependencies, as introduced next.

2.2. Topological Deep Learning (TDL)

Why go beyond graphs? Graphs encode only pairwise links. ICS operate across coupled scales: ***(i)*** individual devices, ***(ii)*** directed cause-effect relations, and ***(iii)*** PLC-grouped control loops. TNNs extend GNNs by operating on higher-order *cells* and by passing messages both within a level and across levels. This facilitates long-range propagation that respects hierarchical structure while avoiding multi-hop graph traversal. *Combinatorial complexes* (CCs) provide a natural input for such models [26], [27]. Figure 2 illustrates the three ranks used as a running example.

Combinatorial complex (CC) in the ICS context. Let S be a finite set. A CC is a triple $\langle S, \mathcal{X}, \mathrm{rk}\rangle$, where $\emptyset \notin \mathcal{X} \subseteq \mathcal{P}(S)$ is a family of nonempty subsets (*cells*) and $\mathrm{rk} : \mathcal{X} \to \mathbb{Z}_{\geq 0}$ is a strictly order-preserving rank: if $x \subset y$ then $\mathrm{rk}(x) < \mathrm{rk}(y)$. For $k \geq 0$, let $\mathcal{X}^k = \{x \in \mathcal{X} : \mathrm{rk}(x) = k\}$ denote the set of k-cells. See *Appendix A.1*, for formal details.

In the ICS setting shown in Figure 2, 0-cells may be created from sensor tags S_i (orange) and actuator tags A_j (green), yielding $\mathcal{X}^0 = S \cup A$. Then, directed 1-cells encode cause-effect links between tags, such as $S \to A$ (a sensor drives an actuator), $A \to A$ (actuator coordination), and $A \to S$ (actuation affects a measured state). These edges may lie within one PLC or span two PLCs through physical process coupling. We formalize $\mathcal{X}^1$ (1-cells) construction in Section 3.2. Finally, $\mathcal{X}^2$ (2-cells) represent PLC zones, encompassing control loops that include all related tags and their internal edges (as in regions PLC_1, PLC_2, PLC_3).

Cochains (signals/features on cells). A k-*cochain* assigns a feature vector to each k-cell:

$$C^k(\mathcal{X}, \mathbb{R}^d) = \{\, H^{(k)} : \mathcal{X}^k \to \mathbb{R}^d \,\}. \tag{1}$$

Let the maximal rank be R. At time t, a cochain tuple is

$$\mathbf{H}_t = \big(\mathbf{H}_t^{(0)}, \mathbf{H}_t^{(1)}, \ldots, \mathbf{H}_t^{(R)}\big), \quad \mathbf{H}_t^{(k)} \in C^k\big(\mathcal{X}, \mathbb{R}^{d_k}\big),$$

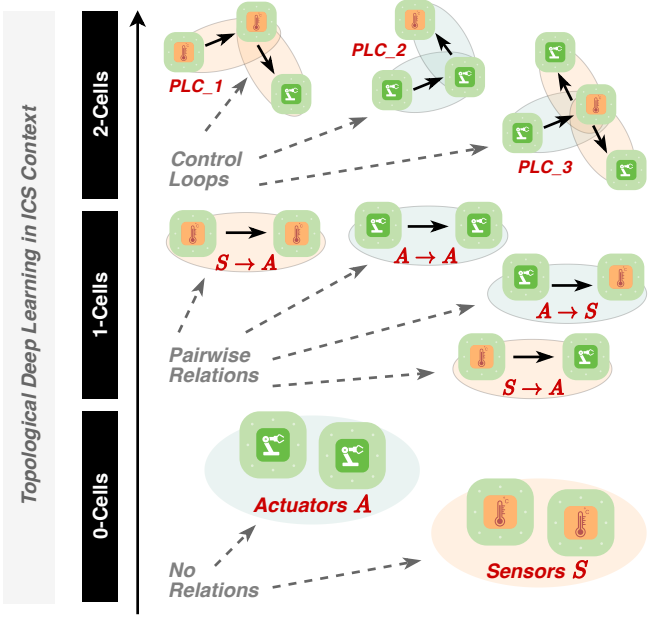


Figure 2: **ICS components via CC.** 0-cells (bottom) are individual sensors/actuators; 1-cells (center) encode direct cause-effect links; 2-cells (top) group all devices under the same PLC control loop.

where d_k may depend on k and on the application. The sequence $\{\mathbf{H}_t\}_{t=1}^T$ provides the rank-wise signals.

ICS example (rank R=2). Typical choices are ***(i)*** 0-cochains: normalized tag values; ***(ii)*** 1-cochains: features from an edge's endpoints (for example concatenation or signed differences); and ***(iii)*** 2-cochains: pooled statistics over a PLC zone (for example mean, variance, or fraction of actuators active). Cochains turn a static CC into a *learnable* object by supplying signals a model reconstructs or predicts (Section 3). Because cochains are user-defined, the formulation is a *framework* in which per-rank features can be tailored without modifying the model's architecture. Formal definitions of cochains and cross-rank maps appear in *Appendix A.2–A.3*.

Neighborhood and incidence matrices. Two matrix families support message passing:

- *Same-rank neighborhoods* $A_k \in \{0,1\}^{|\mathcal{X}^k| \times |\mathcal{X}^k|}$, where $(A_k)_{xy}$=1 when $x, y \in \mathcal{X}^k$ are neighbors. Concretely, we take the union of *adjacency* (share a higher-order coface) and *coadjacency* (share a lower-order face); check *Appendix A.4–A.5*.
- *Incidence (boundary) matrices* $B_k \in \{0,1\}^{|\mathcal{X}^{k-1}| \times |\mathcal{X}^k|}$ with $(B_k)_{yx}$=1 if and only if $y \subset x$ and $\mathrm{rk}(y)$=k−1, $\mathrm{rk}(x)$=k.

Intuitively, A_k passes messages within a level (tag to tag, edge to edge, PLC to PLC), and B_k passes messages across levels (tags to edges to PLCs, and back). For the ICS case studies in Section 3, we focus on $k \leq 2$ because PLC-organized process data map naturally to 0, 1, and 2 cells.

The general construction for arbitrary maximal rank appears in *Appendix A*.

Topological Neural Network (TNN). We implement a *Combinatorial Complex Attention Neural Network* (CCANN) and, for brevity, refer to it as a *TNN* throughout the paper; GNNs on the 1-skeleton arise as a strict special case. In general, TNNs operate on CCs of arbitrary maximal rank and can couple signals across multiple levels. In this work, the highest rank is 2, and we use a TNN of the following form:

$$f_\Theta : \ (C^0 \times C^1 \times C^2) \to (C^0 \times C^1 \times C^2) \tag{2}$$

that updates cochains using message passing along $\{A_k, B_k\}$. One concrete instantiation uses higher-order attention layers that mix same-rank and cross-rank information, followed by small rank-specific decoders (Section 3). *Appendix A* outlines the general case for CCs with higher maximal rank.

Having presented the ICS setting, dominant IDS paradigms, and foundational concepts in TDL, we situate our contribution within the existing literature in Section 5.

3. `TMS-ICS`: Rationale and Methodology

`TMS-ICS` is a hierarchy-aware intrusion-detection system that models an ICS as a combinatorial complex (CC), encodes multivariate process traces as cochains, and learns normal operation with a two-layer higher-order attention (HOA) network (Figure 3). This section details the design choices: CC construction (0/1/2 cells), cochain definitions, HOA encoder–decoder, thresholding, and hierarchical localization.

3.1. Problem Statement

Let the plant hierarchy be encoded as a combinatorial complex $\mathcal{X} = (\mathcal{X}^0, \mathcal{X}^1, \mathcal{X}^2)$ with 0-cells (tags), 1-cells (pairwise relations), and fixed 2-cells (PLC zones). At regularly-sampled time steps $t \in \mathbb{N}$, the control system emits a state vector $\mathbf{s}(t) \in \mathbb{R}^N$ that is mapped to rank-wise cochains $\mathbf{H}_t = (\mathbf{H}_t^{(0)}, \mathbf{H}_t^{(1)}, \mathbf{H}_t^{(2)})$ as in Figure 3 and Equation 4. Given only a benign history $\{\mathbf{H}_t\}_{t=1}^{T_{\text{train}}}$, the objective is to learn a detector f_Θ that, for any future t, maps $\mathbf{H}_t$ to an anomaly decision and a localization set:

$$f_\Theta : \ \mathbf{H}_t \longmapsto \big(A(t),\, \mathcal{Z}_t\big). \tag{3}$$

Here $A(t) \in \{0, 1\}$ and $\mathcal{Z}_t \subseteq \mathcal{X}^{k^\star(t)}$, where $k^\star(t)$ is the smallest rank exhibiting a deviation (check Equations 10–12). The detector should ***(i)*** operate without attack labels (unsupervised); ***(ii)*** provide multi-level localization (tag, interaction, PLC zone); and ***(iii)*** scale across plants with heterogeneous hardware and documentation.

Figure 3 sketches how this objective is obtained in the remainder of the section: Section 3.2 builds the 1-skeleton $\mathcal{X}^{(1)}$, Section 3.3 defines per-rank cochains from the live tag stream, and Section 3.4 specifies the TNN encoder–decoder and thresholding that implement f_Θ.

3.2. Constructing 1-Cells (Edges)

In **`TMS-ICS`**, 0-cells (tags) and 2-cells (PLC zones) follow the Purdue hierarchy and remain fixed, but 1-cells (edges) must be inferred. Because pairwise tag relations are often undocumented or evolve over time, we evaluate three regimes that balance expert effort and statistical discovery.

Manual (Expert-Driven). Edges (i, j) are derived from Piping and Instrumentation (P&ID) diagrams and ladder logic when a change in tag i causally influences tag j within a control horizon. This yields high semantic fidelity and clear interpretability but does not scale to plants with many tags or limited documentation.

Semi-Automatic (State-Space Cross-Prediction). For each target tag j, we fit a local state-space template on benign data [51]. Predictors $i \neq j$ whose fitted coefficients exceed a significance threshold imply edges $i \to j$. This approach reduces manual workload and retains a physics-aligned bias but can take hours to days when mining large tag sets.

Fully Automatic (Embedding-Based Adjacency). Inspired by the *embedding warm-up* phase of GDN [15], a lightweight auto-encoder is trained on benign traces to learn one embedding vector per tag by minimizing a reconstruction loss over its entire time series. After convergence, the learned embeddings are fixed; pairwise cosine similarities are computed, and each tag retains its top-k nearest neighbors as incoming edges. The procedure requires no plant knowledge and runs in minutes, though the purely statistical criterion can introduce spurious links and thus offers the lowest interpretability among the three regimes.

All three regimes yield a complete 1-skeleton $\mathcal{X}^{(1)}$ on top of the unchanged PLC-zone 2-skeleton $\mathcal{X}^{(2)}$. We recompute the incidence and adjacency matrices once, then train **`TMS-ICS`** with an identical hyperparameter schedule in every setting. In Section 4 (***RQ3***), we evaluate the impact of each edge-construction strategy on detection performance and precision.

3.3. Modeling Module

Figure 3 illustrates the pipeline. ***Step 0*** provides the static tag list and PLC map for the plant. In ***Step 1***, we encode this hierarchy as a combinatorial complex $\mathcal{X} = (\mathcal{X}^0, \mathcal{X}^1, \mathcal{X}^2)$ in which $\mathcal{X}^0$ contains all sensors and actuators, $\mathcal{X}^1$ captures their cause-effect links, and $\mathcal{X}^2$ represents fixed PLC zones. ***Step 2*** records a benign multivariate stream $\mathbf{S} = \{\mathbf{s}(t)\}_{t=1}^{T} \in \mathbb{R}^N$; no attack labels are used at the training stage. During ***Step 3***, each data point with timestamp t is mapped to rank-specific cochains, as follows:

$$\mathbf{H}_t^{(k)} : \mathcal{X}^k \ \longrightarrow \ \mathbb{R}^d, \qquad k \in \{0, 1, 2\}, \tag{4}$$

with the default choices of features; ***(i)*** **0-cochains**: normalized tag values; ***(ii)*** **1-cochains**: concatenation of the two endpoint values; and ***(iii)*** **2-cochains**: mean and variance of all tags inside the PLC zone. These cochains turn the static complex into a learnable signal that feeds the TNN in

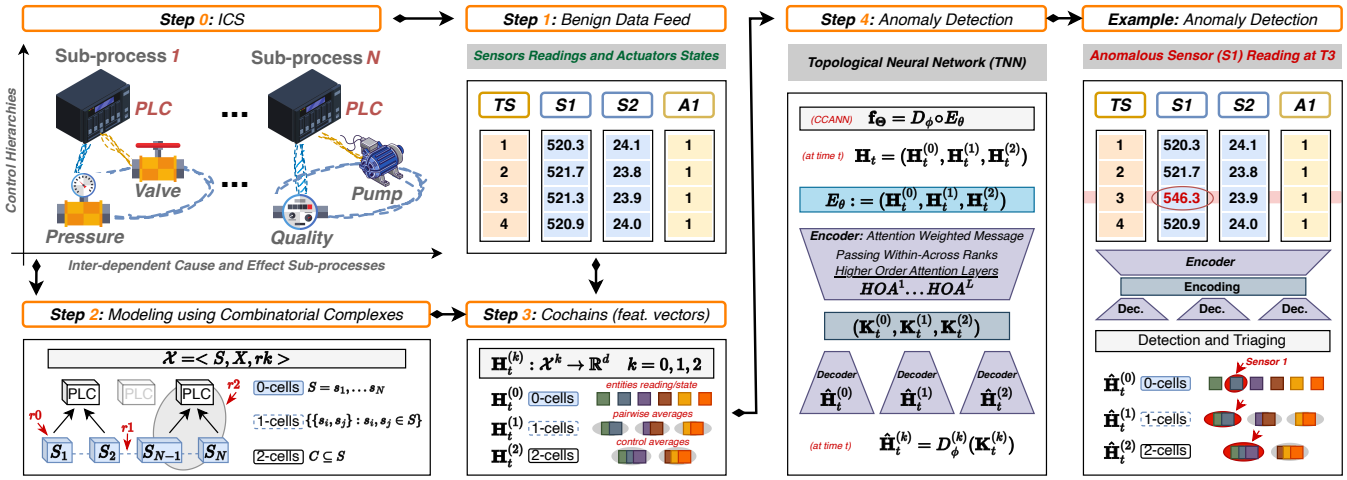


Figure 3: **TMS-ICS pipeline. Step 0** shows a typical ICS with PLC-supervised subprocesses; **Step 1** encodes devices, pairwise links, and PLC zones as a combinatorial complex $\mathcal{X}$; **Step 2** ingests the live tag stream; **Step 3** forms rank-specific cochains $\mathbf{H}_t^{(0)}, \mathbf{H}_t^{(1)}, \mathbf{H}_t^{(2)}$; **Step 4** passes the cochains through the two-layer TNN (Section 2.2, Equation 2) and compares reconstruction residuals with validation thresholds to flag anomalies. The right-hand panel illustrates a detected anomaly and its localization across the three ranks.

Step 4. Developers may swap alternative per-rank features, for example, sliding-window statistics or domain-specific edge filters, without modifying the neural architecture. A thorough exploration of such options is left for future work.

3.4. Intrusion-Detection Module

Following Section 2.1, **TMS-ICS** adopts a *behavior-based* design: training relies solely on benign traces, which are typically abundant, whereas labeled attacks are scarce and plant-specific. Accordingly, we train the model to *reconstruct* normal cochains $\mathbf{H}_t$ (Equation 4), and declare an anomaly whenever the reconstruction error at any rank exceeds a validation calibrated threshold. This self-supervised objective has two practical benefits: ***(i)*** it removes the need for costly, potentially dangerous attack injection, and ***(ii)*** it naturally aligns with the hierarchical PLC/tag structure because errors can be measured at every level and immediately interpreted by operators.

Encoder-decoder architecture. We factor the detector into an *encoder* E_θ that captures cross-rank dependencies and a family of lightweight, rank-specific *decoders* $D_\phi^{(k)}$ that regenerate the original signals:

$$f_\Theta = D_\phi \circ E_\theta = \left(D_\phi^{(0)}, D_\phi^{(1)}, D_\phi^{(2)}\right) \circ E_\theta, \quad \Theta = \{\theta, \phi\}. \tag{5}$$

Encoder. E_θ is a TNN (Section 2.2) built from the HOA message-passing operator introduced in Section 2.2. Each HOA layer receives the triple of cochains $\left(\mathbf{H}_t^{(0)}, \mathbf{H}_t^{(1)}, \mathbf{H}_t^{(2)}\right)$, mixes information *within* each rank via the adjacency matrices (A_0, A_1, A_2), and *across* ranks via the incidence matrices (B_1, B_2). The outcome is a hierarchy-aware latent representation defined as follows:

$$E_\theta : (\mathbf{H}_t^{(0)}, \mathbf{H}_t^{(1)}, \mathbf{H}_t^{(2)}) \longmapsto (\mathbf{K}_t^{(0)}, \mathbf{K}_t^{(1)}, \mathbf{K}_t^{(2)}), \tag{6}$$

where each $\mathbf{K}_t^{(k)}$ embeds not only local features but also multi-hop, multi-rank context (e.g. how a sensor influences its PLC zone through intermediate edges).

Decoders. For every rank $k \in \{0, 1, 2\}$, we attach a two-layer MLP $D_\phi^{(k)}$ that maps the latent vectors back to the feature space of that rank:

$$\hat{\mathbf{H}}_t^{(k)} = D_\phi^{(k)}\left(\mathbf{K}_t^{(k)}\right). \tag{7}$$

Rank-specific decoders serve two purposes: ***(i)*** they let each level learn its own reconstruction nuances (e.g. numeric sensors, binary actuators, PLC aggregates); and ***(ii)*** they yield rank resolved residuals that we later exploit for precise localization.

Training Objective. We minimize a rank-balanced mean-squared error over benign data using the below loss function:

$$\mathcal{L}(\Theta) = \sum_{k=0}^{2} w_k \left\| \hat{\mathbf{H}}_t^{(k)} - \mathbf{H}_t^{(k)} \right\|_2^2, \quad w_k \propto \frac{1}{|\mathcal{X}^k|}. \tag{8}$$

Here, each weight w_k is chosen proportional to the inverse of the number of k-cells, ensuring that tags, edges, and PLC zones each contribute equally to the overall loss regardless of their relative counts.

Inference and Anomaly Flagging. At each timestamp t, we compute the reconstruction error for every cell as follow;

$$e_t^{(k)}(x) = \left\| \hat{\mathbf{H}}_t^{(k)}(x) - \mathbf{H}_t^{(k)}(x) \right\|_2, \quad x \in \mathcal{X}^k. \tag{9}$$

We search from the top of the hierarchy downward (PLC zones, edges, tags) for the highest rank whose error exceeds its threshold τ_k calculated on benign validation errors:

$$k^\star(t) = \min\{\, k \in \{2, 1, 0\} \mid \exists\, x \in \mathcal{X}^k : e_t^{(k)}(x) > \tau_k \}. \tag{10}$$

We also define the binary anomaly indicator as follows:

$$A(t) = \mathbb{1}[k^\star(t) \geq 0]. \tag{11}$$

For localization we return the set of offending cells as defined below:

$$\mathcal{Z}_t = \begin{cases} \{\, x \in \mathcal{X}^{k^\star(t)} \mid e_t^{(k^\star(t))}(x) > \tau_{k^\star(t)} \}, & A(t) = 1, \\ \varnothing, & A(t) = 0. \end{cases} \tag{12}$$

This *top-down* strategy guarantees that a zone level disturbance is reported once, at the PLC rank, rather than as a flood of correlated low-level alerts, while still permitting drill-down capabilities to individual edges or devices when they only deviate.

3.5. Training and Deployment

A continuous benign trace $\mathbf{S} = \{\mathbf{s}(t)\}_{t=1}^{T}$ is first collected during normal operation and split chronologically into an *80%* training portion $\mathcal{D}_{\text{train}}$ and a *20%* validation portion $\mathcal{D}_{\text{val}}$. The HOA topological neural network is optimized on $\mathcal{D}_{\text{train}}$ with Adam, (learning rate $5{\times}10^{-3}$, weight decay 10^{-5}); early stopping halts training once the loss $\mathcal{L}(\Theta)$ in Equation 8 ceases to decrease on $\mathcal{D}_{\text{val}}$. Because each HOA layer propagates messages both *within* a rank and *across* ranks via the incidence matrices, information diffuses through the PLC-tag hierarchy very quickly. In practice, we observe that on all three benchmarks (Section 4) the validation loss flattens and the point-wise F_1 improves by at most 2 percentage points after the first full pass over the data. We therefore train for two epochs by default (the training script allows more, but they make no significant difference on the datasets we studied).

Threshold calibration. For every rank k, we gather the validation set residuals using the following equation:

$$\mathcal{E}_{\text{val}}^{(k)} = \left\{\, e_t^{(k)}(x) \;\middle|\; x \in \mathcal{X}^k,\ t \in \mathcal{D}_{\text{val}} \right\}. \tag{13}$$

Since fewer than 1% of benign observations are expected to deviate substantially from the learned manifold, the detection threshold is set to the 99$^{\text{th}}$ percentile, $\tau_k = \text{perc}_{99}(\mathcal{E}_{\text{val}}^{(k)})$. This simple, static rule yields a false-positive rate of at most 1% on the validation set and introduces no extra hyperparameters. Exploring more sophisticated post-training scoring or adaptive cutoffs is orthogonal to the modeling contribution and left for future work. If a different thresholding policy is desired, the per-rank thresholds τ_0, τ_1, τ_2 can be recomputed from validation residuals without retraining the network.

Site Rollout. Deploying **TMS-ICS** at a new facility involves four steps; ***(i)*** Derive the CC's 0-cells and 2-cells directly from PLC-zone diagrams; ***(ii)*** Choose an edge-construction regime (manual, semi-automatic, or fully automatic) and build the 1-skeleton $\mathcal{X}^{(1)}$; and ***(iii)*** Train the HOA TNN for one to three epochs on a small benign dataset; and ***(iv)*** Calibrate the thresholds τ_0, τ_1, τ_2 on held-out validation data.

4. Evaluation and Results

4.1. Research Question Themes

To rigorously validate **TMS-ICS** for practical ICS deployments, we examine five research question (RQ) themes:

RQ1 Theme: Detection Performance. How does the detection accuracy of **TMS-ICS**, measured at regular interval timestamps, compare to: ***(i)*** graph-centric neural models capturing pairwise tag interactions; ***(ii)*** sequence-oriented neural models treating ICS data purely as multivariate time series; and ***(iii)*** classical statistical and physics-based detectors, when all methods utilize simple, static thresholds calibrated solely on benign validation data?

RQ2 Theme: Comparison with Data-Mining Approaches. Considering specialized invariant- and state-space mining methods (PASAD's SVD residuals [9], MIDAS invariant mining [18], and GeCo state-space mining [51]), how does **TMS-ICS**'s detection performance and scalability compare, particularly given that these data-mining approaches typically require extensive offline preprocessing lasting hours or even days?

RQ3 Theme: Impact of Edge-Construction Regime. Given that the 1-skeleton $\mathcal{X}^{(1)}$ can be constructed manually, semi-automatically, or fully automatically (Section 3.2), how does the degree of automation and domain knowledge incorporated into the edge-inference process influence detection performance, false-alarm rates, and operational practicality?

RQ4 Theme: Role of Hierarchical Structure. (i) How critical is the inclusion of PLC zones (2-cells)? Specifically, how does the performance degrade when PLC loops are removed, leaving only tags (0-cells) and pairwise edges (1-cells)? ***(ii)*** How does performance differ if the reconstruction is restricted to tag-level data (0-cells) while retaining the full complex encoder? and ***(iii)*** In zone-coordinated (multi-component) attacks, does PLC-level aggregation enable earlier and more reliable detection than node/edge signals alone?

RQ5 Theme: Hierarchical Localization. How accurately *and how quickly after an alarm* can **TMS-ICS** pinpoint the root cause across the tag/edge/PLC hierarchy? ***(i)*** Does having dedicated decoders for ranks 0, 1, 2 improve localization relative to a single, flat decoder or attribution methods? ***(ii)*** Within a limited window of w control cycles after the first alarm, how does localization accuracy vary with the window length, and what is the best early-triage duration; and ***(iii)*** Does an operator-facing visualization of informed rank-wise residuals streamline analysis and improve root-cause localization in practice?

4.2. Experimental Setup

Datasets. We initially evaluate **TMS-ICS** on two established and publicly available ICS intrusion-detection benchmarks from the water-treatment and water-distribution domain:

The **SWaT** [22] dataset comprises data from a six-stage scaled-down water-treatment plant equipped with 51 sensors

and 11 actuators. The trace contains seven days of normal operation followed by four days with 31 cyber-physical attack scenarios (13 multi-asset, 18 single-asset), officially split into training and test sets.

The **WADI** [6] dataset is recorded from a three-stage urban water-distribution system controlled by PLCs and RTUs. It provides 16 h of benign data and a second trace with 15 sequential cyber-physical attacks, including sensor spoofing and unauthorized actuator commands. We train on the benign portion and test on the full attack trace. We additionally analyze WADI Attack 4 (Consumers zone), a zone-coordinated manipulation where six motorized valves oscillate in lock-step; we use it as a targeted probe of PLC-level aggregation (Section 4.4).

We further leverage the Tennessee Eastman Process (**TEP**) [11] dataset to assess cross-domain *generalizability* beyond water systems. Although TEP simulates a chemical plant, its multi-unit process architecture, sensing/actuation infrastructure, and hierarchical controller structure closely mirror generic ICS configurations with interconnected control loops. Following the anomaly-generation methodology of Fung *et al.* [20], we execute the public MATLAB simulator and inject single-variable manipulations during normal operations. For this cross-domain evaluation, we examine three representative attacks targeting distinct process components: *A Feed*, *Reactor Level*, and *Stripper Underflow*. Critically, **TMS-ICS** receives no domain-specific adaptation; the architecture, training regimen, and anomaly detection thresholds calibrated on benign data remain identical to those used for SWaT and WADI. Consequently, TEP serves as a generalized ICS stress test rather than a plant-specific case study, functioning solely as a transferability assessment check (see Section 4.6).

Model configuration. Unless stated otherwise, **TMS-ICS** uses two HOA layers. Each layer maps rank-specific inputs to 32-channel hidden states and outputs 64-D latent vectors per rank. We report: ***(i)*** the full 0/1/2-decoder model; ***(ii)*** a graph-only variant without 2-cells; and ***(iii)*** a 0-cell-only decoder model.

Training, inference, and threshold calibration. All models are trained for two epochs with Adam (learning rate 5×10^{-3}, weight decay 10^{-5}). For training and inference, we downsample each dataset to 10-second intervals. Detection thresholds (τ_0, τ_1, τ_2) are set to the 99th percentile of rank-wise residuals on the validation split, targeting a $\approx 1\%$ false-positive rate on benign data.

Implementation details. We implement **TMS-ICS** in Python using the TopoX library [25] for CC structures and HOA layers. All models are trained on a single NVIDIA RTX A6000 GPU (CUDA 12.2, 48 GB VRAM). Detection (inference) experiments run on an Intel Core i9-10900K CPU.

Online deployment and operational efficiency. In live operation, **TMS-ICS** runs on a read-only mirror of the control network switch. On an Intel i9-10900K, a complete forward pass (encoder plus three decoders) over a 10 s polling window requires approximately 70 ms, well-within the 250 ms cycle time of modern SCADA systems [43]. This performance permits real-time intrusion detection without specialized hardware.

4.3. Evaluation Metrics

Detection performance. ICS traces are predominantly benign, with attacks rare and transient. We report Precision, Recall, and F_1 at the timestamp level. All detection thresholds are chosen from a benign-only validation split to avoid dependence on labeled attacks.

Per-rank, per-attack F_1. For selected attacks, we also report rank-specific F_1 within the official attack window. For each rank $k \in \{0, 1, 2\}$, a timestamp is a positive if any k-cell exceeds τ_k; precision/recall are computed against the attack window and summarized as F_1 for that rank.

Effective Operator Hierarchical Localization (EOHL). To assess how well **TMS-ICS** localizes anomalies across hierarchy levels (tags, edges, PLC zones), we define $EOHL_K$, a multi-level generalization of the standard top-K localization metric. Let $\mathcal{D}$ be the set of timestamps at which an alarm is raised. For each $t \in \mathcal{D}$, let $c_t^{(1)}, \ldots, c_t^{(K)}$ be the top-K cells ranked by reconstruction error. Let T_t denote the true faulty device, E_t its incident edges, $\mathrm{Anc}(T_t)$ its parent PLC, and N_t^{above} the set of neighboring devices whose errors exceed the threshold. As such, we define the equation as follows:

$$\mathrm{EOHL}_K = \frac{1}{|\mathcal{D}|} \sum_{t \in \mathcal{D}} \mathbf{1}\Big[\exists\, k \le K :\ c_t^{(k)} \in \mathcal{C}_t\Big], \tag{14}$$

$$\mathcal{C}_t = \{\, T_t \cup N_t^{\mathrm{above}},\ E_t,\ \mathrm{Anc}(T_t)\,\}. \tag{15}$$

The three elements of $\mathcal{C}_t$ correspond to device-level, edge-level, and PLC-level matches, respectively. EOHL_K is the fraction of alarms whose top-K ranked cells contain a correct match at any level of the hierarchy. Restricted to rank-0 cells, Equation 14 reduces to the standard top-K device-localization score.

Time-constrained EOHL (tEOHL). Operators often need the root cause within the first few control cycles. Let $\mathcal{T} = \{t^i_{\mathrm{start}}\}$ be the set of alarm-onset timestamps. For a given window length w, we define:

$$W_w(t_0) = \{\, t_0,\ t_0 + 1,\ \ldots,\ t_0 + w - 1\,\}. \tag{16}$$

Let C_s be the "correct" set at time s (the union of the true faulty device, its incident edges, and its parent PLC). The time-constrained metric $\mathrm{tEOHL}_{K,w}$ is represented as follows:

$$\frac{1}{|\mathcal{T}|} \sum_{t_0 \in \mathcal{T}} \mathbf{1}\Big[\exists\, s \in W_w(t_0), \exists\, k \le K :\ c_s^{(k)} \in C_s\Big]. \tag{17}$$

A larger $\mathrm{tEOHL}_{K,w}$ indicates faster and more actionable localization within the first w samples after the alarm begins.

4.4. TMS-ICS Detection Experiments (RQ1-4)

Table 1 reports timestamp-level Precision, Recall, and F_1 on SWaT and WADI under a common calibration protocol:

TABLE 1: Point-wise detection on SWaT and WADI (Precision/Recall/F_1, in %). All learning methods use a static threshold chosen on a benign-only validation split. Bold indicates the best *within each group*. Baseline numbers are from IPAL [52] and GeCo [51].

Method	Datasets					
	SWaT			WADI		
	Precision	Recall	F_1	Precision	Recall	F_1
Learning-based (static threshold)						
TMS-ICS	**96.74**	**71.89**	**82.49**	62.63	**43.38**	**51.25**
GDN [15]	71.97	69.08	70.49	**82.94**	10.82	19.14
Seq2SeqNN [30]	44.00	10.90	17.50	44.40	13.40	20.50
PASAD [9]	32.40	71.50	44.60	16.40	23.90	19.50
Data-mining / template-based						
Invariant (MIDAS) [18]	**97.30**	69.10	80.80	90.00	21.90	35.20
GeCo [51]	94.80	**79.00**	**86.20**	**92.60**	**32.10**	**47.70**

thresholds are selected only on benign validation data, with no access to attack labels. The aim is to compare *modeling paradigms*: ***(i)*** learning-based detectors with simple static thresholds; and ***(ii)*** mining/template methods that first derive invariants or state-space relations offline.

RQ1 Theme: Detection performance. On SWaT, **TMS-ICS** attains an F_1 of 82.49%, outperforming learning baselines such as GDN (70.49%), Seq2Seq LSTM (17.50%), and PASAD (44.60%). On WADI, **TMS-ICS** achieves 51.25% F_1, again ahead of evaluated learning baselines (GDN 19.14%, Seq2Seq 20.50%, PASAD 19.50%). These results demonstrate the benefit of hierarchy-aware modeling since thresholding is deliberately kept simple and identical across methods. A qualitative, per-rank analysis of a zone-coordinated event (WADI Attack 4) further illustrates why PLC-level aggregation matters (Section 4.4).

RQ2 Theme: Comparison with data-mining approaches. Mining and template methods encode stronger physics priors but incur substantial offline cost. On SWaT, GeCo reaches the highest F_1 (86.20%), with MIDAS close behind (80.80%); **TMS-ICS** is competitive at 82.49%. On WADI, **TMS-ICS** leads with 51.25% F_1, surpassing GeCo (47.70%) and MIDAS (35.20%). GeCo's training/mining time is reported as $\sim$15.1 hours on SWaT and $\sim$46.9 days on WADI before detection runs [51], whereas **TMS-ICS** trains in minutes on commodity GPU (Section 4.2). Thus, while mining methods are informative references, they are not strictly comparable in deployment effort and scalability; **TMS-ICS** matches or exceeds their accuracy on these benchmarks with far lower preparation time.

RQ3 Theme: Impact of Edge-Construction Regime. The tags (0-cells) and PLC zones (2-cells) within **TMS-ICS** remain fixed, directly reflecting the plant's hierarchical structure. However, edges (1-cells) must be inferred from data, documentation, or domain knowledge (Section 3.2). To quantify how different edge-construction strategies affect detection, we evaluate three regimes: *manual* edges, derived directly from P&ID; *semi-automatic* edges, inferred through state-space cross-prediction; and *automatic* edges, based on embedding similarity. We keep all other experimental conditions (architecture, training protocol, thresholds) identical, and report precision, recall, and F_1 scores in Table 2.

TABLE 2: Effect of edge-construction regime on detection performance (Precision/Recall/F_1, in %). Only the 1-skeleton ($\mathcal{X}^{(1)}$) changes; all other settings remain fixed. Precision also serves as an indicator of false-alarm propensity.

Edges in $\mathcal{X}^{(1)}$	Datasets					
	SWaT			WADI		
	Precision	Recall	F_1	Precision	Recall	F_1
Manual (expert-derived)	**96.74**	71.89	**82.49**	53.34	38.73	44.87
Semi-automatic (state-space)	89.06	73.50	80.54	**67.15**	28.31	39.83
Automatic(embedding-based)	75.96	**76.60**	76.28	62.63	**43.38**	**51.25**

The edge-construction strategy exerts a demonstrable and significant influence on performance metrics.

For SWaT, manual edges yield the highest precision (97.67%) and F_1 score (82.49%). This confirms that using carefully derived edges based on detailed process knowledge (P&ID diagrams) results in fewer false alarms and reliable detection. The semi-automatic approach offers comparable F_1 (80.54%) with slightly higher recall, but precision suffers due to denser and potentially spurious inferred connections. Automatic edge inference achieves balanced precision and recall but still trails behind the manual method, likely due to embedding-derived correlations that do not always reflect causal process dependencies.

For WADI, automatic edges produce the best overall F_1 score (51.25%). Given that WADI's public documentation provides less detailed PLC and tag relationships, manual edge inference misses essential couplings, leading to lower recall (38.73%) and weaker overall performance. Automatic inference here identifies broader statistical dependencies and enhances recall substantially. Semi-automatic edges, while achieving the highest precision (67.15%), suffer notably lower recall (28.31%), indicating a tendency toward overly conservative modeling that excludes critical cross-zone or transient attack signatures.

These results point to key trade-offs in operational deployment. Manual edges deliver superior interpretability and precision, making them ideal when accurate, detailed system documentation is available and manageable in scale. Semi-automatic inference significantly reduces expert involvement and maintains high precision but demands substantial computational effort and offline tuning. Automatic inference, requiring no domain expertise and offering rapid deployment, is advantageous for large-scale or insufficiently documented plants, at the expense of interpretability and somewhat higher false-alarm rates. The choice of edge-construction regime directly influences detection effectiveness and false-alarm management. For systems with accurate and accessible documentation, expert-derived edges are strongly recommended. Conversely, automatic edge inference emerges as the best compromise in poorly documented settings, with semi-automatic inference positioned as a viable intermediate option where computational resources and precise modeling are feasible.

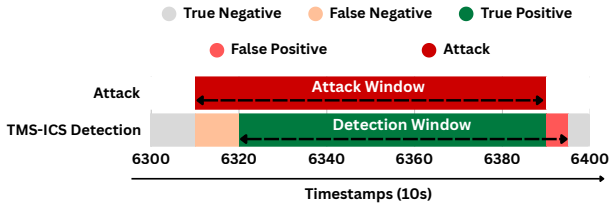
TABLE 3: RQ4: Effect of hierarchy on SWaT (Precision/Recall/F_1, in %).

Architecture	Precision	Recall	F_1
Full hierarchy (enc+dec 0/1/2)	**96.74**	**71.89**	**82.49**
Graph-only (enc+dec 0/1)	90.05	64.22	74.98
Single-decoder (enc 0/1/2, dec 0)	92.94	68.68	78.99

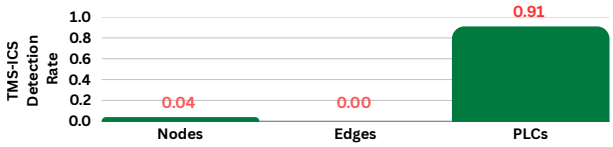
RQ4 Theme: Role of hierarchical structure. Following ***RQ3***, we isolate the contribution of the PLC-level representation and rank-wise decoding on detection performance. For this experiment and all subsequent analyses (except for ***RQ4.iii***), we restrict the evaluation exclusively to the SWaT dataset and the *manual* edge-construction regime described in Section 3.2 until stated otherwise. SWaT provides a richer and more varied set of attack scenarios compared to WADI, enabling more robust statistical conclusions. Additionally, SWaT explicitly defines PLC zones aligned closely with the standard Purdue hierarchy, allowing direct and interpretable evaluation of hierarchical modeling. Using manually constructed edges based on SWaT's detailed P&ID diagrams further enhances interpretability, ensuring that performance variations observed are directly attributable to the hierarchical structure and decoding strategy, rather than artifacts introduced by automated edge inference methods. We report standard Precision, Recall, and F_1 metrics computed at each timestamp on the official SWaT test split.

Removing the PLC-level representation (2-cells) significantly reduces precision (by approximately 6.69 pp) and overall F_1 (by approximately 7.51 pp) relative to the full hierarchical model as observed in Table 3. This degradation highlights the importance of incorporating PLC-loop context for discriminating normal operations from anomalies. Meanwhile, the single-decoder variant, which reconstructs only tag-level data (0-cells) while maintaining the full hierarchical encoder, demonstrates competitive performance, preserving recall to within 2 points of the full model and achieving only a slight F_1 score degradation. This indicates that the encoder effectively embeds hierarchical context into device-level features, even without explicitly reconstructing higher-order cells. Nevertheless, to achieve fine-grained localization across tags, their interactions, and PLC zones, explicit reconstruction at all ranks is essential; only the complete 0/1/2-decoder model provides such detailed attribution capabilities (further explored in ***RQ5***).

RQ4.iii: Zone Coordinated Attack (WADI - Attack 4). This WADI case study complements the SWaT ablations above.[3] At the *Consumers* zone, the attacker forces all six motorized consumer-line valves $(2\_MCV\_101\|201\|301\|401\|501\|601)$ to oscillate between CLOSE and a partly open state. The synchronized actuation induces intermittent bursts at downstream flow meters (2_FQ_*) and large pressure swings across the zone. This is a *zone-coordinated*, actuator-driven attack: control-output (CO) bits flip before flows settle, individual sensor traces lag, and multiple devices move in lock-step. A hierarchy-agnostic detector tends to emit scattered, weak tag/edge alerts (or none) rather than a decisive, zone-level alarm. PLC-level aggregation (2-cells) should detect the *joint* disturbance early and cleanly.



(a) Attack/Detection window.



(b) Hierarchical detection.

Figure 4: **WADI Attack 4 (Consumers zone).** Panel (a) shows the attack and detection windows for a coordinated manipulation of multiple consumer valves within the same PLC zone. Panel (b) summarizes **TMS-ICS** hierarchical detection performance, where the PLC-level decoder achieves a high detection rate while node- and edge-level decoders rarely fire.

We use the same model and calibration as in Section 4.2, with the automatic (embedding-based) edge regime that gave the best overall WADI F_1 in Table 2. Thresholds τ_0, τ_1, τ_2 are set on benign validation data (99th percentile per rank). We report the *per-rank, per-attack* F_1 over the official attack window (Section 4.3).

Figure 4a shows that the **TMS-ICS** fires almost immediately for the coordinated manipulation of six consumer valves in the same PLC zone and sustains high detection quality within the attack window. Figure 4b confirms this quantitatively: the PLC-level decoder attains $F_1 = 0.938$, whereas the node- and edge-level decoders remain largely sub-threshold ($F_1 = 0.035$ and 0.000, respectively). Together, the two panels show that PLC-level aggregation consolidates synchronous component perturbations into a single decisive alarm with minimal false positives.

RQ4.iii. implication. This example reveals a failure mode of flat or pairwise models: when a coordinated actuation pattern perturbs many components slightly and simultaneously, no single tag or edge accumulates enough evidence at the right time. Explicit PLC-level aggregation in **TMS-ICS** captures the shared variance induced by the attack and surfaces it early, aligning with the quantitative drop we observed when removing 2-cells in Table 3.

3. Unlike the SWaT ablations, the WADI case study employs the automatic (embedding-based) edge regime of Table 2; architecture, training schedule, and benign-only per-rank thresholding are otherwise identical to Section 4.2.

TABLE 4: Localization accuracy (EOHL_K) on SWaT. △ denotes hierarchy-based methods; — denotes attribution-based methods.

Method	Type	EOHL_1	EOHL_2	EOHL_5	EOHL_{10}
TMS-ICS *(0-1-2-cells decoders)*	△	**0.503**	**0.586**	**0.586**	**0.639**
TMS-ICS *(0-cells decoders)*	—	0.380	0.430	0.450	0.482
FM [10]	—	0.068	0.072	0.184	0.327
SHAP [34]	—	0.063	0.070	0.213	0.300
LEMNA [23]	—	0.062	0.084	0.204	0.320
LIME [41]	—	0.000	0.000	0.102	0.161

TABLE 5: Responsive localization accuracy ($\text{tEOHL}_{K,t}$) on SWaT.

Time (t)	K=1	K=2	K=3	K=4	K=5	K=10
1 (immediate)	0.60	0.60	0.60	0.60	0.60	**0.70**
5	0.60	0.60	0.60	0.60	0.60	0.65
10	0.58	0.58	0.58	0.58	0.58	0.63
20	0.54	0.54	0.54	0.54	0.54	0.59

4.5. TMS-ICS Localization Experiments (RQ5)

RQ5.i: Hierarchical Localization. As discussed in Section 5, precise anomaly localization remains a major challenge for deep-learning-based IDSs. Fung *et al.* [20] found that conventional attribution methods frequently fail to correctly pinpoint compromised sensors or actuators due to their black-box nature and lack of underlying bias structure.

To evaluate how **TMS-ICS**'s explicit hierarchical design addresses this challenge, we measure localization performance on SWaT using EOHL_K, a hierarchy-aware metric (Section 4.3). Table 4 compares the full three-decoder **TMS-ICS** model against ***(i)*** its 0-decoder ablation (reducing localization to a flat, attribution-like setup); and ***(ii)*** widely-used model-agnostic attribution methods.

Findings. The 0-decoder ablation of **TMS-ICS**, despite reconstructing only the device level (0-cells), already substantially outperforms popular attribution methods (EOHL_5: 45% vs. <22%), demonstrating the effectiveness of embedding hierarchical structure within its latent representations. Adding explicit decoders for interaction-level (1-cells) and PLC-zone-level (2-cells) further increases EOHL_5 to 58.6%. These results clearly highlight the value of explicitly encoding ICS hierarchy to solve the longstanding problem of poor localization in deep-learning-based ICS intrusion detection.

RQ5.ii: Responsive Localization. As highlighted by Fung *et al.* [20], a critical factor limiting practical localization in deep-learning IDSs is the delay between attack onset and accurate localization. To evaluate how rapidly **TMS-ICS** localizes root causes, we employ the responsive localization metric ($\text{tEOHL}_{K,t}$; Section 4.3), which quantifies correct identification within the first t time steps after an initial alert. Results in Table 5 show tEOHL for different values of K (top-ranked cells) and time windows t.

Findings. **TMS-ICS** correctly localizes the root cause of 70% of attacks on the very first SCADA cycle (t=1) when considering the top-10 ranked cells, and achieves 60% accuracy for top 5- localization under the same stringent timing condition. As the response window expands (up to 20 steps), accuracy declines only slightly (to 59% at top-10), underscoring the robustness of hierarchical features in maintaining accurate localization despite propagation effects. Such prompt and robust localization is critical for operators, providing actionable insights immediately upon anomaly detection, and effectively addresses a key practical limitation identified in prior studies [20]. These experiments strongly validate the importance of incorporating explicit hierarchical structure into deep-learning models for ICS cybersecurity. **TMS-ICS**'s hierarchical architecture substantially enhances localization accuracy, directly addressing key limitations identified in prior work while delivering practical operational benefits through rapid root-cause identification of anomalies.

RQ5.iii: Localization case study. To demonstrate how the rank-wise reconstruction error visualization (Equation 12) translates into actionable operational awareness, we analyze a SWaT attack in which the pH analyzer AIT202 in the chemical dosing stage is covertly spoofed. Figure 5 overlays per–cell reconstruction errors on the plant diagram and highlights only those cells whose errors exceed their rank–specific thresholds, which keeps the display uncluttered and easy to read.

Scenario context and immediate impact. At attack onset, **TMS-ICS** surfaces three clear signals above threshold: the supervising PLC (PLC2) shows elevated error ($\bar{e} = 0.22$) that pins down the incident to the dosing stage; the process interaction AIT201–P202 is flagged ($\bar{e} = 21.79$), indicating collateral disruption in NaCl dosing; and the falsified sensor AIT202 exhibits the largest deviation ($\bar{e} = 109.48$), identifying the root cause. All unaffected devices and interactions remain visually subdued, so attention naturally focuses on the relevant cells.

Operator response path. From this single screen an operator can isolate AIT202 and switch dosing to manual, inspect the AIT201-P202 linkage for secondary effects, and monitor the rest of PLC2 to confirm that the anomaly remains contained. The visualization therefore supports rapid triage from a high level PLC alert down to the precise compromised device. A second case study (flow-meter spoofing with downstream propagation) appears in *Appendix B*.

4.6. TMS-ICS Generalizability Experiments

As emphasized in Related Work (Section 5), an IDS must *generalize* across heterogeneous plants and industries [38]; a capability rarely demonstrated in practice. To rigorously assess whether our hierarchy-aware design transfers beyond the water domain, we apply **TMS-ICS** *without core modeling modification* to TEP, a widely-adopted benchmark that simulates a realistic chemical plant with fundamentally different process physics (continuous reactor control vs. discrete actuator sequences), control objectives (product quality and reaction kinetics vs. water level and flow regulation), and operational dynamics. We inject three representative

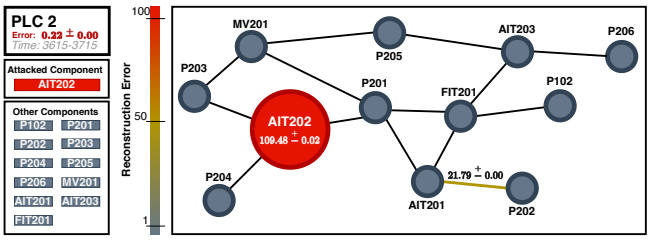


Figure 5: Hierarchical reconstruction-error visualization for the AIT202 sensor-spoofing case. Color intensity reflects normalized reconstruction error for cells exceeding their thresholds; node size and edge thickness scale with the mean error. The view highlights the affected PLC, the abnormal process link, and the spoofed sensor, enabling immediate root–cause identification.

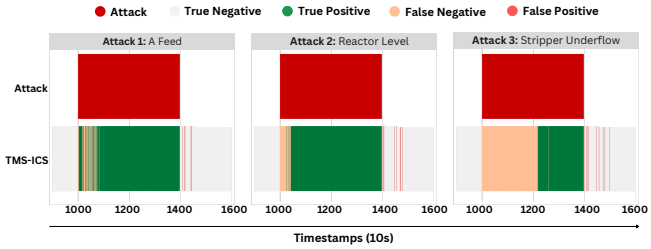


Figure 6: **Cross-domain generalizability to `TEP` (chemical process).** Each panel shows a full run with a single synthetic attack (bottom band = ground-truth window; top band = **`TMS-ICS`** outcomes at each timestamp). True positives concentrate within the attack windows with few false positives outside, using the same model and benign-only thresholding as in `SWaT/WADI`.

manipulations following Fung *et al.* [20] (recall Section 4.2 for dataset details), using the *identical* architecture, training procedure, and threshold calibration as for `SWaT` and `WADI`; no retraining, hyperparameter tuning, or domain-specific adaptation.

Figure 6 shows full-run timelines for each attack: the bottom band marks the ground-truth window; the top band renders **`TMS-ICS`** outcomes (TP, FN, TN, FP) at each timestamp. Across all three cases, true positives align tightly with attack windows and false positives remain sparse, demonstrating that **`TMS-ICS`** successfully detects coordinated manipulations in an out-of-domain process with zero additional tuning. This zero-shot transferability validates our core design principle from Section 5: the detector encodes *structural abstractions* of ICS, including field devices (0-cells), causal relationships (1-cells), and supervisory zones (2-cells), rather than domain-specific physics or signal patterns. This inductive bias captures control system *topology* that generalizes from water treatment and distribution networks to chemical processes and, by extension via `TEP`'s representativeness, to the broader landscape of continuous process control across critical infrastructure sectors.

5. Related Work

An industrial intrusion-detection system must warn operators of a cyber-attack while keeping false alarms low [7]. As explained in Section 2.1, behavior-based detectors that learn normal operation from benign traces are attractive in practice provided they achieve three goals: ***(i)*** *generalizability* across heterogeneous plants [38]; ***(ii)*** *localization* of the root cause [16], [20]; and ***(iii)*** hierarchy-aware comprehensibility, that is, the ability to explain an alert in terms of tags, their interactions, and the supervising PLC zone. Intrusion detection research for ICS has largely progressed along two complementary lines. *Knowledge-centered* methods embed an a-priori description of the plant, whereas *data-centered* methods rely on learning regular behavior directly from historical traces. Table 6 gives representative examples and marks whether a method is hierarchy-aware and whether it supports localization. We outline both strands and clarify how **`TMS-ICS`** builds on their strengths.

Knowledge-centered detectors. Early systems translate expert insight into *process invariants* that must always hold, for instance mass-balance rules in water treatment [1], [2]. Other work derives linear state-space or Kalman observers that track the residual between predicted and measured signals [5], [39], or expresses physical constraints as logic formulas for safety checks [40]. Such rules are easy for operators to read and often provide tag-level localization because each invariant is tied to a small set of variables. However, they omit the hierarchy linking sensors, interactions, and PLC zones, leaving topology implicit and limiting portability: any change in hardware, logic, or operating conditions needs fresh expert rules, which typically cover only part of the process.

Data-centered detectors. In contrast, data-driven detectors learn normal behavior directly from historical logs, significantly reducing the manual work involved.

Statistical Methods. Basic statistical techniques, such as correlation analysis [3] and singular value decomposition methods like `PASAD` [9], are simple to implement but limited. They mainly detect anomalies in individual sensors or pairs, failing to capture more complex interactions.

Data mining. `MIDAS` mines tag-level invariants from historical traces, whereas `GeCo` constructs local state-space models for each sensor and actuator [18], [51]. Although both approaches improve comprehensibility, they remain hierarchy-agnostic and their offline mining phase can take hours or even days; because the search is non-exhaustive, extending the models to larger plants is not easily scalable [51].

Deep sequence models. More advanced approaches employ deep learning, including sequence-to-sequence LSTMs and variational autoencoders [17], [30], to identify complex patterns in multivariate data. However, these models often function as black boxes, making it difficult for operators to identify the exact source of anomalies [19].

Graph neural networks. `GDN` [15] models pairwise dependencies among sensors and improves tag-level localization, but still lacks an explicit PLC-zone layer.

TABLE 6: Representative IDS approaches for ICS anomaly detection. △ = hierarchy-aware, ⊙ = provides localization.

System / Paper	Detection model	Venue	△	⊙
Knowledge-centred				
Adepu *et al.* [2]	Process invariants	AsiaCCS	✗	✓
Ahmed *et al.* [5]	State-space model	AsiaCCS	✗	✓
Cárdenas *et al.* [13]	State-space model	AsiaCCS	✗	✓
Palleti *et al.* [39]	Kalman observer	JPC	✗	✓
Quinonez *et al.* [40]	Physical invariants	USENIX	✗	✓
Data-centred				
Aggarwal *et al.* [3]	Correlation analysis	CPS	✗	✗
Aoudi *et al.* [9]	SVD residual	CCS	✗	✓
Feng *et al.* [18]	Invariant mining	NDSS	✗	✓
Wolsing *et al.* [51]	State-space mining	USENIX	✗	✓
Kim *et al.* [30]	Seq2Seq NN	CyberICPS	✗	✗
Deng *et al.* [15]	GDN	AAAI	✗	✓
`TMS-ICS` (ours)	TNN	—	✓	✓

Open issue: localization. Deep-learning IIDSs can flag attacks but rarely pinpoint which specific sensor or actuator is compromised. Fung *et al.* [20] show that state-of-the-art attribution frequently misidentifies the root cause, citing two core problems: the lag between attack onset and alarm, and inconsistent attack signatures (particularly for binary actuators). This motivates detectors that embed the Purdue hierarchy as an inductive bias, aligning learned representations with tags, interactions, and PLC zones to enable precise multi-level localization without post-hoc attribution.

Gap addressed. Existing IIDSs rarely deliver interpretable alerts across all hierarchy levels. In this work, we address this gap by introducing **`TMS-ICS`** that models the plant as a CC where sensors/actuators form 0-cells, their relationships form 1-cells, and PLC zones form 2-cells. At the core of our framework, TNN learns normal behavior at each rank using only benign data. This provides a unique capability to pinpoint whether an anomaly (or potential attack) stems from a misbehaving tag, a suspicious interaction, or a compromised PLC zone. We obtain these multilevel, semantically grounded alerts solely from benign traces, using fully unsupervised training with no hand-crafted rules or attack annotations.

6. Discussion and Conclusion

This work tests a simple premise: mirroring the Purdue hierarchy inside the detector improves accuracy, lowers false alarms, and guides operators to the cause. Modeling sensors as 0-cells, their cause–effect links as 1-cells, and PLC subsystems as 2-cells provides a useful inductive bias; removing the PLC layer weakens localization, while keeping it enables concise, rank-aware alerts. Edge construction presents clear trade-offs: expert-derived P&ID edges maximize interpretability, embedding-based edges speed onboarding at the cost of transparency, and state-space mining sits between them. The hierarchy also supports prompt, actionable localization rather than scattered low-level alarms, and the approach transfers across domains when a correct connectivity abstraction is available.

Looking ahead, our proposed approach opens several opportunities in exploring alternative thresholding policies to reduce false positives; fusing process tags with network traffic as an additional data to capture coordinated network-physical attacks; and extending the topology beyond PLC zones to higher Purdue layers (for example, add 3-cells for area and 4-cells for site). Accordingly, **`TMS-ICS`**, our proposed framework, establishes a topology-aware foundation for advancing pragmatic ICS security.

Ethical Considerations

This work investigates a defensive intrusion detection approach for Industrial Control Systems (ICS) using publicly available datasets and simulators. No experiments were performed on live plants or production systems, and no human subjects or personally identifiable information were involved. Stakeholders include asset owners, operators, engineers, and vendors. Publishing detection methods can inform adversaries about what may be flagged; the benefits of transparency and reproducibility are judged to outweigh this risk since secrecy is not a reliable security control and community scrutiny enables improvement. The shared artifacts contain only detection and evaluation code and exclude plant-specific topologies, credentials, exploit tooling, or configurations that could enable unauthorized access or actuation. All evaluations use the `SWaT` and `WADI` datasets from the iTrust laboratory and the Tennessee Eastman Process simulator with publicly documented attack scenarios; these datasets contain process telemetry and no personal and sensitive data. Where licenses apply, access instructions are provided rather than redistributing raw traces, and preprocessing preserves original semantics without any attempt at deanonymization or enrichment. The detector operates in read-only mode on a mirrored tag stream and issues no control actions; deployment guidance recommends keeping a human in the loop and using conservative thresholds calibrated on benign validation data. The paper does not describe procedures for bypassing safety systems or crafting attacks; if future work reveals vulnerabilities in real installations, coordinated disclosure with affected parties will precede any public release. Training runs are short and inference is feasible in real time on commodity CPUs, so the compute and energy footprint is modest.

References

[1] Sridhar Adepu and Aditya Mathur. From design to invariants: Detecting attacks on cyber physical systems. In *2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pages 533–540. IEEE, 2017.

[2] Sridhar Adepu and Aditya Mathur. Distributed attack detection in a water treatment plant: Method and case study. *IEEE Transactions on Dependable and Secure Computing*, 18(1):86–99, 2018.

[3] Ekta Aggarwal, Mehdi Karimibiuki, Karthik Pattabiraman, and André Ivanov. Corgids: A correlation-based generic intrusion detection system. In *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy*, pages 24–35, 2018.

[4] Chuadhry Mujeeb Ahmed, Gauthama Raman MR, and Aditya P Mathur. Challenges in machine learning based approaches for real-time anomaly detection in industrial control systems. In *Proceedings of the 6th ACM on cyber-physical system security workshop*, pages 23–29, 2020.

[5] Chuadhry Mujeeb Ahmed, Martin Ochoa, Jianying Zhou, Aditya P Mathur, Rizwan Qadeer, Carlos Murguia, and Justin Ruths. Noiseprint: Attack detection using sensor and process noise fingerprint in cyber physical systems. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pages 483–497, 2018.

[6] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P Mathur. Wadi: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks*, pages 25–28, 2017.

[7] Bushra A Alahmadi, Louise Axon, and Ivan Martinovic. 99% false positives: A qualitative study of {SOC} analysts' perspectives on security alarms. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2783–2800, 2022.

[8] Tejasvi Alladi, Vinay Chamola, and Sherali Zeadally. Industrial control systems: Cyberattack trends and countermeasures. *Computer Communications*, 155:1–8, 2020.

[9] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. 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*, pages 817–831, 2018.

[10] Aleksandar Avdalović, Joseph Khoury, Ahmad Taha, and Elias Bou-Harb. Enhancing network security management in water systems using fm-based attack attribution. In *NOMS 2025-2025 IEEE Network Operations and Management Symposium*, pages 1–10. IEEE, 2025.

[11] Andreas Bathelt, N Lawrence Ricker, and Mohieddine Jelali. Revision of the tennessee eastman process model. *IFAC-PapersOnLine*, 48(8):309–314, 2015.

[12] Srikanth Bellamkonda. Ransomware attacks on critical infrastructure: A study of the Colonial Pipeline incident. *International Journal of Research in Computer Applications and Information Technology*, 7(2):1423–1433, 2024.

[13] Alvaro A Cárdenas, Saurabh Amin, Zong-Syun Lin, Yu-Lun Huang, Chi-Yen Huang, and Shankar Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM symposium on information, computer and communications security*, pages 355–366, 2011.

[14] Markus Dahlmanns, Johannes Lohmöller, Jan Pennekamp, Jörn Bodenhausen, Klaus Wehrle, and Martin Henze. Missed opportunities: measuring the untapped tls support in the industrial internet of things. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, pages 252–266, 2022.

[15] Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4027–4035, 2021.

[16] Sandro Etalle. From intrusion detection to software design. In *European Symposium on Research in Computer Security*, pages 1–10. Springer, 2017.

[17] Daniel Fährmann, Naser Damer, Florian Kirchbuchner, and Arjan Kuijper. Lightweight long short-term memory variational auto-encoder for multivariate time series anomaly detection in industrial control systems. *Sensors*, 22(8):2886, 2022.

[18] Cheng Feng, Venkata Reddy Palleti, Aditya Mathur, and Deeph Chana. A systematic framework to generate invariants for anomaly detection in industrial control systems. In *NDSS*, pages 1–15, 2019.

[19] Clement Fung, Shreya Srinarasi, Keane Lucas, Hay Bryan Phee, and Lujo Bauer. Perspectives from a comprehensive evaluation of reconstruction-based anomaly detection in industrial control systems. In *European Symposium on Research in Computer Security*, pages 493–513. Springer, 2022.

[20] Clement Fung, Eric Zeng, and Lujo Bauer. Attributions for ML-based ICS anomaly detection: From theory to practice. In *Proceedings of the 31st Network and Distributed System Security Symposium (NDSS '24)*. The Internet Society, 2024.

[21] Jairo Giraldo, David Urbina, Alvaro Cardenas, Junia Valente, Mustafa Faisal, Justin Ruths, Nils Ole Tippenhauer, Henrik Sandberg, and Richard Candell. A survey of physics-based attack detection in cyber-physical systems. *ACM Computing Surveys (CSUR)*, 51(4):1–36, 2018.

[22] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. A dataset to support research in the design of secure water treatment systems. In *International conference on critical information infrastructures security*, pages 88–99. Springer, 2016.

[23] Wenbo Guo, Dongliang Mu, Jun Xu, Purui Su, Gang Wang, and Xinyu Xing. Lemna: Explaining deep learning based security applications. In *proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 364–379, 2018.

[24] Dina Hadžiosmanović, Robin Sommer, Emmanuele Zambon, and Pieter H Hartel. Through the eye of the plc: semantic security monitoring for industrial processes. In *Proceedings of the 30th Annual Computer Security Applications Conference*, pages 126–135, 2014.

[25] Mustafa Hajij, Mathilde Papillon, Florian Frantzen, Jens Agerberg, Ibrahem AlJabea, Rubén Ballester, Claudio Battiloro, Guillermo Bernárdez, Tolga Birdal, Aiden Brent, et al. Topox: a suite of python packages for machine learning on topological domains. *Journal of Machine Learning Research*, 25(374):1–8, 2024.

[26] Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Vasileios Maroulas, and Xuanting Cai. Simplicial complex representation learning. *arXiv preprint arXiv:2103.04046*, 2021.

[27] Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, Karthikeyan Natesan Ramamurthy, Tolga Birdal, Tamal K Dey, Soham Mukherjee, Shreyas N Samaga, et al. Topological deep learning: Going beyond graph data. *arXiv preprint arXiv:2206.00606*, 2022.

[28] Zhongkai Hao, Songming Liu, Yichi Zhang, Chengyang Ying, Yao Feng, Hang Su, and Jun Zhu. Physics-informed machine learning: A survey on problems, methods and applications. *arXiv preprint arXiv:2211.08064*, 2022.

[29] Nguyen Xuan Hoang, Nguyen Viet Hoang, Nguyen Huu Du, Truong Thu Huong, Kim Phuc Tran, et al. Explainable anomaly detection for industrial control system cybersecurity. *IFAC-PapersOnLine*, 55(10):1183–1188, 2022.

[30] Jonguk Kim, Jeong-Han Yun, and Hyoung Chun Kim. Anomaly detection for industrial control systems using sequence-to-sequence neural networks. In *International Workshop on the Security of Industrial Control Systems and Cyber-Physical Systems*, pages 3–18. Springer, 2019.

[31] Moshe Kravchik and Asaf Shabtai. Detecting cyber attacks in industrial control systems using convolutional neural networks. In *Proceedings of the 2018 workshop on cyber-physical systems security and privacy*, pages 72–83, 2018.

[32] Dominik Kus, Eric Wagner, Jan Pennekamp, Konrad Wolsing, Ina Berenice Fink, Markus Dahlmanns, Klaus Wehrle, and Martin Henze. A false sense of security? revisiting the state of machine learning-based industrial intrusion detection. In *Proceedings of the 8th ACM on Cyber-Physical System Security Workshop*, pages 73–84, 2022.

[33] Olav Lamberts, Konrad Wolsing, Eric Wagner, Jan Pennekamp, Jan Bauer, Klaus Wehrle, and Martin Henze. Sok: Evaluations in industrial intrusion detection research. *arXiv preprint arXiv:2311.02929*, 2023.

[34] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[35] MITRE. Spoof reporting message (att&ck for ics technique t0856), 2024. https://attack.mitre.org/techniques/T0856/.

[36] MITRE. Unauthorized command message (att&ck for ics technique t0855), 2024. https://attack.mitre.org/techniques/T0855/.

[37] MITRE. Data manipulation (att&ck technique t1565), 2025. https://attack.mitre.org/techniques/T1565/.

[38] Neil Ortiz, Martin Rosso, Emmanuele Zambon, Jerry den Hartog, and Alvaro A Cardenas. From power to water: dissecting scada networks across different critical infrastructures. In *International conference on passive and active network measurement*, pages 3–31. Springer, 2024.

[39] Venkata Reddy Palleti, Yu Chong Tan, and Lakshminarayanan Samavedham. A mechanistic fault detection and isolation approach using kalman filter to improve the security of cyber physical systems. *Journal of Process Control*, 68:160–170, 2018.

[40] Raul Quinonez, Jairo Giraldo, Luis Salazar, Erick Bauman, Alvaro Cardenas, and Zhiqiang Lin. {SAVIOR}: Securing autonomous vehicles with robust physical invariants. In *29th USENIX security symposium (USENIX Security 20)*, pages 895–912, 2020.

[41] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[42] SANS Institute. Introduction to ics security part 2: The purdue model, 2021. https://www.sans.org/blog/introduction-to-ics-security-part-2.

[43] Schneider Electric (Australia) Pty. Ltd. *CitectSCADA v7.20 User Guide*, 2010. Page scan-time parameter: 250 ms default. https://manualzz.com/doc/32934841/citectscada-user-guide.

[44] L Shuaiyi, Kai Wang, Liren Zhang, and Bailing Wang. Process-oriented heterogeneous graph learning in gnn-based ics anomalous pattern recognition. *Pattern Recognition*, 141:109661, 2023.

[45] Keith Stouffer, Keith Stouffer, Michael Pease, CheeYee Tang, Timothy Zimmerman, Victoria Pillitteri, Suzanne Lightman, Adam Hahn, Stephanie Saravia, Aslam Sherule, et al. Guide to operational technology (ot) security. 2023.

[46] Zhiwen Tian, Ming Zhuo, Leyuan Liu, Junyi Chen, and Shijie Zhou. Anomaly detection using spatial and temporal information in multivariate time series. *Scientific Reports*, 13(1):4400, 2023.

[47] David I Urbina, Jairo A Giraldo, Alvaro A Cardenas, Nils Ole Tippenhauer, Junia Valente, Mustafa Faisal, Justin Ruths, Richard Candell, and Henrik Sandberg. Limiting the impact of stealthy attacks on industrial control systems. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1092–1105, 2016.

[48] U.S. Department of Energy. Topic paper 4-14: Purdue model framework for industrial control systems & cybersecurity segmentation, 2022. https://www.energy.gov/sites/default/files/2022-10/Infra_Topic_Paper_4-14_FINAL.pdf.

[49] U.S. Department of Homeland Security. Homeland threat assessment 2025. https://www.dhs.gov/sites/default/files/2024-10/24_0930_ia_24-320-ia-publication-2025-hta-final-30sep24-508.pdf, 2024. Accessed 2025-08-27.

[50] Konrad Wolsing, Eric Wagner, Frederik Basels, Patrick Wagner, and Klaus Wehrle. Deployment challenges of industrial intrusion detection systems. In *European Symposium on Research in Computer Security*, pages 453–473. Springer, 2024.

[51] Konrad Wolsing, Eric Wagner, Luisa Lux, Klaus Wehrle, Martin Henze, and FKIE Fraunhofer. Gecos replacing experts: Generalizable and comprehensible industrial intrusion detection. In *USENIX Security*, 2025.

[52] Konrad Wolsing, Eric Wagner, Antoine Saillard, and Martin Henze. Ipal: breaking up silos of protocol-dependent and domain-specific industrial intrusion detection systems. In *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 510–525, 2022.

Appendix A. Topological Preliminaries

Definition A.1 (Combinatorial complex)**.** A *combinatorial complex* (CC) is a triple $\langle S, \mathcal{X}, \mathrm{rk}\rangle$ where ***(i)*** S is a finite set and $\{s\} \in \mathcal{X}$ for all $s \in S$; ***(ii)*** $\emptyset \notin \mathcal{X} \subseteq \mathcal{P}(S)$ is a family of nonempty subsets, called *cells*; and ***(iii)*** $\mathrm{rk} : \mathcal{X} \to \mathbb{Z}_{\geq 0}$ is order-preserving, i.e., $x \subseteq y \Rightarrow \mathrm{rk}(x) \leq \mathrm{rk}(y)$. Cells of equal rank k form $\mathcal{X}^k$ (the k-cells). Rank is a structural label, not the cardinality of a cell.

Definition A.2 (Cochain space)**.** For a CC $\mathcal{X}$ and feature dimension d, the *k-cochain space* is

$$C^k(\mathcal{X}, \mathbb{R}^d) = \{\, H^k : \mathcal{X}^k \to \mathbb{R}^d \,\}.$$

Elements of C^k are *k-cochains*, i.e., feature vectors attached to k-cells.

Definition A.3 (Cochain maps via incidence)**.** Incidence relations define linear maps between cochain spaces. For $r < k$, the matrix $B^{r,k}$ induces

$$B^{r,k} : C^k(\mathcal{X}) \to C^r(\mathcal{X}),$$

which aggregates signals from rank k to rank r. (Adjacency/coadjacency define analogous same-rank maps.)

Definition A.4 (Adjacency neighborhoods)**.** Two cells of the same rank are *adjacent* if they share a higher-order coface. The k-adjacency neighborhood is

$$N_{a,k}(x) = \{\, y \in \mathcal{X} \mid \mathrm{rk}(x) = \mathrm{rk}(y),\ \exists\, z \in \mathcal{X} \text{ s.t. } \mathrm{rk}(z) = \mathrm{rk}(x) + k,\ x \subsetneq z,\ y \subsetneq z \,\}.$$

Definition A.5 (Coadjacency neighborhoods)**.** Two cells of the same rank are *coadjacent* if they share a lower-order face. The k-coadjacency neighborhood is

$$N_{co,k}(x) = \{\, y \in \mathcal{X} \mid \mathrm{rk}(x) = \mathrm{rk}(y),\ \exists\, z \in \mathcal{X} \text{ s.t. } \mathrm{rk}(z) = \mathrm{rk}(x) - k,\ z \subsetneq x,\ z \subsetneq y \,\}.$$

Definition A.6 (Incidence neighborhoods)**.** Two cells are *incident* if one is a proper subset of the other. For $k \geq 1$:

$$N_{\uparrow,k}(x) = \{\, y \in \mathcal{X} \mid x \subsetneq y,\ \mathrm{rk}(y) = \mathrm{rk}(x) + k \,\},$$

$$N_{\downarrow,k}(x) = \{\, y \in \mathcal{X} \mid y \subsetneq x,\ \mathrm{rk}(y) = \mathrm{rk}(x) - k \,\}.$$

Definition A.7 (Combinatorial Complex Neural Network (CCNN))**.** Let $\mathcal{X}$ be a CC. Let $\mathcal{C}^{i_1} \times \cdots \times \mathcal{C}^{i_m}$ and $\mathcal{C}^{j_1} \times \cdots \times \mathcal{C}^{j_n}$ be Cartesian products of cochain spaces on $\mathcal{X}$. A *CCNN* is a function

$$\mathsf{CCNN} :\ \mathcal{C}^{i_1} \times \cdots \times \mathcal{C}^{i_m} \to \mathcal{C}^{j_1} \times \cdots \times \mathcal{C}^{j_n},$$

mapping input cochains $(\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(m)})$ to output cochains $(\mathbf{K}^{(1)}, \ldots, \mathbf{K}^{(n)})$.

Definition A.8 (Higher-order attention on CCs)**.** Let $\mathcal{X}$ be a CC, $\mathcal{N}$ a neighborhood function on $\mathcal{X}$, and $\mathcal{Y}_0$ a sub-CC. Let $\mathcal{N}(\mathcal{Y}_0) = \{\mathcal{Y}_1, \ldots, \mathcal{Y}_{|\mathcal{N}(\mathcal{Y}_0)|}\}$ be the neighboring sub-CCs. A *higher-order attention* assigns weights $a(\mathcal{Y}_0, \mathcal{Y}_i) \in [0, 1]$ with $\sum_i a(\mathcal{Y}_0, \mathcal{Y}_i) = 1$, enabling the network to focus on the most relevant neighborhoods across ranks.

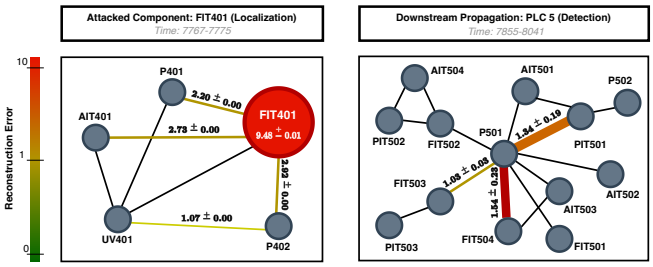


Figure 7: SWaT case: FIT401 spoofing. The visualization first localizes the compromised flow meter FIT401 (left), then surfaces its downstream impact on PLC_5 approximately ninety seconds later (right).

Appendix B. Supplementary Case Study

Figure 7 illustrates a second qualitative example that complements the main text. The attack begins with spoofing of the flow meter FIT401. Within ten samples, the rank-wise reconstruction error map elevates FIT401 above threshold, yielding immediate device-level localization without additional tooling. As the disturbance propagates along the process, the same visualization subsequently highlights increased reconstruction error within the next stage (P501, PLC_5), making the downstream effect explicit. This example shows that the hierarchy-aware display not only pinpoints the initial culprit but also tracks how anomalies migrate across stages, providing operators with a clear, single-view depiction of both origin and propagation.