Trust-Aware Vertical Intrusion Detection for IoT via Evolutionary Graph Neural Networks

Myria Bouhaddi[1,*], Kamel Adi[1]

[1]Computer Security Research Laboratory, Université du Québec en Outaouais, Quebec, Canada

Abstract

The rapid proliferation of IoT technologies has fundamentally reshaped smart infrastructures, spanning from domestic applications to complex industrial systems. However, this layered and geographically distributed architecture introduces multifaceted security challenges, particularly in identifying coordinated or stealthy threats that propagate across different levels of the system. Traditional intrusion detection systems (IDS), typically deployed in isolation either at the edge or within the cloud, struggle to achieve the necessary global visibility and often exhibit delayed responsiveness in such dynamic environments. In this paper, we propose a novel vertical intrusion detection framework tailored to multi-layered IoT networks. Our system enables hierarchical correlation and fusion of alerts across edge, fog, and cloud layers by combining localized anomaly detection with a global graph neural network (GNN) that performs structural and temporal reasoning over an alert graph. To ensure robustness in adversarial settings, we introduce an evolutionary game-theoretic mechanism based on replicator dynamics, which dynamically adjusts the trust levels of inter-node connections. The nodes evaluate the reliability of their neighbors based on accuracy and detection consistency, leading to reinforcement of cooperative behaviors and marginalization of compromised participants. Experimental results show that our architecture improves detection performance, reduces false positives, and adapts effectively to changing threat landscapes.

Keywords

Vertical Intrusion Detection System, Graph Neural Networks (GNN), Internet of Things (IoT), Evolutionary Game Theory

1. Introduction

The proliferation of the Internet of Things (IoT) has profoundly transformed the technological landscape, allowing traditional infrastructures to evolve into smart, interconnected ecosystems. From smart homes and telemedicine to industrial automation and intelligent transportation systems, IoT technologies are now the foundation for critical services across a wide range of sectors. These environments typically rely on a *multi-layered architecture* comprising edge devices, fog nodes, and centralized cloud platforms. This hierarchical structure allows the system to balance trade-offs between latency, computational resources, and storage capabilities [1].

However, this layered architecture also introduces complex security challenges. The distributed nature of IoT leads to *fragmented local observations*, often limited in scope and granularity, which can hinder accurate and timely threat detection. Particularly concerning are *coordinated and stealthy attacks* that exploit the lack of global visibility. An adversary may launch seemingly benign activities at various layers, such as low-frequency anomalies at the edge or subtle misconfigurations at intermediate fog nodes, which appear innocuous in isolation but, when viewed holistically, reveal a sophisticated attack pattern [2]. Moreover, traditional intrusion detection systems (IDS), whether signature-based or anomaly-based, are typically designed for centralized or flat infrastructures. These systems lack the capacity to correlate alerts between layers, reason over distributed information, or adapt to the dynamic trustworthiness of participating entities [3]. Therefore, several recent efforts have sought to address these challenges by distributing IDS functionality across IoT environments [4, 5, 6]. However, most such approaches are based on *horizontal coordination*, where nodes at the same level share

C&ESAR 2025: 32nd Computer & Electronics Security Application Rendezvous

*Corresponding author.

myria.bouhaddi@uqo.ca (M. Bouhaddi); kamel.adi@uqo.ca (K. Adi)

https://uqo.ca/profil/bouhmy01 (M. Bouhaddi); https://uqo.ca/profil/adixka01 (K. Adi)

0000-0002-9941-550X (M. Bouhaddi); 0000-0003-2869-0333 (K. Adi)

alerts or models without leveraging the vertical structure inherent to IoT stacks. This often results in missed opportunities to aggregate semantically related events across the edge-fog-cloud continuum. In addition, these systems commonly assume that local detection agents are *fully trustworthy*, ignoring the possibility of compromised, malfunctioning, or misconfigured nodes that can distort the global analysis. In adversarial settings, such assumptions can make the IDS vulnerable to poisoning or misinformation, degrading both detection accuracy and system robustness.

To overcome these limitations, we propose a new **Vertical Intrusion Detection System (V-IDS)** that is natively designed for the layered nature of IoT infrastructures. Our contributions are threefold:

- We introduce a *graph neural network (GNN)*-based fusion architecture that enables the aggregation and correlation of alerts across the vertical layers of the IoT stack. This design supports both *structural* and *temporal reasoning*, allowing the system to detect distributed and evolving attack patterns that would remain hidden in layer-isolated analyzes.
- We incorporate an *evolutionary game-theoretic mechanism* based on replicator dynamics to continuously adjust *influence (edge weights)* of participating nodes within the GNN. This mechanism evaluates the historical performance and consistency of each node, reinforcing contributions from reliable agents while gradually reducing the impact of untrustworthy or compromised participants.
- We demonstrate that the proposed system significantly improves detection performance and resilience through extensive simulations. Our results highlight the advantages of combining hierarchical alert fusion with trust-based adaptive reasoning under adversarial conditions.

By integrating multi-layered visibility with adaptive trust modeling, V-IDS offers a scalable and robust solution for intrusion detection in complex and dynamic IoT environments. Addresses key limitations of existing systems by enabling global threat inference from fragmented and potentially unreliable local data, a crucial step toward securing the next generation of smart infrastructures.

2. Related work

2.1. Intrusion detection systems for IoT

Intrusion detection systems are essential for monitoring and detecting malicious activity in networked environments [7]. Traditional IDS approaches can be broadly classified as signature-based or anomaly-based. Although signature-based systems are effective against known threats, they struggle with zero-day attacks. Anomaly-based systems, on the other hand, can detect novel patterns but often suffer from high false positive rates.

In the context of IoT, these traditional IDS approaches face significant limitations. IoT environments are characterized by high device heterogeneity, resource constraints, and decentralized architectures. Several works have proposed lightweight or distributed IDS for edge or fog computing layers [8, 9]. However, these systems typically operate in isolation or rely on flat (horizontal) peer coordination, lacking mechanisms to reason across hierarchical layers. As a result, they are ineffective at detecting multi-stage or stealthy attacks that span across edge, fog, and cloud layers.

2.2. Graph-based IDS approaches

Graph-based learning has emerged as a powerful paradigm in intrusion detection, particularly with the adoption of Graph Neural Networks (GNNs) due to their ability to model structured data, such as communication graphs, process dependency graphs, and interaction networks. GNNs enable the propagation and aggregation of information across connected entities, making them well-suited to capture the relational patterns and spatial correlations inherent in networked systems.

In the context of IDS, GNNs have been applied to various tasks: *GCN-IDS* [10] leverages graph convolutional networks to detect intrusions by modeling communication patterns between hosts; *DeepTrack* [11] models lateral movement attacks through dynamic graphs of host behaviors; and

GADIN [12] captures device-to-device interactions for anomaly detection in IoT networks. Other studies have proposed temporal GNNs to incorporate time-evolving features of attack propagation [13], and heterogeneous graph structures to differentiate between types of nodes and edges [14].

Despite their effectiveness, most of the existing GNN-based IDS approaches suffer from structural limitations. They often assume a static, centralized, or flat topology, which does not reflect the layered and distributed nature of modern IoT environments. Node interactions are typically assigned uniform or fixed edge weights, without taking into account dynamic trust levels or reliability, limiting the system's ability to adapt to adversarial conditions.

Moreover, few existing approaches explicitly address the **vertical nature of IoT architectures**, where alert signals and contextual information flow from edge devices to fog nodes and cloud services. Without modeling this hierarchical structure and the corresponding dependencies between layers, such systems lack the ability to perform **holistic threat reasoning** and struggle to maintain robustness in the face of distributed, stealthy, or coordinated attacks [15].

2.3. Trust and weighting mechanisms in distributed detection

Trust-aware computing and dynamic weighting have been explored in distributed systems to mitigate the influence of unreliable nodes. Reputation-based schemes and credibility scores have been used in wireless sensor networks and federated learning [16]. Some works have also applied game-theoretic models to model cooperation among agents [17, 18], aiming to incentivize truthful reporting or penalize malicious behavior.

However, these mechanisms are often implemented outside of the learning process (e.g., pre-filtering nodes), or rely on global reputation tracking, which is impractical for dynamic, large-scale IoT settings. Very few works incorporate local, adaptive trust mechanisms directly into the message-passing operations of a GNN for intrusion detection.

The existing literature on intrusion detection in IoT environments highlights several important limitations. First, most IDS solutions do not take advantage of the inherent vertical structure of IoT architectures. Rather than leveraging the hierarchical organization of edge, fog, and cloud layers for multi-level reasoning, they often operate in a flat or isolated manner, limiting their capacity to detect complex, multi-stage threats. Second, graph-based approaches, including those based on Graph Neural Networks, generally rely on fixed edge weights and lack mechanisms to adapt to dynamic variations in node behavior or reliability. This static configuration renders them vulnerable in adversarial or noisy environments. Finally, while trust and reputation mechanisms have been studied in distributed systems, they are typically implemented externally to the learning model and rarely integrated into the core inference process of the IDS itself. In particular, to our knowledge, no existing approach incorporates localized evolutionary trust dynamics directly within the GNN message-passing framework.

In this work, we address these limitations by introducing a novel vertical IDS framework tailored to the hierarchical structure of IoT environments. Our approach performs multi-layered alert fusion via a graph neural network architecture that explicitly encodes the vertical communication pathways from edge to cloud. Unlike prior work that relies on static topologies and uniform weighting, we incorporate a dynamic, trust-aware weighting mechanism based on local evolutionary dynamics inspired by game theory. This mechanism enables each node to adjust the influence of its neighbors based on observed behavior over time, enhancing resilience to malicious or unreliable participants. As a result, our system can correlate fragmented and partial observations across different layers, suppress the propagation of misleading information, and maintain robustness against distributed and adaptive adversaries.

3. Background

3.1. IoT Stack: Edge, Fog, and Cloud

We consider a three-tier IoT architecture composed of layers *edge*, *fog*, and *cloud*. **Edge devices** resides closest to the physical environment: sensors, actuators, and user endpoints that collect raw telemetry

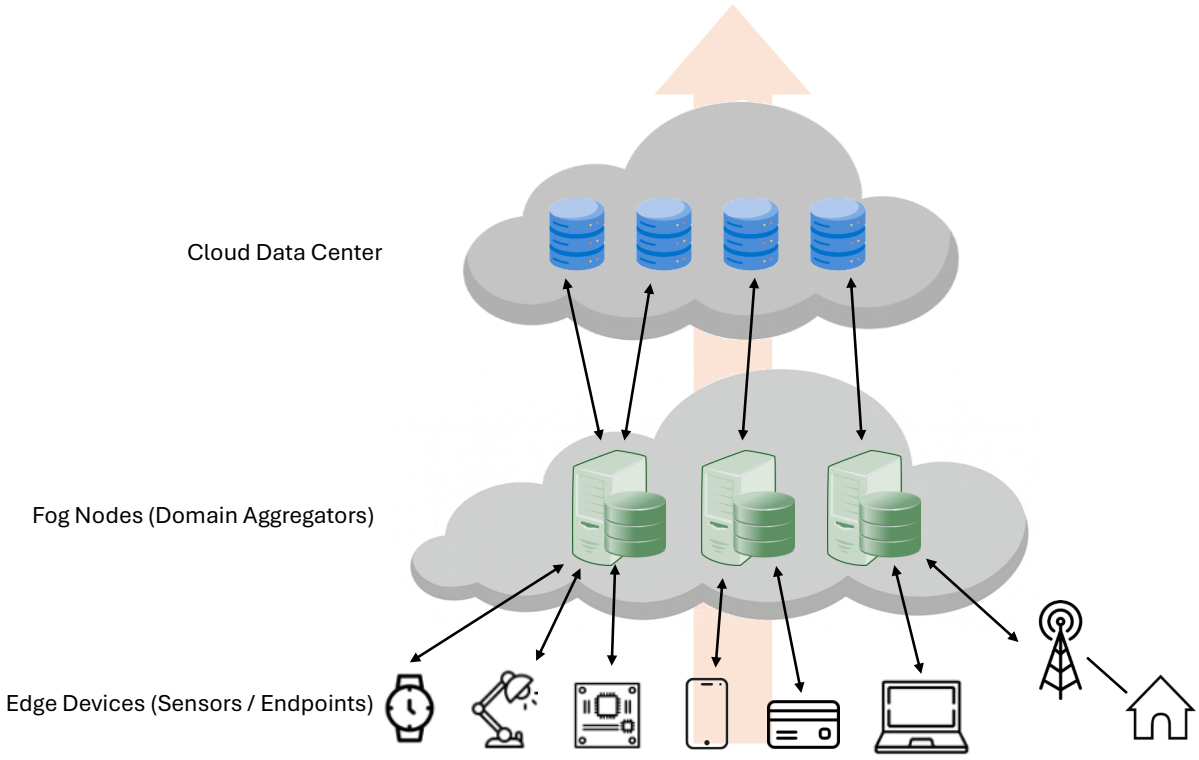

Figure 1: Cloud–Fog–Edge IoT stack with tier-wise aggregation and bidirectional information flow. Solid arrows indicate uplink telemetry and alerts (Edge → Fog → Cloud); downward arrows depict feedback such as policies or model updates.

and perform low-latency processing under constrained resources. **Fog nodes**, positioned between the edge and the cloud, serve as domain-level aggregators; they consolidate data from multiple edge devices, apply pre-processing and filtering, and enforce localized policies to reduce both latency and upstream traffic. The **cloud** layer offers elastic compute and storage across domains, maintains a long-term context, and executes global-scale analytics and learning models.

This vertical architecture is depicted in Figure 1, which highlights the bidirectional flow of information: uplink telemetry and alerts (solid arrows) propagate from edge to fog to cloud, while feedback, updated models and enforcement policies flow downstream.

Within this hierarchical structure, observations are inherently fragmented: edge nodes detect fine-grained but localized patterns; fog nodes uncover domain-level correlations; the cloud synthesizes cross-domain signals into global insights. Our IDS design takes advantage of this vertical segmentation in two synergistic ways. First, each tier performs an adaptive trust assessment by monitoring the consistency and quality of the alerts received from the tier below, adjusting its attention to upstream sources via an evolutionary weighting mechanism. Second, the cloud fuses multi-tier summaries using graph-based message passing within a GNN, enabling global threat detection through the integration of multi-hop, multi-layer dependencies otherwise undetectable by isolated nodes.

3.2. Evolutionary Game Theory: Fitness, Replicator, and ESS

Evolutionary Game Theory (EGT) emerged in the 1970s to explain adaptation in large populations [19]. Rather than perfectly rational players choosing the best responses once, EGT studies how the frequency of strategies changes over time under selection. Two cornerstone ideas are *evolutionarily stable strategy* (ESS), which formalizes resistance to invasion by rare mutants, and *replicator dynamic*, which links the growth rate of a strategy to its relative performance in the current population.

Consider a finite set of pure strategies $\mathcal{S} = \{1, \ldots, m\}$. A population state is a vector $x =$

$(x_1, \ldots, x_m)$ in the probability simplex $\Delta^{m-1} = \{x \in \mathbb{R}^m_{\geq 0} : \sum_i x_i = 1\}$, where x_i is the frequency of the strategy i.

In a symmetric two-player game with payoff matrix $A \in \mathbb{R}^{m \times m}$, the (expected) fitness of strategy i against population x is

$$\pi_i(x) \;=\; (Ax)_i, \qquad \bar{\pi}(x) \;=\; x^\top Ax \;\text{ (mean fitness).}$$

More generally, in a population game, one specifies a continuous fitness map $\pi : \Delta^{m-1} \to \mathbb{R}^m$, $x \mapsto (\pi_1(x), \ldots, \pi_m(x))$.

Replicator dynamic [20]. The continuous-time replicator equation evolves the state $x(t)$ according to payoff differences:

$$\dot{x}_i \;=\; x_i\big(\pi_i(x) - \bar{\pi}(x)\big), \qquad i = 1, \ldots, m.$$

It preserves the simplex ($x(t) \in \Delta^{m-1}$ for all t) and leaves the support of x invariant. A widely used multiplicative update in discrete time is

$$x_i^{(t+1)} \;=\; \frac{x_i^{(t)}\, \pi_i\big(x^{(t)}\big)}{\sum_k x_k^{(t)}\, \pi_k\big(x^{(t)}\big)},$$

which is the discrete analogue of the replicator flow.

Rest points and Nash equilibria [21]. A state x^* is a (symmetric) Nash equilibrium if every strategy in its support achieves maximum payoff: for all i with $x_i^* > 0$ and all j, $\pi_i(x^*) \geq \pi_j(x^*)$. All Nash equilibria are stationary for the replicator dynamic ($\dot{x} = 0$), although not all stationary points are Nash.

Evolutionarily Stable Strategy (ESS) [21]. A state $x^* \in \Delta^{m-1}$ is an ESS if it is a Nash equilibrium and is *resistant to invasion*: there exists $\varepsilon_0 > 0$ such that for any $y \neq x^*$ and all $0 < \varepsilon < \varepsilon_0$,

$$\pi\big(x^*, (1-\varepsilon)x^* + \varepsilon y\big) \;>\; \pi\big(y, (1-\varepsilon)x^* + \varepsilon y\big),$$

i.e., the incumbent earns strictly higher payoff than any nearby mutant when the population is mostly at x^*. In symmetric matrix games, the asymptotically stable rest point of the replicator dynamic; conversely, asymptotic stability of the replicator implies a refinement of Nash that excludes neutrally stable points (non-ESS).

Under replicator flow, the mean fitness $\bar{\pi}(x)$ is a Lyapunov function for potential games, and the dynamics is gradient-like with respect to the Shahshahani metric [21]. Thus, trajectories generically converge to invariant sets composed of Nash equilibria; ESSs are isolated attractors within these sets.

3.3. Graph Neural Networks

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a directed graph, where $\mathcal{V}$ is the set of nodes, with $|\mathcal{V}| = n$, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of directed edges (that is, ordered pairs (j, i) with an edge from j to i). Each node $i \in \mathcal{V}$ is associated with an input vector of characteristics $x_i \in \mathbb{R}^{d_0}$ and a representation $h_i^{(\ell)} \in \mathbb{R}^{d_\ell}$ in the layer ℓ of the network, initialized as $h_i^{(0)} = x_i$. We denote the in-neighborhood of node i by

$$\mathcal{N}(i) = \{\, j \in \mathcal{V} \mid (j, i) \in \mathcal{E} \,\},$$

i.e., the set of nodes with edges directed toward i.

A Graph Neural Network iteratively updates node representations through message passing over $\mathcal{G}$. At each layer ℓ, a node i receives messages from its neighbors $\mathcal{N}(i)$, aggregates them, and updates its state:

$$h_i^{(\ell+1)} = \phi^{(\ell)}\left(h_i^{(\ell)}, \bigoplus_{j \in \mathcal{N}(i)} \psi^{(\ell)}\left(h_i^{(\ell)}, h_j^{(\ell)}, e_{ij}\right)\right),$$

where $\psi^{(\ell)}$ is a message function, $\phi^{(\ell)}$ an update function, e_{ij} optional edge features, and $\oplus$ a permutation-invariant aggregator (for example, sum or mean) [22].

A common instance is the Graph Convolutional Network (GCN), which uses linear messages and symmetric normalization.

$$H^{(\ell+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(\ell)}W^{(\ell)}\right),$$

where $\tilde{A} = A + I$ is the adjacency matrix with self-loops, $\tilde{D}$ its degree matrix, $H^{(\ell)}$ the characteristics of the nodes in the layer ℓ, $W^{(\ell)}$ a learnable weight matrix and σ an activation function (for example, ReLU). This formulation ensures that each node combines its own features and those of its neighbors in a normalized way.

Weighted aggregation. In settings where edges carry reliability scores $w_{ij} \in [0, 1]$, message passing can be adapted to favor trustworthy sources. We define a weighted adjacency $A^{(w)}$ with entries $A^{(w)}_{ji} = w_{ij}$ for the edge $j \to i$, and apply the same normalization:

$$H^{(\ell+1)} = \sigma\left(\hat{D}^{-1/2}\hat{A}^{(w)}\hat{D}^{-1/2}H^{(\ell)}W^{(\ell)}\right),$$

where $\hat{A}^{(w)} = A^{(w)} + I$ and $\hat{D} = \mathrm{diag}(\hat{A}^{(w)}\mathbf{1})$.

Alternatively, we can row-normalize weights per target node:

$$\tilde{w}_{ij} = \frac{w_{ij}}{\sum_{k\in\mathcal{N}(i)} w_{ik} + \varepsilon}, \qquad h_i^{(\ell+1)} = \sigma\left(W_0 h_i^{(\ell)} + \sum_{j\in\mathcal{N}(i)} \tilde{w}_{ij}\, W h_j^{(\ell)}\right),$$

which makes the propagation scale invariant and interpretable as a reliability-weighted attention.

These weights w_{ij} can be learned (as in attention mechanisms) or externally provided, e.g., based on past performance, trust levels, or cross-layer confirmation. In our case, they are computed independently and used to modulate the information flow across the graph.

4. Proposed architecture

Although the concept of a multilayered IDS architecture has been outlined, the present study deliberately focuses on a two-layer Fog–Cloud model, which is sufficient to validate the concepts introduced while maintaining analytical tractability and facilitating simulation. Extending this framework to a three-layer Edge–Fog–Cloud architecture is feasible, for instance, by adopting a GNN-based approach in which nodes are organized into clusters forming subgraphs. In this structure, intra-cluster edges capture interactions among edge nodes, while inter-cluster edges represent interactions at the Fog level. However, such an extension would introduce additional complexity in the IDS design, which requires coordinated detection between edge, fog, and cloud nodes, along with increased computational overhead. This avenue will be explored in future work.

4.1. Overview of the Vertical IDS Framework

We consider a hierarchical IoT architecture composed of three layers: *edge*, *fog*, and *cloud*. Edge devices generate raw data, but lack the capacity for local intrusion analysis. Each fog node F_i supervises a subset $\mathcal{E}_i$ of edge devices and runs a lightweight intrusion detection module. Detected anomalies are forwarded to the cloud layer for further analysis.

At the cloud level, alerts are jointly analyzed using a GNN, where nodes represent fog entities, and edges encode their proximity. This enables the detection of coordinated or distributed intrusions by leveraging spatiotemporal dependencies.

To enhance robustness against unreliable or compromised fog nodes, we propose a *vertical intrusion detection system* that integrates two-level detection and trust feedback. Local fog detection is complemented by global cloud analysis, where trust scores are dynamically adjusted to reinforce reliable sources and downweight inconsistent ones.

The general architecture of our vertical intrusion detection system is illustrated in Figure 2. It shows the two-layer detection flow from edge to cloud, as well as the integration of spatio-temporal analysis and trust-based feedback.

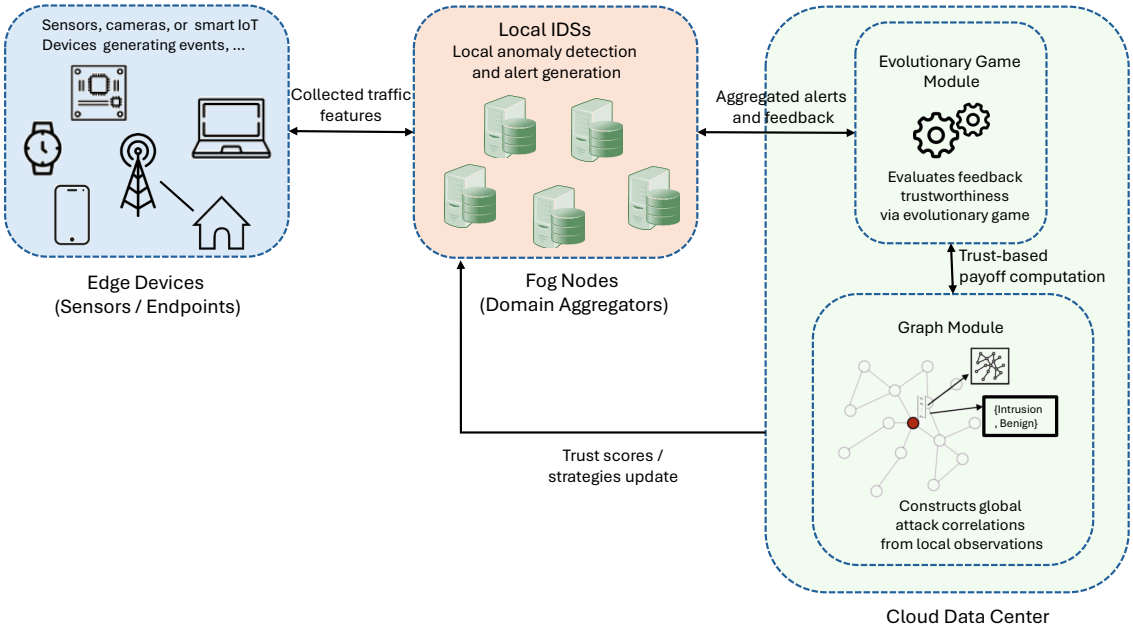

Figure 2: Vertical Intrusion Detection Architecture across Edge, Fog, and Cloud layers. Raw data generated at the edge is processed locally by fog-level IDS modules. Alerts are sent to the cloud for spatio-temporal graph-based analysis and trust-aware aggregation.

4.2. Local Fog Detection and Metric Reporting

In the fog layer, each node F_i is responsible for monitoring a subset of edge devices $\mathcal{E}_i$ and detecting signs of potential intrusions. Due to resource constraints, only lightweight analysis is performed locally. Rather than transmitting raw alerts to the cloud, each fog node computes a set of structured metrics designed to summarize the security context of its monitored region. These metrics form the basis for later global analysis and trust computation.

Let t denote the current detection time window. Each fog node F_i is responsible for monitoring a set of edge devices $\mathcal{E}_i$. During window t, the node collects a set of raw detection events:

$$\mathcal{D}_i(t) = \{d_{i1}, d_{i2}, \ldots, d_{im}\},$$

where m is the number of events detected by the fog node F_i during the time window t, and each event d_{ij} originates from some device in $\mathcal{E}_i$. For each event d_{ij}, the fog node extracts a local feature vector $\mathbf{x}_{ij} \in \mathbb{R}^k$, where k denotes the number of detection-relevant attributes. These features typically include metrics such as anomaly score, packet rate variance, source IP entropy, protocol distribution, and flow duration.

To summarize its local detection activity, each fog node computes the following aggregated metrics on $\mathcal{D}_i(t)$:

- **Anomaly intensity** — average anomaly score across events:

$$A_i(t) = \frac{1}{m} \sum_{j=1}^{m} s_{ij}, \quad \text{denotes the anomaly score of event } d_{ij}, \text{ extracted as a component}$$

of the feature vector $\mathbf{x}_{ij}$.

- **Event diversity** — Shannon entropy over the distribution of event types:

$$H_i(t) = -\sum_{l=1}^{L} p_l \log p_l, \quad \text{where } L \text{ is the number of distinct event types observed in } \mathcal{D}_i(t),$$

and p_l is the proportion of events of type l. Each event d_{ij} is labeled with a type (e.g. port scan, login attempt), and the proportions $\{p_l\}_{l=1}^{L}$ are estimated by counting the frequency of each type among the m events in $\mathcal{D}_i(t)$.

- **Temporal dispersion** — standard deviation of interarrival times:

$$\Delta_i(t) = \mathrm{std}\left(\{t_{i(j+1)} - t_{ij}\}_{j=1}^{m-1}\right), \quad \text{where } t_{ij} \text{ is the timestamp of event } d_{ij},$$

and $\mathrm{std}(\cdot)$ denote the standard deviation. Events $\{d_{ij}\}$ are assumed to be sorted in increasing order of their timestamps.

- **Detection rate** — number of events per device per time unit:

$$R_i(t) = \frac{m}{|\mathcal{E}_i| \cdot \tau}, \quad \text{where } m \text{ is the number of events in } \mathcal{D}_i(t), \ |\mathcal{E}_i| \text{ is the number of devices ,}$$

monitored by F_i and τ is the duration (in seconds) of the detection window t.

The resulting fog-level summary vector is defined as:

$$\boldsymbol{\phi}_i(t) = [A_i(t),\ H_i(t),\ \Delta_i(t),\ R_i(t)] \in \mathbb{R}^4,$$

where each component, respectively, represents the intensity of the anomaly, the diversity of events, the temporal dispersion, and the detection rate calculated by node F_i during the window t.

This compact representation is transmitted to the cloud, where it serves as the input for global verification mechanisms and trust-inference algorithms.

By analyzing the set of vectors $\boldsymbol{\phi}_i(t)$ reported by all fog nodes, the cloud layer can:

- Detect coordinated or distributed attack patterns,
- Compare detection behavior across fog regions,
- Assess the reliability and consistency of local detections.

These insights are used to compute a *replicator coefficient* for each fog node, which quantifies its behavioral alignment with the global detection context. This coefficient serves as a key component of the trust-aware fusion mechanism described in the next section.

5. Evolutionary Game for Trust Reinforcement

To dynamically adjust the trust assigned to each fog node, we model their interactions and behavior consistency through an evolutionary game. The game is played at the cloud level among a population of fog nodes $\mathcal{F} = \{F_1, F_2, \ldots, F_N\}$, where each node acts as a player and N denotes the total number of fog nodes.

5.1. Population and Strategies

Each fog node is associated with a strategy that reflects the consistency of its metrics reported $\boldsymbol{\phi}_i(t)$ compared to those of its neighbors. We consider two main types of strategies:

- **Honest (H):** the node consistently reports indicators that are consistent with both its neighboring nodes and its own past behavior.
- **Unreliable (U):** the node provides metrics that are inconsistent, noisy, or potentially malicious.

At each time step t, the distribution of strategies in the population is represented by the state variables $x_H(t)$ and $x_U(t)$, denoting the proportion of nodes following strategies H and U, respectively. These proportions satisfy the normalization condition: $x_H(t) + x_U(t) = 1$.

5.2. Utility Computation

The utility of a fog node playing strategy $s \in \{H, U\}$ is calculated based on the consistency between its reported indicators $\boldsymbol{\phi}_i(t)$ and those of its neighbors in the GNN graph. These indicators reflect the node's local metrics at time t (for example, anomaly scores, event frequencies).

Formally, for a node F_i, the utility function $u_i(s)$ is defined as:

$$u_i(s) = \sum_{j \in \mathcal{N}(i)} w_{ij} \cdot \text{sim}(\boldsymbol{\phi}_i(t), \boldsymbol{\phi}_j(t))$$

where $\mathcal{N}(i)$ denotes the set of neighbors of node F_i, $w_{ij} \in [0, 1]$ is the trust weight assigned to the edge between F_i and F_j, and sim is a similarity function (e.g., cosine similarity or Pearson correlation) that quantifies the alignment of indicators between the two nodes.

This utility function captures how well the behavior of a node aligns with its local environment. It serves as the basis for the evolutionary game dynamics introduced in the next section, where the nodes adapt their strategies over time according to their relative utilities.

5.3. Replicator Dynamics

The proportion of each strategy in the population evolves according to replicator dynamics, which models the natural selection process, where strategies with higher-than-average utility become more prevalent over time.

Formally, the evolution of the population share $x_s(t)$ of the strategy $s \in \{H, U\}$ is given by:

$$\dot{x}_s(t) = x_s(t) \left(u_s(t) - \bar{u}(t)\right)$$

where $u_s(t)$ is the average utility of the nodes that play the strategy s, and $\bar{u}(t)$ is the mean utility across the entire population:

$$\bar{u}(t) = x_H(t) \cdot u_H(t) + x_U(t) \cdot u_U(t)$$

In this framework, strategies with utility above the population average grow in proportion, while others decline. This evolution provides a global view of how trustworthy behavior propagates or recedes.

In parallel, each node F_i is also assigned a **replicator coefficient** $r_i(t)$, which reflects its alignment with the prevailing trustworthy behavior. It can be derived from the utility deviation of the node relative to the population, for example:

$$r_i(t) = u_i(s) - \bar{u}(t)$$

This coefficient can be used to adjust trust levels, re-weight connections, or trigger mitigation mechanisms in the intrusion detection framework.

5.4. Integration into the Global GNN-Based Detection Model

The final stage consists of building a GNN at the cloud level to detect global intrusion by aggregating the indicators $\boldsymbol{\phi}_i(t)$ received from all fog nodes. The graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined as follows:

- $\mathcal{V} = \{F_1, \ldots, F_N\}$ is the set of fog nodes.
- $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ contains edges based on communication links or similarity relations.
- Each node F_i is associated with a characteristic vector $\boldsymbol{\phi}_i(t)$.
- Each edge $(i, j) \in \mathcal{E}$ is assigned a weight w_{ij} capturing trust or similarity between the two nodes.

To enhance the robustness of the model against unreliable input, the replicator coefficient $r_i(t) \in [0, 1]$ obtained from the evolutionary game is incorporated into the graph structure in two ways:

1. **Node Weighting:** each feature vector $\boldsymbol{\phi}_i(t)$ is scaled by its replicator coefficient:

$$\tilde{\boldsymbol{\phi}}_i(t) = r_i(t) \cdot \boldsymbol{\phi}_i(t)$$

   This ensures that nodes with inconsistent or adversarial behavior have less influence in the feature aggregation process.
2. **Edge Reweighting:** the weights of the edges are also modulated by the product of the replicator coefficients of their endpoints:

$$\tilde{w}_{ij} = r_i(t) \cdot r_j(t) \cdot w_{ij}$$

   This penalizes edges involving less trustworthy nodes, reducing their contribution in the message-passing phase of the GNN.

Final GNN Layer: the GNN is then trained or applied using these reweighted nodes and edges. A generic message-passing layer takes the form:

$$\boldsymbol{h}_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \tilde{w}_{ij} \cdot \mathbf{W}^{(l)} \tilde{\boldsymbol{\phi}}_j^{(l)}\right)$$

where σ is a non-linear activation function, $\mathbf{W}^{(l)}$ the learnable weight matrix at layer l, and $\tilde{\boldsymbol{\phi}}_j^{(l)}$ the transformed representation of node j at layer l.

This trust-aware GNN architecture ensures that nodes with poor historical behavior are dynamically down-weighted, while coordinated indicators among trusted fog nodes dominate global inference. As a result, the intrusion detection system remains robust to data drift, observational noise, and adversarial perturbations originating at edge or fog levels.

Algorithm 1 Vertical Intrusion Detection with Evolutionary Trust and GNN Inference

Input : Raw data streams from edge devices $\mathcal{E}_i$ supervised by each fog node F_i
Output : Global intrusion alerts with dynamic fog trust weighting

foreach *fog node* F_i **do**
  Collect and locally process data from supervised edge devices $\mathcal{E}_i$;
  Compute local anomaly indicators $\phi_i(t)$ (e.g., anomaly score, detection entropy, temporal divergence;
  Transmit $\phi_i(t)$ to the cloud layer;

foreach *fog node* F_i **do**
  At cloud level, evaluate $\phi_i(t)$ across fog nodes;
  Compute utility u_i based on:
  - Consistency with neighboring nodes
  - Agreement with global detection patterns

  Update evolutionary trust weight $\rho_i(t)$ using replicator dynamics:

$$\rho_i(t+1) = \rho_i(t) + \eta \cdot \rho_i(t) \cdot (u_i - \bar{u})$$

  where η is the adaptation rate and $\bar{u}$ is the average utility across all fog nodes;

Construct GNN $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ are fog nodes and $\mathcal{E}$ encodes proximity or functional relationships;
Integrate $\rho_i(t)$ as:

- Node weights or message-passing coefficients in GNN layers

Run global inference via GNN to identify distributed or coordinated intrusions;
return *Final alerts with reliability-weighted confidence scores*

6. Experimental Evaluation

6.1. Dataset and Preprocessing

To assess the effectiveness of our hierarchical and trust-aware intrusion detection framework in an IoT environment, we rely on the **UNSW-NB15** dataset. This dataset, developed by the Australian Center for Cyber Security (ACCS), was generated using the IXIA PerfectStorm tool to simulate modern and diverse network traffic, including both normal behavior and a wide range of attacks.

UNSW-NB15 contains a total of 2,540,044 labeled network flow records, each described by 49 features, including basic connection metadata (for example, duration, protocol, port numbers), content-based metrics (e.g., byte and packet counts), and time-related statistics. The attacks are grouped into the following nine categories: *Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode*, and *Worms*.

Given the constraints of IoT systems, low computational resources, real-time requirements, and partial observability, we selected a representative subset of the dataset, ensuring class balance between benign and malicious traffic. Of the 49 available features, we retained those most suitable for real-time, lightweight anomaly detection at the edge, including the following:

- `dur` – connection duration
- `spkts`, `dpkts` – packet counts in each direction
- `sbytes`, `dbytes` – byte volume in each direction
- `rate` – packet transmission rate
- `sttl`, `dttl` – time-to-live indicators
- `swin`, `dwin` – TCP window sizes

- `ct_state_ttl` – state transition patterns

These features are used to compute local anomaly scores using lightweight statistical estimators (e.g., z-score normalization, adaptive thresholding), suitable for deployment on constrained IoT edge devices.

Vertical Layer Simulation. To reflect the architecture of a realistic IoT deployment, we model a three-tier system composed of **Edge**, **Fog**, and **Cloud** layers:

- **Edge Layer:** a set of low-power IoT nodes (e.g. sensors, smart cameras) collect streaming data in real time and transmit it to their supervising fog node. These edge devices do not perform any local intrusion detection due to resource constraints and lack of global context.
- **Fog Layer:** intermediate fog nodes act as local aggregators, each supervising a group of 10–20 edge nodes. They receive raw data, perform local anomaly detection (for example, temporal divergence based on entropy) and transmit anomaly indicators $\phi_i(t)$ to the cloud.
- **Cloud Layer:** the global coordination layer is implemented using a **spatio-temporal Graph Neural Network (GNN)**, where each fog node is a vertex. The edges between the fog nodes are dynamically weighted using a replicator equation from evolutionary game theory, based on the consistency and precision of the reports received.

To simulate adversarial environments, we deliberately degrade the trustworthiness of a subset of fog nodes by injecting uncertainty and noise into their anomaly indicators. Specifically, we select up to 20% fog nodes to behave adversarially, either through random perturbations, delayed reporting, or inconsistent signaling—mimicking scenarios such as node compromise, software malfunction, or targeted misinformation. The GNN-based model refines threat detection by leveraging spatio-temporal correlations in the graph topology and dynamically adjusting trust weights through evolutionary feedback, thus mitigating the influence of unreliable nodes over time.

This layered simulation allows us to evaluate the **localized detection performance** at the **fog level**, where partial anomaly fusion is performed, as well as the **global correlation capabilities** at the cloud level using GNN inference. In particular, we compare a standard hierarchical IDS with our proposed **evolutionary trust-based IDS (eIDS)** in terms of detection accuracy, false positive rate, and robustness under noisy or adversarial conditions.

6.2. Baselines and Evaluation Metrics

To thoroughly assess the effectiveness of our proposed evolutionary trust-based IDS (eIDS), we compare it against several baseline architectures, each representing a different class of intrusion detection strategies commonly adopted in IoT environments:

(1) Flat Centralized IDS (Central-IDS). Traditional architecture where all edge devices forward raw or preprocessed flow data directly to a centralized cloud-based detection system. This IDS uses a conventional supervised model trained on global features, without hierarchy or distributed intelligence.

(2) Hierarchical IDS without Trust Mechanism (H-IDS). A three-tier architecture similar to ours (Edge–Fog–Cloud), where fog nodes collect flow data from edge devices and perform local anomaly detection independently. The cloud layer performs global correlation based on static topological connections between the fog nodes. This baseline does not incorporate any trust-aware fusion or adaptive graph structure; the cloud layer operates with fixed edge weights.

(3) Hierarchical IDS with Static Trust Weights (S-eIDS). An enhanced hierarchical IDS where fog-level detection is enhanced with pre-assigned trust values between fog nodes. These static trust weights are based on historical metrics such as uptime or past detection reliability. However, the trust graph remains fixed over time, without any dynamic adaptation or feedback mechanism.

(4) Proposed Model – Evolutionary Trust-based IDS (eIDS). Our full system, which includes: (i) unsupervised anomaly detection and signal fusion at the fog layer, (ii) global spatio-temporal correlation

through a GNN at the cloud level, and (iii) dynamic adjustment of trust weights between fog nodes using evolutionary replicator dynamics, allowing the system to penalize unreliable nodes and reinforce trustworthy ones over time.

Evaluation Metrics. We evaluate all models using the following performance indicators:

- **Area Under the ROC Curve (AUC):** measures the general discriminative ability of the model across thresholds.
- **True Positive Rate (TPR):** proportion of correctly detected malicious flows among all malicious ones.
- **False Positive Rate (FPR):** proportion of benign flows incorrectly identified as malicious.
- **Trust Stability Index (TSI):** captures the consistency of the evolution of the trust coefficient over training epochs.

Each experiment is repeated over 10 randomized train-test splits to ensure statistical significance. For all hierarchical models, we maintain the same number of edge and fog nodes and inject synthetic faults into a subset of fog nodes (up to 20%) to simulate adversarial behavior.

Implementation Details. All models are implemented in Python using PyTorch and PyTorch-Geometric for the GNN modules. The replicator dynamics are implemented as a feedback loop between epochs, updating the edge weights in the graph adjacency matrix. Training is carried out for 50 epochs using the Adam optimizer, with early stopping based on AUC validation.

6.3. Results and Analysis

We report the detection performance of all models on the UNSW-NB15 dataset in terms of AUC, TPR, and FPR.

Figure 3 shows the ROC curves averaged over more than 10 runs. Our proposed model, Evolutionary Trust-based IDS (eIDS), consistently outperforms all baselines, achieving an average AUC of 0.85, with a higher true positive rate and a lower false positive rate across all thresholds. In contrast, the Flat Central-IDS model performs significantly worse, with an AUC of only 0.67, highlighting the limitations of centralized detection in dynamic and distributed IoT environments. The Hierarchical IDS without trust and the Static Trust-based IDS obtain intermediate results, with AUCs of 0.73 and 0.78, respectively. These results demonstrate the advantage of integrating an adaptive trust mechanism into the detection process, allowing the system to better handle adversarial or noisy conditions.

Figure 4 compares false positive rates (FPR) in different IDS architectures using a boxplot over multiple runs. The *Flat Central-IDS* shows the highest FPR, illustrating the limitations of centralized decision-making in dynamic environments. Both *Hierarchical IDS (no trust)* and *Static Trust-based IDS* demonstrate moderate variability in FPR, reflecting their limited adaptability when faced with compromised fog nodes. In contrast, the *evolution trust-based IDS (eIDS)* achieves the lowest and most stable FPR, thanks to its ability to dynamically reduce the influence of unreliable nodes through evolving trust scores in the GNN-based correlation mechanism.

Figures 5 and 6 jointly illustrate the evolution of replicator dynamics and the resulting trust coefficients for five Fog nodes over time. A strong positive correlation is observed between replicator values and trust levels, as expected from the softmax-based trust calculation mechanism.

In Figure 5, Fog Nodes 1 and 2 maintain high and stable replicator values throughout the 50 time steps, indicating consistent cooperative behavior. In contrast, Nodes 3, 4, and 5 exhibit a clear and progressive decline in their replicator values, with Node 5 eventually dropping to zero. This decline reflects a sustained deviation from the expected cooperative behavior, which can be attributed to malicious actions or persistent inefficiency.

Figure 6 shows how the trust coefficients evolve accordingly. Nodes 1 and 2 gain progressively higher trust, with their coefficients increasing from approximately 0.20 to above 0.30. However, the trust values for Nodes 3, 4, and 5 gradually decrease below the initial baseline, approaching 0.10 for Node 5. This reflects the direct impact of their decreasing replicator values on the trust computation process.

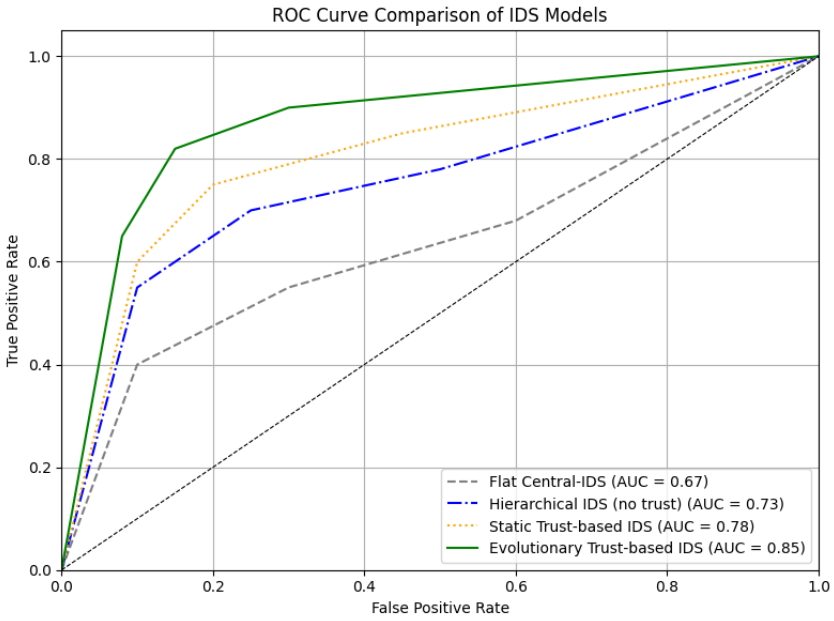

Figure 3: ROC curves for the different IDS architectures evaluated on UNSW-NB15.

Figure 4: Comparison of false positive rates across IDS models. The eIDS architecture achieves the lowest and most stable FPR by dynamically mitigating the influence of compromised fog nodes.

Overall, these results validate the effectiveness of using replicator dynamics as an input to the trust model: cooperative nodes are rewarded with increasing trust, while uncooperative or malicious behavior is penalized with decreasing trust, enabling adaptive and reliable trust management in distributed Fog environments.

7. Conclusion and perspectives

This paper introduced a vertical intrusion detection system (V-IDS) for hierarchical IoT architectures. Combining GNN-based alert fusion with an evolutionary game-theoretic trust mechanism, the model

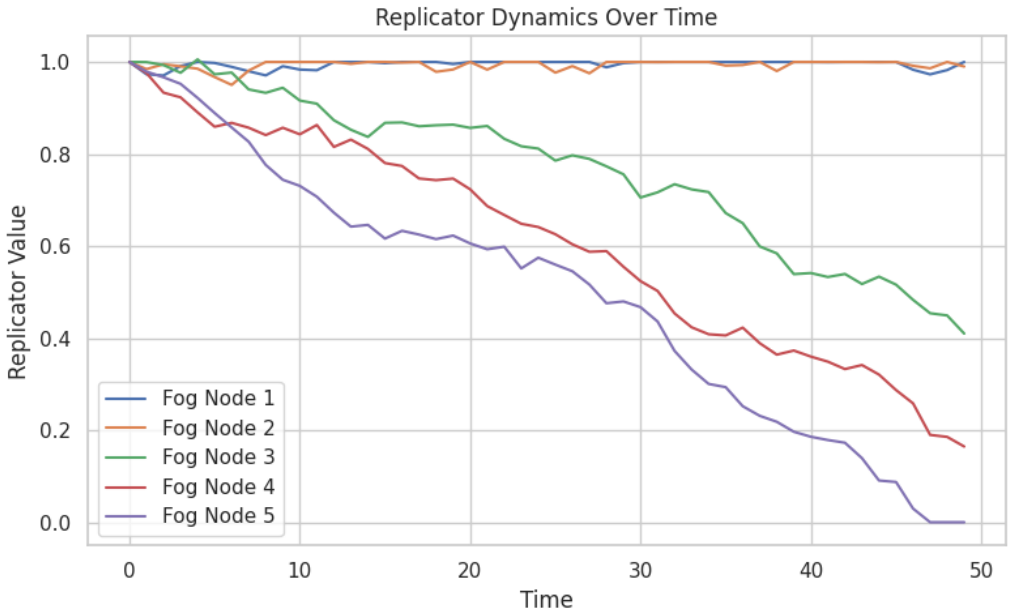

Figure 5: Evolution of the replicator dynamics of five Fog nodes over time. Nodes 1 and 2 maintain a high replicator value, while nodes 3, 4, and 5 experience a steady decline, suggesting increasingly uncooperative or malicious behavior.

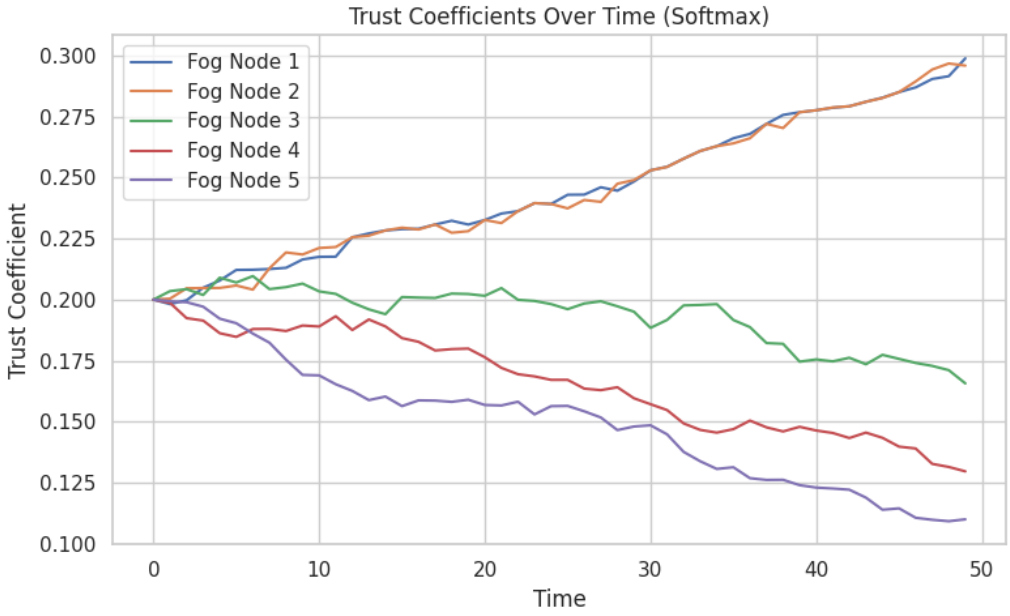

Figure 6: Trust coefficient evolution using softmax normalization. Nodes 1 and 2 gain more trust over time, while nodes 3, 4, and 5 lose trust due to their declining replicator values.

enables structural and temporal reasoning over a two-layer Fog–Cloud setup and improves resilience by dynamically adjusting node influence based on trust. The system mitigates the impact of unreliable or malicious nodes by embedding trust evolution into the GNN via adaptive edge weights, guided by local performance and neighborhood feedback. To ensure tractability, we focus on a two-layer detection model. Extending the architecture to include a third Edge layer is feasible but adds computational complexity, and will be addressed in future work.

The next steps include formal analysis of evolutionary dynamics, dense graph scalability studies, integration with federated learning for privacy, and broader empirical evaluation of real-world data

sets. This research contributes to adaptive, robust, and intelligent IDSs for secure IoT infrastructures.

References

[1] M. A. Al-Garadi, A. Mohamed, A. K. Al-Ali, X. Du, I. Ali, M. Guizani, A survey of machine and deep learning methods for internet of things (iot) security, IEEE communications surveys & tutorials 22 (2020) 1646–1685.

[2] X. C. Yin, Z. G. Liu, L. Nkenyereye, B. Ndibanje, Toward an applied cyber security solution in iot-based smart grids: An intrusion detection system approach, Sensors 19 (2019) 4952.

[3] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, E. Vázquez, Anomaly-based network intrusion detection: Techniques, systems and challenges, computers & security 28 (2009) 18–28.

[4] L. Lin, Q. Zhong, J. Qiu, Z. Liang, E-gracl: an iot intrusion detection system based on graph neural networks, The Journal of Supercomputing 81 (2025) 42.

[5] M. M. Rahman, S. Al Shakil, M. R. Mustakim, A survey on intrusion detection system in iot networks, Cyber Security and Applications 3 (2025) 100082.

[6] A. S. Ahanger, S. M. Khan, F. Masoodi, A. O. Salau, Advanced intrusion detection in internet of things using graph attention networks, Scientific Reports 15 (2025) 9831.

[7] M. Bouhaddi, M. S. Radjef, K. Adi, An efficient intrusion detection in resource-constrained mobile ad-hoc networks, Computers & Security 76 (2018) 156–177.

[8] K. Alsubhi, A. Sagheer, Lightweight intrusion detection system for iot-based healthcare, IEEE Access 8 (2020) 113334–113344.

[9] S. Yin, Y. Yang, S. Sun, S. Wang, Distributed intrusion detection for industrial iot using edge computing and federated learning, IEEE Transactions on Industrial Informatics 17 (2021) 2443–2452.

[10] Y. Li, L. Zhang, Y. Zhang, Gcn-ids: Intrusion detection system based on graph convolutional networks, IEEE Access 8 (2020) 175413–175424.

[11] Y. Feng, X. Ye, J. Zhang, Deep learning for lateral movement detection: An overview, IEEE Access 9 (2021) 143508–143520.

[12] Z. Liu, M. Wang, W. Zhang, Graph anomaly detection for iot networks: A graph neural network approach, Computers & Security 123 (2023) 102994.

[13] X. Zhang, Graph neural networks in network security: From theoretical foundations to applications (2025).

[14] R. Bing, G. Yuan, M. Zhu, F. Meng, H. Ma, S. Qiao, Heterogeneous graph neural networks analysis: a survey of techniques, evaluations and applications, Artificial Intelligence Review 56 (2023) 8003–8042.

[15] O. S. M. B. H. Almazrouei, P. Magalingam, M. K. Hasan, M. Shanmugam, A review on attack graph analysis for iot vulnerability assessment: challenges, open issues, and future directions, IEEE Access 11 (2023) 44350–44376.

[16] J. Zhao, L. Zhang, W. Li, Trust-aware federated learning for iot security: A reputation-based approach, IEEE Internet of Things Journal 9 (2022) 456–466.

[17] M. Bouhaddi, K. Adi, M. S. Radjef, Evolutionary game-based defense mechanism in the manets, in: Proceedings of the 9th International Conference on Security of Information and Networks, 2016, pp. 88–95.

[18] W. Li, H. Jin, H. Zhang, et al., A game theory-based trust model for secure routing in wireless sensor networks, IEEE Transactions on Wireless Communications 17 (2018) 6252–6265.

[19] J. Hofbauer, K. Sigmund, Evolutionary games and population dynamics, Cambridge university press, 1998.

[20] J. W. Weibull, Evolutionary game theory, MIT press, 1997.

[21] L. Samuelson, Evolutionary games and equilibrium selection, volume 1, MIT press, 1997.

[22] G. Corso, H. Stark, S. Jegelka, T. Jaakkola, R. Barzilay, Graph neural networks, Nature Reviews Methods Primers 4 (2024) 17.