*electronics*

MDPI

*Article*

# ATAW-TM: An Adaptive, Threshold-Free, and Automatically Weighted Trust Model for Mitigating Multiple Types of Denial-of-Service Attacks in Software-Defined Wireless Sensor Networks

**Lijuan Wang, Mee Loong Yang and Krassie Petrova ***

School of Engineering, Computer and Mathematical Sciences, Auckland University of Technology, Auckland 1010, New Zealand; lijuan.wang@autuni.ac.nz (L.W.); bobby.yang@aut.ac.nz (M.L.Y.)
* Correspondence: krassie.petrova@aut.ac.nz

**Abstract**

Wireless sensor networks (WSNs), including Software-Defined Wireless Sensors, are particularly vulnerable to Denial-of-Service (DoS) attacks. Trust models are widely acknowledged as an effective strategy to mitigate the threat of successful DoS attacks in WSNs. However, existing trust models commonly rely on threshold configurations that are based on the network administrator's experience and leave the challenging task of weight allocation for various trust metrics to network users. This limits the widespread application of trust models as a WSN defence mechanism. To address that issue, this study proposes and theoretically analyses an Adaptive, Threshold-Free, and Automatically Weighted Trust Model (ATAW-TM) for SDWSNs. The model architecture is aligned with the layered centralized management architecture of SDWSNs, which makes it flexible and enhances its responsiveness. The proposed model does not require manual threshold configuration and weight allocation and allows for rapid trust system recovery. It has significant advantages compared to existing trust models and is potentially more feasible to implement on a large scale.

**Keywords:** denial-of-service attacks; trust model; software-defined wireless sensor networks; SDWSNs; trust value weight allocation; automatic threshold determination



Academic Editors: Baihe Ma and Mst Shapna Akter

Received: 16 November 2025
Revised: 7 December 2025
Accepted: 12 December 2025
Published: 16 December 2025

**Citation:** Wang, L.; Yang, M.L.; Petrova, K. ATAW-TM: An Adaptive, Threshold-Free, and Automatically Weighted Trust Model for Mitigating Multiple Types of Denial-of-Service Attacks in Software-Defined Wireless Sensor Networks. *Electronics* **2025**, *14*, 4933. https://doi.org/10.3390/electronics14244933



## 1. Introduction

A wireless sensor network (WSN) comprises spatially distributed sensor nodes, which include sensing, data processing, and communication components [1]. These nodes self-organize to create a multi-hop wireless network. The sensors can be used to detect and collect environmental or physical data, such as humidity, sound, and temperature [2]. These data are transmitted through the network to a centralized hub (a base station or a sink node) and subsequently forwarded to an Internet-based server for analysis and processing [3,4]. WSNs are widely used in a large variety of applications including military, environmental monitoring, smart homes, agriculture, animal husbandry, and health monitoring.

With Internet of Things (IoT) technology continuously evolving, WSNs have become more widely used. At the same time, their various limitations have become increasingly clear, such as when applications require the WSNs to be tailored to meet unique functional and performance requirements, and the nodes are self-configuring distributed management. These limitations create challenges in terms of dynamic network management, scalability,

and node recycling. Incorporating the concept of Software-Defined Networking (SDN) (which facilitates more dynamic and manageable network operations through the functional separation of the control and the forwarding planes) into WSNs has helped meet these challenges and given rise to a new type of WSN: Software-Defined Wireless Sensor Networks (SDWSNs) [5].

### *1.1. Operational Features and Characteristics of SDWSNs*

SDWSNs incorporate a centralized controller that maintains a global view of the network and dynamically configures sensor nodes through southbound interfaces, similar to OpenFlow. Figure 1 presents the basic structure of an SDWSN, which includes a data plane, an application plane, and a central controller [6].

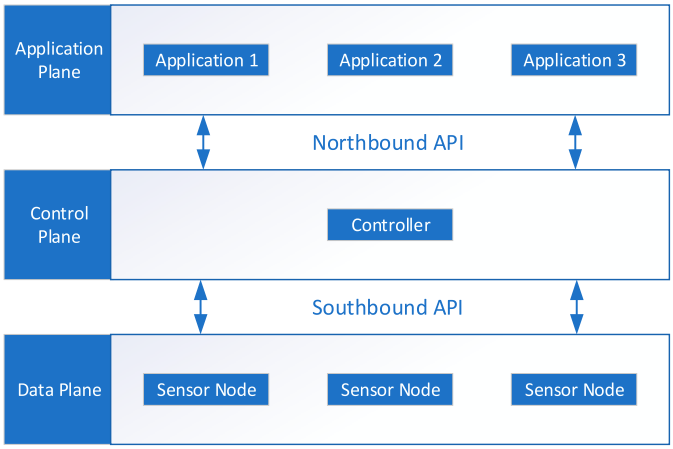


**Figure 1.** The basic structure of an SDWSN.

Unlike conventional WSNs, where decisions are distributed and often rigid, SDWSNs support the following operational features [5,6]:

1. Control–data plane separation:
   The control plane houses the network intelligence, makes global decisions, and manages data plane behaviour. The controller builds global network views, calculates optimal paths and policies. The data plane is responsible for actual packet processing, forwarding, and data transmission based on predefined rules.
2. Flow table-based forwarding:
   The controller installs flow rules in data plane devices' flow tables. These store matching rules and actions. The nodes process packets based on installed flow rules.

SDWSNs combine the flexibility of SDNs with the specific features of WSNs, which makes SDWSNs suitable for a broad range of applications. Compared to WSNs, SDWSNs have the following advantaged characteristics [6,7]:

1. Flexibility and Programmability
   SDWSNs provide increased flexibility and programmability by segregating control plane activities from the data plane. The decoupling supports network adjustments and optimization in response to real-time requirements.
2. Centralized Management
   SDWSNs enable centralized management, including network configuration. The behaviour and data flow of sensor nodes are coordinated by SDN controllers, which improves network management efficiency.
3. Dynamic Optimization
   The control layer enables SDWSNs to dynamically optimize the use of network

resources and to adjust data routing strategies. This improves network performance and reliability.

4. Enhanced Scalability
   The entire set of sensor nodes across the data plane is visible to the central controller, which facilitates efficient network management. This is especially beneficial in the case of large-scale network expansion.

### 1.2. Application Scenarios of SDWSNs

Due to these advantages, SDWSNs are increasingly considered for mission-critical and dynamic application domains [8,9], such as the following:

- Industrial Internet of Things (IIoT), where a rapid reconfiguration and reliability are crucial.
- Environmental and structural health monitoring, requiring adaptive routing in large-scale deployments.
- Emergency and disaster response, where nodes may be mobile and the network topology frequently changes.
- Military and tactical sensing, which demand centralized coordination and strict security controls.

These scenarios highlight the necessity of an architecture capable of handling dynamic conditions and enforcing fine-grained and flexible security policies.

### 1.3. Security Challenges and DoS Vulnerabilities in SDWSNs

Despite their advantages, SDWSNs introduce unique security risks. The centralized controller, while improving coordination, also becomes a prime target for Denial-of-Service (DoS) attacks [10]. Attackers may overwhelm the controller with excessive requests, inject false control messages, manipulate routing rules, or exploit multi-hop wireless communication to generate traffic flooding.

Furthermore, traditional cryptographic mechanisms alone cannot defend against insider nodes that behave maliciously yet possess valid keys [11]. This limitation is particularly problematic in SDWSNs, where compromised nodes can exploit controller–node interactions to degrade network performance or disrupt global routing operations.

### 1.4. Necessity of Trust-Based DoS Mitigation in SDWSNs

Trust models, which establish a network of trust among entities by evaluating and predicting their behaviour based on observed interactions [11], can provide effective defence against DoS attacks under uncertain conditions. These models consider the node behaviour and reputation when making decisions about trustworthiness (like how trust and reputation are established in the context of human behaviour, through continuous interaction and observation). Trust-based mechanisms can differentiate legitimate nodes from misbehaving or compromised ones, thereby achieving the following [10,12,13]:

- Detecting internal DoS behaviours that cryptography cannot address;
- Protecting the controller from unnecessary or malicious traffic;
- Supporting adaptive responses through flow rule adjustments issued by the controller.

Thus, integrating trust evaluation with the SDWSN architecture is a natural and necessary solution for mitigating DoS threats.

### 1.5. Target Scenario of This Study

This study focuses on a typical SDWSN deployment consisting of a central controller and multiple static sensor nodes operating in a multi-hop wireless topology. Nodes forward sensing data toward the controller, and the controller manages routing through flow table

updates. The network is assumed to face internal DoS threats, such as flooding attacks, blackhole attacks, and vampire attacks caused by compromised nodes.

Our work investigates how a trust-based mechanism can be integrated into this SDWSN architecture to detect and mitigate the 14 types of DoS attacks in SDWSNs [14], ensuring stable communication.

Trust models in WSNs have gained increasing recognition and have been widely studied. Numerous trust models have been proposed, with some specifically addressing the challenges of DoS attacks. However, trust models in WSNs are still new and have not set any standards yet [15,16]. There has been extensive research on mechanisms leveraging trust evaluation to defend against DoS threats in traditional WSN environments [11,14], especially in addressing threshold limitations in trust evaluation [10,17], the assignment of weights to trust metrics [18–21], and the loss of trust information [10,22–25].

### *1.6. Contributions*

This study addresses the research gaps above in the context of SDWSNs. It explores the design of innovative trust mechanisms applicable to SDWSNs and proposes a solution to mitigate all the 14 types of DoS attacks identified. This study makes the following research contributions:

1. A three-layer trust framework (node, cluster head, and controller) to address a wide range of DoS attacks in SDWSNs.
2. A combined method of outlier detection and the Bayesian Beta approach to calculate direct trust values within the layered architecture of SDWSNs' trust model. This approach eliminates the dependency on threshold-setting algorithms that rely on network administrators' prior knowledge of specific SDWSNs.
3. A method for integrating reciprocal weighting and entropy-based weighting to automatically assign adaptive weights to various trust metrics across the control and data planes of SDWSNs. This approach eliminates the inaccuracies in combined trust value calculations caused by fixed weights and the difficulties users face in assigning weights.
4. A method for referencing the logistic function that converts the difference between historical combined trust values and current combined trust values into an aging factor. Within the centralized control framework of SDWSNs, this allows for dynamic automatic adjustment of the aging factor. Consequently, when a node exhibits malicious behaviour, its trust value decreases rapidly, and when it returns to normal behaviour, its trust value increases slowly.
5. A trust information retrieval mechanism tailored for SDWSNs' hierarchical structure that enables both member nodes and cluster head (CH) nodes to quickly recover lost trust information from the controller or cluster level. This feature leverages the centralized control and global visibility of SDWSNs to ensure the robustness and resilience of the trust management system after information loss.

The structure of the remainder of this paper is as follows: Section 2 provides an overview of related work. Section 3 presents the proposed trust model (ATAW-TM). Section 4 analyses ATAW-TM within the context of the existing literature, highlighting its contributions. Section 5 discusses the results and concludes the paper; directions for future research are also outlined.

## 2. Related Work

There has been extensive research on mechanisms leveraging trust evaluation to defend against DoS threats in traditional WSN environments, including the identification of trust evidence necessary for trust models to effectively detect DoS attacks [18,22,26,27], developing

methods for trust evidence extraction [28–30], and the trust computing techniques that facilitate the design of robust and reliable trust mechanisms to defend against DoS attacks [23,31,32]. In our extensive literature review [14], we identified existing approaches and challenges in developing adaptive trust mechanisms for detecting and defending WSNs against diverse DoS threats. However, research on using trust-driven approaches to address DoS vulnerabilities in SDWSNs remains limited, with only four studies available to date [10,12,13,33].

### 2.1. ETMRM: Trust-Based Routing and Management in SDWSNs

In 2018, Wang et al. [10] developed a trust-based routing and management scheme (ETMRM) that was designed to enhance energy efficiency in SDWSNs, aimed at mitigating new flow attacks and selective forwarding attacks. This study leverages trust management methodologies initially designed for conventional WSNs. These techniques are combined with SDN flow tables and reporting mechanisms. The integration is carried out within the SDN-WISE model [8,34], an SDN-based framework for wireless sensor networks. Through this approach, the study establishes an initial trust model specifically designed for SDWSNs. This model has seven steps: recording trust, evaluating local trust, reporting trust, combining trust values, evaluating global trust, finding and isolating malicious nodes, and calculating routes based on trust.

To collect trust evidence, the trust record leverages an enhanced flow table to monitor successful and unsuccessful transmissions, encompassing both data and control packets, as well as new flow packets that do not align with existing rules. During the local trust evaluation phase, forwarding trust is determined using a Bayesian Beta approach, while new flow trust is assessed through a threshold-limiting technique. Nodes compile their local trust values for all neighbours into SDN-WISE messages, which are subsequently sent to an aggregation node. This node consolidates the data and relays it to the controller. The controller evaluates network-wide trust scores, identifies nodes exhibiting malicious behaviour, and issues packet drop policies to neighbouring nodes impacted by the threat. When a new flow packet arrives, handling of the relevant rule request is delegated to the controller. It determines the routing path by referencing trust values, considering the node's network-wide trust score and remaining energy level.

The model has undergone prototyping, with experimental results demonstrating its capability to mitigate the effects of new flow and selective forwarding attacks while ensuring more balanced energy consumption. Nevertheless, the model evaluates trust solely within a single cycle during the local calculation phase, disregarding historical trust data. Incorporating past behaviours could enhance the robustness of trust assessments. Furthermore, the model's defence is limited to control plane DoS attacks (new flow attacks) and does not address data plane DoS attacks, as it lacks mechanisms to log and analyse non-new flow data packets.

### 2.2. Hierarchical Trust Management for SDWSNs

Bin-Yahya et al. [12,33] proposed a hierarchical trust management scheme (HTM) for secure communications in SDWSNs. This model calculates and records the trustworthiness of each node at various levels (node, CH, and controller) to promptly detect malicious nodes. In this approach, distinct trust values are maintained for control traffic and data traffic, where the trust evidence is derived from metrics including successful versus unsuccessful forwarding of control and data messages, together with their associated transmission rates.

Forwarding trust is assessed using a Bayesian method, while sending trust is evaluated through a threshold-limiting technique. Each node computes trust values for its neighbours and initiates trust update messages. The cluster head aggregates these updates, forwards them to the sink node, and finally transmits them to the controller. Trust aggregation is

performed at both the cluster head and controller levels, with lower weights assigned to outliers based on score reliability during the aggregation process. To enhance responsiveness to malicious activities, the scheme integrates reward and penalty factors into the Bayesian model. It also capitalizes on specific characteristics of SDWSNs, such as leveraging flow table statistics collected by the controller and using designated control messages to evaluate intermediate nodes' behaviour. This scheme demonstrates high effectiveness in detecting a range of attacks and shows a superior performance compared to ETMRM [10] in detecting DoS attacks.

### 2.3. SDWSN Trust Management Framework with an Intrusion Detection System

Isong et al. [13] introduced an innovative Trust Management Framework (TMF) tailored for SDWSNs. This scheme incorporates two levels of trust evaluation: sensor-to-sensor (S2S) and controller-to-sensor (C2S) trust. It is structured into three key components: the information tracking component, the trust evaluation component, and the control logic component. The information tracking component gathers data from two sources: packet forwarding information collected via the sink node and network statistics obtained from the controller. This information is retained by both the controller and the sink node for further analysis. The trust evaluation component calculates a trust score through both observed and inferred methods. The observed trust relies on the quantity of packets the individual node successfully forwards and receives, while inferred trust is derived through traffic data analysed by an Intrusion Detection System (IDS) model. To ensure continuous monitoring, the controller periodically retrieves statistical data from all network nodes. Finally, the decision-making unit utilizes the computed trust values to determine appropriate actions, such as isolating or monitoring nodes identified as malicious. This systematic approach allows the framework to enhance security by effectively detecting and addressing potential threats.

Although researchers have developed many IDSs based on anomaly detection, signature matching, and deep packet inspection, their applicability in SDWSNs remains limited for several reasons. First, SDWSNs operate under stringent resource constraints, and deep packet inspection or complex machine-learning-based IDS modules often exceed the computational and energy budgets of sensor nodes [35]. Second, most IDS techniques assume stable and high-bandwidth links, whereas SDWSNs frequently involve dynamic topologies, intermittent connectivity, and distributed wireless interference [36]. Third, IDSs typically react after malicious traffic is observed, while DoS attacks in SDWSNs can rapidly exhaust critical control-plane resources [12], such as flow-table entries or southbound communication channels, before IDS mechanisms can respond.

### 2.4. Research Gaps

A major limitation of the models reviewed above is the lack of focus on the integrity of the trust evidence. Moreover, the impact of variations in communication channel conditions on the reliability and precision of trust assessment is overlooked by all above existing approaches. Additionally, the first two implemented and evaluated models have three research gaps we previously identified: setting a threshold in trust evaluation, assigning weights to trust metrics, and addressing the loss of trust information. In the last model, the responsibility for trust evaluation is delegated to the IDS module, but the specific evaluation methods within the IDS module are not discussed in depth. SDWSNs require proactive and lightweight security mechanisms capable of detecting or mitigating DoS behaviour early and continuously, rather than relying solely on traditional IDS approaches. This motivates the exploration of trust-based models.

## 3. Materials and Methods

The proposed Adaptive, Threshold-Free, and Automatically Weighted Trust Model (ATAW-TM) is a trust model specifically designed for SDWSNs to detect DoS attacks within these networks. It eliminates the reliance on human prior knowledge during application, enhancing its usability, and incorporates trust information backup and rapid recovery mechanisms to improve stability. The model is implemented using SDN-WISE as the underlying architecture, with the SDN controller embedded within the sink node to simplify network design and reduce overheads.

ATAW-TM adopts a three-tier hierarchical architecture, comprising the node level, cluster head (CH) level, and controller level. Each level contributes to a layered and cooperative trust evaluation and decision-making process. An overview of the ATAW-TM is presented in Figure 2.

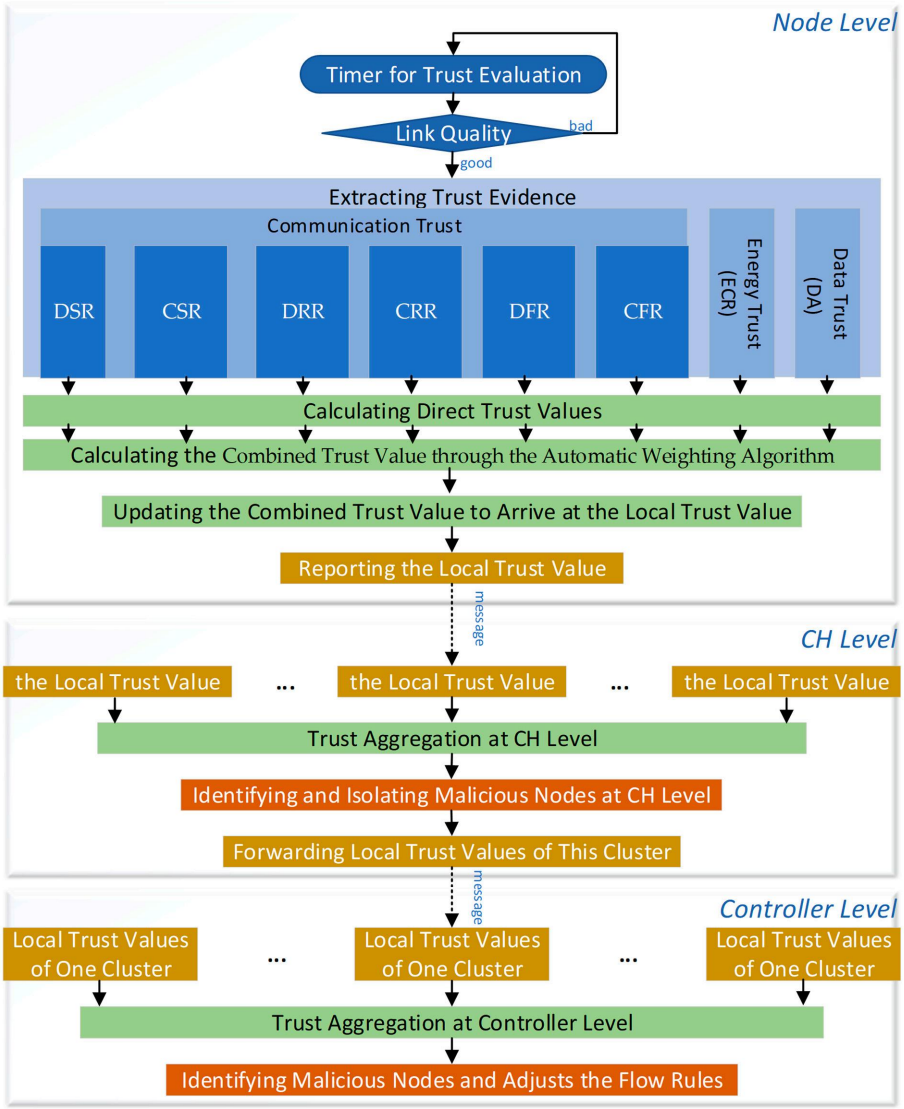


**Figure 2.** Overview of the ATAW-TM. Different colours in the figure represent distinct functional categories in the trust management process: blue blocks denote different sources of trust evidence; dark blue blocks represent initial trust evaluation inputs such as link quality and timer operations; green blocks indicate trust com-putation and aggregation procedures; yellow blocks represent reporting or forwarding operations, such as transmitting local trust values; and orange blocks denote security-related decision actions, specifically identifying and isolating malicious nodes and adjusting flow rules.

At the node level, each sensor node performs its own node-level evaluation process. First, start a timer for trust evaluation to perform periodic trust assessments. Second, nodes are tasked with extracting trust evidence and performing direct trust computation locally. At the beginning of each trust evaluation, assess the link quality between the node and its neighbouring nodes. If the link quality is good, continue with the evaluation process; if the communication link becomes unstable or weak, wait for the next evaluation round. During the evaluation, nodes extract trust evidence and compute the direct trust value locally. Trust evidence encompasses communication, energy, and data indicators. Communication-related trust indicators include metrics such as the transmission and reception rates of both data and control packets, as well as their respective forwarding rates. These indicators were identified in [14] as the eight types of trust evidence, including the data packet sending rate (DSR), control packet sending rate (CSR), data packet receiving rate (DRR), control packet receiving rate (CRR), data packet forwarding rate (DFR), control packet forwarding rate (CFR), energy consumption rate (ECR), and data accuracy (DA).

Next, each direct trust value corresponding to these metrics is then employed to calculate the combined trust value through an automatic weighting algorithm that integrates the direct trust values derived from individual metrics. The historical and current trust metrics are weighed using an aging factor to reflect temporal relevance, resulting in the local trust value. Finally, nodes periodically send their local trust value to the CH of its respective cluster.

The CH is responsible for collecting the local trust value reported by each node and forwarding the report message to the controller. It calculates the aggregated trust value by weighted averaging of the local trust values received from each node using its own calculated local trust value, and it relays trust values to the controller. The CH uses the aggregated trust value to recognize node maliciousness. Subsequently, it halts forwarding any messages to and from malicious nodes, thus segregating detected malicious nodes.

The controller, positioned within the sink node, gathers trust values from all nodes and computes their aggregated trust values from a global perspective. It identifies malicious nodes according to their aggregated trust scores. It then adjusts the flow rules to isolate the identified malicious nodes.

### *3.1. Node-Level Operations: Extracting Trust Evidence*

#### 3.1.1. Assessing Link Quality

Before each trust evaluation, it assesses the quality of connections linking the node with its neighbouring nodes to avoid a low link quality affecting the trustworthiness scores of legitimate nodes. Because of the openness of WSNs' wireless communication, the links between sensor nodes are unstable and can be influenced by the environment. With low-quality links, the communication capability decreases, causing reductions in the rates of packet transmission, reception, and forwarding, thus resulting in a lower calculated trust value.

In ATAW-TM, we use the method proposed in [19] to evaluate the link quality, which relies on the Link Quality Indicator (LQI). In the widely adopted CC2420 ratio, the LQI is calculated from the first eight symbols of a received packet, ranging between 0 and 255. The proposed method involves the evaluation node collecting LQI data over a specific period of time, followed by the calculation of the mean value. If the calculated mean surpasses the threshold, such as 220, the link is considered normal, and the evaluation process continues. If the mean value is below the threshold, the link is considered unstable, and the assessment is temporarily halted and resumed in the next cycle to re-evaluate the link quality.

To prevent attackers from intentionally manipulating link quality indicators (LQIs) to evade trust evaluation, we use a robustness mechanism. This conducts neighbour

consistency checks by comparing LQI patterns across adjacent nodes. If a single node's LQI decreases while most of its neighbours (more than a threshold θ) remain normal, the node's degraded LQI is treated as suspicious, and it continues to undergo trust evaluation. In contrast, if LQI degradation is observed across most nodes in the neighbourhood, the drop is likely caused by environmental interference, and trust evaluation for all nodes in that neighbourhood is suspended. This prevents attackers from exploiting LQI manipulation to bypass detection.

#### 3.1.2. Extracting Trust Evidence

Figure 3 below illustrates the trust evidence included in our trust model, along with the corresponding extraction methods.

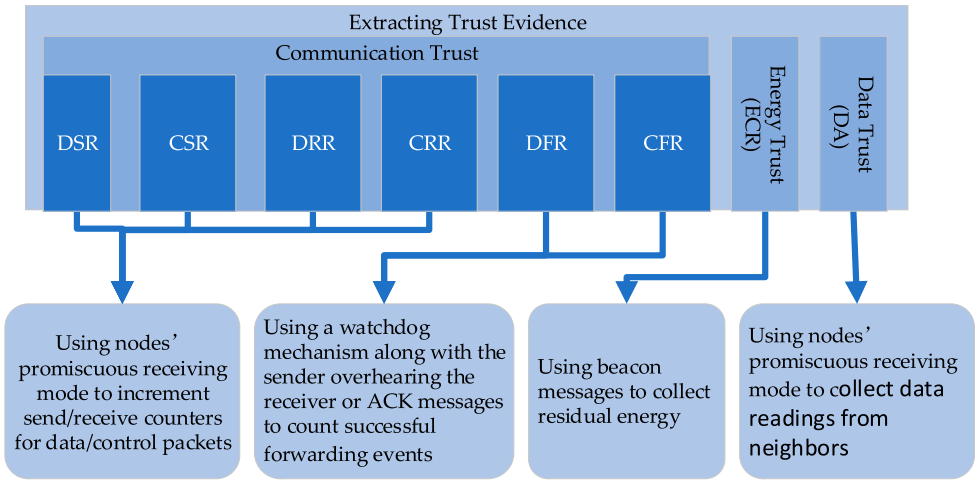


**Figure 3.** Trust evidence and the corresponding extraction methods.

The rates at which data and control packets are sent and received are extracted using the sensor node's promiscuous receiving mode. When this mode is enabled, any packet within the node's receiving range (whether intended for it or another node) is processed. The packet count for a given node is raised by one whenever a data or control packet from that node is identified. Additionally, by examining each incoming packet's destination address and type, the system determines the target node and updates its associated count.

The forwarding rates for data and control packets are determined using listening methods as follows. Promiscuous receiving mode is activated on the sensor node to allow the capture of all surrounding traffic. Once the node is ready to monitor forwarding behaviour, it transmits a packet and starts a watchdog timer. During this monitoring window, the node observes the actions of the intended recipient. Successful forwarding is recorded if the target recipient relays the packet prior to the expiration of the timer; otherwise, failed relaying is logged. Furthermore, to assess the control packet forwarding rate, the ACK-based mechanism can be employed as a supplementary technique alongside the listening method. Forwarding is considered successful if the forwarded packet or its corresponding ACK message is detected.

Using the remaining energy information embedded in beacon messages, the ECR of a node is calculated. This rate is then compared to the ECRs of other nodes to identify any differences.

For data trust evidence, the sensor node collects readings from all its neighbouring nodes using the open receiving mode. It then determines whether there is data deviation by comparing these readings with the neighbourhood data references.

### *3.2. Node-Level Operations: Calculating, Updating, and Reporting Trust Values*

#### 3.2.1. The Process of Direct Trust Value Calculation

To overcome the obstacle of threshold setting in the threshold limitation approach that relies on the network administrator's a priori knowledge of the particular network, we assign a cooperation probability to each value through the outlier detection method when calculating the direct trust values for DSR, CSR, DRR, CRR, ECR, and DA, and then we use the Bayesian Beta method to derive the direct trust values.

In contrast, the forwarding-related metrics, DFR and CFR, represent fundamentally different types of evidence. Forwarding is not a continuous measure but a discrete binary event: a packet is either forwarded successfully or it is not. Consequently, it is neither meaningful nor necessary to compute a probabilistic cooperation value based on distributional assumptions. Instead, forwarding trust is modelled using a binary assignment, where successful forwarding (observed through listening or ACK mechanisms) is assigned a cooperation value of 1 and unsuccessful forwarding is assigned a value of 0. These binary outcomes are then directly incorporated into the Bayesian Beta method. The processes of direct trust value calculation are shown in Figure 4.

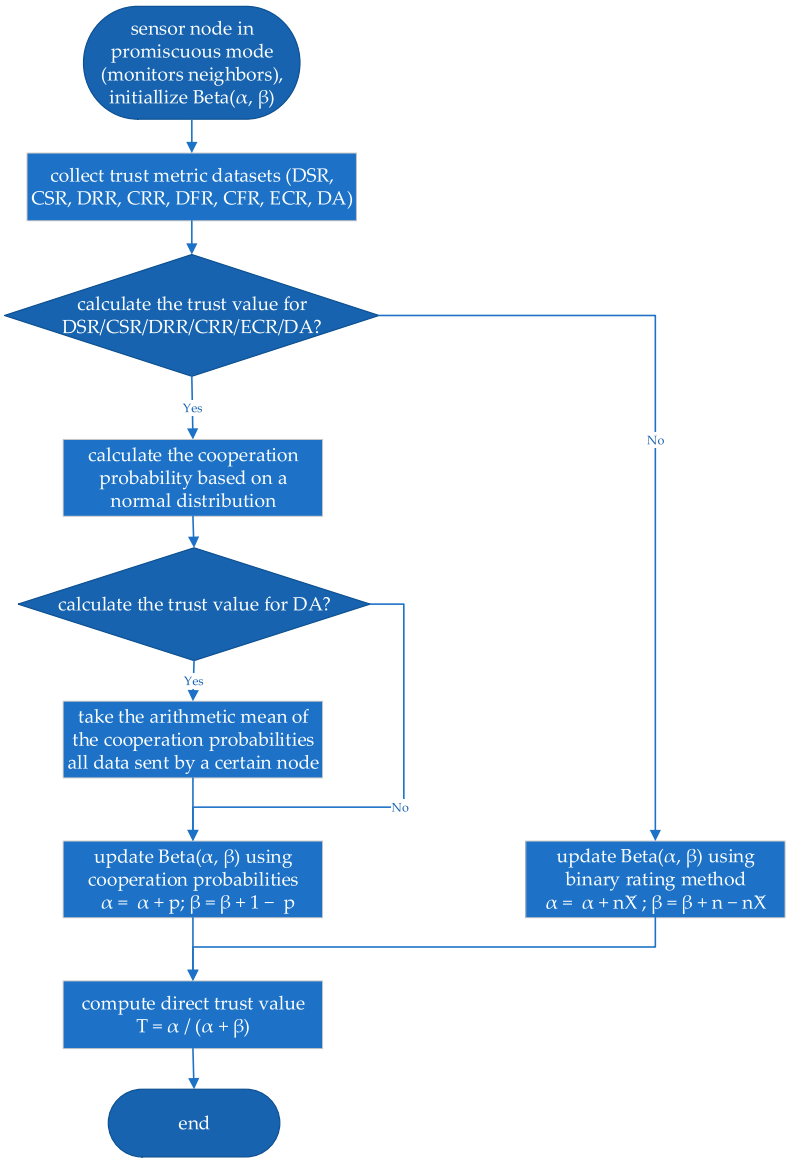


**Figure 4.** The processes of direct trust value calculation.

The evaluation is conducted over a defined period of time using the following framework for each metric. At the start of an evaluation period, we initialize the Beta distribution parameters for each metric: the count of cooperative interactions ($\alpha$) and the count of non-cooperative interactions ($\beta$). These two parameters form the basis for calculating trust values. The initial state for all nodes and all metrics is set to zero ($\alpha = 0$, $\beta = 0$), indicating no prior history.

Next, the node collects trust metric datasets for each metric. The cooperation probability ($p$) is calculated using a Gaussian distribution model based on the collected trust metric datasets. This allows the node to determine the likelihood that its neighbours are behaving cooperatively. It is used to update the $\alpha$ and $\beta$ parameters. The direct trust value ($T$) is computed from $\alpha$ and $\beta$ as the expected value of the Beta distribution at the end of the period, representing the trustworthiness of a node in a specific metric.

#### 3.2.2. The Method of Cooperation Probability Calculation

Multiple studies, through experiments or simulations, have verified that in WSNs where node behaviour is independent, homogeneous, and static, the traffic of nodes and the collected sample data approximately follow a Gaussian distribution when the number of nodes is large enough [37–39]. Therefore, we adopt a cooperation probability calculation method based on a Gaussian distribution.

The packet size is a common feature in traffic analysis and intrusion detection [40]. However, it is not included in the proposed trust model because DoS attacks typically manifest through abnormal transmission rates and forwarding behaviours rather than size anomalies [14]. This design choice reduces the computational overhead while maintaining detection effectiveness. Let us assume that the evaluation node is node $i$, and the set of all $n$ neighbouring nodes being evaluated is represented as $j = \{1, 2, 3, \ldots, n\}$. In an evaluation period, node $i$ monitors its neighbours to build datasets for calculating the cooperation probabilities. The specific data collected for each metric is outlined in Table 1.

**Table 1.** Dataset for each metric.

| Metric ($X$) | Dataset Collected for All Nodes $j$ | Description of Data Collection |
|---|---|---|
| DSR | $DS = \{ds_1, ds_2, \ldots, ds_j, \ldots, ds_n\}$ | Count of data packets sent by each node $j$. |
| CSR | $CS = \{cs_1, cs_2, \ldots, cs_j, \ldots, cs_n\}$ | Count of control packets sent by each node $j$. |
| DRR | $DR = \{dr_1, dr_2, \ldots, dr_j, \ldots, dr_n\}$ | Count of data packets received by each node $j$. |
| CRR | $CR = \{cr_1, cr_2, \ldots, cr_j, \ldots, cr_n\}$ | Count of control packets received by each node $j$. |
| ECR | $EC = \{ec_1, ec_2, \ldots, ec_j, \ldots, ec_n\}$ | Energy consumed by each node $j$, calculated as the starting energy (from previous beacons) minus the remaining energy (from beacons at the end of the period). |
| DA | $D = \{d_{1\_1}, d_{1\_2}, d_{1\_3}, \ldots, d_{1\_m}, d_{2\_1}, d_{2\_2}, d_{2\_3}, \ldots, d_{2\_m}, \ldots, d_{j\_1}, d_{j\_2}, d_{j\_3}, \ldots, d_{j\_m}, \ldots, d_{n\_1}, d_{n\_2}, d_{n\_3}, \ldots, d_{n\_m}\}$ | All data transmitted by each node $j$, where $m$ denotes the count of data sent by node $j$ and may vary across different nodes. |

This method assumes that the data follows a Gaussian distribution, meaning most data points are concentrated around the mean, and the probability of data points appearing decreases as they deviate further from the mean. Assume the data sample set collected by

the sensor node is $X = \{x_1, x_2, \ldots, x_n\}$. First, it is necessary to calculate the mean $\mu$ and standard deviation σ of the dataset as shown in Equations (1) and (2),

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1}$$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2} \tag{2}$$

which are the basic parameters required for calculating the Gaussian distribution. For each data point $x$, the probability density value $P(x)$ can be calculated using the Gaussian distribution probability density function as shown in Equation (3):

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3}$$

where $\frac{1}{\sigma\sqrt{2\pi}}$ is the normalization constant of the Gaussian distribution, ensuring the total probability is 1, and $\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ is the probability density value for each data point, reflecting the degree of deviation of the data point from the mean. $P(x)$ is not the actual probability but the probability density value, indicating the relative frequency of occurrence near that point. The higher the density, the closer the data point is to the center (mean), and the higher the probability of occurrence; the lower the density, the further it is from the mean, possibly indicating an outlier. The curve of a Gaussian distribution probability density function is shown in Figure 5. In this figure, the mean ($\mu$) is 50 and the standard deviation (σ) is 10.

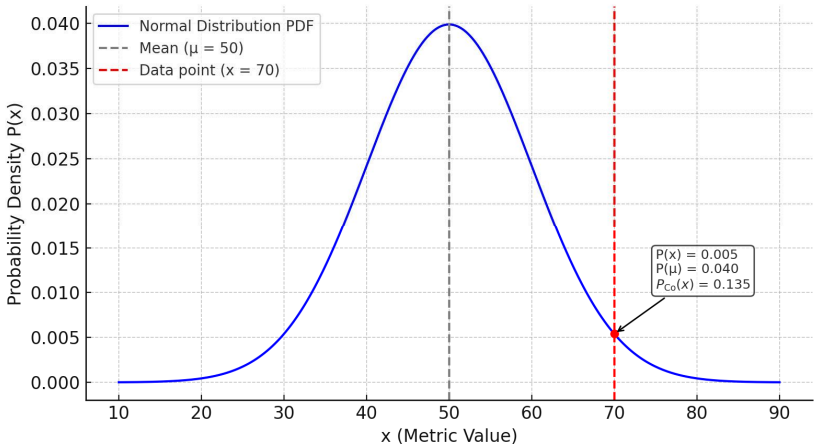


**Figure 5.** Gaussian distribution PDF curve.

Then, we define the cooperation probability of a data point as the ratio between its probability density and that of the mean, as shown in Equation (4),

$$P_{Co}(x) = \frac{P(x)}{P(\mu)} = \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{4}$$

where $P_{Co}(x)$ represents the cooperation probability of $x$. The function curve of our cooperation probability is shown in Figure 6. In this figure, the values of paraments $\mu$ and σ are the same as in Figure 5.

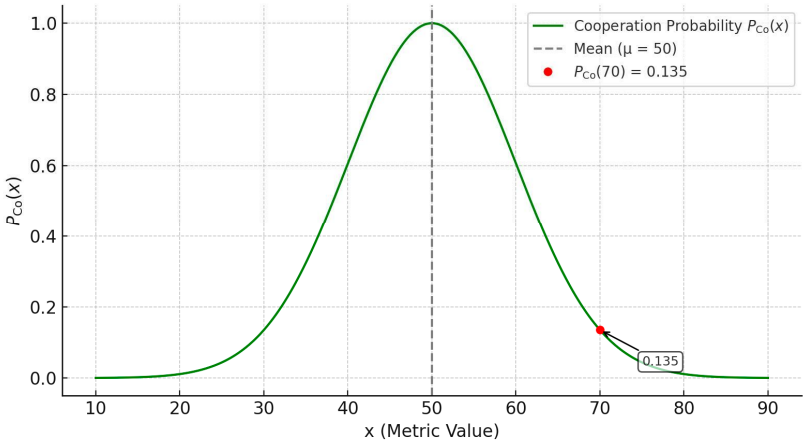


**Figure 6.** Cooperation probability function curve.

From this figure, we can see the range of the cooperation probability is from 0 to 1. The closer the value is to the mean, the closer its cooperation probability is to 1; the further away from the mean, the closer its cooperation probability is to 0.

Since a single node $j$ sends multiple data packets ($m$ packets, where $m$ varies per node), a unique method is used to compute its overall DA cooperation probability. The final $P^{da}_{Co\_ij}$ for a node is the arithmetic mean of the cooperation probabilities calculated for each individual data packet it sent during the period, as shown in Equation (5),

$$P^{da}_{Co\_ij} = \frac{\sum_{k=1}^{m} P^{da}_{Co\_ij\_k}}{m} \tag{5}$$

where $P^{da}_{Co\_ij}$ represents the DA cooperation probability of node $j$, $P^{da}_{Co\_ij\_k}$ represents the cooperation probability of the DA for the $k$th data transmitted by node $j$, and $m$ represents the total number of items sent by node $j$ during the evaluation period.

### 3.2.3. The Method of Direct Trust Value Calculation

The direct trust value is then computed using the classical Bayesian Beta approach. Ganeriwal et al. [11,41] provided a detailed investigation into how Bayesian formulations and Beta distributions can be applied to trust modelling within WSNs. They extended the cooperation metric from a binary rating to an interval rating, meaning it is no longer simply cooperative or non-cooperative but is a probability. They applied the Dirichlet process and derived that the parameter update formulas for binary ratings are similarly applicable to interval ratings. After a single transaction, if the assigned cooperation probability is $P_{Co\_j}$, the Beta parameters are updated as shown as Equation (6).

$$\alpha^{new}_{ij} = \alpha_{ij} + P_{Co\_ij} \quad \beta^{new}_{ij} = \beta_{ij} + 1 - P_{Co\_ij} \tag{6}$$

From the viewpoint of node $i$, the parameters $\alpha_{ij}$ and $\beta_{ij}$ characterize the cooperative and non-cooperative behaviours observed between nodes $i$ and $j$ at the initiation of a transaction. Similarly, $\alpha^{new}_{ij}$ and $\beta^{new}_{ij}$ characterize the cooperative and non-cooperative behaviours of nodes $i$ and $j$ as observed by node $i$ upon the completion of a single transaction. The trust score of node $j$ computed from node $i$'s perspective is the expected value of the beta distributions, which can be easily computed, as shown in Equation (7).

$$T_{ij} = E\left(Beta\{\alpha_{ij}, \beta_{ij}\}\right) = \frac{\alpha_{ij}}{\alpha_{ij} + \beta_{ij}} \tag{7}$$

We consider the behaviour of sending or receiving packets within an evaluation period as a single transaction. By substituting Equation (6) into Equation (7), we can derive the method for calculating the direct trust values for DSR, CSR, DRR, CRR, ECR, and DA at the end of an evaluation period, as shown in Equation (8),

$$\begin{aligned} T_{ij}^{ds} &= \frac{\left(\alpha_{ij}^{ds} + P_{Co_i ij}^{ds}\right)}{\left(\alpha_{ij}^{ds} + \beta_{ij}^{ds} + 1\right)} \\ T_{ij}^{cs} &= \frac{\left(\alpha_{ij}^{cs} + P_{Co_i ij}^{cs}\right)}{\left(\alpha_{ij}^{cs} + \beta_{ij}^{cs} + 1\right)} \\ T_{ij}^{dr} &= \frac{\left(\alpha_{ij}^{dr} + P_{Co_i ij}^{dr}\right)}{\left(\alpha_{ij}^{dr} + \beta_{ij}^{dr} + 1\right)} \\ T_{ij}^{cr} &= \frac{\left(\alpha_{ij}^{cr} + P_{Co-ij}^{cr}\right)}{\left(\alpha_{ij}^{cr} + \beta_{ij}^{cr} + 1\right)} \\ T_{ij}^{ec} &= \frac{\left(\alpha_{ij}^{ec} + P_{Co_{ij}}^{ec}\right)}{\left(\alpha_{ij}^{ec} + \beta_{ij}^{ec} + 1\right)} \\ T_{ij}^{da} &= \frac{\left(\alpha_{ij}^{da} + P_{Coij}^{da}\right)}{\left(\alpha_{ij}^{da} + \beta_{ij}^{da} + 1\right)} \end{aligned} \tag{8}$$

for each metric $X$, where $X$ represents six behavioural metrics (DSR, CSR, DRR, CRR, ECR, and DA), respectively. $\alpha_{ij}^{X}$ and $\beta_{ij}^{X}$ represent the count of cooperative interactions and the count of noncooperative interactions of metric $X$ of node $j$. $P_{Co\_ij}^{X}$ represents the assigned cooperation probability for node $j$ with metric $X$. This probability is calculated based on observed behaviours. It is used to update the $\alpha_{ij}^{X}$ and $\beta_{ij}^{X}$ parameters. $T_{ij}^{X}$ represents the direct trust value of node $j$ in metric $X$. This value is computed from $\alpha_{ij}^{X}$ and $\beta_{ij}^{X}$ at the end of the period. The relationship between these parameters for all six metrics is summarized in Table 2.

**Table 2.** The relationship between these parameters for all six metrics.

| Metric | Cooperative ($\alpha_{ij}^{X}$) | Non-Cooperative ($\beta_{ij}^{X}$) | Direct Trust ($T_{ij}^{X}$) | Cooperation Prob. ($P_{Co\_ij}^{X}$) |
|---|---|---|---|---|
| DSR | $\alpha_{ij}^{ds}$ | $\beta_{ij}^{ds}$ | $T_{ij}^{ds}$ | $P_{Co\_ij}^{ds}$ |
| CSR | $\alpha_{ij}^{cs}$ | $\beta_{ij}^{cs}$ | $T_{ij}^{cs}$ | $P_{Co\_ij}^{cs}$ |
| DRR | $\alpha_{ij}^{dr}$ | $\beta_{ij}^{dr}$ | $T_{ij}^{dr}$ | $P_{Co\_ij}^{dr}$ |
| CRR | $\alpha_{ij}^{cr}$ | $\beta_{ij}^{cr}$ | $T_{ij}^{cr}$ | $P_{Co\_ij}^{cr}$ |
| ECR | $\alpha_{ij}^{ec}$ | $\beta_{ij}^{ec}$ | $T_{ij}^{ec}$ | $P_{Co_{ij}}^{ec}$ |
| DA | $\alpha_{ij}^{da}$ | $\beta_{ij}^{da}$ | $T_{ij}^{da}$ | $P_{Co\_ij}^{da}$ |

#### 3.2.4. The Specific Case of DFR and CFR

The direct trust values for DFR and CFR are also calculated using the classical Bayesian Beta method, as shown in Equation (5). However, unlike other trust metrics, there is no need to calculate the cooperation probability of the dataset. Instead, the cooperation probability is defined as a binary number, either 1 or 0, as in Equation (4), where $P_{Co\_ij}$ is 1 or 0. When it is confirmed through monitoring or ACK methods that node $j$ has successfully forwarded

a packet, $P_{Co\_ij}$ is assigned a value of 1; otherwise, it is assigned a value of 0. At the end of the evaluation period, the Beta parameters are updated, as shown in Equation (9),

$$\alpha_{ij}^{new} = \alpha_{ij} + m\overline{P_{Co\_ij}} \quad \beta_{ij}^{new} = \beta_{ij} + \mathrm{m} \; - m\overline{P_{Co\_ij}} \tag{9}$$

where $m$ represents the total number of data or control packets that node $j$ either successfully forwarded or failed to forward during the evaluation period. For each forwarding action, a cooperation rating is assigned: 1 if the packet was forwarded successfully, and 0 otherwise. These ratings form the set: $P_{Co\_ij} = \{P_{Co\_ij\_1}, P_{Co\_ij\_2}, P_{Co\_ij\_3}, \ldots, P_{Co\_ij\_m}\} \in \{0, 1\}$. By substituting Equation (9) into Equation (5), the formula for the direct trust values of DFR and CFR at the end of an evaluation period is obtained, as in Equation (10),

$$\begin{aligned} T_{ij}^{df} &= \frac{\left(\alpha_{ij}^{df} + m\overline{P_{Co\_ij}^{df}}\right)}{\left(\alpha_{ij}^{df} + \beta_{ij}^{df} + m\right)} \\ T_{ij}^{cf} &= \frac{\left(\alpha_{ij}^{cf} + m\overline{P_{Co\_ij}^{cf}}\right)}{\left(\alpha_{ij}^{cf} + \beta_{ij}^{cf} + m\right)} \end{aligned} \tag{10}$$

where $T_{ij}^{df}$ and $T_{ij}^{cf}$ represent the direct trust values of DFR and CFR for node $j$, respectively. The parameters $\alpha_{ij}^{df}$, $\beta_{ij}^{df}$, $\alpha_{ij}^{cf}$, and $\beta_{ij}^{cf}$ represent the counts of cooperative and non-cooperative forwarding actions for data and control packets, respectively, at the beginning of an evaluation period. For initialization, all Beta parameters are set to zero.

#### 3.2.5. Calculating the Combined Trust Value

When calculating the combined trust value, we use a method that combines the reciprocal method and entropy-based method [42] to assign weights to the direct trust values of each trust metric.

By taking the reciprocal of the direct trust values, we assign higher weights to lower trust values, accelerating the decline of combined trust for anomalous nodes. This helps achieve the goal of quickly identifying malicious nodes. We form a dataset of all direct trust values, organized according to the trust metrics in the sequence: $T_{ij}^{ds}$, $T_{ij}^{cs}$, $T_{ij}^{dr}$, $T_{ij}^{cr}$, $T_{ij}^{ec}$, $T_{ij}^{da}$, $T_{ij}^{df}$, $T_{ij}^{cf}$ represented as $\left\{T_{ij}^{1}, T_{ij}^{2}, T_{ij}^{3}, T_{ij}^{4}, T_{ij}^{5}, T_{ij}^{6}, T_{ij}^{7}, T_{ij}^{8}\right\}$. The weighting scheme for each trust metric, illustrated in Equation (11),

$$\rho_{ij}^{r} = \frac{1}{T_{ij}^{r} + \epsilon} \tag{11}$$

is derived by applying the reciprocal function to each trust value. $\rho_{ij}^{r}$ represents the weighting factor assigned to the direct trust value from node $i$ to node $j$ for the $r$th trust metric. The values of $r$ ranges from 1 to 8. $\epsilon$ is a very small number serving to prevent division by zero.

Building on the reciprocal method, we further use the entropy-based method to adjust the final weights. Entropy-based weighting is often interpreted as reducing the weight of highly uncertain information sources. In contrast, our design adopts a security-oriented interpretation: when a specific trust metric exhibits high volatility within a neighbourhood, this volatility frequently reflects a stronger discriminative power for identifying abnormal behaviour. Thus, after reciprocal weighting penalizes low trust values, entropy is used to amplify metrics that better separate benign from malicious behaviours. By calculating the "volatility" (entropy) of each metric, we automatically identify which metrics are more

important in the network scenario and accordingly increase the weights of these trust values. This allows the trust evaluation to automatically adapt to different network or attack scenarios. For example, in scenarios where the attack affects transmission rates, such as alarm systems, the trust value weights of communication metrics are automatically strengthened. In scenarios where the attack aims to affect DA, such as medical monitoring systems, the impact of DA metrics is automatically amplified.

Shannon's information theory defines entropy as a fundamental metric for quantifying uncertainty in a system [42]. If a random variable $X$ has a set of possible values $\{x_1, x_2, \ldots, x_n\}$ with a corresponding probability distribution $\{p(x_1), p(x_2), \ldots, p(x_n)\}$, then the entropy $(H)$ is defined as shown in Equation (12):

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i) \tag{12}$$

where $H(X)$ represents entropy of the random variable $X$; $p(x_i)$ represents the probability of the event $x_i$; and $b$ represents the base of the logarithm, typically 2 (for bits), $e$ (for natural entropy, in nats), or 10. The more uniform the probability distribution, the greater the entropy, and the higher the uncertainty of the system. The choice of the logarithmic base in entropy calculation significantly affects the sensitivity and discrimination of the resulting weights. In this study, we choose base $b = 2$ for the entropy function, as it offers higher sensitivity to 'probability imbalance'. To illustrate the effect of the logarithm base, consider a simple probability distribution: Let $P = [0.8, 0.2]$. Then, entropy values computed with different bases are show in Table 3.

**Table 3.** Entropy values computed with different bases.

| Bases | Entropy Values |
|---|---|
| 2 | $-(0.8 \log_2 0.8 + 0.2 \log_2 0.2) \approx 0.7219$ |
| e | $-(0.8 \ln 0.8 + 0.2 \ln 0.2) \approx 0.5004$ |
| 10 | $-(0.8 \log_{10} 0.8 + 0.2 \log_{10} 0.2) \approx 0.2173$ |

Among these three entropy values, the binary entropy value is the largest, indicating that it is more sensitive to the degree of value fluctuation. Furthermore, we calculate the information entropy with bases 2, $e$, and 10 for the probability distributions (0.8, 0.2), (0.6, 0.4), and (0.5, 0.5), respectively, and perform a comparative analysis, as shown in Table 4.

**Table 4.** Entropy values of different probability distributions computed with different bases.

| Probability Distribution | Entropy Value with Base 2 | Entropy Value with Base $e$ | Entropy Value with Base 10 |
|---|---|---|---|
| (0.8, 0.2) | 0.7219 | 0.5004 | 0.2173 |
| (0.6, 0.4) | 0.9710 | 0.6730 | 0.2923 |
| (0.5, 0.5) | 1.0000 | 0.6931 | 0.3010 |

From Table 4, we can see that the difference in entropy values for different probability distributions is the largest when the base is 2. Therefore, in our trust model, when using base 2 for the entropy function, can more clearly distinguish which trust metrics are more important. For normalization, we use the factor $k = 1/\log_2 n$, which ensures that the entropy values are scaled to the [0, 1] interval.

First, we 'normalize' each trust metric to form a pseudo-probability distribution, so that it can be used as an input for information entropy to measure the fluctuation of each metric. Suppose the dataset of the $r$th trust metric collected by node $i$ from all its

neighbouring nodes is $X_i^r = \{x_{i1}^r, x_{i2}^r, \ldots, x_{in}^r\}$. Normalize this trust metric as shown in Equation (13),

$$p\left(x_{ij}^r\right) = \frac{x_{ij}^r}{\sum_{j=1}^{n} x_{ij}^r} \tag{13}$$

where $n$ denotes the count of immediate neighbours of node $i$. $P(X_i^r) = \{p(x_{i1}^r), p(x_{i2}^r), \ldots, p(x_{in}^r)\}$ is the pseudo-probability distribution corresponding to $X_i^r = \{x_{i1}^r, x_{i2}^r, \ldots, x_{in}^r\}$, $\sum_{j=1}^{n} p\left(x_{ij}^r\right) = 1$.

To quantify uncertainty in the $r$th trust metric, we employ Equation (12) to compute its entropy and normalize it as shown in Equation (14),

$$\theta_i^r = -\frac{1}{\log_2 n} \sum_{j=1}^{n} p\left(x_{ij}^r\right) \log_2 p\left(x_{ij}^r\right) \tag{14}$$

where $\theta_i^r$ represents the normalized entropy of the $r$th metric calculated by node $i$ within its neighbourhood. The smaller the $\theta_i^r$, the greater the variation, and the more important the metric, which should be given a higher weight. The entropy weight factor calculation method is shown in Equation (15),

$$\lambda_i^r = \frac{1 - \theta_i^r}{\sum_{r=1}^{8} (1 - \theta_i^r)} \tag{15}$$

where $\lambda_i^r$ represents the entropy weight factor for the $r$th metric calculated by node $i$ within its neighbourhood.

#### 3.2.6. Combining Reciprocal Weighting and Entropy-Based Weighting

Following the reciprocal computation, the weights are modified using the adjustment factor $\lambda_i^r$, and normalized to produce the final set of weights, as illustrated in Equation (16),

$$w_{ij}^r = \frac{\rho_{ij}^r \lambda_i^r}{\sum_{\mathrm{r}=1}^{8} \rho_{ij}^r \lambda_i^r} \tag{16}$$

where $w_{ij}^r$ represents the weight of the $r$th trust metric.

Finally, to derive the combined trust score, we compute the weighted average of all trust metrics, as specified in Equation (17),

$$CT_{ij} = \sum_{\mathrm{r}=1}^{8} w_{ij}^r T_{ij}^r \tag{17}$$

where $CT_{ij}$ denotes the combined trust score assigned by node $i$ to node $j$.

#### 3.2.7. Updating the Local Trust Value at the Node

We adopt an asymmetric update strategy in which trust values drop rapidly upon detecting abnormal behaviour and recover gradually during normal operation. This design follows common principles in trust management for WSNs, where rapid decay improves responsiveness to potential attacks while slow recovery prevents malicious nodes from regaining trust too quickly [11]. Previous research showed that the unified decay–recovery strategy provides a stable performance across these behaviours without requiring attack-specific tuning [21,43].

By weighing the historical and current combined trust values with an aging factor, we derive the local trust value. To achieve a rapid decline in trust value when a node exhibits malicious behaviour and a slow recovery of trust value after the node returns to normal behaviour, we use a dynamically and automatically adjusted aging factor. Based on the

logistic function [44] we perform a nonlinear transformation to convert the change from the historical to the current combined trust value into an aging factor. We introduce explicit parameters for the slope ($k$) and midpoint ($x_0$), making the behaviour of the aging factor more transparent and tunable, as shown in Equation (18),

$$w_{ij}^{age} = \frac{1}{1 + \exp\left(-k\left(\Delta CT_{ij} - x_0\right)\right)} \tag{18}$$

where $t$ denotes the current moment, $\Delta CT_{ij} = CT_{ij}(t) - CT_{ij}(t - \Delta t)$, $CT_{ij}(t - \Delta t)$ and $CT_{ij}(t)$ denote the combined trust values from the previous and current evaluation periods, respectively, $\Delta t$ denotes the evaluation period, and $w_{ij}^{age}$ denotes the aging factor.

As indicated by the formula, a higher current combined trust value relative to the historical value results in an aging factor close to 1, while a lower value leads it closer to 0.

Then, the combined trust value updated through the aging factor is used as the local trust value, as shown in Equation (19),

$$LT_{ij} = w_{ij}^{age} CT_{ij}(t - \Delta t) + \left(1 - w_{ij}^{age}\right) CT_{ij}(t) \tag{19}$$

where $LT_{ij}$ represents the value of local trust. When a node exhibits malicious behaviour, the current combined trust value decreases, the aging factor becomes smaller, and the weight of the current combined trust value increases, causing the local trust value to decline rapidly. Conversely, when the node returns to normal behaviour, the current combined trust value increases, the aging factor becomes larger, and the weight of the historical combined trust value increases, causing the local trust value to rise slowly.

#### 3.2.8. Reporting the Local Trust Value

Each node needs to periodically send its local trust values for all its neighbours to the CH and the controller for trust aggregation. To avoid introducing an additional transmission overhead, we use the method from ETMRM [10] for reporting local trust values. Instead of generating separate trust packets, each node embeds the trust values into the SDN-WISE report messages that are sent to the controller as part of the standard control-plane operation. As in the native SDN-WISE architecture, the cluster head forwards the report messages from all nodes in the cluster to the controller, enabling centralized trust aggregation. SDN-WISE report messages use a compact binary format optimized for resource-constrained sensor nodes. Figure 7 shows the message structure used in this work.

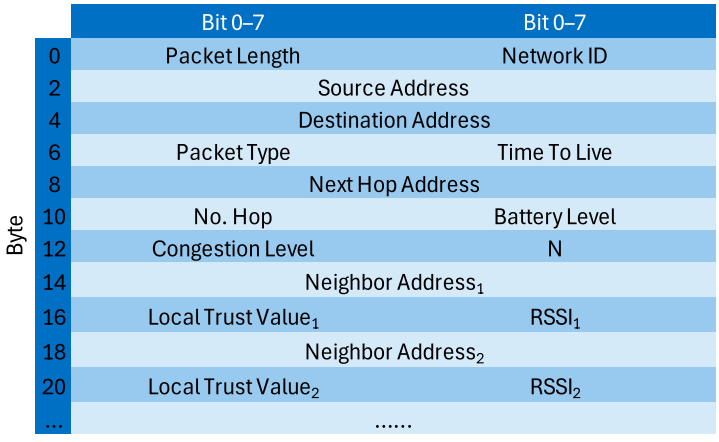


**Figure 7.** The format of the SDN-WISE report message.

To ensure that trust dissemination remains lightweight, the real-valued trust scores computed in the previous section (ranging from 0 to 1 and represented as 4-byte floating-point numbers) are converted into 1-byte unsigned integers in the range of [0, 100], following the approach in ETMRM [10], as shown in Equation (20).

$$LT_{ij} = \lceil 100 \cdot LT_{ij} \rceil \tag{20}$$

This compression significantly reduces the size of the embedded trust data and enables the inclusion of multiple trust values without increasing the overall packet size beyond SDN-WISE's normal control-message budget. Consequently, the trust-reporting mechanism integrates seamlessly with existing SDN-WISE control-plane messages and incurs negligible communication costs.

### *3.3. CH-Level Operations*

#### 3.3.1. Collecting the Local Trust Values of Cluster Member Nodes

The CH node also calculates local trust values for its neighbours. After receiving the SDN-WISE report message from its cluster member nodes, the CH node reads the reported local trust values from these messages. These secondhand trust values, along with the local trust values calculated by the CH node itself, are stored in the global array of the CH in the form of a matrix, as shown in Equation (21),

$$\left[LT_{ij}\right] = \begin{bmatrix} LT_{11} & LT_{12} & \dots & LT_{1n} \\ LT_{21} & LT_{22} & \dots & LT_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ LT_{n1} & LT_{n2} & \dots & LT_{nn} \end{bmatrix} \tag{21}$$

where $LT_{ij}$ denotes the local trust value that node $i$ assigns to node $j$, its neighbour. $n$ is the total number of nodes within the cluster, $i, j \in [1, n]$. If node $j$ is not within the neighbourhood of node $i$, then node $i$ cannot perform a trust evaluation on node $j$. In this case, $LT_{ij}$ equals 0. Node $i$ is not allowed to evaluate itself, so its self-trust value is set to 0.

#### 3.3.2. Trust Aggregation at CH Level

In the trust aggregation process, we adopt the approach proposed by Bin-Yahya et al. [12], which incorporates two key metrics: score reliability and node reliability. To compute score reliability, we measure how much a local trust value deviates from the average of all local trust values assigned to the evaluated node by its neighbours. We then use this metric to calculate node reliability and to assign weights to each trust value during aggregation.

To determine node reliability, we take the arithmetic mean of the score reliability values corresponding to a node's trust evaluations of all its neighbours. This metric enables us to identify nodes that may be engaging in trust manipulation behaviours, such as good-mouthing or bad-mouthing attacks.

To calculate the score reliability metric, we first need to compute the arithmetic mean of the local trust values of node $j$ given by all its neighbour nodes within the cluster. We take all elements greater than 0 in the $j$th column of the matrix and calculate their arithmetic mean, as shown in Equation (22),

$$AT_{avg,j} = \frac{\sum_{i \in X} LT_{ij}}{N_X} \tag{22}$$

where $X$ represents the group of neighbouring nodes associated with node $j$, and $N_X$ represents the total count of node $j$'s neighbours. Then, we calculate the score reliability of the local trust value assigned by node $i$ to node $j$ using Equation (23).

$$SR_{ij} = 100 - \left| LT_{ij} - AT_{avg,j} \right| \tag{23}$$

Then, to compute the node reliability metric, we use the score reliability metrics as inputs, as shown in Equation (24),

$$NR_i = \frac{\sum_{j \in Y} SR_{ij}}{N_Y} \tag{24}$$

where $NR_i$ represents the node reliability metric of node $i$, $Y$ represents the set of neighbour nodes of node $i$, and $N_Y$ represents the total count of node $i$'s neighbours. The node reliability metric is iteratively updated using the historical and current values according to the method in Equation (19).

To aggregate local trust values based on the score reliability metric and the node reliability metric, we set a lower threshold $NR_{th}$ for the node reliability metric, and we only aggregate the local trust values given by nodes whose node reliability metric exceeds this threshold. We derive the aggregated trust value using the method outlined in Equation (25),

$$AT_{C,j} = \frac{\sum_{i \in C} LT_{ij} \cdot SR_{ij}}{\sum_{i \in C} SR_{ij}} \tag{25}$$

where C represents the group of node $j$'s neighbour nodes within the cluster whose node reliability metric exceeds the threshold. $LT_{ij}$ is the local trust value given by node $i$ of node $j$, and $SR_{ij}$ is the score reliability metric of that local trust value. $AT_{C,j}$ is the aggregated trust value of node $j$ at the CH level.

#### 3.3.3. Forwarding the Local Trust Values

CH periodically sends its local trust values for all its neighbours to the controller through the SDN-WISE report message. Additionally, it forwards the SDN-WISE report messages from the cluster member nodes to the controller. These messages contain the local trust values computed by each node for its immediate neighbours.

#### 3.3.4. Identifying and Isolating Malicious Nodes at CH Level

The CH detects malicious nodes by analysing the aggregated trust values of cluster members. It then stops forwarding any messages to the malicious nodes and stops forwarding any messages from the malicious nodes.

### *3.4. Controller-Level Operations*

#### 3.4.1. Collecting the Local Trust Values of All Nodes in the Network

Upon receiving an SDN-WISE message from a network node, the controller extracts the local trust values of neighbouring nodes embedded in the message. These values are then organized into a matrix and stored in the controller's global array, following the method outlined in Equation (21). This trust matrix offers a comprehensive global view of the network's trust landscape.

#### 3.4.2. Trust Aggregation at Controller Level

In conducting trust aggregation at the controller level, we apply the same approach as that used at the CH level. The computation is grounded in the trust matrix, which provides a global perspective, as previously introduced.

3.4.3. Identifying and Isolating Malicious Nodes at Controller Level

The controller detects malicious nodes by evaluating the aggregated trust values within the network. It then adjusts flow rules. If the detected malicious node is the CH, a new CH needs to be selected, and the network topology needs to be re-established. When a malicious node $j$ is identified as a regular node, a control message with a drop rule is sent by the controller to the CH of the corresponding cluster. The CH then propagates the control message throughout its cluster by forwarding it to each member node. Upon receiving this control message, the member nodes modify the flow table stored locally. This modification includes inserting a drop rule <src_addr = 'j', action = 'Drop'> into the flow table and removing the rule with <next_hop = 'j'> from the flow table to avoid forwarding packets to the malicious node. In SDN-WISE, the flow table includes three parts: Matching Rules, Actions, and Statistics. Figure 8 is an example of a flow table in SDN-WISE. For specific explanations of each field, please refer to SDN-WISE [8,34].

| Matching Rule | | | | | Action | | | | | Statistics | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Op. | Size | S | Addr. | Value | Type | M | S | Addr. | Value | TTL | Counter |
| = | 2 | 0 | 2 | A | Drop | 0 | 0 | - | - | 100 | 42 |
| = | 2 | 0 | 2 | B | Forward | 0 | 0 | 0 | D | 100 | 32 |

**Figure 8.** An example of a flow table in SDN-WISE. Different background colours in the table represent different functional categories: the blue back-ground indicates the matching-rule fields used for packet comparison; the light blue background denotes the action fields specifying how the packet is processed; and the orange background rep-resents statistical information related to the rule, such as TTL and the packet counter.

*3.5. Retrieval of Trust Information*

We assume that the controller has unlimited storage resources, allowing for full backup of trust information, and the single point of failure problem of the controller is not addressed in our work. Therefore, only ordinary sensor nodes and CH nodes need to retrieve trust information after it is lost. Since the controller acts as a global repository of trust information for the entire network and each CH manages the trust data of its respective cluster, both cluster heads and ordinary nodes have the ability to recover missing trust records by accessing the controller or the CH. Therefore, we define a pair of new messages based on the existing SDN-WISE messages to request and return trust information. These messages are only exchanged when trust information is lost and do not burden normal communication. The trust information request message does not carry any payload, while the trust reply conveys either cluster-wide trust information or the trust record of the requesting node. The message structures are shown in Figure 9.

The "No. Node" field in the trust information reply message indicates the number of nodes whose trust information is included in the message. If a node requests trust information from the CH, the value of this field in the CH's reply message is 1. When a cluster head (CH) requests trust data from the controller, the reply message contains a field specifying the number of nodes that belong to the CH's cluster. Within the message format, "Node1 Address" identifies the address of Node 1, while "No. Neighbour" indicates how many neighbouring nodes are associated with Node 1. Following this, the reply provides the detailed addresses and trust values of each neighbouring node. Node 2's trust information is added sequentially after Node 1's in the same structure, and this continues until the trust records of all cluster nodes are included.

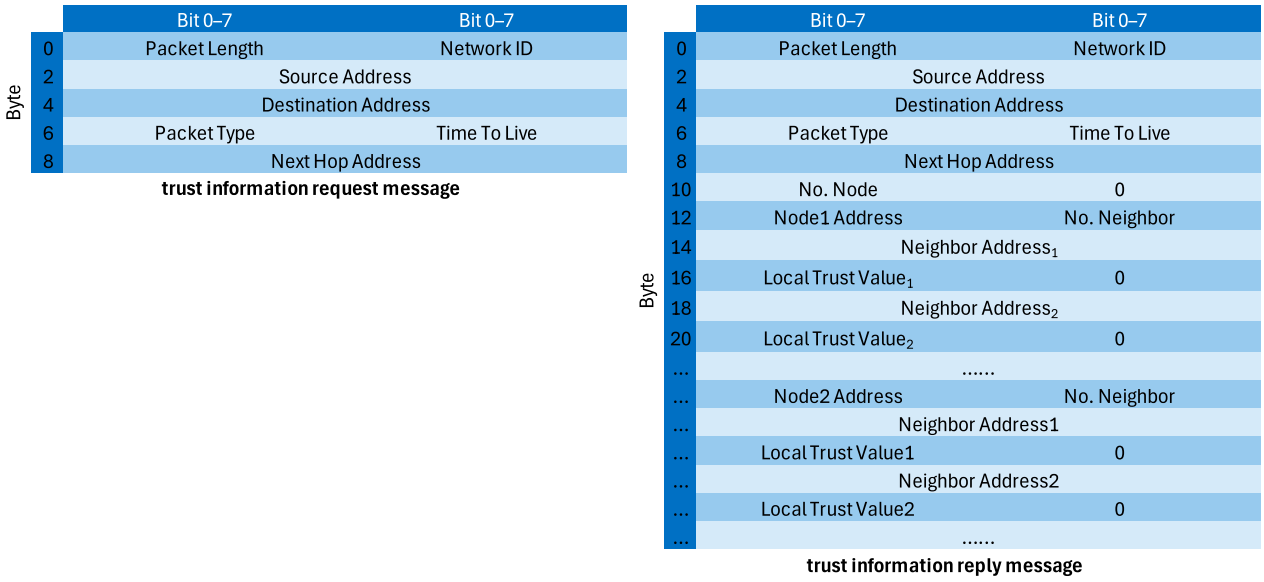


**Figure 9.** The format of the trust information request/reply message.

We incorporate a set of lightweight protection mechanisms suitable for the security of this recovery mechanism. First, all trust retrieval requests and replies should be authenticated by cryptography technologies [45] to prevent forgery. Second, the controller and cluster heads enforce per-node rate limiting through token-bucket mechanisms [46] to restrict excessive requests. Third, suspicious request patterns (e.g., bursts of retrieval messages) are monitored by the controller and can trigger temporary throttling or blacklist actions. These mechanisms provide protection without incurring significant overheads.

## 4. Results and Analysis

In this section, we present the theoretical analysis of the key algorithm in ATAW-TM. We implemented these algorithms in Python 3.13.9 and applied them to representative datasets to analyse and evaluate their performance.

### *4.1. Theoretical Analysis Using Cooperation Probability and Bayesian Inference*

We begin by analysing the calculation of direct trust values of eight trust metrics using cooperation probabilities and Bayesian inference. We take the DSR metric as an example. Cooperation probabilities are computed using a Gaussian distribution-based formula, and trust values are subsequently calculated through Bayesian inference.

#### 4.1.1. Arithmetic Mean and Standard Deviation Calculation

The dataset consists of data packet sending counts from 20 neighbouring nodes in one evaluation period. The values are as follows:

$$\text{Data}: x_i = \{100, 110, 105, 99, 101, 98, 103, 102, 97, 150, 104, 99, 107, 97, 103, 99, 104, 106, 96, 98\}$$

The mean ($\mu$) and standard deviation ($\sigma$) of the dataset are calculated using Equations (1) and (2). The result is $\mu = \frac{1}{20}\sum_{i=1}^{20} x_i = 103.9$, $\sigma = \sqrt{\frac{1}{20}\sum_{i=1}^{20}(x_i - 103.9)^2} = 11.1933$.

4.1.2. Cooperation Probability Calculation

The cooperation probability for each node is computed using Equation (4).

$$P_{Co}(x) = \exp\left(-\frac{(x-103.9)^2}{2*11.1933^2}\right).$$

4.1.3. Using Bayesian Inference to Calculate Direct Trust Values

The Bayesian inference process is used to calculate the trust values of each node. Initially, the parameters of the Beta distribution are set to $\alpha_0 = 0$ and $\beta_0 = 0$. We use Equation (6) to update the parameters of the Beta distribution and use Equation (8) to calculate the direct trust values. The calculated direct trust values for each node are as shown in Table 5.

**Table 5.** Direct trust values for each node.

| Node | Data Packet Sending Count | Direct Trust Value |
|---|---|---|
| Node 1 | 100 | 0.941106 |
| Node 2 | 110 | 0.862004 |
| Node 3 | 105 | 0.995183 |
| Node 4 | 99 | 0.908630 |
| Node 5 | 101 | 0.966995 |
| Node 6 | 98 | 0.870300 |
| Node 7 | 103 | 0.996773 |
| Node 8 | 102 | 0.985697 |
| Node 9 | 97 | 0.826960 |
| **Node 10 *** | **150** | **0.000207** |
| Node 11 | 104 | 0.999960 |
| Node 12 | 99 | 0.908630 |
| Node 13 | 107 | 0.962375 |
| Node 14 | 97 | 0.826960 |
| Node 15 | 103 | 0.996773 |
| Node 16 | 99 | 0.908630 |
| Node 17 | 104 | 0.999960 |
| Node 18 | 106 | 0.982555 |
| Node 19 | 96 | 0.779532 |
| Node 20 | 98 | 0.870300 |

* The bolded values correspond to Node 10, which is the malicious node in the experiment.

4.1.4. Direct Trust Value Calculation Analysis

The calculated direct trust values indicate the level of trust each node has within the network based on their observed cooperation behaviour. The number of data packets sent by Node 10 is 150, much higher than the average of 103.9, which is inconsistent with the behaviour of other nodes. The calculated direct trust value of Node 10's DSR is close to 0. This effectively identifies Node 10 as a potential malicious node exhibiting a flooding attack. For clarity, we have plotted the data from Table 5 in Figure 10.

To demonstrate the model's adaptability, we applied it to a second scenario with a higher baseline data rate:

Data: $x_i =$

$\{1110, 1000, 1020, 995, 1010, 980, 1040, 1020, 970, 1600, 1040, 980, 1060, 970, 1040, 990, 1050, 960, 970, 1030\}$.

In this scenario, the model similarly identified the outlier (Node 10 with a count of 1600), as shown in Figure 11.

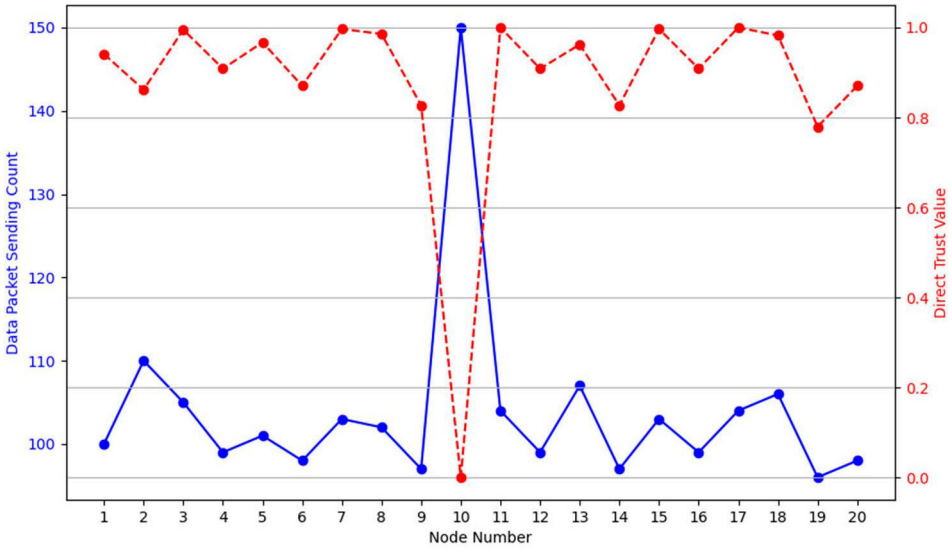


**Figure 10.** Data packet sending count vs. direct trust value in scenario 1. Different colours in the figure represent different metrics: the blue line shows the data packet sending count for each node, and the red line represents the corresponding direct trust value. The sharp deviation at Node 10 highlights the malicious node in this scenario.

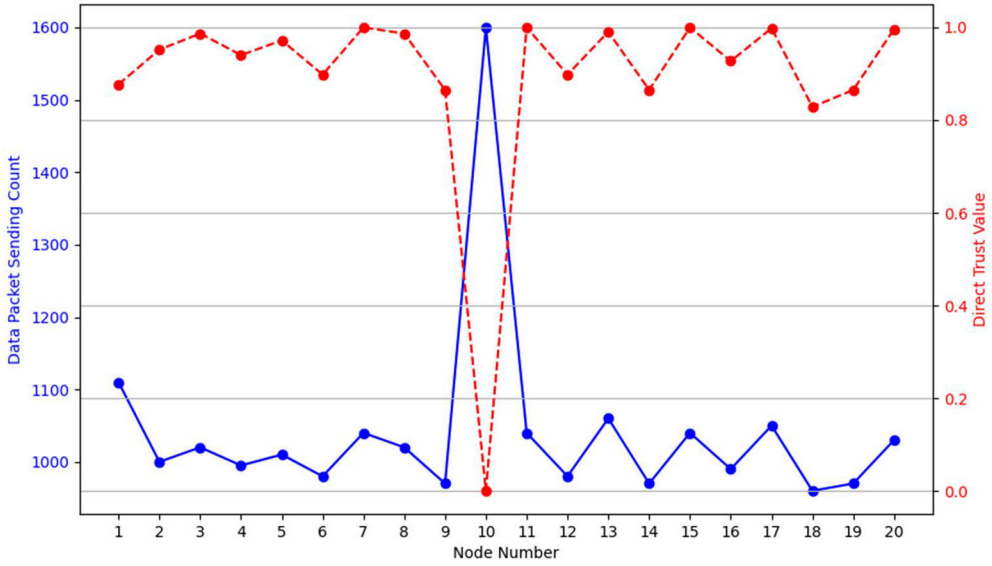


**Figure 11.** Data packet sending count vs. direct trust value in scenario 2. Different colours in the figure represent different metrics: the blue line shows the data packet sending count for each node, and the red line represents the corresponding direct trust value. The sharp deviation at Node 10 highlights the malicious node in this scenario.

These results demonstrate ATAW-TM's ability to function effectively across different network conditions without manual threshold reconfiguration.

### *4.2. Theoretical Analysis of the Hybrid Reciprocal and Entropy-Based Weighting Strategy*

This subsection analyses the hybrid weighting strategy that dynamically assigns importance to the eight-trust metrics, enabling ATAW-TM to adapt to various network conditions and attack patterns.

4.2.1. Direct Trust Value Calculations

Assume we have calculated the direct trust values for eight trust metrics (DSR, CSR, DRR, CRR, ECR, DA, DFR, CFR) for a specific node $j$ from the perspective of node $i$. The trust values are as follows:

$$\text{Direct Trust Values} : T_{ij}^{r} = \{0.9411,\ 0.8952,\ 0.9086,\ 0.9670,\ 0.3269,\ 0.5350,\ 0.8703,\ 0.9857\}$$

where $r = 1$ to 8 represent the eight trust metrics, respectively. The low ECR (0.3269) and moderate DA (0.5350) trust values suggest potential malicious activity in energy consumption and data accuracy. Other metrics' trust values are very high.

4.2.2. Reciprocal Weighting Calculation

The reciprocal weighting factors are computed using Equation (11), where $\epsilon = 0.0001$ to prevent division by zero. The calculated reciprocal weights are as follows:

$$\rho_{ij}^{r} = \{1.062, 1.117, 1.100, 1.034, 3.058, 1.869, 1.149, 1.014\}$$

4.2.3. Entropy-Based Weight Calculation

To compute entropy weights, we first collect trust metric datasets from all neighbouring nodes. Assume node $i$ has five neighbours with the following trust values for each metric; among them, the volatility of ECR and DA is relatively high, as shown in bold:

$$X_i^1 = \{0.9411,\ 0.8620,\ 0.9952,\ 0.8002,\ 0.9099\}\ \text{DSR}$$

$$X_i^2 = \{0.8952,\ 0.9780,\ 0.8870,\ 0.9920,\ 0.9030\}\ \text{CSR}$$

$$X_i^3 = \{0.9086,\ 0.8950,\ 0.9120,\ 0.9510,\ 0.8880\}\ \text{DRR}$$

$$X_i^4 = \{0.9670,\ 0.9540,\ 0.9020,\ 0.8610,\ 0.8580\}\ \text{CRR}$$

$$X_i^5 = \{\mathbf{0.3269},\ \mathbf{0.8950},\ \mathbf{0.9120},\ \mathbf{0.9010},\ \mathbf{0.8880}\}\ \text{ECR}$$

$$X_i^6 = \{\mathbf{0.5350},\ \mathbf{0.7150},\ \mathbf{0.8980},\ \mathbf{0.6560},\ \mathbf{0.8740}\}\ \text{DA}$$

$$X_i^7 = \{0.8703,\ 0.8560,\ 0.9820,\ 0.8410,\ 0.9670\}\ \text{DFR}$$

$$X_i^8 = \{0.9857,\ 0.9720,\ 0.8910,\ 0.9780,\ 0.8040\}\ \text{CFR}$$

We normalize these values using Equation (13) to form pseudo-probability distributions. The normalized entropy is calculated using Equation (14). The entropy weight factors are computed using Equation (15):

$$\lambda_i^r = \frac{1-\theta_i^r}{\sum_{r=1}^{8}\left(1-\theta_i^r\right)} = \{0.034, 0.014, 0.004, 0.016, 0.650, 0.218, 0.027, 0.037\}$$

4.2.4. Combined Weight Calculation

The final weights are computed by combining reciprocal and entropy weights using Equation (16):

$$w_{ij}^r = \frac{\rho_{ij}^r \lambda_i^r}{\sum_{r=1}^{8} \rho_{ij}^r \lambda_i^r} = \{0.014, 0.006, 0.002, 0.006, 0.784, 0.161, 0.012, 0.015\}$$

#### 4.2.5. Combined Trust Value Calculation

The combined trust score is calculated using Equation (17):

$$CT_{ij} = \sum_{r=1}^{8} w_{ij}^{r} T_{ij}^{r} = 0.3940$$

The weighting analysis results are consolidated in Table 6 and Figure 12.

**Table 6.** Weighting analysis results.

| Trust Metric | Direct Trust Value | Reciprocal Weight | Entropy Weight | Combined Weight | Weighted Trust Value |
|---|---|---|---|---|---|
| DSR | 0.9411 | 1.062 | 0.034 | 0.014 | 0.0136 |
| CSR | 0.8952 | 1.117 | 0.014 | 0.006 | 0.0056 |
| DRR | 0.9086 | 1.100 | 0.004 | 0.002 | 0.0014 |
| CRR | 0.9670 | 1.034 | 0.016 | 0.006 | 0.0062 |
| ECR | 0.3269 | 3.058 | 0.650 | 0.784 | 0.2563 |
| DA | 0.5350 | 1.869 | 0.218 | 0.161 | 0.0860 |
| DFR | 0.8703 | 1.149 | 0.027 | 0.012 | 0.0105 |
| CFR | 0.9857 | 1.014 | 0.037 | 0.015 | 0.0144 |
| Combined Trust Value 0.3940 | | | | | |

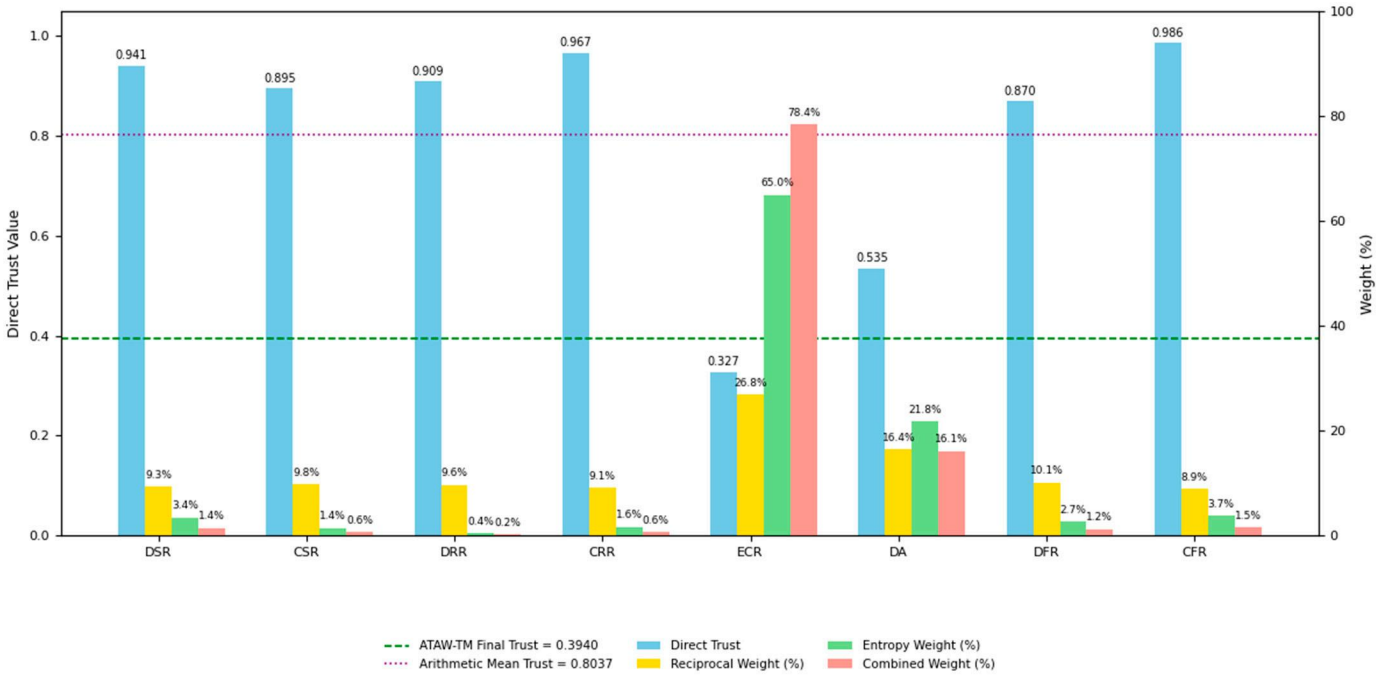


**Figure 12.** Direct trust value and weight comparison.

#### 4.2.6. Analysis

The results demonstrate the effectiveness of the hybrid weighting strategy. The ECR metric, despite having the lowest direct trust value (0.3269), receives the highest final weight (0.784) due to the combined effect of reciprocal weighting (emphasizing low values) and entropy weighting (reflecting its importance in energy-constrained scenarios). The ECR metric and DA metric have higher entropy weighting due to their higher volatility. Although the direct trust values of most metrics are very high, the combined trust value (0.3940) calculated using our weighting method accurately reflects the node's overall malicious behaviour. Compared to fixed-weight averaging (0.8037), ATAW-TM achieved a combined trust score of 0.3940, representing a 51% increase in detection sensitivity.

### *4.3. Theoretical Analysis of the Dynamically Adjusted Aging Factor*

Finally, we analyse the aging factor mechanism, which enables the trust model to respond rapidly to malicious behaviour while allowing for slow, stable recovery. In this subsection, we set the parameters in the aging factor calculation equation (Equation (18)) to $k = 1.0$ and $x_0 = 0.0$. A parameter and sensitivity analysis will be conducted in the next subsection.

#### 4.3.1. Trust Value Inputs

Assume that for a given node $j$, the historical combined trust value from the previous evaluation period is $CT_{ij}(t - \Delta t) = 0.850$, and the current combined trust value is $CT_{ij}(t) = 0.394$. These values reflect a significant decline in trust, suggesting potential malicious activity during the current period.

#### 4.3.2. Aging Factor Calculation

The aging factor $w_{ij}^{age}$ is computed using Equation (18):

$$w_{ij}^{age} = \frac{1}{1 + \exp(0.456)} \approx \frac{1}{1 + 1.578} \approx 0.388$$

#### 4.3.3. Updating the Local Trust Value

The local trust value $LT_{ij}$ is then updated using Equation (19):

$$LT_{ij} = 0.388 \times 0.850 + (1 - 0.388) \times 0.394 = 0.329 + 0.241 = 0.570$$

To further illustrate the behaviour of the aging factor under different scenarios, we consider two additional scenarios:

Scenario 1: Trust Improvement

If $CT_{ij}(t - \Delta t) = 0.400$ and $CT_{ij}(t) = 0.800$, then

$$w_{ij}^{age} = \frac{1}{1 + \exp(-0.400)} \approx \frac{1}{1 + 0.670} \approx 0.599$$

$$LT_{ij} = 0.599 \times 0.400 + 0.401 \times 0.800 = 0.240 + 0.321 = 0.561$$

Scenario 2: Stable Trust

If $CT_{ij}(t - \Delta t) = 0.700$ and $CT_{ij}(t) = 0.710$, then

$$w_{ij}^{age} = \frac{1}{1 + \exp(-0.010)} \approx \frac{1}{1 + 0.990} \approx 0.502$$

$$LT_{ij} = 0.502 \times 0.700 + 0.498 \times 0.710 = 0.351 + 0.354 = 0.705$$

The results are summarized in Table 7.

**Table 7.** Trust change vs. the aging factor.

| Scenario | CT(t-Δt) | CT(t) | ΔCT | Aging Factor | Local Trust |
|---|---|---|---|---|---|
| Trust Decline | 0.850 | 0.394 | −0.456 | 0.388 | 0.570 |
| (additional) Trust Improvement | 0.400 | 0.800 | 0.400 | 0.599 | 0.561 |
| (additional) Stable Trust | 0.700 | 0.710 | 0.010 | 0.502 | 0.705 |

We have plotted the relationship between the trust change $\Delta CT = CT_{ij}(t) - CT_{ij}(t - \Delta t)$ and the aging factor $w_{ij}^{age}$ in Figure 13.

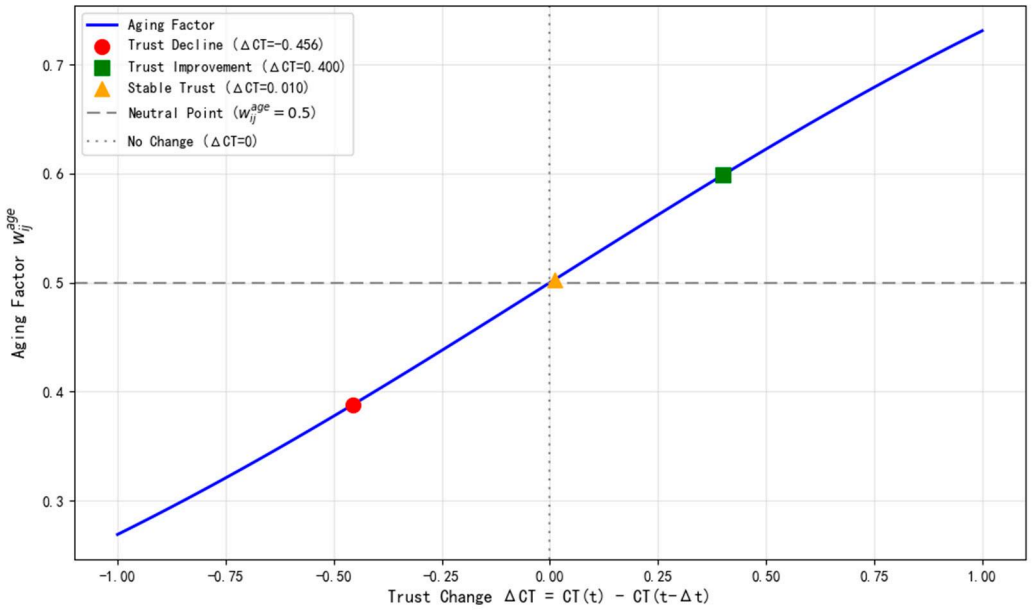


**Figure 13.** Trust change vs. aging factor.

#### 4.3.4. Parameter and Sensitivity

This subsection investigates the impact of the logistic aging parameters $k$ and $x_0$ and evaluates their sensitivity.

Parameter interpretation:

- Slope parameter $k$: A larger $k$ increases the steepness of the logistic curve. This leads to faster trust decay when $\Delta CT$ becomes negative, improving attack responsiveness but making the model more sensitive to transient noise.
- Midpoint parameter $x_0$: This value determines the tolerance centre. A positive $x_0$ shifts the logistic curve so that only sufficiently large negative deviations trigger rapid trust decay.

Sensitivity study:

To further analyse the impact of the logistic aging parameters and justify the selection of $(k, x_0)$, Figure 14 presents a detailed sensitivity study of the aging factor $w_{ij}^{age}$ and its effect on the local trust update process.

Panel (a) shows that the slope parameter $k$ directly controls the steepness of the logistic curve. A small $k$ results in a smooth and gradual change in $w_{ij}^{age}$, making the system tolerant to small fluctuations in $\Delta CT$. As $k$ increases, the curve becomes sharper, allowing the model to react aggressively to deviations in trust behaviour. This capability is beneficial when rapid isolation of malicious nodes is required, but it may introduce false positives in traffic with high variability.

Panel (b) illustrates how the midpoint $x_0$ shifts the logistic response horizontally. A positive $x_0$ delays the decrease in $w_{ij}^{age}$, providing resilience against bursty but benign traffic patterns, whereas a negative $x_0$ increases the sensitivity of the trust decay mechanism. For instance, when $\Delta$CT ranges from $-0.25$ to $-0.75$, the aging factor with $x_0 = -0.10$ decreases faster than that with $x_0 = 0.00$. This parameter therefore allows the trust system to be tuned for different SDWSN deployments and levels of environmental noise.

Panel (c) quantitatively evaluates the sensitivity of $w_{ij}^{age}$ with respect to $k$ under a representative trust decrease $\Delta$CT $= -0.2$. As $k$ increases, $w_{ij}^{age}$ drops sharply toward zero, demonstrating how steep logistic functions accelerate the suppression of trust values after

anomalous behaviour is observed. This analysis supports selecting moderate $k$ values (e.g., $k = 5$) to maintain a balance between responsiveness and robustness.

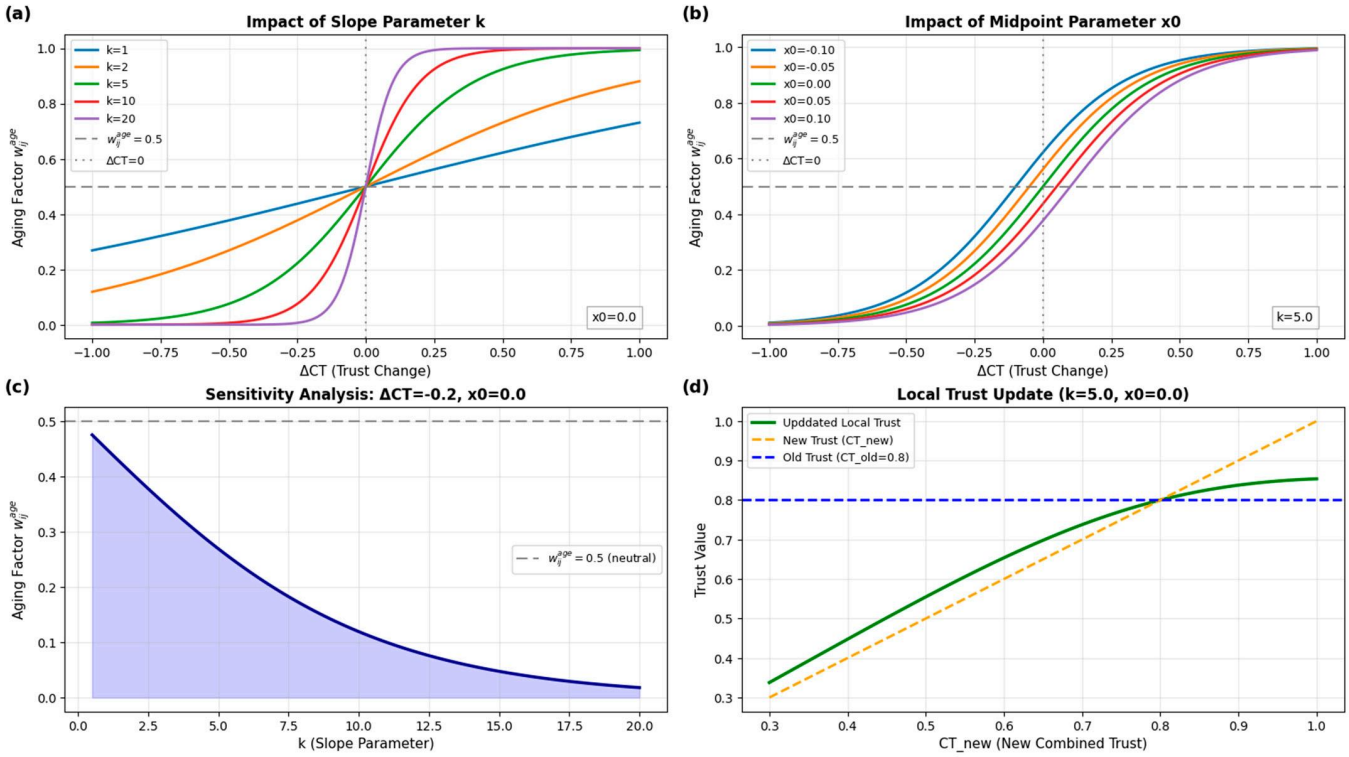


**Figure 14.** Sensitivity analysis of the logistic aging function under different slope parameters $k$, midpoints $x_0$, and trust update behaviours. (**a**) Impact of slope parameter $k$: Larger $k$ makes the aging factor curve steeper and increases the sensitivity of the system to trust changes, while smaller $k$ produces smoother transitions. (**b**) Impact of midpoint parameter $x_0$: Shifting $x_0$ moves the logistic curve horizontally, adjusting the tolerance range so that small fluctuations near $\Delta CT = 0$ can be either emphasized or suppressed. (**c**) Sensitivity analysis: When ΔCT is fixed (e.g., $\Delta CT = -0.2$), increasing $k$ leads to a monotonic decrease in the aging factor, revealing how $k$ controls the reaction strength to a given trust change. (**d**) Local trust update: The updated local trust reflects the weighted combination of historical trust and new trust.

Panel (d) shows the resulting impact on the local trust update process. When $k = 5$ and $x_0 = 0$, the updated trust curve remains stable under small negative fluctuations but reacts more firmly when continuous negative trust evidence accumulates. This behaviour aligns with the design goal of penalizing persistent malicious behaviour while avoiding excessive punishment for normal variation. It also reflects the design goal of a quick response to malicious activity coupled with a slow, steady recovery.

Overall, Figure 14 demonstrates that the logistic aging mechanism provides tuneable flexibility. By appropriately selecting $k$ and $x_0$, the proposed model can balance responsiveness to actual attacks and robustness to transient fluctuations, ensuring reliable operation in diverse SDWSN environments.

If the deployment environment is prone to transient fluctuation or bursty traffic, smaller $k$ values are advisable to avoid unnecessary trust oscillation. Conversely, if rapid identification of malicious nodes is critical, larger $k$ values help accelerate trust decay.

#### 4.3.5. Dynamic Ageing Mechanism Analysis

In terms of robustness, the results demonstrate the effectiveness of the dynamically adjusted aging factor in balancing historical and current trust values. When trust declines significantly (e.g., due to malicious behaviour), the aging factor decreases, giving more

weight to the current low trust value. Conversely, when trust improves, the aging factor increases, emphasizing historical trust. Importantly, this asymmetric aging factor also can mitigate false positives during transient disturbances by incorporating historical trust values into the final trust score. For example, in the Trust Decline scenario shown in Table 7, when trust drops from 0.850 to 0.394, the adjusted final local trust value becomes 0.570, which smooths the abrupt change compared to using only the current value. This mechanism enhances the model's robustness in dynamic SDWSN environments.

In terms of detection sensitivity, this dynamically adjusted aging factor is a crucial defence against On–Off Attacks, where a malicious node switches between good and bad behaviour to build trust and then exploit it. ATAW-TM counters this with an asymmetric response:

- During the "Off" (malicious) phase, a sharp drop in current trust leads to a small aging factor, causing the local trust value to plummet rapidly for quick detection and isolation.
- During the "On" (normal) phase, recovery is intentionally slow. A larger aging factor prioritizes the node's poor history, preventing it from quickly regaining trust after brief good behaviour.

This mechanism significantly raises the cost and reduces the effectiveness of on–off attacks, thereby mitigating the associated risk.

As discussed above, this dynamic aging mechanism achieves a balance between model robustness and detection sensitivity.

### *4.4. Computational Complexity Analysis*

A key advantage of the proposed trust model is its low computational complexity, which makes it suitable for resource-constrained SDWSNs. The local trust computation for each node is based on lightweight operations, including counting forwarding events, calculating simple deviations, and performing scalar updates of trust values. For a node with $d$ neighbours, the complexity of local trust evaluation is $O(d)$, as each neighbour is processed once per trust update cycle.

The cooperation probability calculation also relies on basic arithmetic using precomputed mean and deviation terms, avoiding expensive operations such as matrix computation or iterative training as required in learning-based approaches. Trust aggregation at the cluster head only involves summing or averaging trust values received from cluster members, resulting in $O(n_c)$ complexity, where $n_c$ is the number of nodes in the cluster.

The overall computational cost of the full trust update process is therefore linear in the number of neighbour relationships in the network and does not depend on the network size beyond local connectivity. This lightweight design ensures that the model can operate efficiently on SDWSN sensor nodes with limited CPU, memory, and energy resources, while maintaining responsiveness to DoS-related misbehaviour.

## 5. Discussion

The ATAW-TM model proposed above was designed specifically for SDWSNs. The model addresses key limitations of existing trust models, particularly those related to threshold configurations, weight allocation, and the loss of trust information in the presence of DoS attacks. ATAW-TM enhances the detection of DoS attacks and supports dynamic, self-adjusting trust evaluations by adopting a layered architecture with components at the node, CH, and controller levels. While the sensor nodes in the node layer evaluate their trust locally by assessing the link quality, extracting trust evidence, and calculating direct trust values, in the CH layer, these values are processed to calculate aggregated trust scores

and to identify and isolate malicious nodes. IN the controller layer, the process of global trust aggregation isolates malicious nodes by adjusting flow rules.

The core principle of ATAW-TM is adaptability in response to changing node network behaviour; the depictable trust evaluation mechanism removes the need for manual threshold configuration, through outlier detection and Bayesian inference. The model automatically computes trust values and assigns weights to various trust metrics using a hybrid reciprocal and entropy-based weighting strategy. This ensures that the model can dynamically adjust its focus based on the prevailing network conditions, making it highly adaptable to different attack types. Additionally, the incorporation of a logistic function-based aging factor balances the influence of historical and current trust valueS, allowing for more accurate evaluations of node behaviour over time. The model features a trust information retrieval mechanism as well to facilitate rapid recovery when trust information is lost, ensuring the resilience and stability of the network. ATAW-TM improves reliability in dynamic SDWSN environments by integrating link quality checks and trust information retrieval mechanisms, ensuring robustness against intermittent connectivity.

### *5.1. Comparison with Other Models*

We propose a resilient trust model for SDWSNs, designed to effectively identify various types of DoS attacks. Compared to the previous trust models designed for SDWSNs [10,12,13,33], the proposed model has the following advantages.

First, our trust model considers and addresses the issue of wireless link qualities between sensor nodes being affected by the environment, while others did not consider link qualities at all. Moreover, our trust model is comprehensively designed to account for multiple types of DoS attacks, whereas other models typically consider only a subset of these attacks.

Second, our trust model does not use thresholds for evaluating packet transmission behaviour, eliminating the need for network administrators to configure thresholds, thus making it easily applicable to various WSNs. Previous models such as the ones described in [10,12] used threshold-based algorithms, which had issues with threshold configuration. The model proposed in [13] did not use threshold-based algorithms but considered using a complex IDS module, whereas our model is more lightweight.

Third, in our trust model, the weights of various trust metrics are dynamically and automatically assigned based on their values through an algorithm when calculating the combined trust value. This prevents a malicious node's good performance in one aspect from masking its malicious behaviour in another aspect. In previous trust models [12,33], researchers left the task of assigning weights to users, which raised a significant obstacle standing in the way of the widespread application of trust models. In [10], fixed weights were used for the metrics of the data packet forwarding rate and control packet forwarding rate, resulting in poor flexibility and accuracy. The model proposed in [13] did not involve this weight assignment issue but considered using a complex IDS module.

Furthermore, our trust model uses a dynamically adjusted aging factor in the iterative update process of historical and current trust values, effectively balancing historical and current trust values. The models described in [10,13] did not consider historical trust values, while the model introduced in [12] considered historical trust values but used a fixed aging factor, resulting in poor flexibility and accuracy.

Additionally, we designed a trust information retrieval mechanism that allows for the quick retrieval of trust information for sensor nodes and cluster head nodes after trust information is lost, enabling rapid recovery of the trust system. Previous models did not consider the issue of trust information loss.

Finally, for ATAW-TM, local trust computation yields a per-node time complexity of $O(d)$. Cluster head aggregation incurs $O(n_c d)$ operations for a cluster of size $n_c$. The communication overhead is $O(d)$ bytes per reporting interval due to 1-byte trust compression. ETMRM and HTM exhibit similar per-node computational costs. Compared to ETMRM and HTM, ATAW-TM introduces two additional steps, namely Gaussian collaboration and entropy weighting, while still preserving linear time complexity. Moreover, the constant factor related to exponential operations can be reduced using lookup table methods.

### *5.2. Limitations and Directions for Further Research*

While the theoretical analysis of the ATAW-TM trust model provides insights into its practical application for evaluating node trustworthiness in SDWSNs, a major limitation of this work is that the ATAW-TM has not yet been implemented or evaluated experimentally. This limitation will be addressed in our future work, including the implementation and evaluate of ATAW-TM in real-world scenarios and conducting an in-depth analysis of its performance. The first step is to develop a prototype using the SDN-WISE platform and the Contiki operating system within the Cooja simulation environment. The second step will involve simulating various (14) attack scenarios to test the effectiveness of the proposed trust model. Following this, we will evaluate the model using performance metrics such as detection accuracy, false positive/negative rates, trust convergence speed, communication overhead, and energy efficiency. A rigorous analysis will be conducted through quantitative simulation experiments.

Furthermore, we will explore the integration of the ATAW-TM trust model with SDWSN routing protocols to improve their efficiency and expedite the network's response to DoS attacks, ensuring stable operations even during malicious disruptions.

Additionally, future work will explore controller redundancy and failover strategies to complement ATAW-TM's trust retrieval mechanism.

Although Gaussian modelling was selected based on prior studies demonstrating its suitability for large-scale WSN traffic patterns, many WSN applications generate bursty or event-driven traffic that may follow Poisson or heavy-tailed distributions [47–49]. Our future work will investigate these alternative distributions to enhance adaptability in diverse environments. We also acknowledge that Gaussian modelling may introduce false positives under highly bursty workloads, where mean–standard-deviation-based outlier detection can misclassify legitimate traffic spikes as malicious. To mitigate this risk, non-parametric outlier detection techniques will be explored in future work to relax the normality assumption and better accommodate heavy-tailed or event-driven traffic patterns.

In addition, integrating adaptive trust update strategies that adjust decay rates or penalty weights based on the detected anomaly type represents a valuable direction for future work. Such adaptive mechanisms could incorporate behaviour signatures or temporal characteristics to further improve detection accuracy and reduce false positives.

**Author Contributions:** Conceptualization, L.W., M.L.Y. and K.P.; methodology, L.W., M.L.Y. and K.P.; validation, M.L.Y.; formal analysis, L.W.; investigation, L.W.; resources, M.L.Y.; data curation, L.W.; writing—original draft preparation, L.W., M.L.Y. and K.P.; writing—review and editing, L.W., M.L.Y. and K.P.; visualization, L.W.; supervision, M.L.Y. and K.P.; project administration, M.L.Y. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The data supporting the conclusions of this article are available in the text.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References


1. Akyildiz, I.F.; Su, W.; Sankarasubramaniam, Y.; Cayirci, E. Wireless sensor networks: A survey. *Comput. Netw.* **2002**, *38*, 393–422. [CrossRef]
2. Bono, F.M.; Polinelli, A.; Radicioni, L.; Benedetti, L.; Castelli-Dezza, F.; Cinquemani, S.; Belloli, M. Wireless Accelerometer Architecture for Bridge SHM: From Sensor Design to System Deployment. *Future Internet* **2025**, *17*, 29. [CrossRef]
3. Yick, J.; Mukherjee, B.; Ghosal, D. Wireless sensor network survey. *Comput. Netw.* **2008**, *52*, 2292–2330. [CrossRef]
4. Gungor, V.C.; Hancke, G.P. Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches. *IEEE Trans. Ind. Electron.* **2009**, *56*, 4258–4265. [CrossRef]
5. Miyazaki, T.; Yamaguchi, S.; Kobayashi, K.; Kitamichi, J.; Song, G.; Tsukahara, T.; Hayashi, T. A software defined wireless sensor network. In Proceedings of the 2014 International Conference on Computing, Networking and Communications (ICNC), Honolulu, HI, USA, 3–6 February 2014; pp. 847–852.
6. Kobo, H.I.; Abu-Mahfouz, A.M.; Hancke, G.P. A Survey on Software-Defined Wireless Sensor Networks: Challenges and Design Requirements. *IEEE Access* **2017**, *5*, 1872–1899. [CrossRef]
7. Bukar, U.A.; Othman, M. Architectural Design, Improvement, and Challenges of Distributed Software-Defined Wireless Sensor Networks. *Wirel. Pers. Commun.* **2022**, *122*, 2395–2439. [CrossRef]
8. Galluccio, L.; Milardo, S.; Morabito, G.; Palazzo, S. SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks. In Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM), Hong Kong, China, 2015; pp. 513–521.
9. Al-Hamid, D.Z.; Karegar, P.A.; Chong, P.H.J. A novel SDWSN-based testbed for IoT smart applications. *Future Internet* **2023**, *15*, 291. [CrossRef]
10. Wang, R.; Zhang, Z.; Zhang, Z.; Jia, Z. ETMRM: An Energy-efficient Trust Management and Routing Mechanism for SDWSNs. *Comput. Netw.* **2018**, *139*, 119–135. [CrossRef]
11. Ganeriwal, S.; Balzano, L.K.; Srivastava, M.B. Reputation-based framework for high integrity sensor networks. *ACM Trans. Sens. Netw.* **2008**, *4*, 1–37. [CrossRef]
12. Bin-Yahya, M.; Alhussein, O.; Shen, X. Securing Software-Defined WSNs Communication via Trust Management. *IEEE Internet Things J.* **2022**, *9*, 22230–22245. [CrossRef]
13. Isong, B.; Manuel, M.; Dladlu, N.; Abu-Mahfouz, A. Trust Management Framework for Securing Software-Defined Wireless Sensor Networks. In Proceedings of the 2023 International Conference on Electrical, Computer and Energy Technologies (ICECET), Cape Town, South Africa, 16–17 November 2023; pp. 1–6.
14. Wang, L.; Petrova, K.; Yang, M.L. Trust Models in Wireless Sensor Networks for Defending Against Denial-of-Service Attacks: A Literature Review. *Appl. Sci.* **2025**, *15*, 3075. [CrossRef]
15. Alhandi, S.A.; Kamaludin, H.; Alduais, N.A.M. Trust Evaluation Model in IoT Environment: A Comprehensive Survey. *IEEE Access* **2023**, *11*, 11165–11182. [CrossRef]
16. Tyagi, H.; Kumar, R.; Pandey, S.K. A detailed study on trust management techniques for security and privacy in IoT: Challenges, trends, and research directions. *High-Confid. Comput.* **2023**, *3*, 100127. [CrossRef]
17. Usman, M.; Asghar, M.R.; Ansari, I.S.; Granelli, F.; Qaraqe, M. Trust-Based DoS Mitigation Technique for Medical Implants in Wireless Body Area Networks. In Proceedings of the ICC 2019—2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–6.
18. Ahmed, A.; Qureshi, K.N.; Anwar, M.; Masud, F.; Imtiaz, J.; Jeon, G. Link-based penalized trust management scheme for preemptive measures to secure the edge-based internet of things networks. *Wirel. Netw.* **2024**, *30*, 4237–4259. [CrossRef]
19. Wu, X.; Huang, J.; Ling, J.; Shu, L. BLTM: Beta and LQI Based Trust Model for Wireless Sensor Networks. *IEEE Access* **2019**, *7*, 43679–43690. [CrossRef]
20. Zhang, M.; Feng, R.; Zhang, H.; Su, Y. A recommendation management defense mechanism based on trust model in underwater acoustic sensor networks. *Future Gener. Comput. Syst.* **2023**, *145*, 466–477. [CrossRef]
21. Almutairi, A.; Carpent, X.; Furnell, S. Towards a Mobility-Aware Trust Model for the Internet of Underwater Things. In Proceedings of the ICT Systems Security and Privacy Protection, Edinburgh, UK, 26 July 2024; pp. 1–15.
22. Anwar, R.W.; Zainal, A.; Outay, F.; Yasar, A.; Iqbal, S. BTEM: Belief based trust evaluation mechanism for Wireless Sensor Networks. *Future Gener. Comput. Syst.* **2019**, *96*, 605–616. [CrossRef]
23. Jinhui, X.; Yang, T.; Feiyue, Y.; Leina, P.; Juan, X.; Yao, H. Intrusion Detection System for Hybrid DoS Attacks using Energy Trust in Wireless Sensor Networks. *Procedia Comput. Sci.* **2018**, *131*, 1188–1195. [CrossRef]
24. Isaac Sajan, R.; Jasper, J. A secure routing scheme to mitigate attack in wireless adhoc sensor network. *Comput. Secur.* **2021**, *103*, 102197. [CrossRef]
25. Rahamathullah, U.; Karthikeyan, E. A lightweight trust-based system to ensure security on the Internet of Battlefield Things (IoBT) environment. *Int. J. Syst. Assur. Eng. Manag.* **2021**. [CrossRef]

26. Anand, C.; Vasuki, N. Trust Based DoS Attack Detection in Wireless Sensor Networks for Reliable Data Transmission. *Wirel. Pers. Commun.* **2021**, *121*, 2911–2926. [CrossRef]
27. Cao, Z.; Zhou, X.; Xu, M.; Chen, Z.; Hu, J.; Tang, L. Enhancing Base Station Security Against DoS Attacks in Wireless Sensor Networks. In Proceedings of the 2006 International Conference on Wireless Communications, Networking and Mobile Computing, Wuhan, China, 22–24 September 2006; pp. 1–4.
28. Cho, Y.; Qu, G. Detection and Prevention of Selective Forwarding-Based Denial-of-Service Attacks in WSNs. *Int. J. Distrib. Sens. Netw.* **2013**, *9*, 205920. [CrossRef]
29. Gautam, A.K.; Kumar, R. A Robust Trust Model for Wireless Sensor Networks. In Proceedings of the 2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON), Gorakhpur, India, 2–4 November 2018; pp. 1–5.
30. Han, G.; Shen, W.; Duong, T.Q.; Guizani, M.; Hara, T. A proposed security scheme against Denial of Service attacks in cluster-based wireless sensor networks. *Secur. Commun. Netw.* **2014**, *7*, 2542–2554. [CrossRef]
31. Lyu, C.; Zhang, X.; Liu, Z.; Chi, C.H. Selective Authentication Based Geographic Opportunistic Routing in Wireless Sensor Networks for Internet of Things Against DoS Attacks. *IEEE Access* **2019**, *7*, 31068–31082. [CrossRef]
32. Qureshi, K.N.; Iftikhar, A.; Bhatti, S.N.; Piccialli, F.; Giampaolo, F.; Jeon, G. Trust management and evaluation for edge intelligence in the Internet of Things. *Eng. Appl. Artif. Intell.* **2020**, *94*, 103756. [CrossRef]
33. Bin-Yahya, M.; Shen, X. HTM: Hierarchical Trust Management for Software-Defined WSNs. In Proceedings of the 2019 IEEE Globecom Workshops (GC Wkshps), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6.
34. Anadiotis, A.-C.; Galluccio, L.; Milardo, S.; Morabito, G.; Palazzo, S. SD-WISE: A Software-Defined WIreless SEnsor network. *Comput. Netw.* **2019**, *159*, 84–95. [CrossRef]
35. Mitchell, R.; Chen, I.-R. A survey of intrusion detection techniques for cyber-physical systems. *ACM Comput. Surv. (CSUR)* **2014**, *46*, 1–29. [CrossRef]
36. Alaba, F.A.; Othman, M.; Hashem, I.A.T.; Alotaibi, F. Internet of Things security: A survey. *J. Netw. Comput. Appl.* **2017**, *88*, 10–28. [CrossRef]
37. Niu, R.; Varshney, P.K.; Cheng, Q. Distributed detection in a large wireless sensor network. *Inf. Fusion* **2006**, *7*, 380–394. [CrossRef]
38. Azad, A.K.M.; Kamruzzaman, J. Energy-Balanced Transmission Policies for Wireless Sensor Networks. *IEEE Trans. Mob. Comput.* **2011**, *10*, 927–940. [CrossRef]
39. Rabbat, M.; Nowak, R. Distributed optimization in sensor networks. In Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks, Berkeley, CA, USA, 26–27 April 2004; pp. 20–27.
40. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* **2018**, *1*, 108–116.
41. Han, Y.; Wang, H.; Li, Y.; Zhang, L. Trust-aware and improved density peaks clustering algorithm for fast and secure models in wireless sensor networks. *Pervasive Mob. Comput.* **2024**, *105*, 101993. [CrossRef]
42. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
43. Ye, J.; Jiang, W. Routing Protocol for Underwater Wireless Sensor Networks Based on a Trust Model and Void-Avoided Algorithm. *Sensors* **2024**, *23*, 7614. [CrossRef] [PubMed]
44. Hastie, T.; Tibshirani, R.; Friedman, J. *The elements of statistical learning*; Springer: New York, NY, USA, 2009; pp. 79–127.
45. Yang, M.L.; Al-Anbuky, A.; Liu, W. An Authenticated Key Agreement Scheme for Wireless Sensor Networks. *J. Sens. Actuator Netw.* **2014**, *3*, 181–206. [CrossRef]
46. Fu, H.; Sun, M.; He, B.; Li, J.; Zhu, X. A survey of traffic shaping technology in internet of things. *IEEE Access* **2022**, *11*, 3794–3809. [CrossRef]
47. Alizai, M.H.; Landsiedel, O.; Link, J.A.B.; Götz, S.; Wehrle, K. Bursty traffic over bursty links. In Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, Berkeley, CA, USA, 4–6 November 2009; pp. 71–84.
48. Siam, M.A.Y.; Hasan, F.; Sharmin, S.; Saha, S. An Efficient Burst Traffic-Aware Clustering and Mobility Management model for mobile-sink based Heterogeneous Wireless Sensor Networks. In Proceedings of the 11th International Conference on Networking, Systems, and Security, Khulna, Bangladesh, 19–21 December 2024; pp. 96–103.
49. Chiasserini, C.-F.; Garetto, M. Modeling the performance of wireless sensor networks. In Proceedings of the IEEE INFOCOM 2004, Hong Kong, China, 7–11 March 2004.