The Scientific Temper (2024) Vol. 15 (spl): 170-181

E-ISSN: 2231-6396, ISSN: 0976-8653

Doi: 10.58414/SCIENTIFICTEMPER.2024.15.spl.21

https://scientifictemper.com/

RESEARCH ARTICLE

Trust aware clustering approach for the detection of malicious nodes in the WSN

Rekha R.*, P. Meenakshi Sundaram

Abstract

Wireless sensor networks (WSNs) are pivotal in a range of applications such as environmental monitoring, healthcare, and defense. However, their decentralized and resource-constrained nature makes them vulnerable to various security threats, particularly from malicious nodes that can disrupt the network's functionality. To address this issue, this paper proposes a novel trust aware clustering (LEACH) approach integrated with an optimization-based technique for the detection of malicious nodes in WSNs. The proposed model leverages the low-energy adaptive clustering hierarchy (LEACH) protocol for efficient clustering and energy management while incorporating a trust-based mechanism to evaluate the behavior of nodes. Additionally, an optimization algorithm is employed to enhance the accuracy of malicious node detection and improve the overall network performance. The trust model dynamically updates based on node interactions, ensuring that compromised nodes are detected and isolated promptly. Simulation results demonstrate the efficacy of the proposed approach in terms of increased detection accuracy, reduced energy consumption, and prolonged network lifetime, making it a robust solution for securing WSNs against malicious attacks.

Keywords: Wireless sensor networks, Clustering approach, Low-energy adaptive clustering hierarchy, Malicious node detection.

Introduction

Wireless sensor networks (WSNs) have gained significant attention due to their potential to support a wide range of applications, from environmental monitoring, healthcare, and military surveillance to industrial automation and smart cities. A WSN typically consists of a large number of sensor nodes that are distributed in an area to collect and transmit data to a central base station (BS) or sink node. These sensor nodes, which are resource-constrained in terms of energy, memory, and computational power, communicate with each other and the base station via wireless channels. While WSNs offer a flexible and scalable means of gathering and processing data, they are also highly vulnerable to various security threats, particularly due to their wireless nature, decentralized structure, and unattended deployment in hostile environments, Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020), Ibrahim, D. S., Mahdi, A. F., & Yas, Q. M. (2021), Fahmy, H. M. A., & Fahmy, H. M. A. (2020), Temene, N., Sergiou, C., Georgiou, C., & Vassiliou, V. (2022).

One of the most critical threats in WSNs is the presence of malicious nodes. Malicious nodes can degrade the network's performance by disrupting communication, tampering with data, draining the network's energy resources, and compromising the overall security of the network. Malicious node detection has become a crucial research area, aiming to ensure the integrity, reliability, and efficiency of WSN operations, Gomathi, S., & Gopala Krishnan, C. (2020), Jane Nithya, K., & Shyamala, K. (2022), Anand, C., & Vasuki, N. (2021).

Overview of Security Challenges in WSNs

Before delving into malicious node detection, it is important to understand the inherent vulnerabilities of WSNs, which make them susceptible to attacks, Sharma, S., Bansal, R. K., & Bansal, S. (2013, December), Olakanmi, O. O., & Dada, A. (2020):

Department of Computer Science, Maruthupandiyar College (Affiliated to Bharathidasan University, Tiruchirappalli), Thanjavur, Tamilnadu, India.

***Corresponding Author:** Rekha R, Department of Computer Science, Maruthupandiyar College (Affiliated to Bharathidasan University, Tiruchirappalli), Thanjavur, Tamilnadu, India, E-Mail: npramki@gmail.com

How to cite this article: Rekha, R., Sundaram, P.M. (2024). Trust aware clustering approach for the detection of malicious nodes in the WSN. The Scientific Temper, **15**(spl):170-181.

Doi: 10.58414/SCIENTIFICTEMPER.2024.15.spl.21

Source of support: Nil

Conflict of interest: None.

Received: 18/09/2024 **Accepted:** 25/09/2024 **Published**: 16/10/2024

Limited resources

Sensor nodes have limited energy, processing capabilities, and memory, which makes it difficult to implement computationally heavy security protocols. This limitation makes them prone to energy-draining attacks like the denial of service (DoS).

Unreliable communication

WSNs typically use wireless communication channels, which are inherently insecure. Adversaries can eavesdrop, intercept, or manipulate messages transmitted between sensor nodes.

Physical vulnerability

Since sensor nodes are often deployed in hostile or remote environments, they can easily be captured or physically tampered with by attackers, allowing them to be compromised or turned into malicious nodes.

Lack of centralized control

WSNs often operate in an ad-hoc manner without a central authority to monitor or secure the network. This decentralized structure increases the difficulty of detecting and managing malicious activity.

Malicious Nodes in WSNs

Malicious nodes are sensor nodes within the network that exhibit abnormal or harmful behavior. These nodes can either be compromised by attackers or deployed intentionally as rogue devices. Malicious nodes can perform a variety of attacks that disrupt the network's functionality, such as, Ramasamy, L. K., KP, F. K., Imoize, A. L., Ogbebor, J. O., Kadry, S., & Rho, S. (2021), Lai, Y., Tong, L., Liu, J., Wang, Y., Tang, T., Zhao, Z., & Qin, H. (2022), Nagarjun, S., Anand, S., & Sinha, S. (2019):

Sinkhole attack

In this attack, a malicious node attracts all the traffic in the network by pretending to be a high-quality node (e.g., with the shortest path to the sink). Once it has control over the traffic, it can drop packets or selectively forward malicious information.

Wormhole attack

Two or more malicious nodes collaborate to create a tunnel or shortcut, misleading the network into routing packets through them. This allows attackers to intercept and manipulate the data.

Malicious attack

A malicious node presents multiple fake identities or nodes to the network, thereby disrupting routing protocols, resource allocation, and voting mechanisms.

Blackhole attack

A malicious node drops all packets that it receives, effectively disrupting communication in the network.

Hello flood attack

The malicious node sends high-powered HELLO messages to multiple nodes in the network, tricking them into believing it is a neighboring node. This causes energy depletion as the nodes attempt to communicate with a non-existent neighbor.

Detection Techniques for Malicious Nodes

Various techniques have been proposed to detect malicious nodes in WSNs, each with its advantages and limitations, Yang, H., Zhang, X., & Cheng, F. (2021), Morsi, A. M., Barakat, T. M., & Nashaat, A. A. (2020):

Trust-based systems

Trust-based systems monitor the behavior of nodes to assign a trust score based on their interactions with other nodes. Malicious nodes, which tend to behave abnormally (e.g., dropping packets or forwarding incorrect data), receive lower trust scores and can be isolated from the network.

Intrusion detection systems (IDS)

IDS in WSNs are designed to monitor the network for suspicious activities or policy violations. They can be implemented as either signature-based (detecting known patterns of attacks) or anomaly-based (detecting deviations from normal behavior).

Clustering and cooperative detection

Clustering techniques can group nodes into clusters, with cluster heads responsible for monitoring and detecting malicious activity within their cluster. This hierarchical approach reduces the communication overhead and enhances the scalability of detection mechanisms.

Optimization-based approaches

Optimization techniques, such as genetic algorithms, swarm intelligence, or machine learning, can be used to fine-tune the parameters of detection systems, improving their accuracy and minimizing false positives.

Reputation systems

In these systems, nodes build reputations based on their behavior over time. Nodes with consistently malicious behavior lose reputation points and may eventually be excluded from network activities.

Low-Energy Adaptive Clustering Hierarchy

The low-energy adaptive clustering hierarchy (LEACH) is a widely adopted clustering protocol in WSNs designed to reduce energy consumption and prolong the network lifetime. LEACH enables the self-organization of nodes into clusters, where a subset of nodes, known as cluster heads, are responsible for aggregating and forwarding data to the base station. By rotating the role of cluster heads among

nodes in a probabilistic manner, LEACH aims to distribute energy usage evenly across the network and mitigate premature energy depletion in specific nodes, Kumar, N., Desai, J. R., & Annapurna, D. (2020, July), Sampoornam, K. P., Saranya, S., Mohanapriya, G. K., Devi, P. S., & Dhaarani, S. (2021, February).

Cluster Formation

Setup phase

In the setup phase, each node in the network decides whether to become a cluster head for the current round based on a probabilistic model. The probability of a node becoming a cluster head in a given round is calculated using a predetermined threshold and a random number generated by each node. Nodes elect themselves as cluster heads if their calculated probability exceeds the threshold.

Cluster formation

Once cluster heads are elected, non-cluster-head nodes join the cluster of the nearest cluster-head based on signal strength or other proximity metrics. Each cluster forms a local communication subnet, and nodes communicate with their respective cluster heads.

Data Aggregation and Forwarding

Data collection

Member nodes within each cluster collect data from their surroundings through sensing or data generation processes. These nodes transmit their data to the cluster head for aggregation and forwarding to the base station.

Aggregation and forwarding

Cluster heads aggregate data received from member nodes and transmit the aggregated data to the base station. By reducing redundant transmissions and consolidating data at the cluster level, LEACH minimizes energy consumption and bandwidth usage, thereby prolonging network lifetime.

Cluster Rotation

Energy balancing

To ensure energy usage is balanced across nodes and prevent premature depletion of energy in cluster heads, LEACH employs a rotation mechanism. In each round, cluster heads are re-elected probabilistically based on the predetermined threshold. Nodes that have served as cluster heads in previous rounds are less likely to be elected as cluster heads again, promoting energy balance among nodes.

Dynamic adaptation

The rotation of cluster heads allows nodes to share the energy-intensive task of data aggregation and forwarding over time. Nodes take turns serving as cluster heads, allowing energy-depleted nodes to recover while others take on the cluster head role. This dynamic adaptation mechanism enhances the overall resilience and longevity of the network.

Advantages of LEACH

Energy efficiency

LEACH reduces energy consumption by minimizing redundant data transmissions and aggregating data at the cluster level, thereby prolonging network lifetime.

Decentralized operation

LEACH operates in a decentralized manner, allowing nodes to self-organize into clusters without the need for centralized control or coordination.

Scalability

LEACH is scalable and can accommodate a large number of nodes in the network by dynamically adjusting cluster formations and cluster-head rotations.

Proposed Trust Aware Clustering (TAC) Approach For Malicious Node Detection

Cluster formation

WSN nodes are grouped into clusters using clustering algorithms like LEACH or HEED. Each cluster elects a cluster head responsible for coordinating cluster operations and security monitoring.

Node trust calculation

Each node calculates trust scores for its neighboring nodes based on observed behavior, interactions, and communication patterns. Trust metrics may include factors such as node reputation, reliability, and consistency in adhering to network protocols. The trust calculation algorithm assigns a numerical trust value to each neighboring node, indicating its level of trustworthiness.

Cluster-based trust aggregation

Cluster heads aggregate trust scores from cluster members to compute cluster-level trust values. Trust aggregation techniques such as weighted averaging or reputation propagation may be used to combine individual node trust scores into a collective trust assessment for the entire cluster.

Anomaly detection

Cluster heads analyze aggregated trust values and network behavior to detect anomalies or deviations indicative of Malicious attacks. Anomaly detection algorithms compare observed trust values with expected norms and thresholds to identify suspicious nodes or clusters.

Malicious attack identification

Nodes or clusters exhibiting suspicious behavior or

abnormally low trust scores are flagged as potential Malicious attackers. Malicious nodes are identified based on criteria such as having multiple low-trust identities, inconsistent communication patterns, or unauthorized access to resources.

Response and mitigation

Upon detecting Malicious attacks, appropriate response and mitigation measures are implemented to neutralize the threat and protect the network. Response strategies may include isolating Malicious nodes, revoking their privileges, updating routing tables to avoid compromised paths, or notifying higher-level authorities for further action.

Algorithm: Trust Aware Clustering Approach for Malicious Node Detection

Step 1: Cluster Formation with LEACH

- *Step 1.1: Initialization*

This involves setting up various parameters and variables related to the network, such as node properties, communication range, energy levels, etc. These parameters will be used throughout the algorithm for decision-making and calculations.

- *Step 1.2: Set the desired percentage of cluster heads P*

In clustering algorithms like LEACH (Low Energy Adaptive Clustering Hierarchy), the selection of cluster heads is probabilistic. The parameter P represents the desired percentage of nodes that will become cluster heads. It influences the probability of each node becoming a cluster head.

- *Step 1.3: Set the desired percentage of cluster heads P*

This step initializes a data structure, typically a list, to store information about the clusters formed in the network. Each entry in the list represents a cluster and contains information such as the cluster head node, member nodes, and possibly other metadata related to the cluster.

Step 2: Cluster Head Election (LEACH): For each node i in the network

- *Step 2.1: Calculate the probability Pi of node i becoming a cluster head using LEACH*

LEACH is a randomized algorithm where each node in the network computes its probability of becoming a cluster head based on its residual energy level. The probability Pi is calculated using the formula:

$$P_i = \frac{P}{1 - PX\left(mod\ \left(R_i, \frac{1}{P}\right)\right)}$$

Where P is the desired percentage of cluster heads in the network. R_i is the residual energy of node i. This formula ensures that nodes with higher residual energy have a higher probability of being selected as cluster heads, but it also introduces randomness into the process to evenly distribute the cluster head roles across the network.

- *Step 2.2: Generate random number r_i between 0 and 1*

Each node generates a random r_i number uniformly distributed between 0 and 1.

- *Step 2.3: If $r_i < P_i$, node i becomes a cluster head*

The randomly generated number is r_i compared to the probability P_i calculated for node i.

- *Step 2.4: Add cluster head i to the clusters list*

If node i is selected as a cluster head, it is added to the list of clusters in the network. This list maintains information about the cluster heads selected in the current round, which will be used in subsequent steps of the clustering process.

Step 3: Cluster Formation: For each non-cluster head node

- *Step 3.1: For each non-cluster head node*

This step iterates over all non-cluster head nodes in the network, meaning nodes that have not been selected as cluster heads in the Cluster Head Election phase.

- *Step 3.2: Calculate the distance to each cluster head*

For each non-cluster head node, calculate the distance to every cluster head in the network. The distance metric can be based on various factors such as Euclidean distance, signal strength, or hop count.

- *Step 3.3: Join the cluster of the nearest cluster head*

After calculating distances to all cluster heads, the non-cluster head node selects the cluster head that is closest to it. This selection is typically based on the shortest distance or strongest signal strength, depending on the chosen metric. The non-cluster head node then joins the cluster of the selected nearest cluster head.

- *Step 3.4: Add the node to the respective cluster in the clusters list*

Once a non-cluster head node has determined the nearest cluster head and joined its cluster, the node is added to the respective cluster in the clusters list. The clusters list maintains information about all clusters formed in the network, including the cluster head and member nodes of each cluster.

Step 4: Node Trust Calculation

- *Step 4.1*

This step involves setting up the necessary infrastructure and parameters to perform trust calculations for each node

in the network.

- *Define trust metrics and parameters for trust calculation*

Trust metrics refer to the factors or attributes used to evaluate the trustworthiness of nodes in the network. These metrics may include historical behavior, communication patterns, reliability, and adherence to network protocols. Parameters for trust calculation specify how trust scores will be computed based on the defined metrics. For example, the weight assigned to each metric, thresholds for trust level categorization, and the mathematical formulae for trust score computation.

- *Initialize trust_scores dictionary*

Create a data structure, typically a dictionary, to store the calculated trust scores for each node in the network. The keys of the dictionary represent the node identifiers, and the corresponding values represent the computed trust scores. Initializing the trust_scores dictionary ensures that trust scores can be efficiently recorded and updated for each node during the trust calculation process.

Step 4.2: Calculate Trust Scores

- *For each node i in the network*

This step iterates over each node in the network to compute its trust score based on observed behavior metrics.

- *Calculate the trust score TSi based on observed behavior metrics using a trust calculation function*

For each node i, a trust score TS_i is computed using a trust calculation function. This function takes into account various observed behavior metrics $(B_{i1}, B_{i2}, \ldots, B_{in})$ associated with node i and combines them to produce a numerical value representing the node's trustworthiness. The trust calculation function ff can vary depending on the specific requirements of the WSN and the chosen trust metrics. It may involve weighting different metrics, applying mathematical transformations, or using machine learning algorithms to infer trust scores. The trust score TS_i reflects the node's reputation, reliability, and adherence to network protocols based on its observed behavior. A higher trust score indicates a more trustworthy node, while a lower trust score suggests potential untrustworthiness.

- *Store the trust score TSi in the trust_score directly*

Once the trust score TS_i is computed for node i, it is stored in a data structure such as a dictionary, specifically designed to store trust scores for each node in the network. The trust_scores dictionary maintains a mapping between node identifiers and their corresponding trust scores, allowing for efficient retrieval and updating of trust information during subsequent steps of the algorithm.

Step 5: Cluster based Trust Aggregation

- *Step 5.1: Initialization*

This step involves preparing the necessary infrastructure and data structures for aggregating trust scores at the cluster level.

Initialize variables for trust aggregation

This involves setting up any variables or parameters required for the trust aggregation process. These variables may include counters, accumulators, or other data structures used during the aggregation. For example, variables may be used to keep track of the sum or average of trust scores within each cluster.

- *Initialize cluster_trust_values dictionary*

Create a data structure, typically a dictionary, to store the aggregated trust values for each cluster in the network. Each entry in the cluster_trust_values dictionary represents a cluster and contains the aggregated trust value computed for that cluster. Initializing the cluster_trust_values dictionary ensures that aggregated trust values can be efficiently recorded and updated for each cluster during the trust aggregation process.

Step 5.2: Aggregate Trust Scores

- *For each cluster j in the clusters list*

This step iterates over each cluster in the network, where each cluster consists of a group of nodes led by a cluster head.

- *Calculate the cluster-level trust value CT_j using weighted averaging*

Within each cluster j, the cluster-level trust value CT_j is calculated by aggregating the trust scores (TS_j) of all nodes in the cluster. The trust scores of individual nodes within the cluster are combined using a weighted averaging technique. Each node's trust score is weighted equality in the case. The formula for calculating the cluster-level trust value (CT_j) is

$$\boldsymbol{CT_j} = \frac{\sum_{i \in j} TS_j}{N}$$

- Where TS_j represents the trust scores of node i in cluster j. N is the total number of nodes in cluster j.
- This calculation results in a single numerical value representing the overall trustworthiness of the cluster, derived from the individual trust scores of its member nodes.

- *Store the cluster-level trust value CT_j in the cluster_trust_values dictionary*

Once the cluster-level trust value CT_j is computed for cluster *jj*, it is stored in a data structure such as a dictionary. The cluster_trust_values dictionary maintains a mapping between cluster identifiers and their corresponding cluster-level trust values. Storing the cluster-level trust values allows

for easy access to and retrieval of trust information for each cluster during subsequent steps of the algorithm.

Step 6: Anomaly Detection

- *Step 6.1: Initialization*

This step involves setting up the parameters and thresholds necessary for anomaly detection.

- *Define thresholds and parameters for anomaly detection*

Anomaly detection relies on predefined thresholds and parameters to identify deviations from expected norms in trust values. Thresholds may be set based on empirical observations, historical data, or theoretical models to distinguish between normal and suspicious trust values. Parameters may include criteria for categorizing clusters with low trust values as potential candidates for Malicious attacks.

Step 6.2: Detect anomalies

This step involves analyzing the cluster-level trust values to identify clusters exhibiting anomalous behavior indicative of potential Malicious attacks.

- *For each cluster j in the cluster_trust_values dictionary*

Iterate over each cluster in the network and examine its cluster-level trust value (CT_j).

- *Compare the cluster-level trust value CT_j with expected norms and thresholds*

Compare the computed cluster-level trust value (CT_j) with predefined thresholds and expected norms. Thresholds may indicate the minimum acceptable trust level for a cluster to be considered normal. Clusters with trust values below this threshold are flagged as potential anomalies.

- *Identify clusters with suspiciously low trust values as potential candidates for Malicious attacks*

If the cluster-level trust value (CT_j) falls below the defined threshold, the cluster is identified as exhibiting suspicious behavior. Clusters with low trust values may indicate the presence of Malicious nodes or other malicious activities within the cluster.

Step 7: Malicious Attack Identification

- *Step 7.1: Initialization*

This step involves setting up the criteria and data structures necessary for identifying Malicious nodes**.**

- *Define criteria for identifying Malicious nodes based on anomalous behavior and trust scores*

Criteria are established to distinguish between normal nodes and potential Malicious attackers based on their behavior and trust scores. Anomalous behavior may include inconsistent communication patterns, unauthorized access to resources, or abnormally low trust scores. Trust scores play a crucial role in identifying Malicious nodes, as they indicate the trustworthiness of nodes within the network**.**

- *Initialize malicious_nodes list*

Create a data structure, such as a list, to store the identifiers of nodes flagged as potential Malicious attackers. Initializing the malicious_nodes list allows for the efficient tracking and management of identified Malicious nodes during the identification process.

Step 7.2: Identify Malicious Nodes

This step involves analyzing clusters flagged as potentially containing Malicious nodes and identifying individual nodes exhibiting suspicious behavior.

- *For each cluster flagged as potentially containing Malicious nodes*

Iterate over each cluster identified during the anomaly detection phase.

- *Analyze cluster members for anomalous behavior*

Examine the behavior of nodes within the cluster for characteristics indicative of Malicious attacks. Look for patterns such as multiple identities associated with the same node, inconsistent communication behavior, or unauthorized access attempts.

- *Flag nodes exhibiting suspicious behavior as potential Malicious attackers*

If nodes within the cluster demonstrate behavior consistent with Malicious attacks, flag them as potential Malicious attackers. Add the identifiers of these nodes to the malicious_nodes list for further analysis or mitigation.

Step 8: Response and Mitigation

- *Step 8.1: Input*

This step involves gathering input in the form of response strategies aimed at mitigating Malicious attacks.

- *Response strategies for mitigating Malicious attacks*

Identify and define a set of response strategies tailored to neutralize the threat posed by Malicious attacks. Response strategies may encompass a range of actions aimed at isolating, containing, or eliminating Malicious nodes from the network. These strategies are designed to disrupt the malicious activities of Malicious attackers and restore the integrity and functionality of the WSN.

- *Step 8.2: Response strategies*

This step involves implementing the identified response strategies to neutralize the threat of Malicious attacks.

- *Implement appropriate response and mitigation measures*

Execute the predefined response strategies to counteract Malicious attacks and minimize their impact on the network. Response measures may include:

- *Isolating Malicious nodes*

Preventing Malicious nodes from participating in network activities by disconnecting or blocking their communication.

- *Revoking their privileges*

Removing the network privileges and access rights of identified Malicious nodes to prevent further malicious actions.

Updating routing tables

Modifying routing tables or network configurations to avoid routing traffic through compromised paths associated with Malicious nodes.

- *Notifying higher-level authorities*

Reporting detected Malicious attacks to higher-level network administrators or security authorities for further investigation or intervention.

The selection and execution of response strategies should be guided by the severity of the Malicious attacks, network conditions, and the potential impact on network performance and stability.

Result And Discussion

In this section, the performance of the proposed TAC approach is evaluated with the other trust-aware clustering techniques like K-Means, spectral clustering, and density-based clustering approaches. The performance of the proposed TAC is evaluated with performance metrics like packet delivery ratio (in %), packet loss (in %), end-to-end delay (in %), energy consumption (in Joules) and throughput (in mpbs) with varying percentage of malicious nodes in the network.

Performance Analysis with 10% Malicious Nodes in WSN

Table 1 depicts the packet loss (in %) obtained by the proposed TAC, K-means, spectral and density with 10% malicious nodes in the network.

The table compares packet loss (in %) between the proposed trust-aware clustering (TAC) approach and three existing clustering techniques: K-Means, Spectral, and Density when 10% malicious nodes are present in the network. The results are presented for varying numbers of nodes, ranging from 80 to 150.

The proposed TAC approach consistently demonstrates the lowest packet loss across all node configurations, ranging from 2.5 to 4.0%, as the number of nodes increases from 80 to 150. This highlights the superior efficiency of the TAC approach in mitigating the impact of malicious nodes, leading to more reliable packet transmission. K-means clustering exhibits higher packet loss, starting at 6.8% with 80 nodes and gradually increasing to 8.8% with 150 nodes. While it performs better than spectral and density-based clustering in some cases, it is significantly less effective than the TAC approach. Spectral clustering shows the highest packet loss, starting at 7.2% for 80 nodes and rising to 9.3% for 150 nodes. This indicates that the spectral clustering approach struggles to handle malicious nodes effectively, leading to higher packet loss. Density-based clustering performs better than Spectral but worse than K-Means, with packet loss values ranging from 5.9 to 7.8% as the number of nodes increases. While it is more efficient than spectral, it still lags behind the proposed TAC approach in mitigating packet loss.

Table 1: Packet Loss (in %) obtained by the proposed TAC, K-means, spectral and density with 10% malicious nodes

Number of nodes	*Packet Loss (in %)*			
	Proposed TAC	*K-means*	*Spectral*	*Density*
80	2.5	6.8	7.2	5.9
90	2.8	7.1	7.5	6.3
100	3.0	7.4	7.9	6.5
110	3.2	7.8	8.2	6.8
120	3.4	8.1	8.5	7.0
130	3.6	8.3	8.7	7.3
140	3.8	8.5	9.0	7.5
150	4.0	8.8	9.3	7.8

Table 2 depicts the packet delivery ratio (in %) obtained by the proposed TAC, K-means, spectral and density with 10% malicious nodes in the network.

The table compares the packet delivery ratio (PDR in %) between the proposed trust-aware clustering (TAC) Approach and three existing clustering methods: K-Means, Spectral, and Density-based clustering when 10% malicious nodes are present. The PDR is measured across networks with different numbers of nodes, ranging from 80 to 150.

Proposed TAC Approach consistently achieves the highest PDR, ranging from 97.5% with 80 nodes to 96.0% with 150 nodes. This demonstrates the TAC approach's superior ability to ensure successful packet delivery despite the presence of malicious nodes, highlighting its efficiency in managing network reliability. K-means clustering performs moderately, with PDR values starting at 93.2% for 80 nodes and decreasing to 91.2% for 150 nodes. While it maintains a relatively high PDR, it is clearly outperformed by the proposed TAC approach. Spectral clustering consistently exhibits the lowest PDR, ranging from 92.8% with 80 nodes to 90.7% with 150 nodes. The decreasing PDR shows that Spectral Clustering is less effective at maintaining high packet delivery in the presence of malicious nodes, leading to more packet drops. Density-based clustering achieves better PDR than Spectral Clustering, with values between 94.1% and 92.2% across the varying number of nodes. However, it is still outperformed by the TAC approach in all cases.

Table 3 depicts the End-to-End Delay (in %) obtained by the Proposed TAC, K-Means, Spectral and Density with 10% malicious nodes in the network.

Table 2: Packet Delivery Ratio (in %) obtained by the proposed TAC, K-means, spectral and density with 10% malicious Nodes

Number of Nodes	*Packet Delivery Ratio (in %)*			
	Proposed TAC	*K-Means*	*Spectral*	*Density*
80	**97.5**	**93.2**	**92.8**	**94.1**
90	**97.2**	**92.9**	**92.5**	**93.7**
100	**97.0**	**92.6**	**92.1**	**93.5**
110	**96.8**	**92.2**	**91.8**	**93.2**
120	**96.6**	**91.9**	**91.5**	**93.0**
130	**96.4**	**91.7**	**91.3**	**92.7**
140	**96.2**	**91.5**	**91.0**	**92.5**
150	**96.0**	**91.2**	**90.7**	**92.2**

Table 3: End-to-end delay (in %) obtained by the proposed TAC, K-means, spectral and density with 10% malicious Nodes

Number of nodes	*End-to-end delay (in %)*			
	Proposed TAC	*K-Means*	*Spectral*	*Density*
80	1.2	4.5	4.8	3.9
90	1.3	4.7	5.0	4.2
100	1.5	5.0	5.3	4.5
110	1.6	5.3	5.6	4.7
120	1.8	5.5	5.8	5.0
130	1.9	5.7	6.0	5.2
140	2.0	6.0	6.2	5.5
150	2.2	6.3	6.5	5.8

The table compares the End-to-End Delay (in %) between the Proposed Trust-Aware Clustering (TAC) Approach and three existing clustering techniques: K-means, spectral, and density-based clustering when 10% malicious nodes are present. The results are shown for different network sizes, ranging from 80 to 150 nodes.

Proposed TAC Approach achieves the lowest end-to-end delay across all configurations, with values ranging from 1.2% for 80 nodes to 2.2% for 150 nodes. The low delay highlights the TAC approach's efficiency in reducing latency during packet transmission, even in the presence of malicious nodes, ensuring faster data delivery across the network. K-Means Clustering has significantly higher delays compared to the TAC approach, starting at 4.5% for 80 nodes and increasing to 6.3% for 150 nodes. The gradual increase in delay suggests that K-means clustering is less efficient in maintaining fast communication as the network size grows and malicious nodes affect the network. Spectral clustering performs the worst in terms of delay, with values increasing from 4.8 to 6.5% as the number of nodes rises from 80 to 150. This indicates that spectral clustering introduces higher delays in packet transmission, making it less suitable for latency-sensitive applications. Density-based clustering achieves better results than spectral clustering but still lags behind K-means and the proposed TAC approach. The end-to-end delay for density-based clustering starts at 3.9% for 80 nodes and increases to 5.8% for 150 nodes, indicating moderate performance in minimizing latency.

Table 4 depicts the average energy consumption (in Joules) obtained by the proposed TAC, K-means, spectral and density with 10% malicious nodes in the network.

The table compares the average energy consumption (in Joules) between the proposed trust-aware clustering (TAC) approach and three existing clustering techniques: K-means, spectral, and density-based clustering, in the presence of 10% malicious nodes. The average energy consumption is measured for varying numbers of nodes, ranging from 80 to 150.

Proposed TAC approach consistently demonstrates the lowest energy consumption, starting at 0.35 Joules with 80 nodes and rising to 0.53 Joules with 150 nodes. The lower energy consumption reflects the TAC approach's efficiency in managing network operations, even in environments with malicious nodes, leading to extended network lifetime. K-Means Clustering has higher energy consumption compared to TAC, starting at 0.50 Joules for 80 nodes and increasing to 0.70 Joules for 150 nodes. The rising energy consumption indicates that K-Means clustering requires more resources to maintain the network, especially as the number of nodes increases. Spectral clustering shows the highest energy consumption, ranging from 0.54 to 0.75 Joules as the number of nodes grows. This suggests that Spectral Clustering is the least energy-efficient method, consuming more power to perform network tasks and handle malicious nodes. Density-based clustering performs better than Spectral but worse than K-Means, with energy consumption increasing from 0.46 to 0.67 Joules across the node range. Although its energy usage is lower than Spectral Clustering, it is still significantly higher than the proposed TAC approach.

Table 5 depicts the throughput (in Mbps) obtained by the proposed TAC, K-means, spectral and density with 10% malicious nodes in the network.

The table compares the throughput (in Mbps) between the proposed trust-aware clustering (TAC) approach and three existing clustering techniques, K-means, spectral, and density-based clustering, in the presence of 10% malicious nodes. Throughput is measured for networks with different node sizes, ranging from 80 to 150.

Proposed TAC Approach consistently demonstrates the highest throughput, starting at 7.5 Mbps for 80 nodes and decreasing to 6.1 Mbps for 150 nodes. This indicates that the TAC approach ensures more efficient data transmission, even in the presence of malicious nodes, resulting in better network performance and higher data flow. K-Means Clustering exhibits moderate throughput, starting at 5.8 Mbps with 80 nodes and decreasing to 4.4 Mbps with 150

Table 4: Average energy consumption (in Joules) obtained by the proposed TAC, K-means, spectral and density with 10% malicious Nodes

Number of nodes	*Average Energy Consumption (in Joules)*			
	Proposed TAC	*K-Means*	*Spectral*	*Density*
80	**0.35**	**0.50**	**0.54**	**0.46**
90	**0.37**	**0.53**	**0.57**	**0.49**
100	**0.40**	**0.56**	**0.60**	**0.52**
110	**0.43**	**0.59**	**0.63**	**0.55**
120	**0.46**	**0.62**	**0.66**	**0.58**
130	**0.48**	**0.64**	**0.69**	**0.61**
140	**0.50**	**0.67**	**0.72**	**0.64**
150	**0.53**	**0.70**	**0.75**	**0.67**

Table 5: Throughput (in Mbps) obtained by the proposed TAC, K-means, spectral and density with 10% malicious nodes

Number of nodes	*Throughput (in Mbps)*			
	Proposed TAC	*K-Means*	*Spectral*	*Density*
80	7.5	5.8	5.4	6.2
90	7.3	5.6	5.2	6.0
100	7.1	5.4	5.0	5.8
110	6.9	5.2	4.8	5.6
120	6.7	5.0	4.6	5.4
130	6.5	4.8	4.4	5.2
140	6.3	4.6	4.2	5.0
150	6.1	4.4	4.0	4.8

nodes. The steady decline in throughput as the number of nodes increases shows that K-Means clustering is less efficient than the TAC approach in maintaining a high data transmission rate. Spectral clustering consistently shows the lowest throughput, with values starting at 5.4 Mbps for 80 nodes and dropping to 4.0 Mbps for 150 nodes. This suggests that Spectral Clustering is less effective at handling malicious nodes and maintaining network efficiency, leading to a significant decrease in data transmission rates. Density-based clustering achieves better throughput than spectral clustering but remains lower than both K-Means and the TAC approach. Throughput starts at 6.2 Mbps for 80 nodes and decreases to 4.8 Mbps for 150 nodes, indicating that it performs moderately in maintaining data transmission rates as the network grows.

Performance Analysis with 20% Malicious Nodes in WSN

Table 6 gives the packet loss (in %) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes in the network.

The table compares the packet loss (in %) between the proposed trust-aware clustering (TAC) approach and three existing clustering techniques: K-means, spectral, and density-based clustering, in the presence of 20% malicious nodes. The results are shown for networks with varying sizes, ranging from 80 to 150 nodes.

Table 6: Packet loss (in %) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes

Number of nodes	*Packet Loss (in %)*			
	Proposed TAC	*K-Means*	*Spectral*	*Density*
80	4.2	12.5	13.1	11.3
90	4.5	13.0	13.5	11.7
100	4.7	13.5	14.0	12.1
110	4.9	13.9	14.5	12.5
120	5.1	14.3	15.0	12.9
130	5.3	14.6	15.4	13.2
140	5.5	15.0	15.8	13.6
150	5.7	15.4	16.2	14.0

Proposed TAC approach demonstrates the lowest packet loss, ranging from 4.2% for 80 nodes to 5.7% for 150 nodes. This indicates that the TAC approach effectively minimizes packet loss even when the percentage of malicious nodes increases, reflecting its robustness in maintaining network reliability. K-Means Clustering exhibits significantly higher packet loss, starting at 12.5% for 80 nodes and rising to 15.4% for 150 nodes. The increasing trend in packet loss indicates that K-Means is less capable of handling the challenges posed by malicious nodes, leading to a deterioration in network performance. Spectral clustering shows the highest packet loss among the techniques analyzed, ranging from 13.1% with 80 nodes to 16.2% with 150 nodes. The substantial packet loss underscores the inefficiency of Spectral Clustering in managing malicious activities within the network, resulting in a marked decrease in data transmission reliability. Density-based clustering performs better than Spectral Clustering but remains less effective than both the Proposed TAC and K-Means approaches. Packet loss ranges from 11.3% for 80 nodes to 14.0% for 150 nodes, suggesting that while it is somewhat resilient to malicious nodes, it still suffers from higher packet loss compared to the TAC approach.

Table 7 gives the packet delivery ratio (in %) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes in the network.

The table compares the packet delivery Ratio (in %) between the proposed trust-aware clustering (TAC) approach and three existing clustering techniques: K-means, spectral, and density-based clustering, in the presence of 20% malicious nodes. The results are presented for networks with varying node counts, ranging from 80 to 150.

Proposed TAC approach consistently achieves the highest packet delivery ratio, starting at 95.8% for 80

Table 7: Packet delivery ratio (in %) obtained by the proposed TAC, K-means, spectral and density with 20% malicious Nodes

Number of nodes	*Packet Delivery Ratio (in %)*			
	Proposed TAC	*K-means*	*Spectral*	*Density*
80	95.8	87.5	86.9	88.7
90	95.5	87.0	86.5	88.3
100	95.3	86.5	86.0	87.9
110	95.1	86.1	85.5	87.5
120	94.9	85.7	85.0	87.1
130	94.7	85.4	84.6	86.8
140	94.5	85.0	84.2	86.4
150	94.3	84.6	83.8	86.0

nodes and decreasing slightly to 94.3% for 150 nodes. This demonstrates the TAC approach's effectiveness in ensuring reliable data transmission, even with a higher proportion of malicious nodes in the network. K-means clustering shows a significantly lower packet delivery ratio, beginning at 87.5% for 80 nodes and declining to 84.6% for 150 nodes. The steady decrease indicates that K-means is less capable of maintaining high delivery rates when faced with malicious nodes, leading to reduced network efficiency. Spectral clustering presents an even lower packet delivery ratio, starting at 86.9% for 80 nodes and dropping to 83.8% for 150 nodes. The consistently lower performance underscores the challenges of using spectral clustering in environments where malicious activity is prevalent, as it struggles to maintain reliable communications. Density-based clustering performs slightly better than spectral clustering but still lags behind K-means and the proposed TAC approach. The packet delivery ratio ranges from 88.7% for 80 nodes to 86.0% for 150 nodes, indicating moderate reliability but still significantly lower than the TAC approach.

Table 8 depicts the End-to-End Delay (in %) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes in the network.

The table presents the end-to-end delay (in %) observed for the proposed trust-aware clustering (TAC) Approach compared to three existing clustering techniques: K-means, spectral, and density-based clustering in the context of 20% malicious nodes. The results are displayed for networks with varying sizes, from 80 to 150 nodes.

The proposed TAC approach exhibits the lowest end-to-end delay, beginning at 1.5% for 80 nodes and increasing to 2.6% for 150 nodes. This indicates that the TAC approach is efficient in facilitating timely data transmission, even in the presence of a higher percentage of malicious nodes. K-Means Clustering shows a significantly higher end-to-end delay, starting at 5.2% for 80 nodes and rising to 7.0% for 150 nodes. The continuous increase in delay suggests that K-means is less effective at ensuring prompt data delivery, likely due to its struggles with the presence of malicious nodes. Spectral clustering demonstrates the highest end-to-end delay among the methods evaluated, with delays starting at 5.6% for 80 nodes and climbing to 7.5% for 150 nodes. This indicates that spectral clustering is particularly inefficient in maintaining low latency under adverse conditions, such as with the presence of malicious nodes. Density-based clustering offers a moderate performance compared to K-means and Spectral Clustering, with end-to-end delays ranging from 4.7% for 80 nodes to 6.8% for 150 nodes. While it performs better than the latter two techniques, it still falls short of the TAC approach's efficiency.

Table 9 depicts the average energy consumption (in Joules) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes in the network.

The table compares the average energy consumption (in Joules) among the proposed trust-aware clustering (TAC) approach and three existing clustering techniques, K-means, spectral, and density-based clustering, in the context of 10% malicious nodes. The results are presented for networks with varying sizes, ranging from 80 to 150 nodes.

The proposed TAC approach demonstrates the lowest average energy consumption, starting at 0.42 Joules for 80 nodes and gradually increasing to 0.58 Joules for 150 nodes. This indicates that the TAC approach is highly efficient in energy utilization, which is critical in wireless sensor networks where energy resources are limited. K-Means Clustering exhibits higher energy consumption, beginning at 0.62 Joules for 80 nodes and increasing to 0.83 Joules for 150 nodes. This trend signifies that K-Means is less efficient in managing energy resources, resulting in higher energy usage as the number of nodes increases. Spectral clustering shows a similar pattern, with average energy consumption starting at 0.67 Joules for 80 nodes and rising to 0.88 Joules for 150 nodes. The energy consumption values indicate that Spectral Clustering is even less efficient than K-Means, leading to greater energy depletion in the network. Density-based clustering performs slightly better than Spectral Clustering but still consumes more energy than the Proposed TAC approach. The average energy

Table 8: End-to-end delay (in %) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes

Number of nodes	*End-to-end delay (in %)*			
	Proposed TAC	*K-means*	*Spectral*	*Density*
80	1.5	5.2	5.6	4.7
90	1.7	5.4	5.8	5.0
100	1.9	5.7	6.1	5.3
110	2.0	6.0	6.4	5.6
120	2.2	6.2	6.6	5.9
130	2.3	6.5	6.9	6.2
140	2.5	6.8	7.2	6.5
150	2.6	7.0	7.5	6.8

Table 9: End-to-end delay (in %) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes

Number of nodes	*End-to-end delay (in %)*			
	Proposed TAC	*K-means*	*Spectral*	*Density*
80	1.5	5.2	5.6	4.7
90	1.7	5.4	5.8	5.0
100	1.9	5.7	6.1	5.3
110	2.0	6.0	6.4	5.6
120	2.2	6.2	6.6	5.9
130	2.3	6.5	6.9	6.2
140	2.5	6.8	7.2	6.5
150	2.6	7.0	7.5	6.8

Table 10: Throughput (in Mbps) obtained by the proposed TAC, K-means, spectral and density with 20% malicious Nodes

Number of Nodes	*Throughput (in Mbps)*			
	Proposed TAC	*K-Means*	*Spectral*	*Density*
80	6.8	4.9	4.6	5.2
90	6.6	4.7	4.4	5.0
100	6.4	4.5	4.2	4.8
110	6.2	4.3	4.0	4.6
120	6.0	4.1	3.8	4.4
130	5.8	3.9	3.6	4.2
140	5.6	3.7	3.4	4.0
150	5.4	3.5	3.2	3.8

consumption ranges from 0.58 Joules for 80 nodes to 0.79 Joules for 150 nodes, highlighting its moderate efficiency in comparison to TAC.

Table 10 depicts the throughput (in Mbps) obtained by the proposed TAC, K-means, spectral and density with 20% malicious nodes in the network.

The table outlines the throughput (in Mbps) achieved by the proposed trust-aware clustering (TAC) approach compared to three existing clustering techniques: K-means, spectral, and density-based clustering under conditions where 20% of the nodes are malicious. The results are provided for networks ranging from 80 to 150 nodes.

Proposed TAC approach consistently achieves the highest throughput, beginning at 6.8 Mbps for 80 nodes and decreasing to 5.4 Mbps for 150 nodes. This suggests that the TAC approach effectively maintains data transmission rates even with a significant proportion of malicious nodes, indicating its robustness in handling network disturbances. K-means clustering demonstrates a significantly lower throughput, starting at 4.9 Mbps for 80 nodes and declining to 3.5 Mbps for 150 nodes. The substantial drop in throughput reflects the inefficiencies of K-means in mitigating the effects of malicious nodes, leading to reduced data delivery rates. Spectral Clustering records even lower throughput values, with results beginning at 4.6 Mbps for 80 nodes and falling to 3.2 Mbps for 150 nodes. This decline highlights the method's inability to sustain throughput levels in the face of challenges posed by malicious nodes. Density-based clustering shows moderate throughput performance, ranging from 5.2 Mbps for 80 nodes to 3.8 Mbps for 150 nodes. While better than K-means and spectral clustering, it still does not match the effectiveness of the proposed TAC approach.

Conclusion

The proposed trust-aware clustering (TAC) approach has demonstrated significant advantages over traditional clustering techniques such as K-means, spectral, and density-based clustering in addressing the challenges posed by malicious nodes. Throughout the evaluation of various performance metrics, including packet loss, packet delivery ratio, end-to-end delay, average energy consumption, and throughput, the TAC approach consistently outperformed existing methods. The results indicate that TAC not only minimizes packet loss and energy consumption but also maximizes throughput and packet delivery, even in scenarios where a considerable proportion of nodes (20 and 10%) are compromised. This highlights the effectiveness of the TAC approach in fostering a resilient and efficient WSN, capable of sustaining high levels of data integrity and network performance in the face of malicious interference.

The TAC approach enhances network security by integrating trust metrics into the clustering process, allowing for a more informed selection of reliable nodes for communication. This capability is crucial in WSNs, where resource constraints and energy efficiency are paramount. Furthermore, the proposed method's adaptability to varying node densities and malicious node ratios demonstrates its robustness and applicability in diverse real-world scenarios.

The trust-aware clustering approach presents a promising solution for the detection and management of malicious nodes in WSNs. Its ability to improve overall network performance while ensuring security makes it a valuable contribution to the field. Future work may focus on further refining the TAC algorithm, exploring its integration with other security measures, and testing its effectiveness in more complex network topologies and dynamic environments.

References

Anand, C., & Vasuki, N. (2021). Trust based DoS attack detection in wireless sensor networks for reliable data transmission. *Wireless Personal Communications*, 121(4), 2911-2926.

Fahmy, H. M. A., & Fahmy, H. M. A. (2020). Energy Management Projects for WSNs. *Wireless Sensor Networks: Energy Harvesting and Management for Research and Industry*, 611-637.

Gomathi, S., & Gopala Krishnan, C. (2020). Malicious node detection in wireless sensor networks using an efficient secure data aggregation protocol. *Wireless Personal Communications*,

113(4), 1775-1790.

Ibrahim, D. S., Mahdi, A. F., & Yas, Q. M. (2021). Challenges and issues for wireless sensor networks: A survey. *J. Glob. Sci. Res*, 6(1), 1079-1097.

Ibrahim, D. S., Mahdi, A. F., & Yas, Q. M. (2021). Challenges and issues for wireless sensor networks: A survey. *J. Glob. Sci. Res*, 6(1), 1079-1097.

Jane Nithya, K., & Shyamala, K. (2022). A systematic review on various attack detection methods for wireless sensor networks. *In International Conference on Innovative Computing and Communications: Proceedings of ICICC 2021*, Volume 3 (pp. 183-204). Springer Singapore.

Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020). Applications of wireless sensor networks: an up-to-date survey. *Applied system innovation*, 3(1), 14.

Kumar, N., Desai, J. R., & Annapurna, D. (2020, July). ACHs-LEACH: Efficient and Enhanced LEACH protocol for wireless sensor networks. In *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)* (pp. 1-6). IEEE.

Lai, Y., Tong, L., Liu, J., Wang, Y., Tang, T., Zhao, Z., & Qin, H. (2022). Identifying malicious nodes in wireless sensor networks based on correlation detection. *Computers & Security*, 113, 102540.

Morsi, A. M., Barakat, T. M., & Nashaat, A. A. (2020). An efficient and secure malicious node detection model for wireless sensor networks. *International Journal of Computer Networks & Communications (IJCNC)* Vol, 12.

Nagarjun, S., Anand, S., & Sinha, S. (2019). A research on the malicious node detection in wireless sensor network. *International Journal of Engineering and Advanced Technology*, 2249-8958.

Olakanmi, O. O., & Dada, A. (2020). Wireless sensor networks (WSNs): Security and privacy issues and solutions. *Wireless mesh networks-security, architectures and protocols*, 13, 1-16.

Ramasamy, L. K., KP, F. K., Imoize, A. L., Ogbebor, J. O., Kadry, S., & Rho, S. (2021). Blockchain-based wireless sensor networks for malicious node detection: A survey. *IEEE Access*, 9, 128765-128785.

Sampoornam, K. P., Saranya, S., Mohanapriya, G. K., Devi, P. S., & Dhaarani, S. (2021, February). Analysis of LEACH routing protocol in wireless sensor network with wormhole attack. *In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* (pp. 147-152). IEEE.

Temene, N., Sergiou, C., Georgiou, C., & Vassiliou, V. (2022). A survey on mobility in wireless sensor networks. *Ad Hoc Networks*, 125, 102726.

Yang, H., Zhang, X., & Cheng, F. (2021). A novel algorithm for improving malicious node detection effect in wireless sensor networks. *Mobile Networks and Applications*, 26, 1564-1573.

Zijie, F., Al-Shareeda, M. A., Saare, M. A., Manickam, S., & Karuppayah, S. (2023). Wireless sensor networks in the internet of things: review, techniques, challenges, and future directions. *Indonesian Journal of Electrical Engineering and Computer Science*, *31*(2), 1190-1200.