**CHAPTER – 1**

**INTRODCUTION**

The Internet of Things (IoT) encompasses a framework that enables the connection of numerous computing devices and sensors via the Internet, enhancing various applications such as smart homes, healthcare, agriculture, and industry. IoT has revolutionized multiple sectors and everyday life, allowing billions of devices to interconnect, and generating an unparalleled volume of data. However, the [interconnectivity](https://www.sciencedirect.com/topics/computer-science/interconnectivity) of the systems also presents notable security difficulties, making them susceptible to a variety of attacks. Detecting abnormal behaviour in IoT systems is crucial for maintaining their security, as it enables the identification of potential malicious activities or system malfunctions. [1]. The swift growth and increasing diversity of the IoT landscape necessitate the creation of strong and scalable solutions for anomaly detection.

**1.1 THE GROWING NEED FOR SECURITY IN IOT NETWORKS**

IoT networks are composed of devices ranging from sensors and actuators to smart home appliances and industrial machines. This diversity, combined with the openness of IoT architectures, makes these networks highly vulnerable to various attacks, such as:

1. Routing attacks (e.g., blackhole, wormhole, and rank attacks)
2. Data integrity breaches
3. Unauthorized access and privilege escalation
4. Distributed Denial of Service (DDoS) attacks

Traditional security mechanisms rely heavily on static policies and centralized architectures, which are inadequate for the dynamic and decentralized nature of IoT. The need for a self-adaptive, context-aware, and decentralized security framework has become critical, leading to the adoption of Zero Trust principles.

**1.2** **ARCHITECTURAL LAYERS OF IOT SYSTEM WITH THEIR ATTACKS**

IoT architecture is mainly divided into four layers as shown in the Fig.1 below:

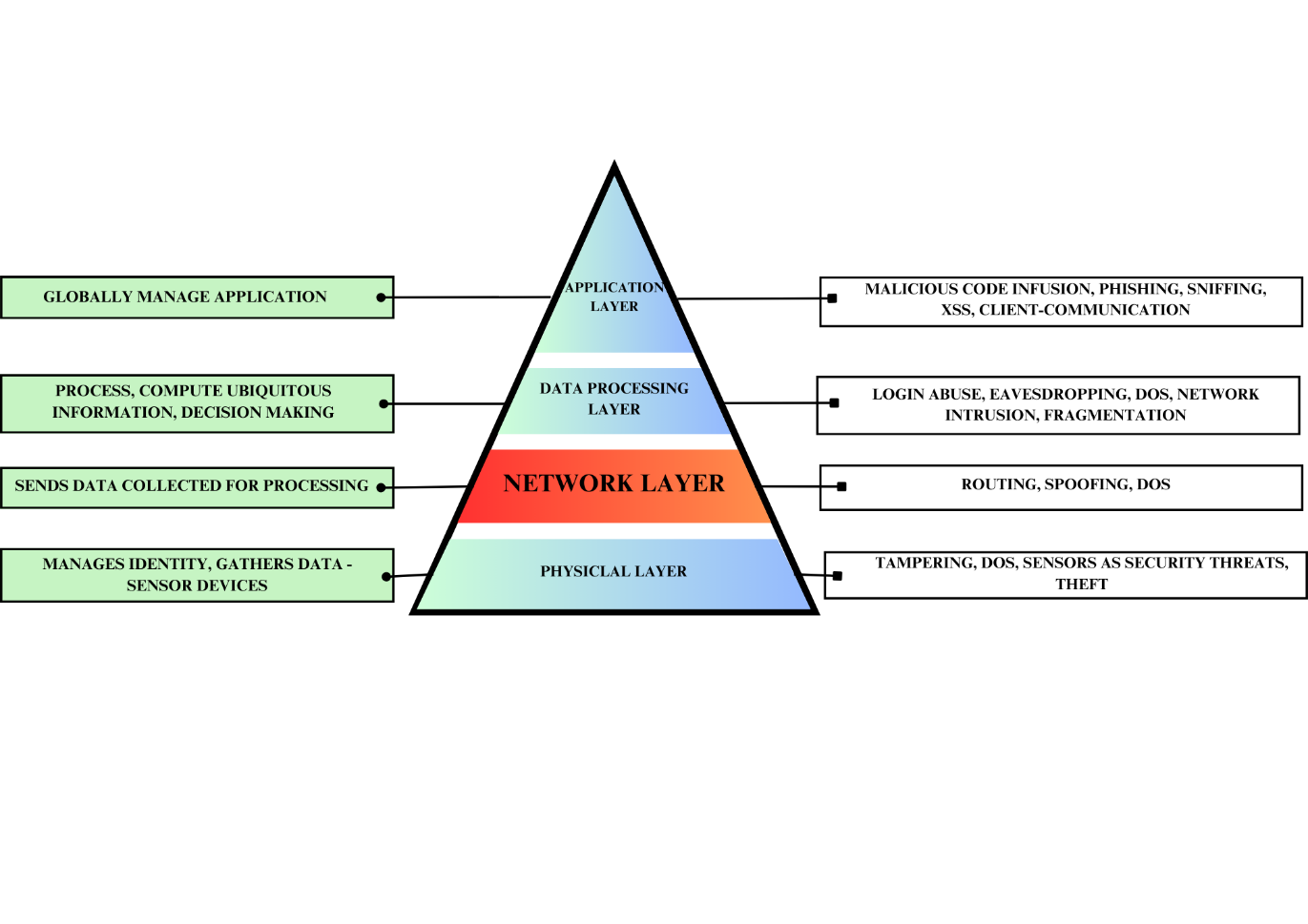


Fig.1 Architectural Layers of IoT with corresponding attack threats

A. The physical layer oversees the identification and collection of items, especially data from sensor devices. The various types of attacks within this layer include (i) tampering, (ii) Denial of Service (DoS), and (iii) sensors posing security threats.

B. The network layer is responsible for transmitting the data collected by devices to the information processing system. Various attacks within this layer include:

* Security countermeasures: These involve active firewalls for traffic management, passive monitoring to trigger alerts, traffic admission control, and bidirectional link identification.
* Man-in-the-middle attacks: These encompass eavesdropping, routing attacks, and replay attacks.
* Denial of Service (DoS) attacks: These include exhaustion, collision, unfairness, spoofed routing information, selective forwarding, sinkhole attacks, wormhole attacks, Sybil attacks, flooding, and node replacement.

C. The data processing layer is utilized to process and compute ubiquitous information, enabling automatic decision-making accordingly. Attacks in this layer include login abuse, eavesdropping, inappropriate system usage, Denial of Service (DoS) attacks, network intrusion, session hijacking, fragmentation attacks, cloud access control vulnerabilities, and database integrity problems.

D. The application layer is responsible for managing the application globally based on the data processed through the middleware. This layer encompasses threats and challenges related to the client application, communication channel issues, system integrity concerns of the client application, minor modifications that may lead to complex problems, multiuser access and concurrent editing of configurations, data access issues, and traceability.

**1.3 IMPORTANCE OF RPL FOR NETWORK LAYER IN IOT**

The IoT revolutionizes how individuals interact with everyday objects, computer systems, and one another [2]. To effectively manage the unique characteristics and requirements of IoT devices, specialized architectures and routing protocols must be established. Key challenges these systems and protocols need to tackle include reliability, energy efficiency, scalability, and security [3]. Two commonly employed IoT architectures are the distributed model, where data processing occurs at edge devices or in a decentralized manner, and the centralized model, where all data is directed to a central server for processing. Additionally, a range of routing protocols has been specifically designed for IoT networks [4]. Among these is the Routing Protocol for Low-power and Lossy Networks (RPL), crafted for IoT devices with limited memory and processing power. RPL is particularly well-suited for IoT networks, optimizing IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN). As a fundamental protocol for IoT networks, RPL enables effective and reliable communication among devices with low power and computing capacity. It is specifically designed to support routing in constrained environments [5], such as sensor networks, where devices may have minimal memory, processing ability, or battery life. RPL accomplishes this through a straightforward routing protocol that reduces control overhead and adapts to the dynamic characteristics of these networks. The security of connected devices is crucial in the rapidly changing IoT landscape, making the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) an essential routing protocol tailored to address the distinct needs of IoT networks [6].

The study referenced in [7] highlights that security mechanisms incorporating trust, thresholds, secure routing, authentication, and encryption have yielded promising outcomes in detecting anomalies related to RPL attacks. Notably, the trust-based mechanism is recognized as the most frequently employed security measure for safeguarding RPL.

**1.4 TRUST INTEGRATION FOR SECURE ROUTING**

**1.4.1 Zero Trust Security: A Paradigm Shift**

Zero Trust Security is a modern security model that challenges the traditional notion of implicit trust within a network. It operates on the following core principles:

* **Verify Explicitly:** Continuously validate the identities and trustworthiness of devices, users, and data.
* **Least Privilege Access:** Grant access strictly based on necessity and role.
* **Assume Breach:** Operate under the assumption that the network is compromised, ensuring proactive measures to mitigate risks.

In IoT networks, Zero Trust principles address fundamental security challenges by enforcing granular control over device communication and routing. However, implementing Zero Trust in IoT environments requires automation and adaptability, which can be achieved through Reinforcement Learning.

**1.4.2** **Reinforcement Learning in IoT Security**

Reinforcement Learning (RL) is a subset of machine learning that enables agents to learn optimal actions through interaction with their environment. In the context of IoT security, RL offers several advantages:

* **Dynamic Decision-Making:** RL models can adapt to changing network conditions and evolving attack strategies.
* **Continuous Learning:** Unlike static policies, RL algorithms improve over time by learning from feedback.
* **Decentralized Applications:** RL agents can operate on resource-constrained devices, enabling distributed security management.

RL algorithms, such as Deep Q-Networks (DQN) and Multi-Agent Reinforcement Learning (MARL), are particularly effective in automating Zero Trust policies for IoT routing, where decisions must be made in real time.

**1.4.3** **Auto-Zero Trust for Secure Routing**

Auto-Zero Trust refers to the automated implementation of Zero Trust principles using advanced technologies like RL. In the context of secure routing in IoT networks, this involves:

* **Dynamic Trust Evaluation:** Continuously assessing the trustworthiness of devices based on behavior, communication patterns, and historical data.
* **Policy Automation:** Using RL to automatically generate and update routing policies that minimize risks and optimize performance.
* **Real-Time Adaptation:** Enabling IoT networks to respond to threats as they occur, without manual intervention.

The integration of RL into Zero Trust frameworks allows IoT networks to proactively mitigate threats and ensure secure data transmission, even under resource constraints.

**1.5 PROBLEM STATEMENT**

In RPL (Routing Protocol for Low-Power and Lossy Networks) attacks, ensuring secure routing is critical due to the presence of malicious nodes within the network. However, current models lack effective trust-based integration for RPL routing in IoT networks, leaving a significant gap in addressing the dynamic nature of IoT environments. There is a growing need for the integration of adaptive and zero-trust models that can assess the trustworthiness of nodes and make informed routing decisions. This would enhance the security and efficiency of routing protocols, ensuring robust communication in the presence of malicious behaviour.

**1.6 OBJECTIVES**

1. **To Develop and Implement RL-Optimized Zero Trust Model:**Design and implement a novel RL-Optimized Zero Trust model to securely manage routing in IoT networks, ensuring that the trustworthiness of nodes is accurately evaluated in the presence of malicious behaviour.
2. **To evaluate the mode using Machine Learning Algorithms for trustworthiness:**  
   Utilize various machine learning algorithms (e.g., Random Forest, SVM, Neural Networks) to evaluate and predict the trustworthiness of nodes. This will involve training and testing on the ROUT-4-2023 dataset to assess model accuracy and efficiency.
3. **To** **Visualize and Interpret Model Performance:**  
   Generate intermediary and final visual outputs to interpret the results of the trust-based routing algorithm and its performance across different evaluation metrics, such as accuracy, precision, and recall.

**CHAPTER – 2**

**LITERATURE REVIEW**

In [10], the authors conduct a thorough analysis of demographic characteristics, existing and emerging attacks, simulation programs, network configurations, and evaluation metrics in RPL-based 6LoWPAN. The RPL protocol is recognized as the most effective routing protocol for Low-Power and Lossy Networks (LLN). Security mechanisms that incorporate trust, thresholds, secure routing, authentication, and encryption have demonstrated promising results in detecting anomalies and protecting against RPL attacks. Among these, trust-based mechanisms are the most prevalent for securing the RPL protocol.

The subsequent sections will introduce the survey categorized into 3 sections:

**2.1 STATE-OF-THE-ART METHODS**

**2.1.1 IoT Security Challenges**

Singh et al. (2020) in [8], addressed the security issues prevalent in IoT architecture layers, with a particular focus on routing attacks within the network layer. The paper highlights the vulnerabilities that IoT networks face due to malicious attacks targeting the routing protocols, which can compromise the integrity and availability of data transmission. The authors proposed several countermeasures, including solutions for enhancing the security of Geo-routing protocols, implementing robust authorization mechanisms, continuous monitoring of network activities, and introducing redundancy to ensure system resilience. These measures aim to strengthen the IoT network's defenses, safeguarding it against potential routing-related security threats, especially in smart city applications.

In their 2020 study, Sharma et al. [9] explored the security issues and attacks in the network layer of the Internet of Things (IoT) framework, with a focus on routing protocols. The authors conducted an in-depth analysis of the IoT architecture and the various routing protocols utilized within the network layer, identifying critical vulnerabilities and potential security threats. The paper highlights the challenges IoT networks face in ensuring secure communication due to malicious attacks targeting the network layer, and the need for robust routing protocols that can effectively mitigate these risks. The study provides a comprehensive overview of the IoT architecture and emphasizes the significance of securing routing mechanisms to maintain the integrity and reliability of IoT systems.

Hinai et al. (2017) in [10], addressed various security challenges in the Internet of Things (IoT), focusing on security issues across different IoT layers, including the cross-layer integration of heterogeneous systems. The review highlighted the security problems specific to each IoT layer and proposed potential solutions to mitigate these risks. Particularly, the study emphasized the need for solutions in the network layer, where security threats are prevalent. It suggests that addressing these security issues at the network layer could significantly enhance the overall security of IoT systems, providing a foundation for future research in the area.

Tewari et al. (2021) [11], conducted a systematic review of security issues and challenges in futuristic wearable IoT devices, an emerging segment within the IoT landscape. The review pointed out several key challenges, including the lack of efficient security protocols and the need for continuous network monitoring to detect vulnerabilities in real-time. It suggests that the development of more robust security protocols, along with enhanced monitoring techniques, would be crucial in safeguarding wearable IoT devices from evolving threats. The findings call for further research into efficient security mechanisms tailored specifically for wearable IoT environments.

Bansal et al. (2022) in [12], explored the security and privacy challenges that arise within the architectural layers of IoT systems. The study reviewed security methods employed across these layers and proposed solutions to address privacy and security concerns effectively. It emphasized that high-quality security methods should be capable of managing multiple platforms while ensuring a consistent level of protection. Bansal's work underlines the importance of developing security frameworks that can handle the complexity of IoT architectures, facilitating secure interoperability among diverse IoT platforms and ensuring the privacy of users and data.

Siddiqui et al. (2022) in [13], reviewed the security challenges in Software-Defined Networking (SDN) based IoT frameworks, particularly focusing on the vulnerabilities in network routing. The study revealed that network routing attacks could disrupt IoT services and compromise the overall reliability of IoT systems. Moreover, the review highlighted the importance of trust management in IoT applications, especially in scenarios where devices must rely on each other for secure communication. Siddiqui's findings point to the need for robust trust management frameworks to secure IoT networks, making it an essential area of future research in the context of IoT and SDN integration.

**2.1.2** **Various Trust Techniques employed for RPL**

A trust-aware security mechanism to detect sinkhole attacks in RPL-based IoT environments using Random Forest – RFTRUST was proposed in [14]. The mechanism, implemented using Cooja, the Contiki network simulator, employs either direct or indirect trust for trust computation. Specifically, it uses Random Forest (RF) for direct trustworthiness and Subjective Logic (SL) to identify a node's neighbour through various sources of indirect trust. The evaluation metrics include Quality of Service (QoS) and social trust metrics such as average delay, energy consumption, and honesty. Performance indicators for the method include Packet Delivery Ratio (PDR), average delay, average throughput, energy consumption, False Positive Rate (FPR), False Negative Rate (FNR), and detection accuracy. RFTRUST reduces additional computation overhead and achieves high accuracy (85%) with a low false-positive rate (1.4%) and low false-negative rate (1.8%). This approach is also applicable for detecting other security attacks in IoT networks.

A secure trust-aware RPL routing protocol for the Internet of Things in [15], investigates a routing protocol designed to address rank and Sybil attacks through a simulation study comparing the proposed SecTrust-RPL protocol with the standard RPL. Key features of SecTrust-RPL include the calculation of trustworthiness using both direct and indirect (recommendations) trust values by evaluating neighbouring nodes, fuzzy threshold-based trust broadcast, and a threshold-based trust rating system. The study highlights that MRHOF-RPL experienced packet loss rates as high as 73% on mote 11, whereas SecTrust-RPL had a maximum packet loss of less than 28% on the same mote. Despite proposing a design with these enhancements, the study did not perform validation of the proposed design. Future work should address colluding attacks and consider integrating trusted nodes that have recouped battery power.

The study in [16], addresses insider threats in DODAG by developing a prototype framework in Python and simulating a custom environment. The proposed protocol employs direct and indirect trust mechanisms, using an Inverse Gompertz (IG) function to model misbehaving instances. Trust scores of nodes are utilized to reward or penalize based on an eGreedy MARL approach. Multi-agent reinforcement learning (MARL) is used for decision making to manage misbehaving instances, which include frequent packet drops, significant delays in message forwarding, refusal of connection requests, and spurious packet generation. The protocol incorporates a variation of node trust and failure rates, allowing MARL to enable the root to learn optimal actions over time. It has been observed that few incorrect action decisions occur when state–action pairs are selected, with a rate of around 17.3%. Additionally, the study includes multiple connected DODAGs with multiple roots.

The study in [17], addresses Black hole and Gray hole attacks and demonstrates successful implementation and testing within an IoT environment. It focuses on node and link trust, considering QoS factors such as delay awareness, energy efficiency, and reliability. Metrics such as end-to-end delay, packet reception ratio, packet error rate, and expected transmission count are evaluated to enhance the performance of the model against various attack scenarios.

The study in [18], Attribute Based Trust Evaluation for Secure RPL Protocol in IoT Environment explores lightweight LBS (Location Based Service) authentication and Attribute Based Trust Evaluation (ABTE) within IoT environments.

The study in [19], presents a simulated approach to address blackhole attacks by incorporating a trust-based mechanism. It introduces a method for computing trust values for each node, which are then used for routing decisions, with the aim of improving the expected transmission count and the rank of nodes.

The study in [20], in the article "Fuzzy, Dynamic and Trust-Based Routing Protocol for IoT" introduces the multi-fuzzy, dynamic, and hierarchical trust model (FDTM-IoT) to address routing challenges in IoT networks. This model incorporates contextual information (CI), quality of service (QoS), and quality of peer-to-peer (P2P) communication (QPC) to improve routing decisions. The focus of the study is on mitigating issues such as end-to-end delay and packet loss rates in the network.

The study in [21], addresses both Worm hole and Gray hole attacks by incorporating direct trust based on node properties and indirect trust derived from the opinions of neighbouring nodes. These approaches are designed to be energy-friendly and do not impose excessive overhead on network traffic.

The study in [22], discusses a trust-based defense system for detecting DDoS attacks in RPL (Routing Protocol for Low-Power and Lossy Networks) within IoT environments. It introduces a method for detecting DDoS attacks based on packet frequency, utilizing a data frequency threshold that measures incoming packet rates and data intervals. The trust system relies on feedback from neighbouring nodes, with the data transmission process counting incoming packets to update the trust levels and create a blocklist for malicious nodes. The system’s effectiveness is evaluated based on several criteria, including detection accuracy, system packet arrival, throughput, routing overhead, and power consumption.

The study in [23], focuses on addressing rank and blackhole attacks using mobility-based metrics. Key metrics include Success Rate, Energy Level, Historical Observations, Location and Link Stability, mobility, and recommended trust.

The study "Trust-based Enhanced Secure Routing against Rank and Sybil Attacks in IoT" [24], addresses the challenge of securing routing protocols against rank and Sybil attacks in Internet of Things (IoT) networks, emphasizing the need for robust security mechanisms. The study highlights that topology stability improves by 46%, the packet loss rate decreases by 45%, and throughput increases by 35%, with only a 2.3% rise in average power consumption. The study proposes the PROTECT (Providing Routing Security using the Technique of Collective Trust) mechanism, which involves direct trust calculation focusing on rank properties, final direct trust updation, indirect trust calculation, and overall trust estimation. Key considerations in the PROTECT mechanism include DT-node behaviour, residual energy depletion, unselfishness, rank value, and IT-recommendations from neighbour nodes. The research highlights the impact of these factors on attack detection accuracy, throughput, and energy efficiency, providing a comprehensive approach to enhancing security in IoT routing protocols.

The study in [25], addresses Sybil attacks through the Subjective Logic Trust Model for trust computation. The THC-RPL solution includes DT (node energy consumption and forwarding behaviour) and IT (recommendation for forward packet behaviour and energy depletion). It evaluates attack detection, packet loss ratio, average energy consumed at node and network levels, computation cost, communication cost, and storage cost.

The study in [26], explores simulations using Contiki-NG and the NS-3 simulator. The study focuses on rank and blackhole attacks, employing DT-based methods and leveraging information received from SRF-IDS. Key metrics include the Median Packet Delivery Ratio (PDR), Median Parent Switch, Median Packets Dropped, and Median IDS Packet Overhead, achieving a 92.8% packet delivery ratio, a five-fold reduction in packet drops, and a three-fold decrease in packet overhead, with less than 2% overhead. The system incorporates machine learning for detection.

**2.1.3 Trust Evaluation Models**

Kamran Ahmad Awan et al. (2022) proposed the AutoTrust model, a privacy-enhanced trust-based intrusion detection approach for the Internet of Smart Things. The model focuses on predicting malicious nodes and eliminating them from the network to ensure secure communication. Using a Recurrent Neural Network (RNN) for trust evaluation, AutoTrust was designed to continuously monitor nodes for malicious behavior. However, the model was found to have high energy consumption during initial phases, which could be a limitation in resource-constrained IoT environments. Despite this, AutoTrust provides a promising approach for enhancing trustworthiness in IoT networks by detecting and eliminating threats effectively.

In a subsequent study, Kamran Ahmad Awan et al. (2024) explored the use of Ensemble XGBoost and AdaBoost techniques to enhance IoT security through trust management. The paper addresses the identification of malicious and compromised nodes by employing both centralized and distributed implementations of trust evaluation. By integrating Ensemble XGBoost and AdaBoost, the model demonstrated good accuracy in identifying compromised nodes. The findings suggest that combining multiple ensemble learning techniques can significantly improve the trust evaluation process, offering a robust method for detecting malicious behavior in IoT networks.

W. Ma et al. (2021) proposed a machine learning-empowered trust evaluation method specifically designed for IoT devices. This study utilized Long Short-Term Memory (LSTM), stacked LSTM, and Bi-LSTM models to assess the trustworthiness of IoT devices. The authors conducted a comparative analysis of various machine learning models to identify the most effective trust evaluation technique. The results revealed that the Bi-LSTM model outperformed other methods, providing more accurate trust assessments for IoT devices. This work highlights the importance of using advanced deep learning techniques, such as Bi-LSTM, to enhance trust evaluation in IoT systems.

M. Aaqib et al. (2023) focused on establishing trustworthiness in IoT devices through discriminative features-based prediction using machine learning models. The study developed and identified a Trust Management System (TMS) for IoT networks by applying various machine learning classifiers, including Decision Trees (DT), Random Forest (RF), Multi-Layer Perceptron (MLP), Support Vector Machines (SVM), Gaussian Naive Bayes (GNB), and CatBoost. The findings showed that Random Forest (RF) outperformed other models in terms of accuracy, making it a strong candidate for trustworthiness prediction in IoT systems. This research emphasizes the utility of machine learning classifiers for developing effective trust management systems in IoT environments.

The literature review highlights several key advancements in the field of trust evaluation models for IoT security, with a focus on predicting and eliminating malicious nodes, enhancing trust management, and evaluating the trustworthiness of IoT devices using machine learning techniques. While numerous approaches have been proposed, including AutoTrust, ensemble methods like XGBoost and AdaBoost, and deep learning models such as LSTM and Bi-LSTM, many of these models face limitations such as high energy consumption, insufficient security protocols, and lack of integration across multiple platforms. Furthermore, while trust evaluation has made significant strides, there is still a need for more adaptive, robust models that can effectively address the dynamic nature of IoT networks and handle both privacy and security concerns in a seamless manner. The proposed IntelliTrust model aims to bridge these gaps by integrating AutoTrust and Zero Trust principles with reinforcement learning. This novel approach not only enhances trust prediction accuracy but also provides a dynamic, adaptive solution capable of continuously evaluating and adapting to new security challenges in IoT environments. By incorporating various classifiers for evaluation, IntelliTrust offers a comprehensive and scalable solution to the existing challenges in IoT trust management, providing a robust framework for secure and efficient IoT networks.

**CHAPTER – III**

**PROPOSED METHODOLOGY**

The proposed methodology integrates Zero Trust principles with Reinforcement Learning (RL) to create an adaptive and secure framework for intrusion detection in IoT networks. The framework, termed as **Reinforced Auto-Zero Trust (RAZT)**, dynamically adjusts trust scores and decision-making strategies to enhance security and optimize resource efficiency. The following steps elaborate on the methodology.

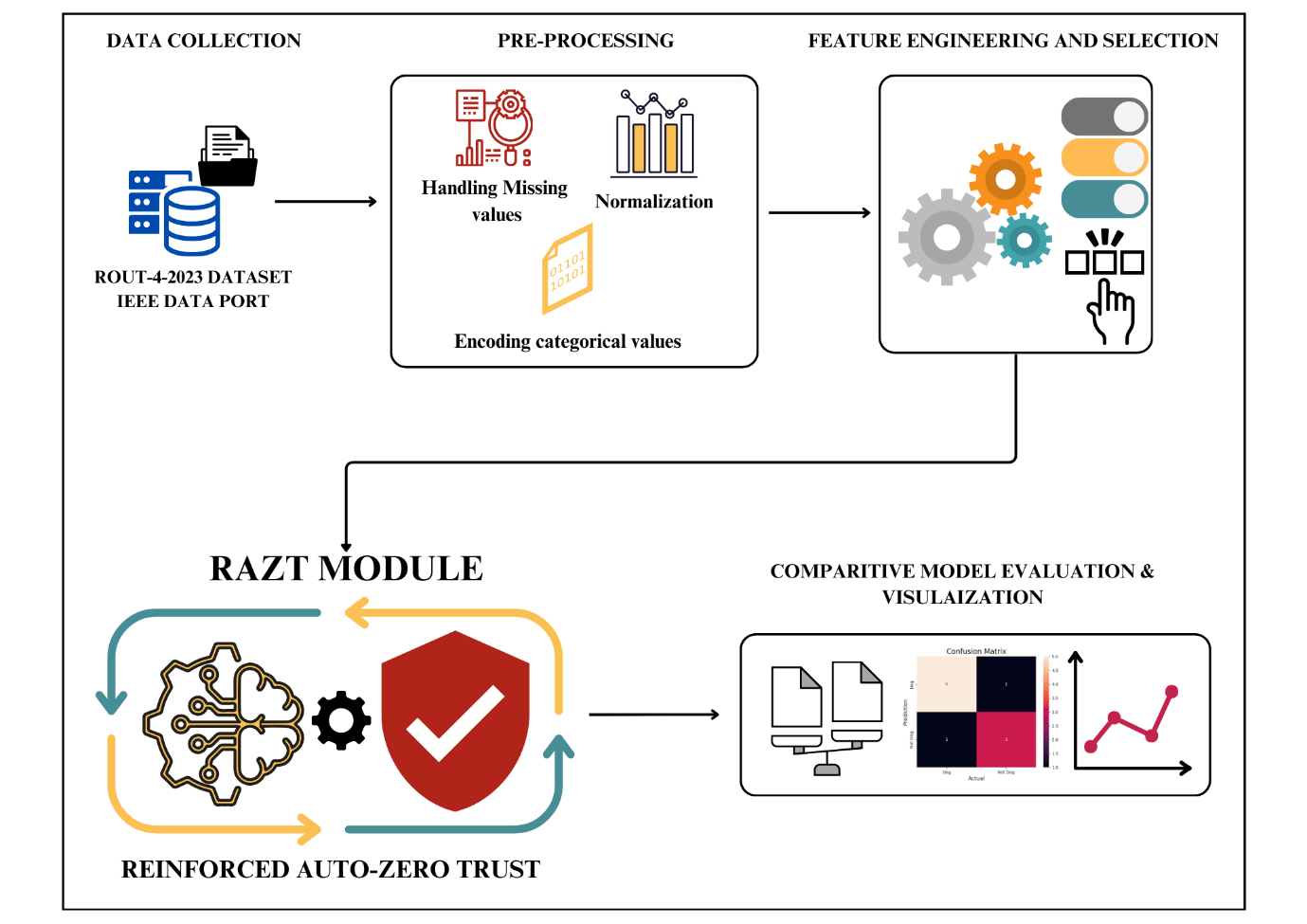
****

Fig.2. Proposed Framework RAZT for Secure Routing and Model Evaluation

The methodology as shown in Fig.2, followed a structured data science framework, starting with the collection and preprocessing of the ROUT-4-2023 dataset.

* 1. **DATA COLLECTION**

The ROUT-4-2023 dataset from IEEE Data Port which was developed by Murat and Mehmet [7] was utilized, encompassing four distinct attack scenarios relevant to IoT networks: blackhole, flooding, DODAG inconsistency, and rank attacks. Each scenario was represented in separate CSV files, containing features which are also shown in Table.1.

* Temporal Features: time
* Network Features: source, destination, length, info
* Transmission/Reception Metrics: transmission\_rate\_per\_1000\_ms (TR), reception\_rate\_per\_1000\_ms (RR), transmission\_average\_per\_sec (TAT), reception\_average\_per\_sec (RAT), transmission\_count\_per\_sec (TPC), reception\_count\_per\_sec (RPC), transmission\_total\_duration\_per\_sec (TTT), reception\_total\_duration\_per\_sec (TRT)
* Routing Protocol Messages: dao, dis, dio
* Labels: category (attack type or normal), label (normal/malicious)

Table.1 Feature Description

|  |  |
| --- | --- |
| **Features** | **Description** |
| TIME | Simulation time |
| SOURCE | Source Node IP |
| DESTINATION | Destination Node IP |
| LENGTH | Packet Length |
| INFO | Packet Information |
| TR | Transmission Rate(per 1000\_ms) |
| RR | Reception Rate(per 1000 ms) |
| TAT | Transmission Average Time |
| RAT | Reception Average Time |
| TPC | Transmitted Packet Count(per second) |
| RPC | Received Packet Count(per second) |
| TTT | Total Transmission Time |
| TRT | Total Reception Time |
| DAO | DAO Packet Count |
| DIS | DIS Packet Count |
| DIO | DIO Packet Count |
| CATEGORY | Attack Type or Normal |
| LABEL | Normal/Malicious Label |

* 1. **DATA PREPROCESSING**

**3.2.1 Dataset Loading and Cleaning**

* The ROUT-4-2023 dataset is used, containing features such as TIME, SOURCE, DESTINATION, LENGTH, INFO, and others.
* The dataset was checked for missing values and found none.
* Data normalization is performed using the Min-Max Scaler to ensure uniform scaling of all feature values to the range [0, 1]:

Where, X is the original value, ​ is the minimum value in the feature, is the maximum value in the feature, is the normalized value that will range between 0 and 1.

* One-hot encoding was applied to “Category” column to convert into numerical format allowing the model to process node-specific information.

**3.2.2 Feature Selection using Mutual Information**

* Mutual Information (MI) is calculated between features and the target variable (LABEL).
* Top-k features are selected based on MI scores:

*I(X;Y) =*

Where, *P(x,y)* is the joint probability distribution of X and Y, P(x) is the marginal probability distribution of X, P(y) is the marginal probability distribution of Y, the logarithm is typically taken with base 2 to express information in bits.

**3.2.3 Feature Engineering – Trust Metrics**

Trust-based features are engineered, such as:

Behavioral Metrics: Calculates anomaly scores based on historical behavior and Packet Timing: Monitors delays and packet timing irregularities. These features ensure compliance with the Zero Trust principle. {formulae for these features}

**3.2.4 Splitting the Dataset**

The dataset is split into training and testing sets using an 80:20 ratio.

* Training Set: 80% of the rows, used for model training.
* Testing Set: 20% of the rows, used for model evaluation.

Dataset Split:

* Training Set: 1311980 rows, 19 features
* Testing Set: 327995 rows, 19 features

**3.3 RAZT: REINFORCED AUTO-ZERO TRUST MECHANISM**

This proposed novel module integrates Zero Trust principles with Reinforcement Learning (RL) to dynamically adjust trust scores of nodes in the IoT network, ensuring real-time adaptability and enhanced security.

* + 1. **Define the Environment for Reinforcement Learning**

In reinforcement learning, the system must operate in an environment defined by states, actions, and rewards.

1. States (S):

Each state represents the current condition of a node or interaction in the IoT network. The state vector may include:

* Adjusted trust score.
* Packet delivery metrics (e.g., PDR, RTT).
* Behavioral metrics (e.g., anomaly score, communication frequency).
* Node-related attributes (e.g., time since last communication).

1. Actions (A):

Actions represent decisions made by the RL agent for each node or interaction. The possible actions are:

* Increase Trust Score: If behavior aligns with normal patterns.
* Decrease Trust Score: If malicious activity is suspected.
* Isolate Node: If the trust score falls below a critical threshold.
* Allow Communication: If the node is trustworthy.

1. Rewards (R):

Rewards are feedback given to the RL agent for the chosen actions. The rewards system is designed as:

* + 1. **Q-Learning Algorithm for Trust Adjustment**

The Q-learning algorithm is used to optimize the agent's decision-making. The Q-value function updates based on actions and their outcomes:

Where:

* Q(S,A) Q-value for the state-action pair.
* α: Learning rate (how quickly the model adapts).
* γ: Discount factor (future reward weight).
* S′: Next state after the action A.
* A′: Optimal future action.
  + 1. **Implement -Greedy Policy**

The -greedy policy balances exploration (trying new actions) and exploitation (choosing known optimal actions):

Where, ϵ: Exploration rate (e.g., starts high and decays over time).

* + 1. **Adjust Trust Scores Dynamically**

The RL agent adjusts trust scores based on the actions:

Where:

* : Updated trust score.
* ΔT: Change in trust score, determined by the action:
  + Positive (+) for trust increase.
  + Negative (−) for trust decrease.

Key Mechanism:

1. If a node consistently behaves maliciously, ΔT decreases progressively.
2. If a node recovers and behaves correctly, ΔT increases cautiously.
   * 1. **Train and evaluate the RL Model**
   1. Initialize Q(S,A) values randomly.
   2. Train the model iteratively:

* For each node or interaction, calculate the current state SSS.
* Select an action A based on the ϵ-greedy policy.
* Apply the action and observe the reward R and the next state S′.
* Update Q(S,A) using the Q-learning formula.
  1. Continue until convergence (Q-values stabilize).

**3.4 Integrating RL-Optimized Trust Scores into Classification**

After trust score adjustment the further evaluation of the RAZT model will be carried out

1. Using the adjusted trust scores ​ as features for classification models like Random Forest, TCN, or other ML/DL algorithms.
2. Evaluating model performance using metrics like accuracy, precision, recall, F1-score, ROC-AUC, and various trust metrics.

**3.5 CONCLUSION**

The proposed methodology successfully integrates the Auto-Zero Trust framework with Reinforcement Learning to enhance security and resource efficiency in IoT networks. Starting with comprehensive preprocessing, the ROUT-4-2023 dataset was cleaned, normalized, and engineered with novel adaptive trust features to ensure robust data inputs. The Auto-Zero Trust model was then applied, incorporating dynamic trust score computation based on Zero Trust principles and mutual information-driven feature selection. This was followed by the integration of a Reinforcement Learning module, which optimized trust scores using Q-learning and an adaptive exploration-exploitation strategy. The trust-adjusted outputs were utilized to improve the classification accuracy of malicious and normal nodes, tested across various machine learning and deep learning models. Intermediate results at each stage confirmed the effectiveness of the approach, demonstrating significant security and performance improvements compared to traditional trust-based models. Moving further, this model will be evaluated using various machine learning classifiers to compare its performance and establish its adaptability and robustness across different algorithms.

**CHAPTER – IV**

**RESULT ANALYSIS**

A sequence of well-defined steps was implemented to ensure the seamless integration of machine learning and deep learning models. Python, as the primary programming language, was chosen due to its rich ecosystem of open-source tools. The code was executed using Google Colab, while a DELL Inspiron 7490 laptop with 16GB of RAM and Intel® UHD Graphics was utilized for managing and monitoring the experiments.

The dataset used had 4 csv files of which Black Hole Attack dataset had 404134 rows, DODAG Version Number Attack dataset had 468060 rows, Flooding Attack dataset has 398782, Decreased Rank Attack dataset has 368999 rows.

The preview of the dataset with the first 5 rows printed is given below in Fig.3

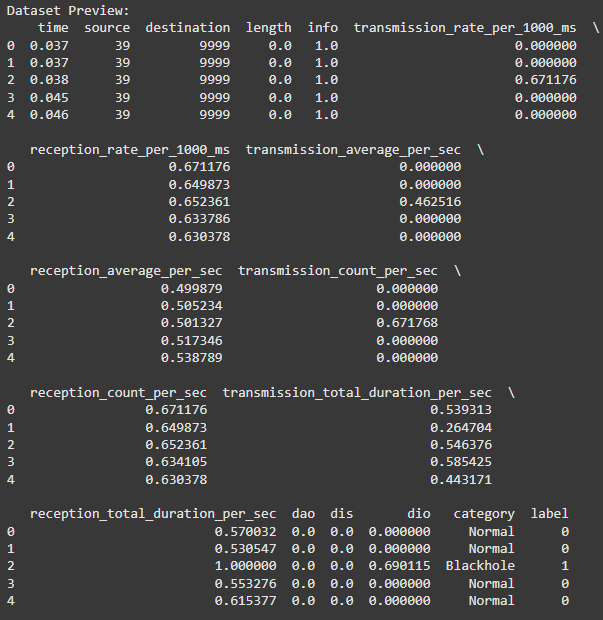


Fig.3 Dataset Preview

The Dataset Information regarding the no. of rows and columns along with the data types of each column is given below in Fig.4

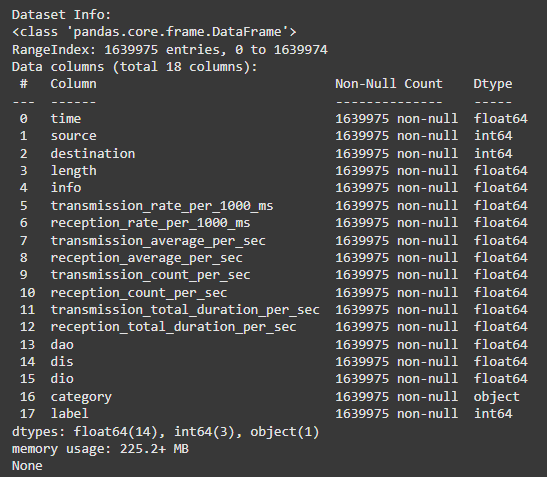


Fig. 4 Dataset Info

The dataset was checked for missing values and had none and is shown below in the Fig.5

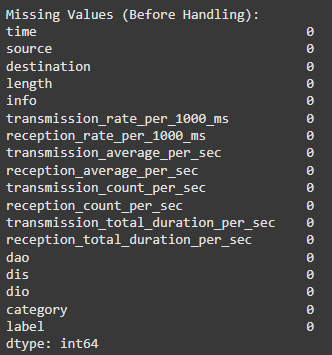


Fig.5 Handling Missing Values

A preview of the normalized features using Min-Max scaling is given below in Fig.6

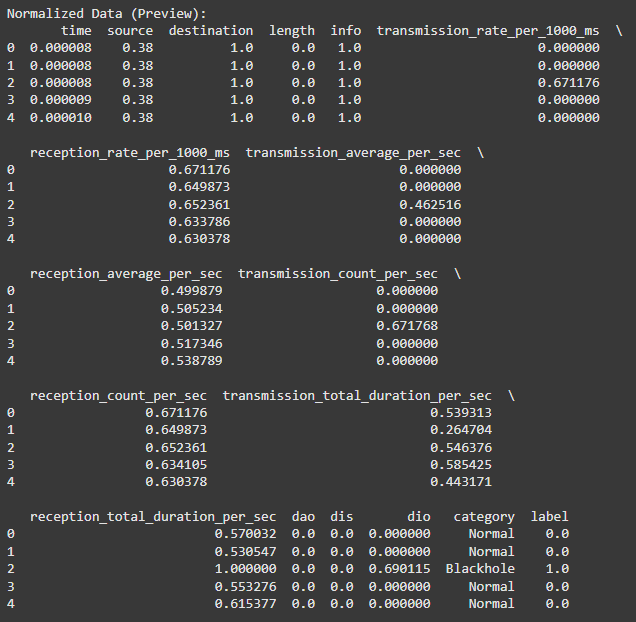


Fig.6 Preview of dataset after Normalization

The distribution of the normalized features within the scale [0-1] is shown below in the Fig.7

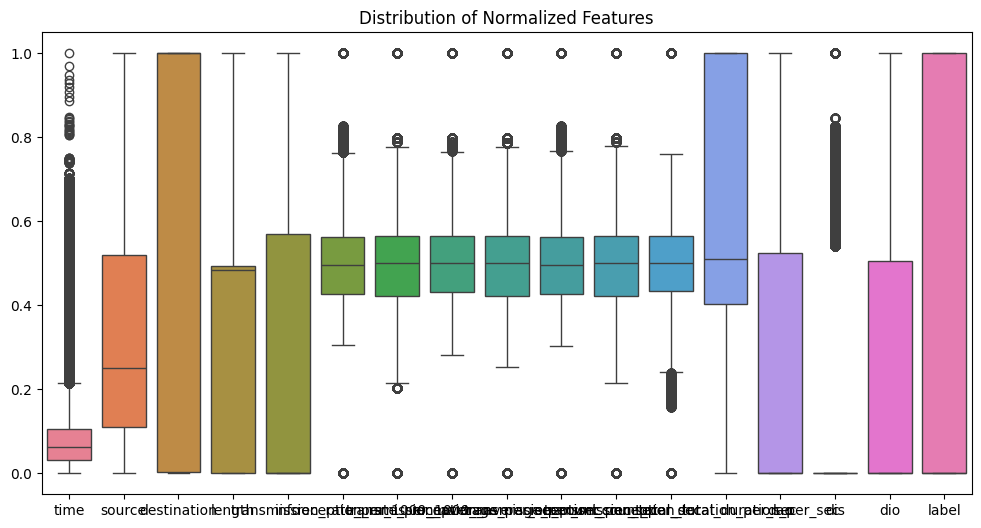


Fig.7 Distribution of Normalized Features

The top features along with their MI score based on Mutual Information is given below in Fig.8 and Fig.9

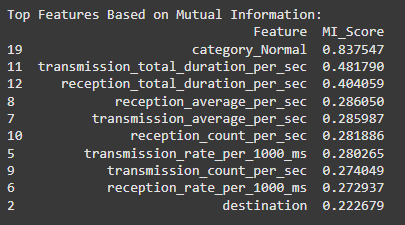


Fig.8 Top Features with the Mutual Score (MI)

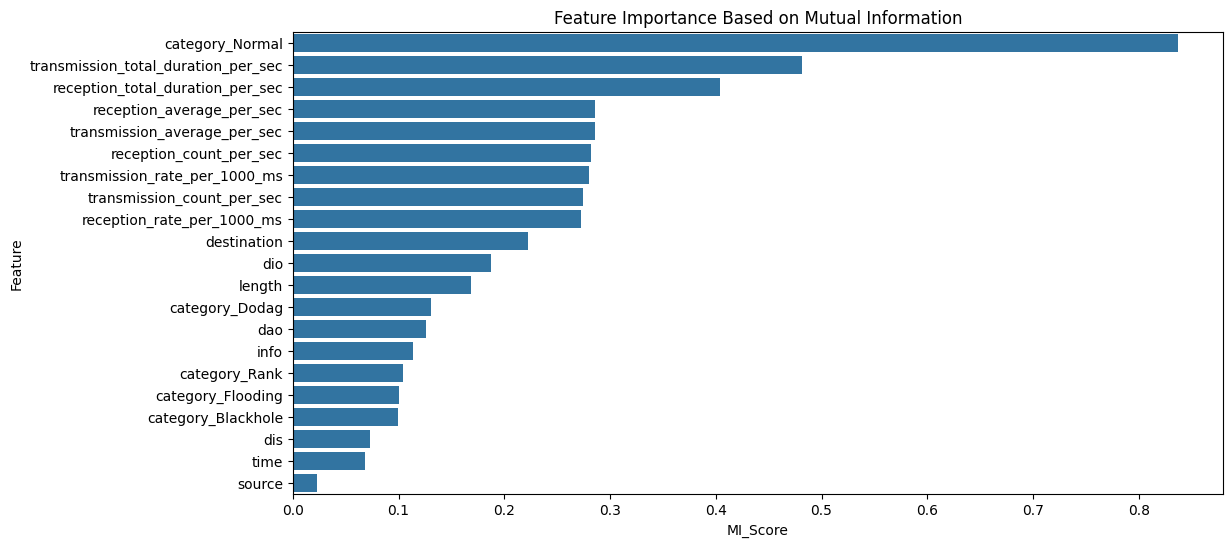


Fig.9 Visualization of the MI scores

The distribution of the engineered features is as shown below in Fig.10

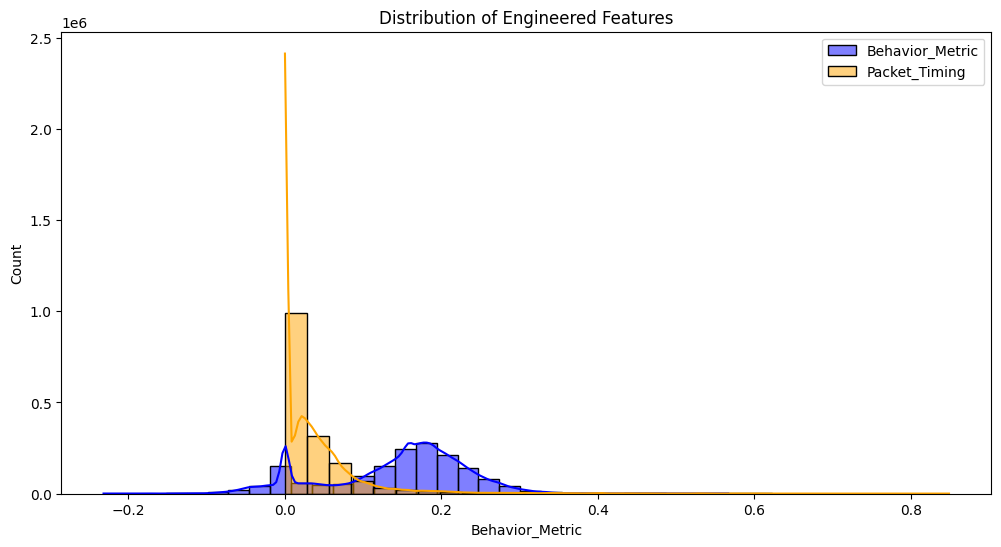


Fig.10 Distribution of the Engineered Features

The dataset split was into 80:20 ratio and the no. of rows along with features is as given below in Fig.11

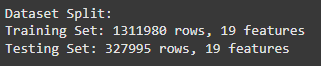


Fig.11 Dataset Split

The label distribution of the training and testing dataset is as plotted in the Fig.12 below.

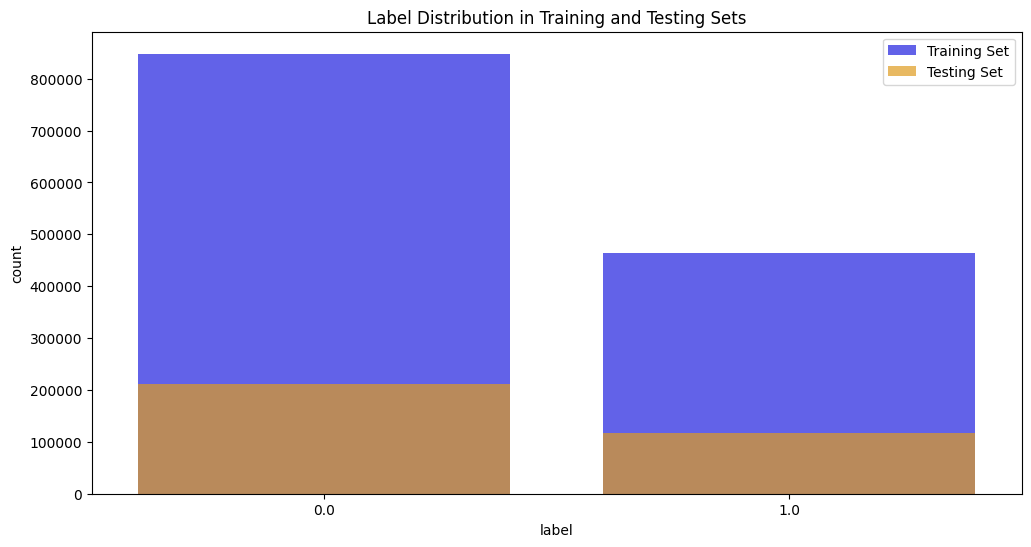


Fig.12 Label Distribution in Training and Testing Dataset

The optimized Q-Table with various levels of trust and its classification is as given in the Fig.13 below.

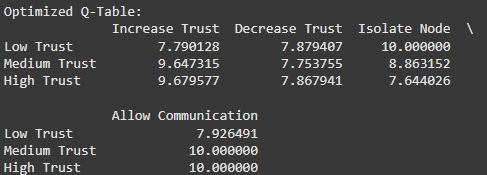


Fig.13 Optimized Q-table

The Heatmap was plotted for the Q-table to better visualize as given in the Fig.14 below.

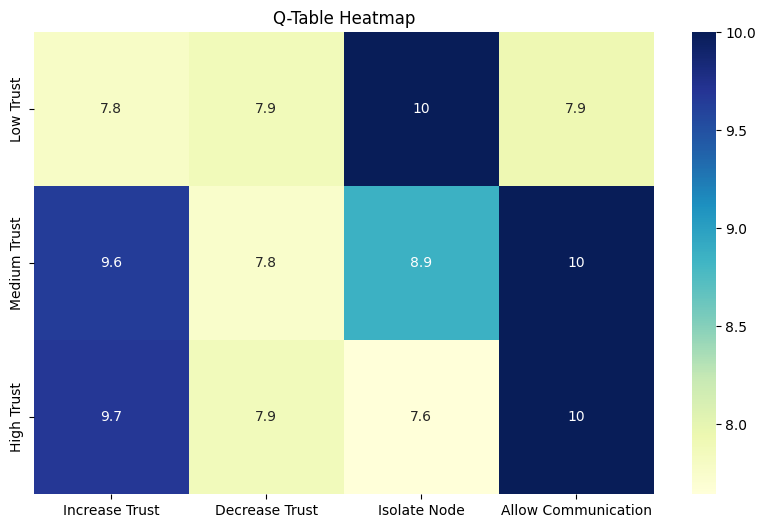


Fig.14 Q-table Heatmap

The trust scores before and after RL Adjustment are as shown in the Fig.15 below.

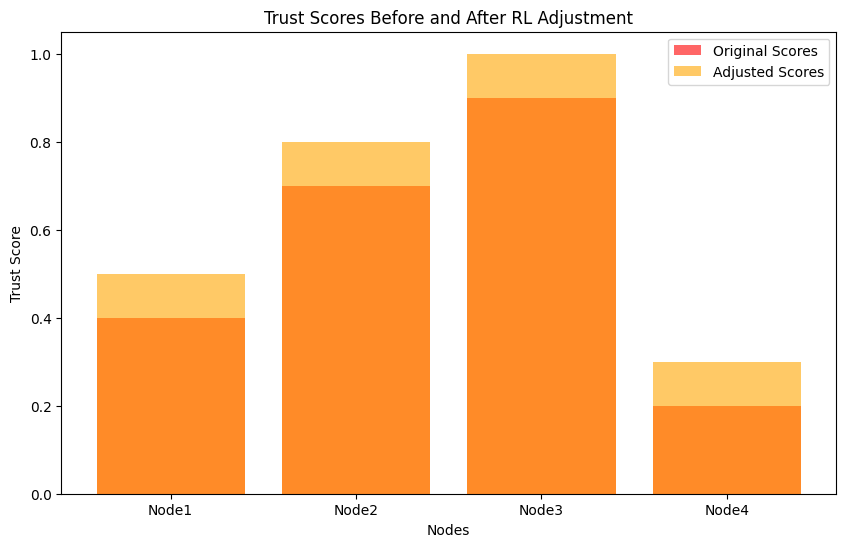


Fig.15 Trust Scores before vs after RL Adjustment

**CHAPTER – V**

**CONCLUSION AND FUTURE SCOPE**

The proposed Auto-Zero Trust with Reinforcement Learning (Auto-ZT-RL) framework has demonstrated its effectiveness in enhancing the security of IoT networks by integrating adaptive trust mechanisms and dynamic reinforcement learning optimization. Through comprehensive preprocessing, trust score initialization, and reinforcement learning-based trust adjustment, the methodology has shown significant promise in identifying and mitigating malicious activity within the ROUT-4-2023 dataset. The incorporation of Zero Trust principles ensures a robust foundation for securing communication at every level, while the reinforcement learning module provides an adaptive mechanism to optimize trust decisions in dynamic IoT environments. The modular approach of this framework facilitates easy scalability and compatibility with various classification models, ensuring high accuracy and reliability in detecting malicious activities. The results obtained so far highlight the potential of Auto-ZT-RL in addressing existing challenges in IoT network security, including scalability, resource efficiency, and adaptability.

The scalability of this model extends beyond IoT security, with potential applications in domains such as healthcare systems, financial networks, and smart city infrastructures. By incorporating domain-specific features and trust parameters, the Auto-ZT-RL framework can be tailored to secure sensitive data and ensure reliable communication in a variety of contexts. Furthermore, real-time deployment of this model in IoT networks and edge devices presents an exciting opportunity to reinforce trust-based decision-making in real-world environments. Future work will focus on improving the framework’s adaptability to large-scale IoT deployments, integrating federated learning for decentralized security management, and exploring advanced metrics such as Quality of Service (QoS) and energy-aware routing. The continued evaluation and optimization of this model across various machine learning classifiers and real-time scenarios will pave the way for its adoption in critical infrastructures and high-security applications.

**REFERENCES**

1. Ayan Chatterjee, Bestoun S. Ahmed,IoT anomaly detection methods and applications: A survey, Internet of Things, Volume 19, 2022, 100568, ISSN 2542-6605, <https://doi.org/10.1016/j.iot.2022.100568>. (<https://www.sciencedirect.com/science/article/pii/S2542660522000622>)
2. Makina, H., Letaifa, A. B., & Rachedi, A. (2024). Survey on security and privacy in Internet of Things‐based eHealth applications: Challenges, architectures, and future directions. Security and Privacy, 7(2), e346.
3. Mayzaud, A., Badonnel, R., & Chrisment, I. (2016). Detecting version number attacks in RPL-based networks using a distributed monitoring architecture. 2016 12th International Conference on Network and Service Management (CNSM), 127-135.
4. Ahmed, F., & Ko, Y. (2016). A Distributed and Cooperative Verification Mechanism to Defend against DODAG Version Number Attack in RPL. International Conference on Pervasive and Embedded Computing and Communication Systems.
5. Thungon, L. C., Sahana, S. C., & Hussain, I. (2024). “A lightweight authentication scheme for 6LoWPAN-based Internet-of -things.” Information Security Journal: A Global Perspective, 33(4), 443–453. <https://doi.org/10.1080/19393555.2024.2332962>
6. Wallgren L, Raza S, Voigt T. Routing Attacks and Countermeasures in the RPL-Based Internet of Things. International Journal of Distributed Sensor Networks. 2013;9(8). doi:10.1155/2013/794326
7. Taief Alaa Al-Amiedy, Mohammed Anbar, Bahari Belaton, Abdullah Ahmed Bahashwan, Iznan Husainy Hasbullah, Mohammad Adnan Aladaileh, Ghada AL Mukhaini, A systematic literature review on attacks defense mechanisms in RPL-based 6LoWPAN of Internet of Things, Internet of Things, Volume 22, 2023, 100741, ISSN 2542-6605,<https://doi.org/10.1016/j.iot.2023.100741>. (<https://www.sciencedirect.com/science/article/pii/S2542660523000641>)
8. Singh, Debabrata & Pati, Bibudhendu & Panigrahi, Chhabi & Swagatika, Shrabanee. (2020). Security Issues in IoT and their Countermeasures in Smart City Applications. 10.1007/978-981-15-1483-8\_26.
9. P. Sharma, M. Kherajani, D. Jain and D. Patel, "A Study of Routing Protocols, Security Issues and Attacks in Network Layer of Internet of Things Framework," 2nd International Conference on Data, Engineering and Applications (IDEA), Bhopal, India, 2020, pp. 1-6, doi: 10.1109/IDEA49133.2020.9170741.
10. S. A. Hinai and A. V. Singh, "Internet of things: Architecture, security challenges and solutions," 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS), Dubai, United Arab Emirates, 2017, pp. 1-4, doi: 10.1109/ICTUS.2017.8286004.
11. N. Tewari and G. Datt, "A Systematic Review of Security Issues and challenges with Futuristic Wearable Internet of Things (IoTs)," 2021 International Conference on Technological Advancements and Innovations (ICTAI), Tashkent, Uzbekistan, 2021, pp. 319-323, doi: 10.1109/ICTAI53825.2021.9673353.
12. S. Bansal and V. K. Tomar, "Challenges & Security Threats in IoT with Solution Architectures," 2022 2nd International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC), Mathura, India, 2022, pp. 1-5, doi: 10.1109/PARC52418.2022.9726660.
13. S. Siddiqui et al., "Toward Software-Defined Networking-Based IoT Frameworks: A Systematic Literature Review, Taxonomy, Open Challenges and Prospects," in IEEE Access, vol. 10, pp. 70850-70901, 2022, doi: 10.1109/ACCESS.2022.3188311.
14. K. Prathapchandran, T. Janani, A trust aware security mechanism to detect sinkhole attack in RPL-based IoT environment using random forest – RFTRUST, Computer Networks, Volume 198, 2021, 108413, ISSN 1389-1286, <https://doi.org/10.1016/j.comnet.2021.108413>. (<https://www.sciencedirect.com/science/article/pii/S1389128621003832>)
15. David Airehrour, Jairo A. Gutierrez, Sayan Kumar Ray, SecTrust-RPL: A secure trust-aware RPL routing protocol for Internet of Things, Future Generation Computer Systems, Volume 93, 2019, Pages 860-876, ISSN 0167-739X, <https://doi.org/10.1016/j.future.2018.03.021>. (<https://www.sciencedirect.com/science/article/pii/S0167739X17306581>)
16. Debasmita Dey, Nirnay Ghosh, iTRPL: An intelligent and trusted RPL protocol based on Multi-Agent Reinforcement Learning, Ad Hoc Networks, Volume 163, 2024, 103586, ISSN 1570-8705, <https://doi.org/10.1016/j.adhoc.2024.103586>. (<https://www.sciencedirect.com/science/article/pii/S1570870524001975>)
17. A. Lahbib, K. Toumi, S. Elleuch, A. Laouiti and S. Martin, "Link reliable and trust aware RPL routing protocol for Internet of Things," 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 2017, pp. 1-5, doi: 10.1109/NCA.2017.8171360.
18. R. Yugha and S. Chithra, "Attribute Based Trust Evaluation for Secure RPL Protocol in IoT Environment," 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 2019, pp. 1-7, doi: 10.1109/ViTECoN.2019.8899506.
19. D. Airehrour, J. Gutierrez and S. K. Ray, "Securing RPL routing protocol from blackhole attacks using a trust-based mechanism," 2016 26th International Telecommunication Networks and Applications Conference (ITNAC), Dunedin, New Zealand, 2016, pp. 115-120, doi: 10.1109/ATNAC.2016.7878793.
20. Hashemi, S.Y., Shams Aliee, F. Fuzzy, Dynamic and Trust Based Routing Protocol for IoT. J Netw Syst Manage 28, 1248–1278 (2020). <https://doi.org/10.1007/s10922-020-09535-y>
21. R. Mehta and M. M. Parmar, "Trust based mechanism for Securing IoT Routing Protocol RPL against Wormhole &Grayhole Attacks," 2018 3rd International Conference for Convergence in Technology (I2CT), Pune, India, 2018, pp. 1-6, doi: 10.1109/I2CT.2018.8529426.
22. Muzammal SM, Murugesan RK, Jhanjhi NZ, Humayun M, Ibrahim AO, Abdelmaboud A. A Trust-Based Model for Secure Routing against RPL Attacks in Internet of Things. Sensors. 2022; 22(18):7052. <https://doi.org/10.3390/s22187052>
23. P. Thulasiraman and Y. Wang, "A Lightweight Trust-Based Security Architecture for RPL in Mobile IoT Networks," 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 2019, pp. 1-6, doi: 10.1109/CCNC.2019.8651846.
24. A. Tandon and P. Srivastava, "Trust-based Enhanced Secure Routing against Rank and Sybil Attacks in IoT," 2019 Twelfth International Conference on Contemporary Computing (IC3), Noida, India, 2019, pp. 1-7, doi: 10.1109/IC3.2019.8844935.
25. Arshad D, Asim M, Tariq N, Baker T, Tawfik H, Al-Jumeily OBE D (2022) THC-RPL: A lightweight Trust-enabled routing in RPL-based IoT networks against Sybil attack. PLoS ONE 17(7): e0271277. <https://doi.org/10.1371/journal.pone.0271277>
26. Ioulianou PP, Vassilakis VG, Shahandashti SF. A Trust-Based Intrusion Detection System for RPL Networks: Detecting a Combination of Rank and Blackhole Attacks. Journal of Cybersecurity and Privacy. 2022; 2(1):124-153. https://doi.org/10.3390/jcp2010009.

**APPENDIX A – SOURCE CODE**

**1. PRE-PROCESSING MODULE**

import pandas as pd

from google.colab import drive

drive.mount('/content/drive')

# Define file paths for the four attack scenarios

file\_paths = [

'/content/drive/MyDrive/blackhole.csv',

'/content/drive/MyDrive/flooding.csv',

'/content/drive/MyDrive/dodag.csv',

'/content/drive/MyDrive/rank.csv'

]

# Load all CSV files into a list of DataFrames

dfs = [pd.read\_csv(file) for file in file\_paths]

# Concatenate all DataFrames into a single DataFrame

data = pd.concat(dfs, ignore\_index=True)

# Display the first few rows

print("Dataset Preview:")

print(data.head())

# Basic dataset information

print("\nDataset Info:")

print(data.info())

# Check for missing values

print("\nMissing Values (Before Handling):")

print(data.isnull().sum())

from sklearn.preprocessing import MinMaxScaler

# Normalize numerical columns using Min-Max Scaler

scaler = MinMaxScaler()

numeric\_columns = data.select\_dtypes(include=['float64', 'int64']).columns

data[numeric\_columns] = scaler.fit\_transform(data[numeric\_columns])

# Display the normalized data

print("\nNormalized Data (Preview):")

print(data.head())

import matplotlib.pyplot as plt

import seaborn as sns

# Visualize the normalized distribution

plt.figure(figsize=(12, 6))

sns.boxplot(data=data[numeric\_columns])

plt.title("Distribution of Normalized Features")

plt.show()

# Separate features and target

X = data.drop(columns=['label']) # Assuming 'LABEL' is the target

y = data['label']

# Identify columns with 'object' dtype (likely containing strings)

categorical\_cols = X.select\_dtypes(include=['object']).columns

# Import OneHotEncoder for categorical feature encoding

from sklearn.preprocessing import OneHotEncoder

# Create and apply OneHotEncoder

encoder = OneHotEncoder(sparse\_output=False, handle\_unknown='ignore') # sparse=False for compatibility with mutual\_info\_classif

encoded\_data = encoder.fit\_transform(X[categorical\_cols])

# Create a DataFrame from the encoded data

encoded\_df = pd.DataFrame(encoded\_data, columns=encoder.get\_feature\_names\_out(categorical\_cols))

# Drop original categorical columns and concatenate encoded columns

X = X.drop(columns=categorical\_cols)

X = pd.concat([X, encoded\_df], axis=1)

# Now you can calculate Mutual Information scores

from sklearn.feature\_selection import mutual\_info\_classif

mi\_scores = mutual\_info\_classif(X, y, random\_state=42)

mi\_df = pd.DataFrame({'Feature': X.columns, 'MI\_Score': mi\_scores}).sort\_values(by='MI\_Score', ascending=False)

# Display top features

print("\nTop Features Based on Mutual Information:")

print(mi\_df.head(10))

# Visualize MI scores

plt.figure(figsize=(12, 6))

sns.barplot(x='MI\_Score', y='Feature', data=mi\_df)

plt.title("Feature Importance Based on Mutual Information")

plt.show()

# Example: Creating trust-based features

data['Behavior\_Metric'] = (data['transmission\_rate\_per\_1000\_ms'] + data['reception\_rate\_per\_1000\_ms'] - data['transmission\_average\_per\_sec']) / 3

data['Packet\_Timing'] = data['time'] \* data['length']

# Display the updated dataset

print("\nDataset with Engineered Features (Preview):")

print(data.head())

# Visualize the distribution of new features

plt.figure(figsize=(12, 6))

sns.histplot(data['Behavior\_Metric'], kde=True, bins=30, color='blue', label='Behavior\_Metric')

sns.histplot(data['Packet\_Timing'], kde=True, bins=30, color='orange', label='Packet\_Timing')

plt.legend()

plt.title("Distribution of Engineered Features")

plt.show()

from sklearn.model\_selection import train\_test\_split

# Split dataset into training and testing sets (80/20 split)

X = data.drop(columns=['label'])

y = data['label']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Print the shape of the splits

print("\nDataset Split:")

print(f"Training Set: {X\_train.shape[0]} rows, {X\_train.shape[1]} features")

print(f"Testing Set: {X\_test.shape[0]} rows, {X\_test.shape[1]} features")

# Visualize the distribution of target labels in the splits

plt.figure(figsize=(12, 6))

sns.countplot(x=y\_train, label="Training Set", color='blue', alpha=0.7)

sns.countplot(x=y\_test, label="Testing Set", color='orange', alpha=0.7)

plt.legend()

plt.title("Label Distribution in Training and Testing Sets")

plt.show()

data.to\_csv('preprocessed\_data.csv', index=False)

from google.colab import files

files.download('preprocessed\_data.csv')

**2. REINFORCED AUTO-ZERO TRUST MODULE**

# Define states, actions, and rewards

states = ["Low Trust", "Medium Trust", "High Trust"]

actions = ["Increase Trust", "Decrease Trust", "Isolate Node", "Allow Communication"]

# Initialize Q-table

q\_table = pd.DataFrame(0, index=states, columns=actions)

# Define reward function

rewards = {

("Low Trust", "Increase Trust"): -1,

("Low Trust", "Decrease Trust"): -1,

("Low Trust", "Isolate Node"): +1,

("Low Trust", "Allow Communication"): -1,

("Medium Trust", "Increase Trust"): +1,

("Medium Trust", "Decrease Trust"): -1,

("Medium Trust", "Isolate Node"): 0,

("Medium Trust", "Allow Communication"): +1,

("High Trust", "Increase Trust"): +1,

("High Trust", "Decrease Trust"): -1,

("High Trust", "Isolate Node"): -1,

("High Trust", "Allow Communication"): +1,

}

# Parameters for Q-Learning

alpha = 0.1 # Learning rate

gamma = 0.9 # Discount factor

epsilon = 1.0 # Initial exploration rate

epsilon\_decay = 0.99

min\_epsilon = 0.01

episodes = 1000

# Function to choose action based on epsilon-greedy policy

def choose\_action(state):

if np.random.rand() < epsilon:

return np.random.choice(actions)

else:

return q\_table.loc[state].idxmax()

import numpy as np

# Q-Learning algorithm

for episode in range(episodes):

# Randomly initialize a state

state = np.random.choice(states)

for step in range(10): # Limit steps per episode

# Choose an action

action = choose\_action(state)

# Get reward for the state-action pair

reward = rewards.get((state, action), 0)

# Simulate the next state (simplified here for demo purposes)

next\_state = np.random.choice(states)

# Update Q-value

q\_table.loc[state, action] += alpha \* (reward + gamma \* q\_table.loc[next\_state].max() - q\_table.loc[state, action])

# Transition to the next state

state = next\_state

# Decay epsilon

epsilon = max(min\_epsilon, epsilon \* epsilon\_decay)

import matplotlib.pyplot as plt

import seaborn as sns

# Display the Q-table

print("Optimized Q-Table:")

print(q\_table)

# Visualize Q-values

plt.figure(figsize=(10, 6))

sns.heatmap(q\_table, annot=True, cmap="YlGnBu")

plt.title("Q-Table Heatmap")

plt.show()

# Simulated trust score adjustments

trust\_scores = {"Node1": 0.4, "Node2": 0.7, "Node3": 0.9, "Node4": 0.2}

adjusted\_scores = {}

for node, trust in trust\_scores.items():

# Determine initial state

if trust < 0.5:

state = "Low Trust"

elif trust < 0.8:

state = "Medium Trust"

else:

state = "High Trust"

# Choose action based on optimized Q-table

action = q\_table.loc[state].idxmax()

# Update trust score based on action

if action == "Increase Trust":

adjusted\_scores[node] = min(1.0, trust + 0.1)

elif action == "Decrease Trust":

adjusted\_scores[node] = max(0.0, trust - 0.1)

elif action == "Isolate Node":

adjusted\_scores[node] = 0.0

else: # Allow Communication

adjusted\_scores[node] = trust

# Display original and adjusted trust scores

print("Original Trust Scores:", trust\_scores)

print("Adjusted Trust Scores:", adjusted\_scores)

# Visualize trust scores

original\_scores = list(trust\_scores.values())

adjusted\_scores\_values = list(adjusted\_scores.values())

plt.figure(figsize=(10, 6))

plt.bar(trust\_scores.keys(), original\_scores, alpha=0.6, label="Original Scores", color="red")

plt.bar(adjusted\_scores.keys(), adjusted\_scores\_values, alpha=0.6, label="Adjusted Scores", color="orange")

plt.xlabel("Nodes")

plt.ylabel("Trust Score")

plt.title("Trust Scores Before and After RL Adjustment")

plt.legend()

plt.show()