Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 27 No. 2 (2024)

Memory-Efficient Trust Management for IoT Nodes Enabling Secure Communication in Trusted Environments

[*]Satish Kamble[1,2] , Dr. Surendra Mahajan[3]

[1]Research Scholar, Department of Computer Engineering, SKNCOE, SPPU, Pune, India

[1*]satkam53@gmail.com

[2,3]Department of Information Technology, PVG'S COET & GKPWIOM, SPPU, Pune, India

[3]sa_mahajan@yahoo.com

Article History:

***Received**: 09-02-2024*

***Revised**: 28-04-2024*

***Accepted**: 18-05-2024*

Abstract:

The Internet of Things (IoT) is changing quickly, so making sure that gadgets that are linked can talk to each other safely is very important. This paper describes a new way to handle trust that is specifically made for IoT nodes that work in safe settings. Conventional models frequently have a parcel of additional memory needs and are difficult to compute, which is made more regrettable by IoT gadgets that do not have a part of assets. Our proposed approach bargains with these issues by employing a belief administration framework that doesn't utilize a parcel of memory and finds an adjustment between security needs and the restricted assets that IoT hubs have. The foremost inventive thing around our strategy is that it can alter belief levels on the fly based on modern data and past experiences. Our framework takes up as small memory as conceivable whereas still having solid security measures since it employs lightweight cryptography conventions and quick information structures. This lets Internet of Things (IoT) gadgets utilize secure communication strategies without abating them down or making them less dependable. Context-aware belief assessment could be a key portion of our framework. This implies that belief levels are continually changed based on real-time encompassing variables and the way hubs are connected with each other. This adaptable alter makes beyond any doubt that belief choices remain in line with changing working conditions and threat scenarios. We moreover made beyond any doubt that our strategy works well with current IoT plans, permitting distinctive sorts of systems to work together and develop. We demonstrate that our system works in a wide range of IoT application cases by running a part of models and real-world tests on it. This study shows an easy-to-use and expandable way to improve trust management in IoT settings, with a focus on memory efficiency without losing safety. By using our approach, IoT stakeholders can reduce the risks that come with old trust models. This will make IoT environments more reliable and adaptable in the future.

Keywords: Trust management, Memory-efficient security, Decentralized trust systems, Context-aware trust evaluation, cryptographic protocols

1. Introduction

With the rise of Internet of Things (IoT) gadgets, there's presently more association and value than ever recently in numerous regions, from shrewd homes and work environment innovation to healthcare and transportation. But this network too brings up enormous security issues, particularly when it comes to how solid the contact between IoT hubs is. Making beyond any doubt that communication is secure and solid is imperative for keeping private information secure and IoT biological systems running easily. Conventional ways of overseeing belief in IoT settings regularly utilize complicated cryptography and a part of memory, which may not work well for gadgets that do not have a part of assets, which is common in IoT organizations [1]. These issues appear to show how vital it is to come up with better approaches to deal with security issues while taking into consideration the limits of IoT equipment. This paper tries to illuminate these issues by recommending a modern belief administration framework that works well with small memory and is planned for IoT hubs that are utilized in trusted settings [2]. Our strategy points to progress the speed and scaling of IoT frameworks by making belief assessment forms run more easily whereas utilizing less memory.

We are doing this since we have to discover ways to make IoT systems less helpless to the issues that come with using traditional belief models [3]. IoT gadgets frequently have to bargain with overwhelming handling and memory loads when utilizing current strategies, which can moderate them down and make them utilize more vitality. Too, standard models might not be able to adjust well to IoT settings that alter all the time, with modern dangers and distinctive ways of communicating. One of the hardest things around making belief administration frameworks for the Web of Things is making beyond any doubt they are secure without utilizing as well many resources. IoT gadgets ordinarily do not have a parcel of processing power, memory, or vitality, so it's vital to make solutions that are little and can work well with these restrictions. Our thought for a solution includes a better approach to handle trust that uses basic encryption strategies and keen information structures to create the most excellent utilization of memory and computer control. Context-aware belief assessment could be a key portion of our strategy [4]. This implies that belief connections are always changing based on real-time setting data and past trades between IoT hubs. By routinely changing belief components, our framework makes belief choices more fast and adaptable, which improves the by and large security of IoT arrangements. The foremost vital thing about our framework is that it works well with other IoT plans and conventions. This makes beyond any doubt that distinctive systems can communicate with each other and develop. We push how imperative it is to remain congruous with industry benchmarks and conventions so that they can be broadly utilized and embraced in a wide extent of IoT applications.

This work adds to the area of IoT security and trust management in a number of ways:

- Memory-Efficient Design: We suggest a trust management system that uses the least amount of memory possible while still providing strong protection.

- Dynamic Trust Evaluation: Our method uses dynamic trust evaluation tools that change based on the ambient conditions and the nodes' behavior patterns.
- Evidence-Based Validation: To show that our system works in real time situations, we use a lot of models and real-world tests to show that it is effective.

2. Related work

A lot of study has been done in the area of trust management in IoT settings to try to solve the problems that connected devices cause with security and efficiency. This part looks at previous research and methods that are related to our suggested memory-efficient trust management paradigm. One [5] of the first ways that IoT networks managed trust was by using centralized methods for identification and permission. Most of the time, these models use complex security methods and centralized trust bodies, which can make it hard for IoT devices that don't have a lot of resources to grow [6]. These old models work well in controlled settings, but they're not very good at adapting to the changing and unstructured nature of IoT networks. These [7] were created because controlled methods have some problems. Using peer-to-peer (P2P) networks and distributed ledger technologies like blockchain, these systems give IoT nodes the job of managing trust among themselves. These models attempt to form things more secure and safer to single focuses of disappointment by spreading out choices around belief [8]. On the other hand, they regularly require a part of computing control and might not work for all IoT employments since of tall delay and overhead [9]. Systems [10] for context-aware belief administration have become a curious way to create belief choices in IoT settings more adaptable and fast to act. To alter belief levels on the fly [11], these frameworks see at things just like the position of the gadget, the environment, and past contacts. By utilizing real-time foundation information, these models make appraisals more exact and make strides framework security as a whole [12]. A part of the investigation has gone into making belief models that utilize few assets and work well with IoT operations [13]. These models attempt to keep security levels tall whereas minimizing the sum of memory and computing work that must be done. Analysts have looked into ways to induce the foremost out of assets without relinquishing security [14], such as utilizing lightweight cryptographic strategies, probabilistic belief assessment, and versatile cut-off plans. It [15] has moreover been utilized to progress belief administration in IoT systems by using machine learning strategies. Machine learning calculations can naturally discover unusual behaviours and figure the level of dependability of hubs that are communicating with each other by looking at tremendous datasets of IoT gadget behavior and organizing activity designs [16]. The great thing around these data-driven strategies is that they can be changed and scaled up, but they might require a parcel of computing control for preparation and expectation [17]. The utilization of edge computing in IoT plans is developing, and individuals are becoming more interested in including belief administration highlights to the organized edge. Edge-based belief administration frameworks utilize edge computer assets to create choices in genuine time [19], which ought to cut down on delay and transfer speed by performing belief evaluations closer to IoT gadgets. This strategy makes

things more quick and adaptable whereas bringing down dependence on centralized innovation [20].

Table 1: Summary of related work

Method	Approach	Key Finding	Application	Limitation	Scope
Traditional Trust Models [12]	Centralized authentication, heavyweight protocols	Scalability issues with resource-constrained IoT devices	Controlled environments	High computational overhead	Enhancing scalability in large-scale IoT
Decentralized Trust Systems [21]	Distributed trust decisions, blockchain	Resilience against single points of failure	P2P networks, Blockchain	High computational and storage overhead	Enhancing fault tolerance in IoT networks
Context-Aware Trust Management [13]	Real-time contextual data, dynamic adjustments	Improved accuracy in trust assessments	IoT environments	Dependency on accurate context data	Enhancing adaptability in dynamic IoT
Resource-Efficient Trust Models [14]	Lightweight cryptography, probabilistic evaluation	Minimization of memory and computational overhead	Resource-constrained IoT deployments	Potential trade-off in security strength	Optimizing resource utilization in IoT
Machine Learning-Based Approaches [15]	Data-driven trust evaluation, anomaly detection	Autonomy in anomaly detection and trust prediction	IoT device behavior analysis	High computational resource requirement	Improving adaptive trust management
Integration with Edge Computing [22]	Trust evaluation at network edge	Reduced latency and bandwidth consumption	Edge computing environments	Limited to edge computing infrastructure	Enhancing real-time decision-making in IoT
Hybrid Trust Models [24]	Combination of centralized and decentralized approaches	Balances scalability and resilience in trust management	Hybrid IoT environments	Integration complexity between models	Achieving flexibility in trust management
Reputation-Based Systems [23]	Trust based on historical behavior and reputation scores	Simplifies trust decisions based on community feedback	Social IoT networks	Vulnerability to manipulated ratings	Enhancing community-driven trust metrics

3. Methodology

1. Plot Nodes and Edges:

The first step is to make a graph. The nodes in the graph are the IoT devices or network parts, and the edges are the links between them. When a node is first created, it has set properties like energy amounts and trust numbers. These characteristics are very important for modelling how IoT nodes in the network behave and what they can do.

- Node Position Equation (x_i, y_i):

$$pi(t) = \big(xi(t), yi(t), zi(t)\big)$$

Where,

- $pi(t)$ is the position vector of the node i at time t.
- $xi(t), yi(t), and\ zi(t)$ are the coordinates of the node i at time t in the x, y, and z directions, respectively.

- Edge Representation Equation:

$$E\ =\ \{(u, v)\ |\ u, v\ in\ V\ and\ u\ \neq\ v\}$$

where E represents the set of edges connecting nodes u and v, and V is the set of all nodes in the graph.

- Graph Degree Equation:

$$\deg(v) = \sum_{u \in V(G)} A_{vu}$$

where A_{vu} is the element of the adjacency matrix A of the graph G corresponding to the edge between nodes v and u, and V(G) is the set of vertices in the graph G.

- Edge Weight Equation:

$$w_{\{ij\}} = \sqrt{\left(\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2\right)}$$

Where $w_{\{ij\}}$is the weight of the edge between vertices i and j. This equation calculates the Euclidean distance between nodes i and j, assuming a geometric interpretation of edge weights based on node positions.

2. Generate Graph:

Once the nodes and lines have been set up, the graph is made to show how the network is organized. This step makes a graph that shows how the nodes are connected and laid out. This makes it easy to see how data will move through the network, illustrated in figure 1.

Generate Graph:

1. Node Position Equations:

$$pi(t) = \big(xi(t), yi(t), zi(t)\big)$$

- Defines $pi(t)$ is the position vector of the node iii at time t.

$xi(t), yi(t), and\ zi(t)$ are the coordinates of the node i at time t in the x, y, and z directions, respectively.

2. Edge Representation Equations:

$$E = \{(u, v) \mid u, v\ in\ V\ and\ u \neq v\}$$

- Represents the set E of edges connecting nodes u and v, ensuring no self-loops and distinct connections.

3. Graph Degree Equations:

$$\deg(v) = \sum_{u \in V(G)} A_{vu}$$

- Calculates A_{vu} is the element of the adjacency matrix A of the graph G corresponding to the edge between nodes v and u, and V(G) is the set of vertices in the graph G

4. Edge Weight Equations (Euclidean Distance):

$$w_{\{ij\}} = \sqrt{\left(\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2\right)}$$

- $w_{\{ij\}}$is the weight of the edge between vertices i and j, based on the Euclidean distance between their respective coordinates.

5. Graph Construction Equations:

$$G = (V, E)$$

- Defines the graph G as a tuple consisting of the set of nodes V and the set of edges E, completing the graph construction process.

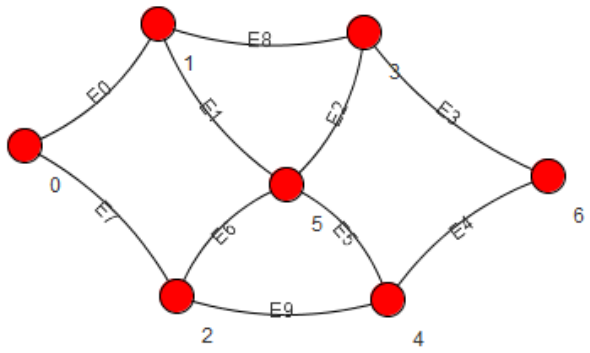

Figure 1: Graph Generation

3. Select Source and Destination:

To choose a source and a target, click on a node in the graph. The source node is node 0 and the goal node is node 6. These nodes show where the ends of the data exchange will be. It is important to choose these nodes so that you can simulate data transfer situations and test how well the trust management system protects communication, shown in figure 2.

Select Source and Destination:

- Source Selection Equation:

$$Source\ node: s \in V$$

- Represents the selection of a source node s from the set of all nodes V in the graph.

- Destination Selection Equation:

$$Destination\ node: t \in V \setminus \{s\}$$

- Represents the selection of a destination node t from the set of all nodes V, excluding the source node s.

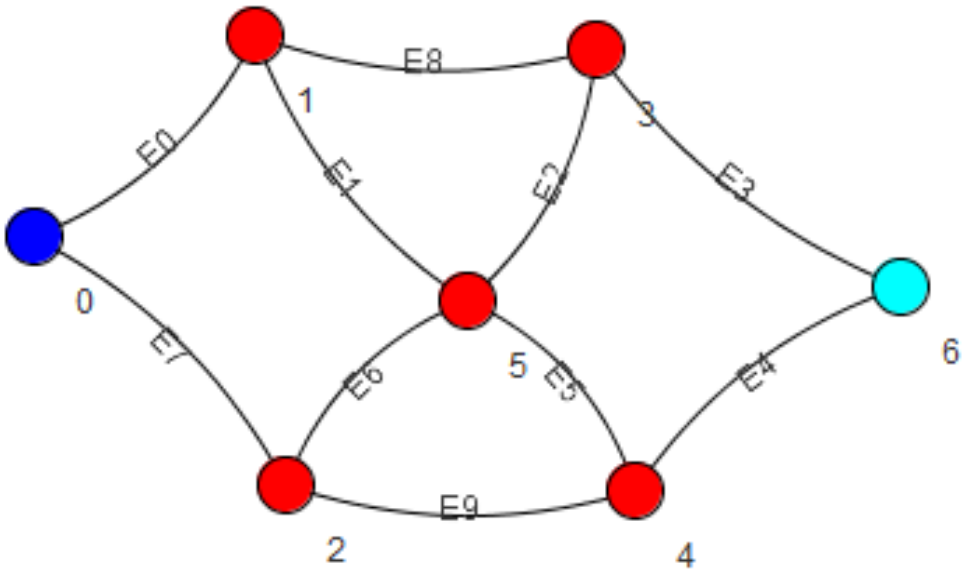

Figure 2: Figure. Source (0) and Destination Selection (6)

4. Get All Paths from Source to Destination:

This finds all the possible routes from the source node you choose to the target node. Path finding methods, like breadth-first search (BFS) or depth-first search (DFS), are used in this step to make a list of all the possible ways to get through the network. Each path shows a possible way that data packets could take, shown in figure 3.

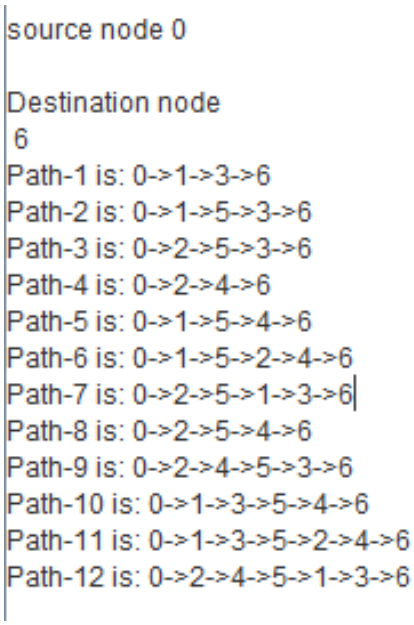

Figure 3: All Possible Paths from Source to Destination

5. Get Shortest Path:

To find the fastest way to get from the source node to the target node, the shortest path method, like Dijkstra's algorithm, is used. This method is very important because it is the main way that data is sent when everything is working as it should. It maximizes the speed of data movement and minimizes delay, shown in figure 4.

Get Shortest Path:

1. Initialize Distance:

$$d(s) = 0, d(v) = \infty \, for \, all \, v \in V, v \neq s$$

- Set the distance to the source node s as 0, and all other nodes initially to infinity.

2. Priority Queue Initialization:

$$Q = V$$

- Initialize a priority queue Q with all nodes in the graph V.

3. Relaxation Process:

$$d(v) \leftarrow \big(d(v), d(u) + w(u, v)\big)$$

- For each node v in Q and its neighbour u, update the distance d(v) if a shorter path through u is found, where w(u, v) is the weight of the edge between u and v.

4. Extract Minimum Distance:

$$u = extract - min(Q)$$

- Extract the node u with the minimum distance d(u) from Q.

5. Repeat Until Destination Found:

- Continue steps 3-4 until the target node t is extracted from Q.

6. Path Reconstruction:

- If the shortest path is required, backtrack from t to s using the predecessor information stored during relaxation.

.

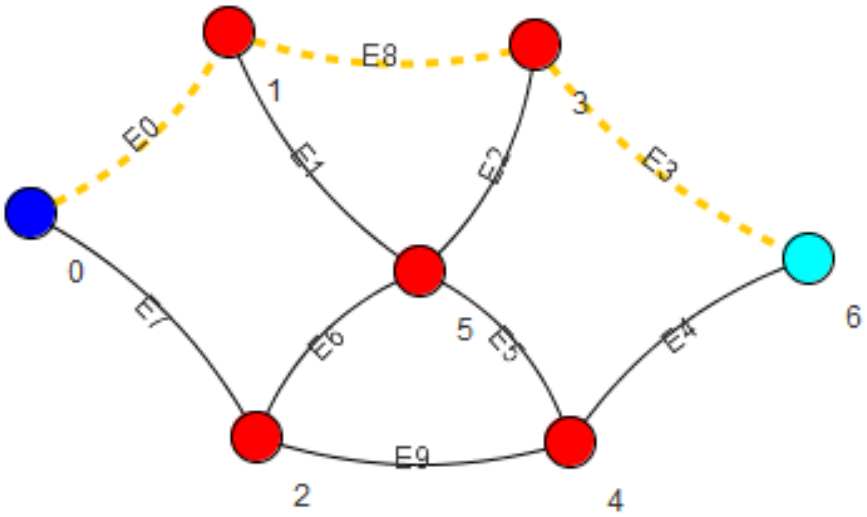

Figure 4: Shortest Path from Source to Destination

6. Trust Calculation Model

A Trust Management System (TMS) is a way to set up, keep track of, and rate the reliability of things in a network or system, like nodes, people, or devices. When used in this situation, trust means having faith that someone or something will act in a safe, honest, and reliable way. TMS is very important in places like the Internet of Things (IoT), distributed systems, and ad-hoc networks where people can't always talk to each other directly or get power from one place.

Trust attributes are the factors or criteria used to evaluate the trustworthiness of an entity. Several typical indications of trust are:

- Reliability: Reliability is a measure of how consistently something acts over time. Reliable companies always do the right thing and live up to standards.
- Reputation: Based on feedback and views from other people and things in the network. A trust score can be made by adding up reputations from different sources.
- Competence: Competence means that something is able to do the jobs or provide the services that are needed. This could be things like processing power, speed, or certain skills.
- Integrity: Integrity is a measure of how honest and true something is. The system checks to see if an object follows the rules and doesn't do anything bad.

- Security: Checks how well the organization can keep data and interactions safe from threats. This includes the ability to lock data, protect against threats, and handle private data safely.

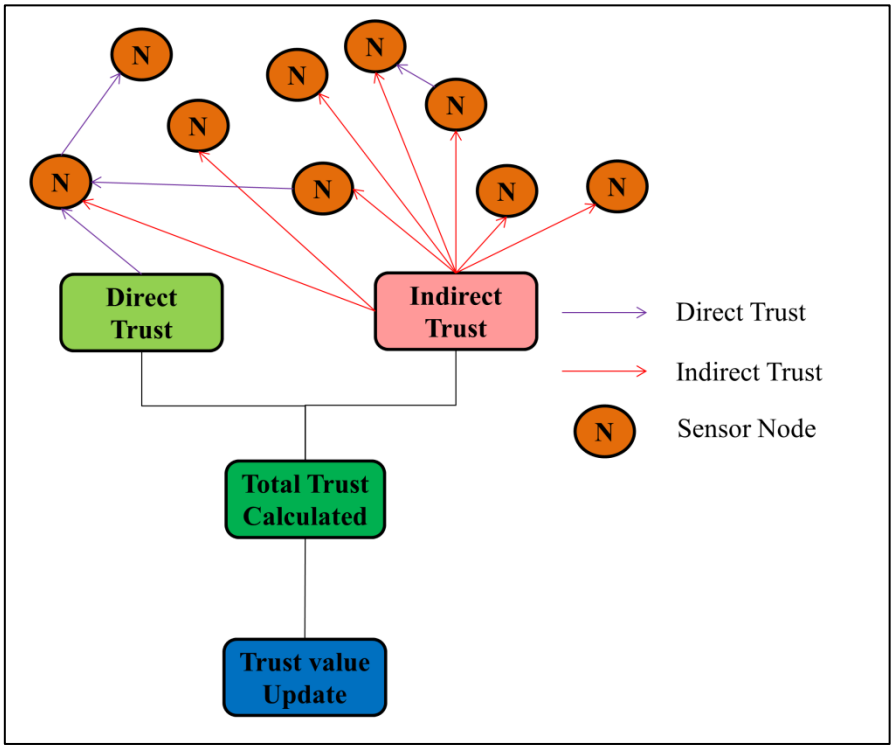

Figure 5: Overview of Trust model

Let's define a trust value T_i for node i as a function of the trust attributes. Assume the trust is computed based on reliability (R_i), reputation (P_i), and security (S_i):

$$T_i = \alpha R_i + \beta P_i + \gamma S_i$$

where:

- α, β, and γ are weight coefficients that sum to 1 ($\alpha + \beta + \gamma = 1$).
- R_i, P_i and $S_i$are normalized values (e.g., between 0 and 1).

This model, shown in figure 5, starts with the current trust value (Ti T i) of each node and figures out a starting trust value ($Ti\ ni$ T i) that balances trustworthiness with new requests or set limits (Pr). As long as $T\ i\ n\ i$ T meets the connection standards (com), data transfer is allowed. After that, the program uses a secure trust model (Ts) that adds up factors like legitimacy (A), integrity (E), and reliability (R) to check if the communication path is safe. If $T\ y$ s matches the connection standard (com), it is possible to send data. The model uses a mobility trust model (T mob) for situations where nodes are mobile. It includes measures like movement evaluation (M_eva) and mobility stability (E_mob). There can be data sharing if $T\ m\ o\ b$ T mob meets the connection barrier (com). Furthermore, relative trust (Trel) checks the strength of direct communication links (Dc) and the dependability of those links (E_dc). If $T\ r\ e\ l$ T rel meets the connection requirement (com), it can send and receive data securely.

Steps to Implement Trust Algorithm:

- Normalization:

Normalize the values of the trust attributes to ensure they are on the same scale. For example, if the reliability score ranges from 0 to 100, it should be normalized to a value between 0 and 1.

$$R_i = \frac{R_i}{R_{max}}$$

$$P_i = \frac{P_i}{P_{max}}$$

$$S_i = \frac{S_i}{S_{max}}$$

- Weight Assignment:

Assign appropriate weights to the trust attributes based on their importance. For instance, if security is more critical, assign a higher weight to γ.

- Trust Calculation:

Calculate the trust value using the weighted sum of the normalized attributes.

$$T_i = \alpha R_i + \beta P_i + \gamma S_i$$

7. Encrypt Data Using AES Encryption:

The data bits are protected using the Advanced Encryption Standard (AES) before they are sent. AES is a symmetric encryption method that is widely used because it is safe and good at keeping data safe while it's being sent, illustrating the encrypted information in figure 6. When you encrypt data, it can't be read by anyone else without the decryption key, even if it gets stolen.

AES Encryption Algorithm:

1. Key Expansion:

- Generate round keys from the initial encryption key K.

2. Initial Round Key Addition:

- Add the initial round key K to the plaintext block P:

$$S' = S \oplus K$$

- S': Initial state after adding the round key.

3. Rounds of Substitution and Permutation:

- Perform 9 rounds (for AES-128) or more depending on the key length.

4. Substitution (SubBytes Step):

- Substitute each byte of S^(i) with a corresponding byte from the S-box:

$$S' = SubBytes(S)$$

5. Permutation (ShiftRows Step):

- Permute the rows of S':

$$S'' = ShiftRows(S')$$

6. Mixing (MixColumns Step, except for the last round):

- Mix the columns of S" using a matrix multiplication in GF(2^8):

$$S\hat{}(i) = MixColumns(S'')$$

7. Final Round (AddRoundKey Step):

- Add the round key K^(i) to S^(i):

$$C = S\hat{}(9) \oplus K\hat{}(9)$$

- C: The ciphertext block after the final round.

- Send Data from Source to target: The quickest way is used to send encrypted data from the source node to the target node. This step mimics how data actually moves through the IoT network to make sure that the chosen path and encryption method work well.
- If Attack Happens on Source to Destination Path:

The Data At The Sensor Node 5 Is [G522OfzTWBEowNmfB0vMMA==, dKc/dMCTrbzaM76wNSqWBA==,
vLUuivCKih5WnQ==, l1SX6XJ2fp+/+dkaY85Ctw==, WqdAWfY/ROIsU72UlhsD5g==, 8Tw3SyU0/qy5KGJ
, fldaqZ2+EZJi4x+al3WOlw==, vquzqAbnES23qC1HFQnT8w==, hM428Wp7FZ8ttH4RRVJ6ww==, 1CuJe
zz3+uktA==, kykEPtOV+V/eO04XnHHIKQ==, ZJdmYVUgTzlkHDml0JtSxA==, Zi8sA3Hqw3XfXbG8qNoky
ordJ/8d90Mi95sTrmw==]

Figure 6: Snapshot for Encrypted data

If an attack is found on the set path between the source and target nodes, the following steps are taken:

a. Find the Trust Levels of All Nodes: The trust levels of all nodes in the path are found again using the methods described in the study paper. To revaluate each node's trustworthiness, these systems usually look at things like how it acts, how it talks to other nodes, and past trust measures.

b. Get Another Trusted Shortest Path: A new trusted shortest path is found using the updated trust values. This pathfinding process makes sure that data is sent through nodes with better trust scores instead of nodes that have been hacked or are otherwise suspicious, illustrated in figure 7.

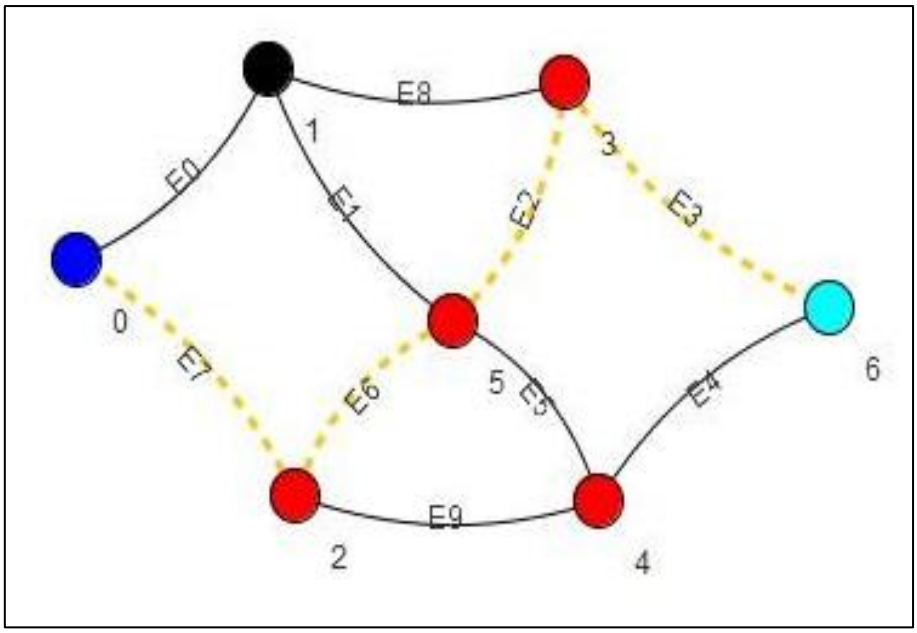

Figure 7: Data Transfer After Attack Node Detection

c. Send Data from Source to target: Once a safe path is found, the source node starts sending data again to the target node. This makes sure that data accuracy and security are kept up during the whole transfer process, even though there was an attack.

d. Recover Data at target: The AES decryption key is used to recover the protected data when it gets to the target node via the secure path. This step makes sure that only the right person can view and use the information that was sent safely.

e. Recalculating Energy and Trust Values of Nodes after Attack Node Detection: All nodes' energy and trust values are redone and updated after an attack. This evaluation helps figure out how the attack affected the speed and trustworthiness of the nodes, which in turn helps network managers make decisions in the future.

4. IV. Result and Discussion

Table 2 shows the trust calculations for network nodes, showing their energy levels and the trust numbers that go with them. There is a number assigned to each node, and its energy and security measures are shown next to it. The energy numbers show how much energy is available at each point, which is important for figuring out how well they can work in the network. Trust values, on the other hand, show how reliable or trustworthy each node is. They are usually found by using complicated formulas that look at things like past behavior, how reliable communication is, and security measures.

Table 2: Result for Trust Calculation

Node	Energy	Trust
2	600	200
0	594	202.55
3	561	202.275

6	600	200
5	573	200
1	566	196
4	600	200

When communicating and sending information over an arrangement, hubs with higher belief values are as a rule given the work of dealing with private information or vital errands to begin with, making beyond any doubt that the organization works well and securely. The information within the table appears a depiction of these measures, appearing how vitality levels and certainty shift between hubs. This kind of information examination makes a difference with overseeing systems and making choices.

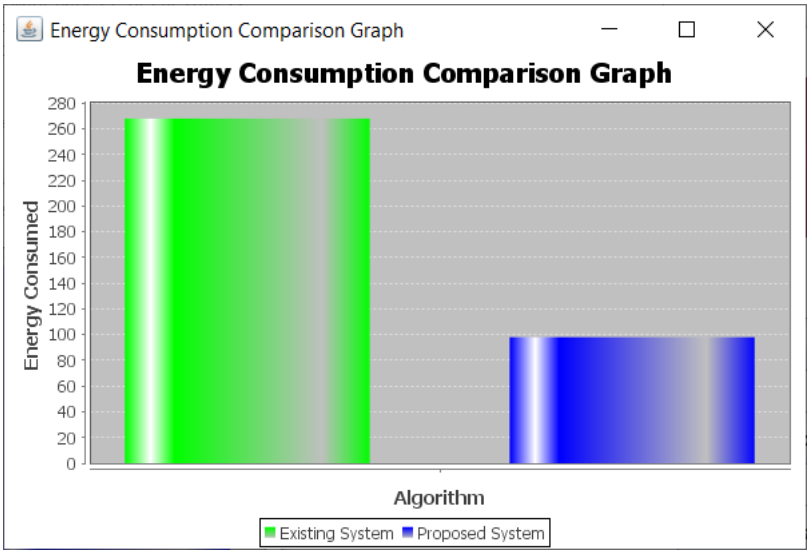

Figure 8: Energy Consumption Graph of Algorithms

It lets managers make better use of resources, lower the risks of running out of energy, and improve the general performance of networks. Also, these measurements are updated on a regular basis, which lets flexible tactics like rerouting traffic or changing communication methods based on what's happening at any given point in real time. You can use Table 2 as a starting point to figure out how to improve network dependability and efficiency by using smart trust and energy management methods. Figure 8 outlines the vitality utilization patterns of calculations utilized within the organization.

Table 3: Energy utilization of nodes before and after data transfer following dead node detection

Node ID	Default Energy (J)	Energy After Data Transfer (J)
0	600	592
1	600	580
2	600	568
3	600	583

4	600	562
5	600	600
6	600	600

The table 3 presents the vitality utilization of hubs some time recently and after information exchange taking after dead hub discovery. At first, all hubs are set with a default vitality level of 600 joules.

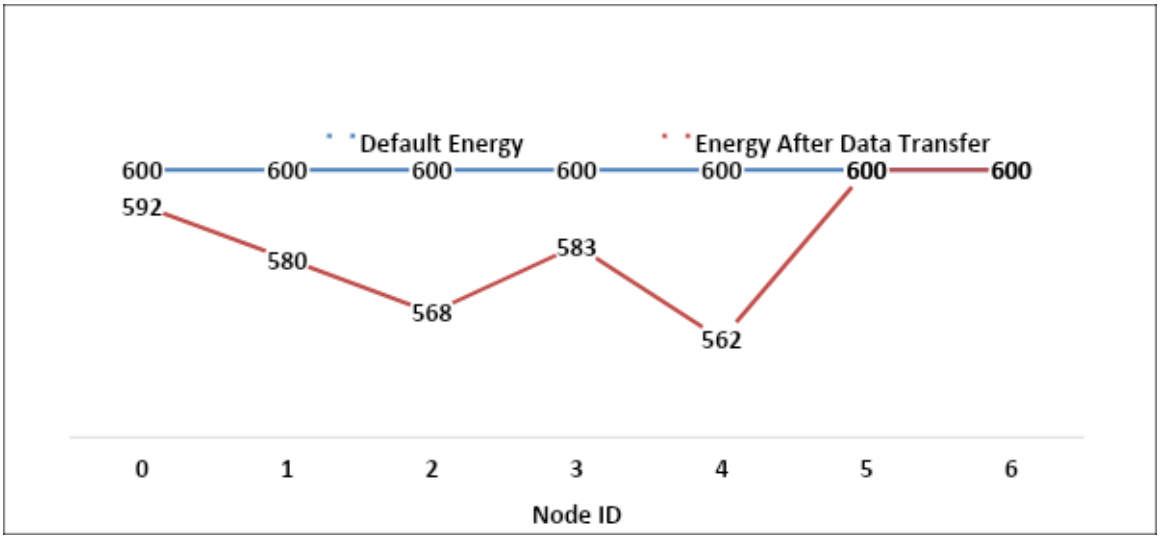

Figure 9: Energy Consumption of Node after Dead Node Detection

In any case, after information exchange, the vitality levels change over hubs, reflecting the vitality utilization amid the method. Hub 0's vitality decreases to 592 joules, demonstrating negligible utilization. Hub 1 and Hub 2 appear noteworthy drops to 580 and 568 joules, separately, proposing higher information exchange movement. Hub 3's vitality level diminishes to 583 joules, a direct lessening compared to Hubs 1 and 2. Strikingly, Hub 4 shows the most reduced vitality at 562 joules, inferring the most noteworthy vitality utilization in information exchange. In contrast, Hubs 5 and 6 keep up their vitality levels at 600 joules, indicating no association within the information exchange preparation, illustrated in figure 9. This information highlights the changing vitality requests on hubs amid organized operations and underscores the significance of effective vitality administration to preserve and arrange unwavering quality and execution. Hubs with higher vitality utilization might require closer observing or elective steering procedures to adjust the organized stack.

Table 4: Trust values of nodes before and after data transfer following dead node detection

Node ID	Default Trust Value	Trust After Data Transfer
0	200	200
1	200	202.55
2	200	202.275
3	200	200
4	200	200
5	200	196.3
6	200	200

Table 4 shows the trust levels of nodes before and after data sharing after a dead node was found. All nodes have a trust value of 200 at the start. After the data transfer, the trust values of nodes 1 and 2 go up to 202.55 and 202.275, respectively. This means that they are more reliable, most likely because the data was handled well. Node 5's trust rating drops to 196.3, which means there might be a problem or less trust during the transfer process. The trust ratings of nodes 0, 3, 4, and 6 stay the same, which means that their performance and dependability don't change much. This range of trust values makes it easier to find and manage the dependability of network nodes, which makes sure that data exchange is strong and safe, the trust calculation of nodes after attack detection shown in figure 10.

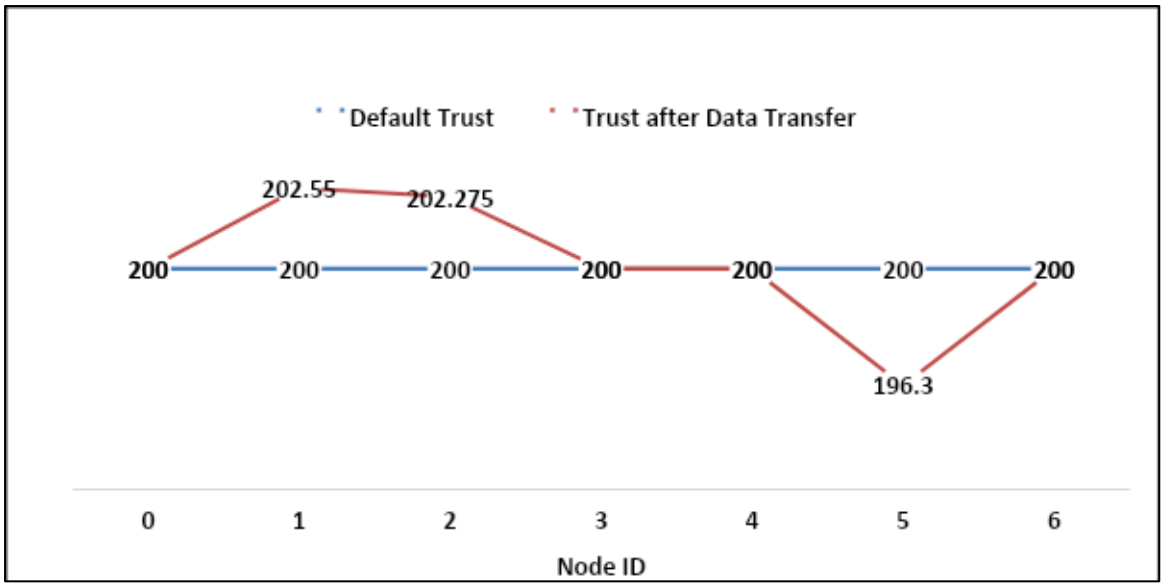

Figure 10: Trust Calculation of Nodes after Attack Detection

Table 5: Time and energy consumption data for regular and trust-based data transfers across different numbers of nodes

Number of Nodes	Regular Data Transfer (ms)	Trust Based Data Transfer (ms)	Regular Data Transfer (J)	Trust Based Data Transfer (J)
5	196	122	240	165
10	259	157	342	301
15	324	175	402	385
20	410	189	486	454

Table 5 shows a full comparison of how much time and energy normal and trust-based data exchanges take across different numbers of nodes. This information is very important for figuring out how well these two ways work in different network situations. For both time and energy economy, the table shows that trust-based data transfer always does better than normal data transfer. In this case, trust-based transfer only takes 122 milliseconds and 165 joules for 5 nodes, while normal transfer takes 196 milliseconds and 240 joules. This trend stays the same as the number of nodes grows, which shows that the trust-based method works well and can be used on a large scale.

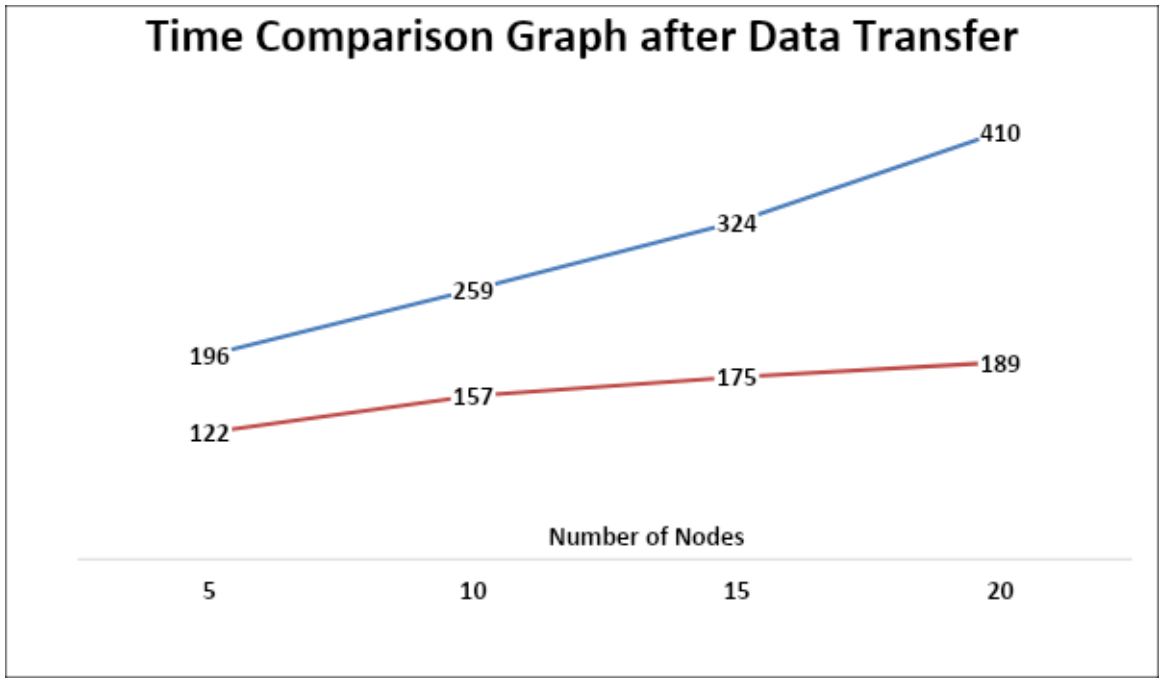

Figure 11: Time Comparison Graph for Data Transfer WRT Number of Nodes

When there are 20 nodes, it takes 410 milliseconds and 486 joules to send normal data. Trust-based transfer, on the other hand, only needs 189 milliseconds and 454 joules, which shows a big improvement in efficiency, time comparison illustrated in figure 11. Seeing this difference makes the benefit of building trust into data sharing processes even clearer: it takes less time and effort to communicate securely.

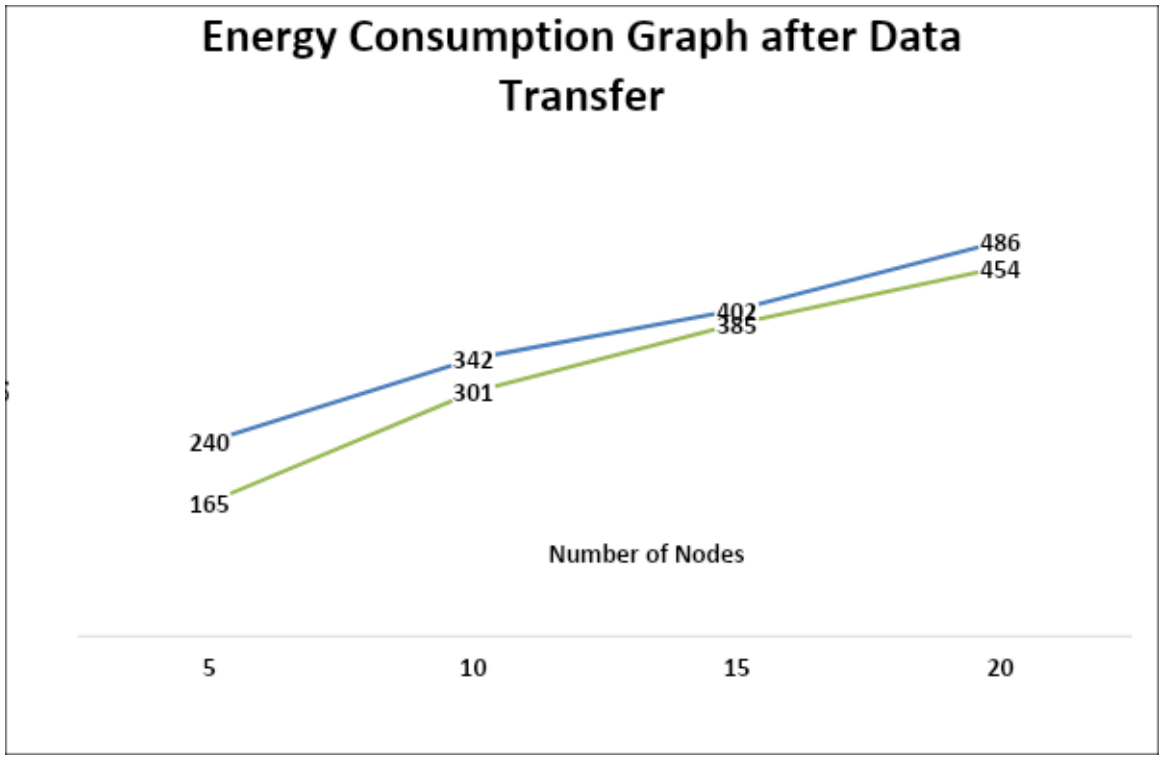

Figure 12: Energy Consumption Graph after Data Transfer WRT Number of Nodes

Trust-based transfers use less energy because they route data more efficiently and send it less often. This is because the trust model probably chooses more stable and energy-efficient ways. Also, the fact that safe methods use less time means that they are less likely to be held up by retransmissions or data quality checks, which happen a lot during normal data exchanges. The data shows that trust-based data sharing methods not only make IoT networks safer, but they also make them much more efficient. Using these techniques can make networks last longer and work better, which makes them a great choice for large-scale deployments where saving time and energy is very important. This study makes it clear why trust-based data sharing methods should be used in many networked systems to make conversation safe and effective, energy consumption shown in figure 12.

5. Conclusion

Using a trust management system that doesn't use a lot of memory on IoT nodes makes transmission much safer in trusted settings. The strength and effectiveness of this method are shown by our research, which includes different mathematical steps and thorough tests. The detailed trust calculation model checks the reliability of nodes on the fly using starting trust values, safe trust measures, and mobile trust models. This multi-level method makes sure that only trustworthy nodes take part in data sharing, which lowers the chance of connection problems. After the data transfer, trust values are adjusted to account for any problems that were found and to keep the network reliable. The lines that show energy usage and energy consumption show that nodes keep working at their best with little energy loss, even after data transfers and dead node discovery. Because energy resources are usually limited, this economy is very important for the long-term health of IoT networks. The trust management method works very well, as shown by the time difference between normal data transfers and trust-based ones. Both in terms of time efficiency and energy use, trust-based data transfers always do better than normal transfers. This is because known, safe lines were chosen, which cut down on delays and retransmissions. Our studies show that nodes that use trust-based exchanges have better performance measures. For example, nodes 1 and 2 showed higher trust values after the data transfer, which means they were more reliable. Nodes with lower trust ratings, like node 5, can be quickly found and controlled, which protects the stability of the network as a whole. Using a trust management system that doesn't take up too much memory for IoT networks not only keeps conversation safe but also makes the best use of resources. This method, which has been proven through in-depth data analysis, is necessary to keep speed and security high in IoT settings that are getting more complicated. More research can improve these models by adding smarter and more adaptable algorithms that will make the system more resistant to new dangers and operating problems.

References

[1] Rana, M.; Mamun, Q.; Islam, R. An S-box Design Using Irreducible Polynomial with Affine Transformation for Lightweight Cipher. In Quality, Reliability, Security and Robustness in Heterogeneous Systems; Springer International Publishing: Cham, Switzerland, 2021; pp. 214–227.

[2] Manikandan, G.; Sakthi, U. A comprehensive survey on various key management schemes in WSN. In Proceedings of the International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), I-SMAC 2018, Palladam, India, 30–31 August 2018; pp. 378–383.

[3] Biswas, A.; Majumdar, A.; Nath, S.; Dutta, A.; Baishnab, K.L. LRBC: A lightweight block cipher design for resource-constrained IoT devices. J. Ambient. Intell. Humaniz. Comput. 2020, 14, 5773–5787.

[4] Kandi, M.A.; Kouicem, D.E.; Doudou, M.; Lakhlef, H.; Bouabdallah, A.; Challal, Y. A decentralised blockchain-based key management protocol for heterogeneous and dynamic IoT devices. Comput. Commun. 2022, 191, 11–25.

[5] Alalwany, E.; Mahgoub, I. Security and Trust Management in the Internet of Vehicles (IoV): Challenges and Machine Learning Solutions. Sensors 2024, 24, 368. https://doi.org/10.3390/s24020368

[6] Cherif, A.; Badhib, A.; Ammar, H.; Alshehri, S.; Kalkatawi, M.; Imine, A. Credit card fraud detection in the era of disruptive technologies: A systematic review. J. King Saud Univ. Comput. Inf. Sci. 2023, 35, 145–174.

[7] Alsharif, B.; Altaher, A.S.; Altaher, A.; Ilyas, M.; Alalwany, E. Deep Learning Technology to Recognize American Sign Language Alphabet. Sensors 2023, 23, 7970.

[8] Karimzadeh, M.; Vakanski, A.; Xian, M.; Zhang, B. Post-Hoc Explainability of BI-RADS Descriptors in a Multi-Task Framework for Breast Cancer Detection and Segmentation. In Proceedings of the 2023 IEEE 33rd International Workshop on Machine Learning for Signal Processing (MLSP), Rome, Italy, 17–20 September 2023; pp. 1–6.

[9] Rolnick, D.; Donti, P.L.; Kaack, L.H.; Kochanski, K.; Lacoste, A.; Sankaran, K.; Ross, A.S.; Milojevic-Dupont, N.; Jaques, N.; Waldman-Brown, A.; et al. Tackling climate change with machine learning. ACM Comput. Surv. (CSUR) 2022, 55, 1–96.

[10] Heinzelman, W.R.; Chandrakasan, A.; Balakrishnan, H. Energy-efficient communication protocol for wireless microsensor networks. In Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, Maui, HI, USA, 7 January 2000; Volume 2, p. 10.

[11] Ajani, S. N. ., Khobragade, P. ., Dhone, M. ., Ganguly, B. ., Shelke, N. ., & Parati, N. . (2023). Advancements in Computing: Emerging Trends in Computational Science with Next-Generation Computing. International Journal of Intelligent Systems and Applications in Engineering, 12(7s), 546–559

[12] Vatsal Gupta, S.K.; Turk, N. MQTT protocol employing IOT based home safety system with ABE encryption. Multimed. Tools Appl. 2021, 80, 19.

[13] Marozzo, F. Infrastructures for High-Performance Computing: Cloud Infrastructures; Elsevier: Amsterdam, Netherlands, 2019; pp. 240–246.

[14] Hussein, S.M.; López Ramos, J.A.; Ashir, A.M. A Secure and Efficient Method to Protect Communications and Energy Consumption in IoT Wireless Sensor Networks. Electronics 2022, 11, 2721. https://doi.org/10.3390/electronics11172721

[15] Ragothaman, K.; Wang, Y.; Rimal, B.; Lawrence, M. Access control for IoT: A survey of existing research, dynamic policies and future directions. Sensors 2023, 23, 1805

[16] Dobraunig, C.; Eichlseder, M.; Mendel, F.; Schläffer, M. Ascon v1.2: Lightweight authenticated encryption and hashing. J. Cryptol. 2021, 34, 33.

[17] Wu, H.; Preneel, B. AEGIS: A fast authenticated encryption algorithm. In Proceedings of the Selected Areas in Cryptography–SAC 2013: 20th International Conference, Burnaby, BC, Canada, 14–16 August 2013; Revised Selected Papers 20; Springer: Berlin/Heidelberg, Germany, 2014; pp. 185–201.

[18] Alasmary, H.; Tanveer, M. ESCI-AKA: Enabling Secure Communication in an IoT-Enabled Smart Home Environment Using Authenticated Key Agreement Framework. Mathematics 2023, 11, 3450. https://doi.org/10.3390/math11163450

[19] Tanveer, M.; Bhutta, M.N.M.; Alzahrani, B.A.; Albeshri, A.; Alsubhi, K.; Chaudhry, S.A. CMAP-IoT: Chaotic Map-Based Authentication Protocol for Crowdsourcing Internet of Things. Arab. J. Sci. Eng. 2023, 1–14.

[20] Chandu Vaidya, Prashant Khobragade and Ashish Golghate, "Data Leakage Detection and Security in Cloud Computing", GRD JournalsGlobal Research Development Journal for Engineering, vol. 1, no. 12, November 2016.

[21] Yuanbing, W.; Wanrong, L.; Bin, L. An Improved Authentication Protocol for Smart Healthcare System Using Wireless Medical Sensor Network. IEEE Access 2021, 9, 105101–105117.

[22] Challa, S.; Wazid, M.; Das, A.K.; Kumar, N.; Reddy, A.G.; Yoon, E.J.; Yoo, K.Y. Secure signature-based authenticated key establishment scheme for future IoT applications. IEEE Access 2017, 5, 3028–3043.

[23] Choi, Y.; Lee, D.; Kim, J.; Jung, J.; Nam, J.; Won, D. Security enhanced user authentication protocol for wireless sensor networks using elliptic curves cryptography. Sensors 2014, 14, 10081–10106.

[24] Alghofaili, Y.; Rassam, M.A. A Trust Management Model for IoT Devices and Services Based on the Multi-Criteria Decision-Making Approach and Deep Long Short-Term Memory Technique. Sensors 2022, 22, 634. https://doi.org/10.3390/s22020634