International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V71I5P229

Volume 71 Issue 5, 272-281, May 2023

Original Article

A Novel Algorithm of Safe-Route Traversal of Data for Designing the Secured Smart City Infrastructures

Arpit Chhabra[1], Niraj Singhal[2], Syed vilayat Ali Rizvi[3]

[1]*Shobhit Institute of Engineering and Technology (Deemed to be University), Meerut, India and, Sir Chottu Ram Institute of Engineering & Technology, Chaudhary Charan Singh University, Meerut, India.*
[2,3]*SCRIET, Chaudhary Charan Singh University, Meerut, India*

[1]*Corresponding Author : arpit.0121@gmail.com*

Received: 02 March 2023 Revised: 19 April 2023 Accepted: 28 April 2023 Published: 25 May 2023

Abstract *- Here, a special feature-rich safe route data traversal algorithm research model has been developed for use in data transfer preparation tasks. The characteristics include the format for three-dimensional data (memory is only one dimension). However, the method expertly transforms it into a 3-D entity (cube string construction), which is essential for establishing a novel data traversal model that deviates from conventional wisdom and has multiple DRDs. In this paper, a method for decrypting, encrypting data that calls for multiple DRDs. The DRDs include the following: a string containing cube data is called a cube string. Delete List (a list that contains cube string moves), Character Position List (List of all character's initial positions in the cube string, including the plain string) because it requires multiple dependencies to crack or decode the data, these dependencies make it extremely difficult for hackers to predict the nature of data traversal.*

Keywords *- Security, Learning algorithm, Rubik cube, Multiple Data-Retrieval Dependencies (DRDs).*

1. Introduction

Smart cities are complicated systems. Every member node of a smart city (i.e., smart homes, smart offices, smart factories, etc.) is interconnected and shares the same lineage towards data sharing and data manifestation[16]. Smart cities also generate a bulk amount of data during their operations. This bulk data is not only vital but important to make a complex peer-dependent infrastructure like a smart city function normally. The above can be achieved by implementing mechanisms like "Systems Reliability Concepts" and "Secure Data Transmission Mechanisms" like the Rubiks Cube algorithm. However, being good but these mechanisms required to be implemented and share strong advantages, and they cannot be implemented in a 3-rd level or 5-th level smart cities, which are architecturally complicated and rely heavily upon data redundancy and data integrity. In this research paper, a format of data restructuring is presented that promises optimum security and reliability. Now, it will proceed with the application part, in which it will demonstrate the actual use case and impact of the aforementioned technology.

2. Literature Review

Today, cyber-attacks occur on a daily basis. To stay aware of the rising recurrence of attacks and breaks as well as the way that human work hours can not grow endlessly, the digital protection area needs forefront and adaptable innovation and ways to deal with robotize network safety guarded exercises. Conventional methods are unable to protect complex networks with a large number of users and endpoints with multiple identities[3] because of the variety, complexity, and effectiveness of zero-day attacks. Committed urban communities improve urban areas' viability, efficiency, and livability. A smart city can use and monitor its basic infrastructure, including roads, tunnels, and other infrastructure. Control support exercises can aid in the modernization of the assets while also monitoring for security issues. To get past any text-based hindrances to creating and utilising brilliant metropolitan regions, a strategy has been intended for finding out about them. Based on a comprehensive study of writing from various fields[4].

The information-driven society is being undermined by digital attacks on even the most advanced frameworks and framework models. Because the more sophisticated and unexpectedly the most complex advancements are common to contemporary criminals and are the same on a daily basis, there is always the possibility of swindlers indicating over the escape clauses in such encryption and security calculations. A more modern, 2-D encryption style that is difficult to track is also difficult to design because each encryption strategy for a similar encryption method is unique [5].

Digital attacks on even the most advanced frameworks and framework models threaten the information-driven society. Because the more sophisticated and suddenly the most complicated progressions are common to current criminals and typically the same, there are typical risks of criminals demonstrating over-the-getaway conditions in such encryption/security estimations. Cybercriminals use hash tables and clever rainbow tables to review and decipher the objective data set's associated jumbled data. Because each encryption strategy for a similar encryption method is unique, a more modern, 2-D encryption style that is difficult to track is also difficult to design [6].

It should not be forgotten that, despite their intricate design, Smart City environments can be connected, and data can be transferred between their components thanks to communication technology. The abilities of remote correspondences empower innovations for savvy urban communities, working with their quick and viable reception and development. The appearance of multiple technologies is the apparent trend in that field[7].

A brilliant city is one in which existing offices and administrations are upgraded through virtual innovation to the advantage of people and organizations. The most important infrastructures in this town are connected. Through increased record changes, municipal domain names aim to control Critical property, automate city governance and optimizing Dynamic provided services. Modeling those record flows, on the other hand, is not governed by any established principle or standard[8].

The Internet of Things (IoT) relies heavily on communication technology for data transmission. The Internet of Things (IoT) heavily depends on communication technology to move data around [2].

People are already connected to one another via smartphones and other devices. Smart energy meters, security systems, and home appliances are being used in many cities. The overall connectivity that is referred to as the "net of factors" is actually on its way to social structures like homes, automobiles, public areas, and others. Standards are evolving for all of these likely related structures. They will result in upgrades unlike any other within the first class of life. New interconnected structures for observing, control, and computerization are being added to city foundations and administrations to exploit them.

A web of connected records, including visitor updates, GPS location, and weather, will be accessible to smart public and private transportation. Useful resources will be beneficial to public safety, emergency responders, and integrated systems in disaster recovery. It notices two basic and convoluted difficulties: security and privacy Examples of protection include physical disruptions to service availability and illegal access to data. As digital residents become increasingly equipped with information about their region and sports, privacy appears to disappear[9]. Integrative regulations are required to strengthen connections and improve people's lives in cities and rural areas based on existing monetary, social, and environmental ties[10].

Another includes pace monitoring devices to encourage safer driving in residential areas. However, poor policymaking may also result from improper use of this sophisticated technology. Planners, integration partners, and businesses managing these new groups face significant political, technological, and socioeconomic challenges due to the intricate and interconnected nature of smart towns[11].

In order to successfully and safely plan safe travel, smart self-sustaining vehicles (AVs) are networks of cyber-physical systems (CPS) that wirelessly communicate with other CPS substructures (such as intelligent vehicles and devices). Due to their unreliable Wi-Fi communication, these vehicles are easy prey for malware attacks that can compromise their autonomy, increase inter-vehicle communication latency, and drain their power. Passengers' safety may be put at risk as a result of these compromises, and there may be financial loss. Therefore, real-time detection of such assaults is necessary for secure smart transportation and delivery systems (ITS)[12].

Malware that targets IoT devices has recently increased in frequency. They prompt utilizing a productive assault discovery and Live Malware Examination Framework (Chartbook), which gives exceptional data on the dangers of IoT. The parts of the Map book incorporate a mixture of IoT, honey-pot engineering, assault attribution, malware downloader, and a constant malware examination framework. ATLAS has identified 859 distinct malware files that target 17 actual IoT devices since its launch. 65 percent of these samples were first observed by the infrastructure or are currently unknown to Virus Total, compared to timestamps from Virus Total[1].

The utilization of tremendous amounts of organization traffic measurements by gadgets getting to realize methods can bring about security issues for residents, in spite of the way that attack recognition is fundamental for perceiving legitimate shrewd urban areas. They propose a federated mastering strategy that incorporates the hazard and manufacturer utilization Description files into the context of the net of factors-enabled smart towns as a means of resolving this issue[14].

3. Problem Identification

3.1. Packet Loss and Latency Issues

From the initial start of data packet propagation at the start node all the way to the destination, packet loss refers to data loss. To put it succinctly, packet loss occurs when only a portion of the total data payload reaches the target node.

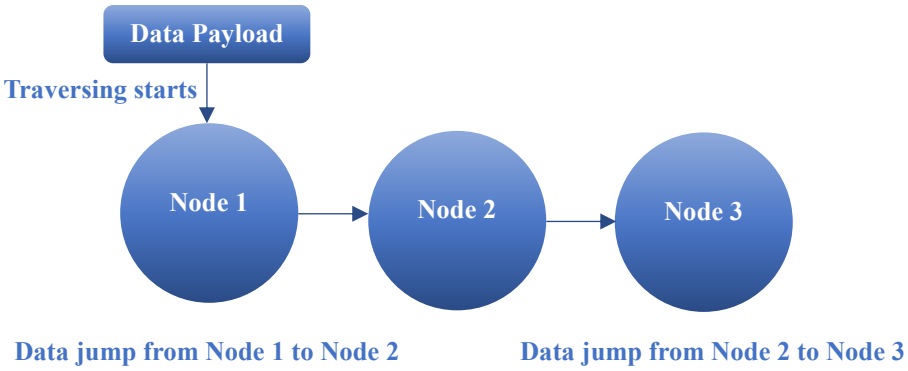

Fig. 1 (a) The schematic diagram of data flow in a normal network

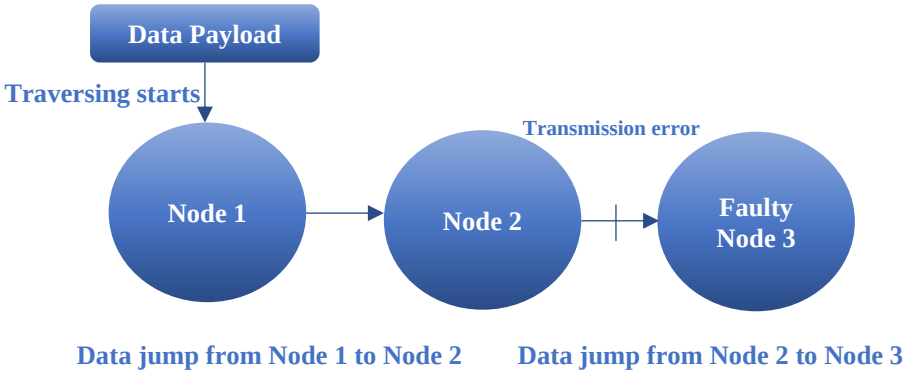

Fig. 1 (b) The schematic diagram of blocker encounter in data flow in a normal network

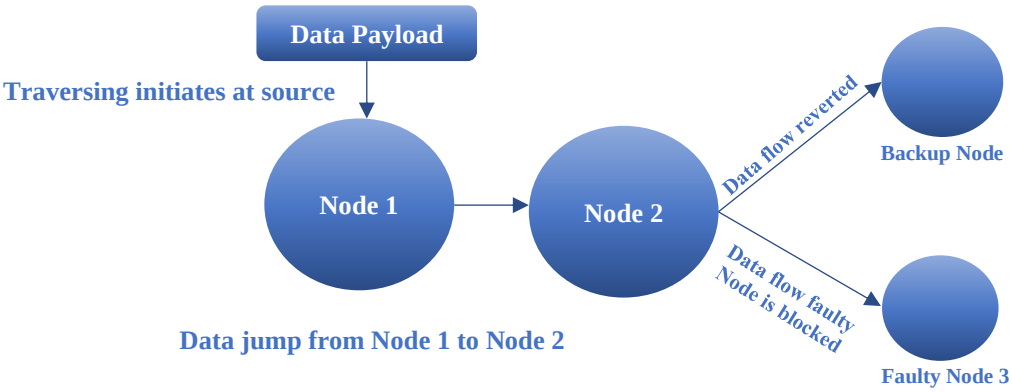

Fig. 1 (c) The schematic diagram of blocker resolution in data flow in a normal network

Infrastructures in smart cities are very busy, with a lot of data payload moving from one endpoint to another. This indicates that smart city networks experience congestion on a regular basis, which is uncommon for most other infrastructures. The entire smart city data flow is impacted when a network endpoint or node fails. Transmission may occur with invalid (incomplete) data or with no data during packet loss. Preventing data flow between the affected node and the node that came before it.

Assuming that there is N number of Nodes in a network for a particular path of data flow, and it takes T seconds for a Node to parse, process and redirect data packets to the next node, then the performance of blocker resolution can be estimated to be *NT seconds in the *worst case scenario*.

Networks are impacted not only by packet loss in terms of performance but also by other important factors like availability and security.

Data must typically be retransmitted from the source following a network failure to reach its final destination, which may take a significant amount of time. The data flow in a working network and an affected network is depicted schematically in Figure 1.0, with the same data latency of approximately *3T milliseconds (assuming that T

milliseconds is the average time it takes for a data packet to move between two nodes).

3.2. Infrastructural Failures

Because smart cities are known to accommodate a variety of user interfaces, communication networks, and communication channels, their infrastructure dependence increases exponentially with network size or scale. The problem with a network failure occurs when data-deadlock situations caused by a network failure prevent the sub-networks, protocols, or data reservoirs that make up that part of the network from retrieving data. In this sense, a minor error in the network could result in an infrastructure failure.

Example - There exists a network that comprises 2 sub-networks; let's name them Network A and Network B. Let's assume that both networks - A and B are inter-coupled, meaning both depend closely on one another for data. If Network A fails, Network B will face data deadlock, infrastructure collapses, and vice-versa.

3.3. Data Security

As previously stated, succinct errors or a network failure may tend to expose the network to some network-related vulnerabilities, particularly identity theft and data breaches, when they accumulate into an infrastructure failure. False or inaccurate data can be easily introduced with little network trace and high effectiveness because some parts of the network may be experiencing data deadlock.

4. Approach Applied for Data Traversal Procedure

Data traversal in a network via the proposed mechanism is carried out through a spontaneous set of operations occurring on every node. Smart cities are an amalgamation of many communication technologies ranging from micro-sensors with a range of about 10 cm to WANs with a range of as much as 1 kms. So for fair throughput, we will consider only sub-medium to medium-level computation solutions inside the smart cities.

There are sub-processes required to implement the entire mechanism:

4.1. Data Packaging

Initially, data is modeled into a cube data structure. Data packing refers to modularizing data in a compact format, making its traversal over *a given network feasible.*

4.2. Data Batching

Data batching refers to making data traversal ready. In this phase, all lost information related to the encryption will be locked and safeguarded with an existing encryption algorithm for transmission in the *network.*

4.3. Key Duplication

Key duplication is the process of copying and forwarding the key along with the network data payload.

4.4. Data Transfer

Data transfer is the final process in which the data payload is compressed and encrypted following any general encryption algorithm, e.g., RSA, AES, DES, etc.

5. Proposed Algorithmic Process

The proposed algorithmic mechanism is designed to safely transport data from one endpoint to another in a network while safeguarding the data's integrity and confidentiality by leveraging a rubik's cube data re-modeling technique.

5.1. Initiation Process

In the initiation process, basically modeling data into a cube-like data structure and performing the mundane process of mixing and transforming the credentials into a secured format so that the result is a much more secure and pro-traversal medium. The initiation process is sub-categorized into three phases, namely:

5.1.1. Data Re-Modeling

Data re-modeling is the step in which data strings are first fed into the cube data structure, which is dynamically allocated and controlled via a control engine.

Note: A control engine or a memory engine is a low-level memory management and memory-allocation program that makes and manipulates cubes for the infrastructure.

5.1.2. Meta-Data Structuring

The control engine will mix the cube data structure, resulting in the metadata cipher cube string, move-list array, and character position array. Restructuring is required for these meta-data, which are received as objects or discrete memory structures. Because the move list's length can exceed 100 elements within the dynamically allocated array, this meta-data is first compressed.

5.2. Data Preprocessing

Encrypted meta-data can be extremely cumbersome to transport over a network. Therefore, the cube string is compressed using various compression methods to reduce complications (I anticipate that gzip would be an excellent starting point for prototyping this technology); After that, a symmetrical encryption algorithm, such as the AES algorithm, will be used to encrypt the data. After all of the necessary meta-data has been compressed and encrypted, it is brought together to be combined and encrypted once more to create something similar to a Merklehash but with an encryption algorithm. Figure 2.0 depicts the entire data flow.

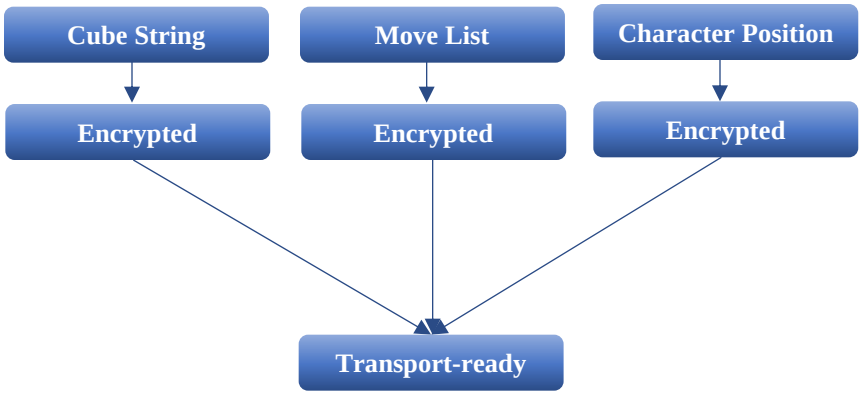

Fig. 2 The schematic diagram of data flow in data preprocessing

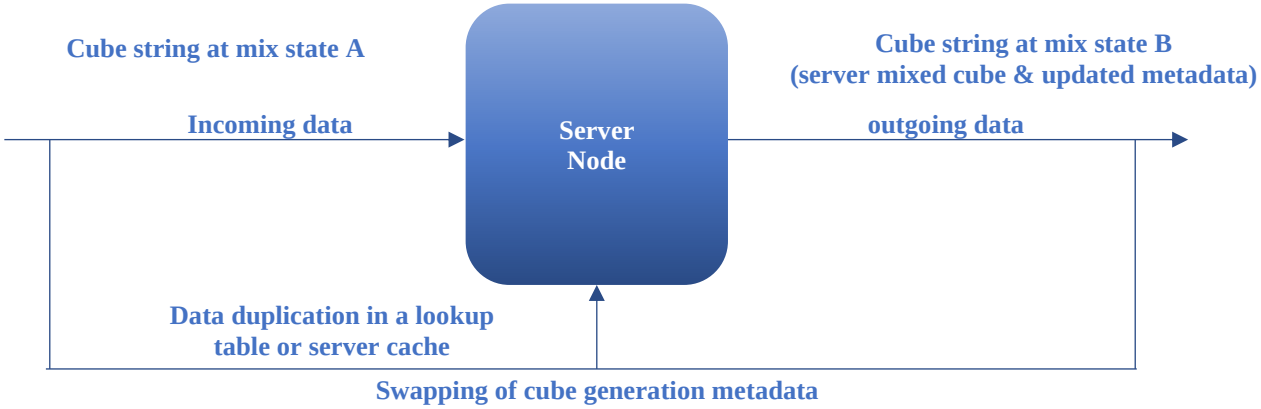

Fig. 3 The schematic diagram of data exchange in a proposed algorithm

5.3. Transmission Process

A mechanism will forward data to the network's target node during the transmission process. Compressed and encrypted data is first transferred during the transmission process to a transmission service or mechanism (for example, REST APIs, TCP servers, and Web-Socket interfaces are plausible options; web socket-based solutions will be regarded as the best for real-time data transfer). After that, upon reaching the node, the data will undergo a key exchange, undergo any necessary modifications, archive the data, and continue their network propagation.

This whole process can be broken into two sub-processes that constitutes this process, namely:

5.4. Data Batching

In the previous set of processes, after the data has been encrypted and compressed by the aforementioned mechanism, it will be batch processed for further transfer into the network using a networking resource or medium for that use case. A web server with appropriate endpoints or an open web socket connection is preferred. Figure 3.0 depicts how data is formatted, loaded as a data payload, and sent to the next network location.

5.5. Key Exchange and Modification

Inside the batching data configuration, if it is specified that the next node is subjected to change or modify the mixing pattern of the cube. The aforementioned case will be handled as the following set of synchronized mechanisms:

5.1.1. Data is decrypted, and its components are stored in respective memory buffers. Then the data's cube string is formed using the memory engine used in the very first step in the process.

5.1.2. Then data is mixed by arbitrarily chosen moves by the engine.

5.1.3. The moves are recorded and pushed onto the move list.

5.1.4. Metadata structuring and data preprocessing occur again.

5.1.5. Keys are duplicated, one is saved in the node archive, and one is set onto the next endpoint for the next node in the network.

Algorithm

```
LET RECEIVED_PACKAGE -> RESPONSE.PACKAGE;
  LET RECEIVED_KEY ->
  RESPONSE.RECEIVED_PACKAGE.KEY;
// running decryption functions (for symmetric ones)
//  not decrypting the cube but the metadata
```

```
LET DATA ->
RECEIVED_PACKAGE.REVERSE_PREPROCESS();

// data is an object, so it needs de-structuring
LET MOVE_LIST, CUBE_STRING = DATA.MOVE_LIST.
CUBE_STRING;
/*
• Application modifications and key exchange.
• Suppose there is a function named MIX over the cube
  object, which takes the cube_string and returns a modified
  instance of the mixed cube with applied modifications
  and a modified move_list.
*/
LET CUBE = NEW CUBE(CUBE_STING, MOVE_LIST);
LET MOD_MOVE_LIST, MOD CUBE_STRING =
CUBE.MIX();
/*
• Now considering that one endpoint of an API or web
  socket is open so that it can transfer data onto the next
  node.
• New metadata is encrypted again using a proposed
  algorithm and forwarded to the node using an arbitrary
  server endpoint.
*/
LET NEW_CUBE -> NEW CUBE(MOD_CUBE_STRING,
MOD_MOVE_LIST);
LET META_DATA -> NEW_CUBE.ENCRYPT();
LET PAYLOAD -> META_DATA.PREPROCESS();
/*
• The payload format must contain a key.
• Key encrypted product is the amalgamation of metadata
  such as cube_move and character position only.
*/
LET NEW_KEY -> PAYLOAD.KEY;

        /*
• Archiving key in the server
• Assuming there is a local state database for server logs.
        */
ARCHIVE(NEW_KEY, CUBE_STRING);

// forwarding data to the next)address server address

POST 'https:<next_address>/endpoint' -payload PAYLOAD
```

6. Analysis of Proposed Algorithm

The algorithm's behavior during various encryption and data transmission stages is intriguing. Whereas the majority of systems are either completely stateless or stateful. The proposed calculation shows both - state-full and stateless ways of behaving at stages where they are ideal to be shown.

Processes carried on over a constant state in the memory are known as state-full processes. To put it another way, processes that are state-full indicate that there is a memory of the past. The current transaction may be affected by previous ones because they are remembered. A state-full environment is ideal for use cases involving caching or where the requirement of previous data for pattern matching or verifying the exclusivity of current data necessitates state.

Specifically use case is worried about reserving the shape string, so in the event of a course or crossing hindrance, the beginning of the sign does not need to start the time-exorbitant retry system to divert the information bundles to the objective hub in the organization through a work-around way. In addition, the proposed system requires part-state-fullness due to the need for accurate Data Pattern Matching (DPM) to determine whether they lost data during traversal or if a third party intervened in the network. For instance, - utilizing example matching is a customary means of tracking down blunders in information parcels on a TCP/IP network by utilizing the equality bit technique. The proposed system is significantly more effective at initiating recovery mechanisms thanks to state-fullness.

During its time in a server node, the cube string enters a state-full environment and is serialized and pattern-matched to ensure its authenticity and integrity. It is state-full because the cube string exists in an environment that has access to a persistent memory archive. At the same time, the cube string is mapped to persistent memory before being mixed up once more for the purpose of being transferred to the subsequent server node in the network.

The proposed framework, state-full assignments, can be done utilizing indexers while managing huge organizations. Because data pattern matching is simple and efficient, state-fullness makes the proposed system more efficient and threat-evident. Integrations like process containerization, visualizations, and even the creation of distributive network protocols are all made possible by the proposed system's state-fullness.

Then again, Stateless cycles are processes that work without needing any tenacious information in the memory. The cube string exhibits state-fullness in the server node environment of the proposed system but not throughout its entire length. When a cube string reaches a server node, it goes through a number of steps before being sent again across the network.

6.1. Metadata Preprocessing (Stateless)

Metadata preprocessing refers to the process of converting compressed data packets into workable data structures and data types. This is obviously why it is stateless, because it does not require any persistent memory or cache records, and there are no past transactions of how initial data was formed using original plain data and original set of metadata.

Table 1. Analysis of Various Networking Protocols

S.No	The proposed system	Network Time Protocol
1	Uses a combined restorable, tamper-evident encryption method with a wide variety of security options.	Have no security options; although it includes security provisions, those are rarely used. It uses an MD5 algorithm which is popular as a weak *encryption method*.
2	Authentication can be implemented at two distinct levels, node-level or adding token (signer) meta-data along with cube string. Hence, improved security and better availability with zero complexity overhead.	Authentication is carried out by AutoKey, which is an asymmetric authentication mechanism. AutoKey is problematic as well as an insecure protocol.
3	It performs integrity checks by comparing cube states with metadata. If an anomaly is encountered, it discards the current data payload and requests the last server node to resend the data packet with other mix patterns.	It relies on stateless User Datagram Protocol **(UDP)**. Statelessness allows an attacker to send spoofed NTP responses to an NTP client. Because there is no check over potential changes in the data payload. Hence, it is not tamper-evident nor includes
4	Dos attack is impossible as the server node workers can easily identify cube string data packets' unique package data format. If not matched, it rejects its processing, hence preventing loss of computation power and resources.	By advancing the system time on a validating resolver, it is, therefore, possible to execute a DoS attack or to flush the cache by causing the TTLs of the records to expire prematurely. Conversely, putting a machine's clock back would allow for a replay attack.

6.2. Cube String Regeneration (Stateless)

Cube string regeneration is a pretty straightforward task. Cube string reaches the server node; it gets processed into data structures and types which are supported by the server computer and are easy to be worked with. This may seem like a process which may require a state in order to perform its task, but it does not. It is because cube generation is done via pre-defined and constant mandates governed across a network.

6.3. Mixing and Regeneration of New Metadata (State-Full)

It is the only state-full process in the mechanism, and it is important for it to be. Metadata preprocessing refers to the process of converting compressed data packets into workable data structures and data types. This is obviously why it is stateless, because it does not require any persistent memory or cache records, and there are no past transactions of how initial data was formed using original plain data and original set of metadata.

6.4. Preparing Data into Compact Data Packets (Stateless)

In the proposed mechanism for data security on distributed network systems, we combine meta-data to the cube string and then serialize the obtained data into a compact, secure and transmission-ready format. This process is also stateless as the protocol for handling the serialization is taken care of by the protocol over which transmission has to be done.

6.5. Transmission (Stateless)

Prepared data packets and transmitted them into the network.

7. Solutions to the Problem Associated with

7.1. Solution to Packet Loss and Latency Issues

The proposed method caches and duplicates data at the node level; each time data moves from one node to another, a copy of the cube string is saved, reinitialized in a memory buffer, and cached for use in the future.

Therefore, even though the most recent copy of the data is cached in the preceding node in the event of a network fault, the propagating data may be lost. After that, the cache cube string is made into a cube instance inside the node server, thoroughly mixed as an additional security measure, batched, and sent to a new gateway or a new node. Figure 4.0 depicts the operation of the same as a schematic diagram.

In such a case, there will be significantly lesser latency, network downtime and increased security after a network flow fluctuation.

Taking into consideration the previous example, and as can be seen in Figure 4.0, in the event that certain nodes or endpoints in a network experience failures, the data will proceed normally, copying the most recent cube string instance, creating a new cube instance, saving it in a cache, and then batching the data to move on to the next node. Data, on the other hand, is lost at the broken network node. The most recent node becomes the source, as opposed to data packets repeating from the beginning from the source.

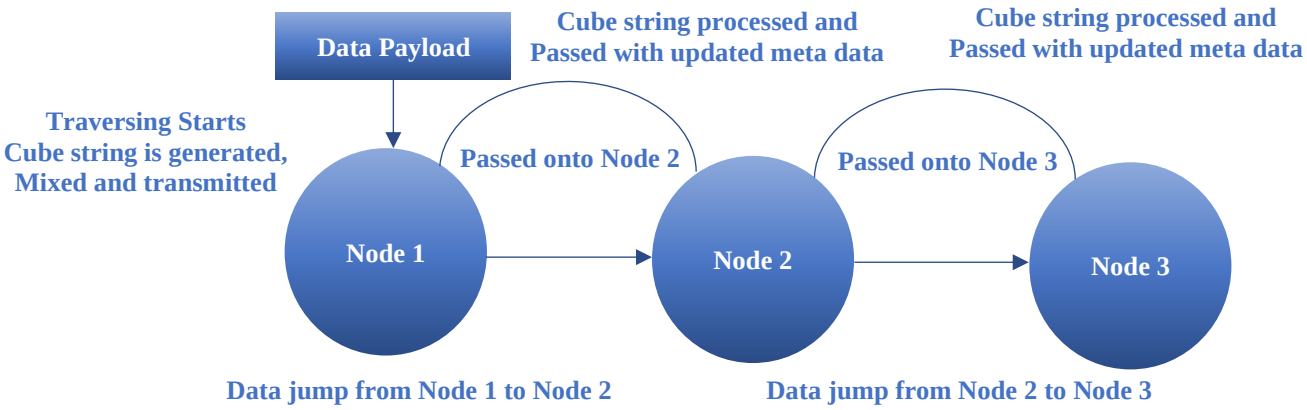

Fig. 4 (a) The schematic diagram of normal data flow in a network implementing the proposed mechanism

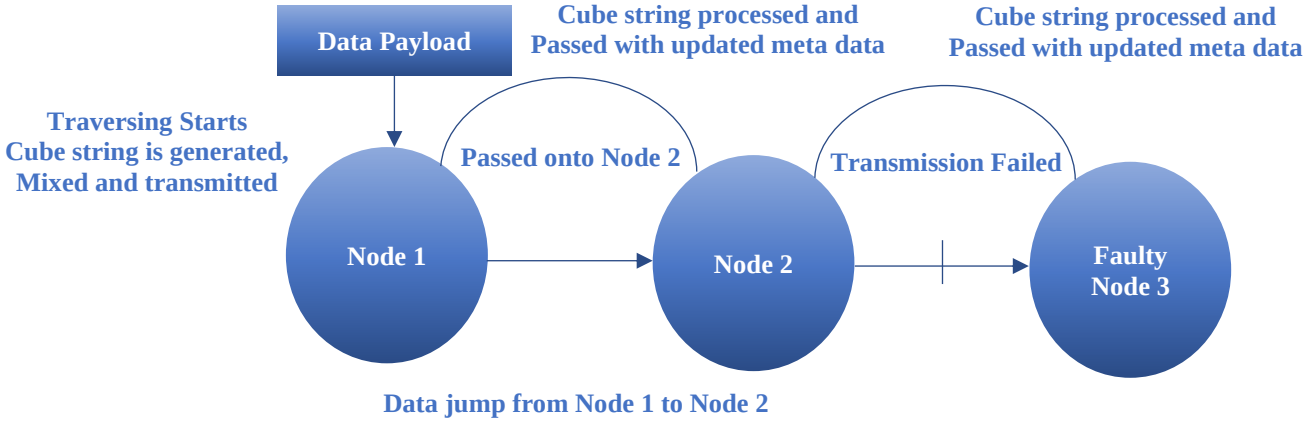

Fig. 4 (b) The schematic diagram of blocker encounter data flow in a network implementing the proposed mechanism

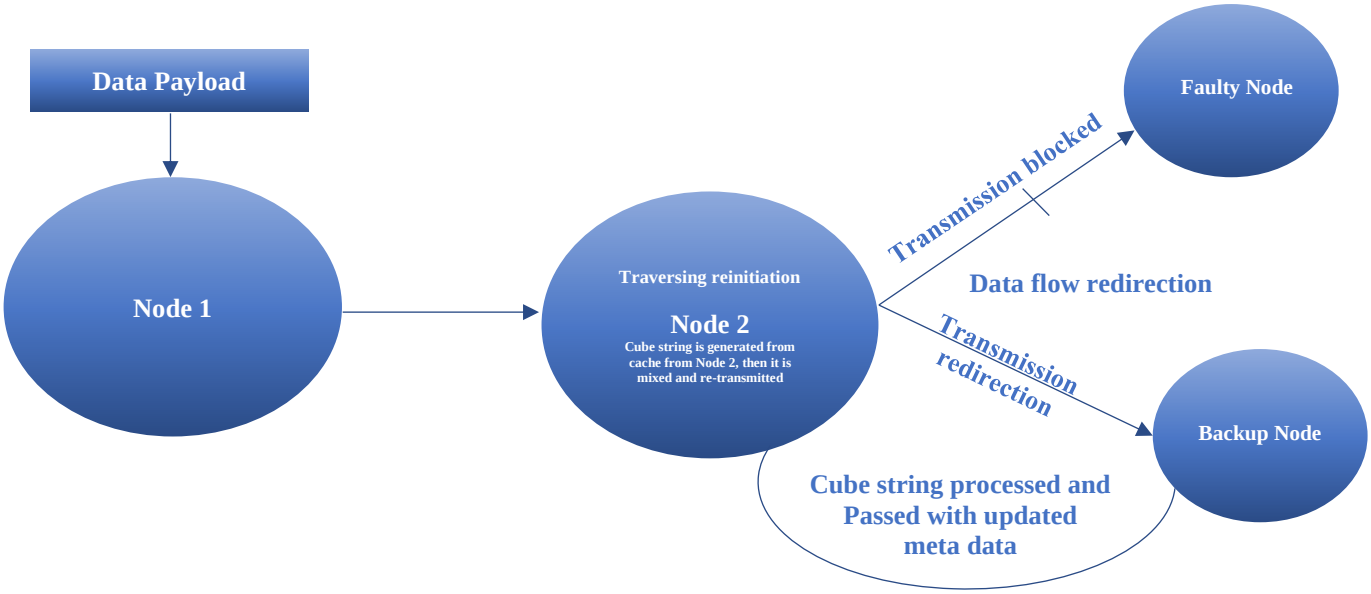

Fig. 4 (c) The schematic diagram of blocker resolution data flow in a network implementing the proposed mechanism

In this example, the latency is reduced from *3T milliseconds to just *1T milliseconds (assumption - T milliseconds is the average time it takes for a data packet to jump between any two nodes), and it mixes the cube for additional security and with updated meta-data before sending it to the new target node.

7.2. Prevention of Infrastructural Failures

Smart cities are known to accommodate multiple user interfaces, communication networks, and communication channels; their infrastructure dependence increases exponentially with network size or scale.

The problem with a network failure occurs when data-deadlock situations caused by a network failure prevent the sub-networks, protocols, or data reservoirs that make up that part of the network from retrieving data. In this sense, a minor error in the network could result in an infrastructure failure.

7.3. Increased Data Security

The proposed mechanism follows a significant rule of changing the cube state or data's metadata regarding encryption whenever a system experiences a network or security fault. The proposed mechanism provides more security than other traditional data caching or secure traversal algorithms. In addition, the constant caching, modification, and batching cycle (CMB cycle), which limits the total combination of character position field data to approximately 43 quintillions for a 54-character string, makes the nature of the data packet's metadata unpredictable.

8. Conclusion & Future Scope

The proposed approach can be used in any network application (or solution) as a middleware service, in large-scale enterprise-grade application solutions, in second layer block-chain security solutions, and so on. With low latency, high availability, and data security, the mechanism guarantees optimal security and throughput for any network requirements.

The proposed approach is a new, all-encompassing standard for managing and protecting networks at the node level rather than the network. This algorithm and mechanism become data and user-centric as a result of this shift from a macro to a micro-level perspective. This indicates that by prioritizing data management and security planning, the proposed approach can accommodate any use case with high performance and the advantage of meeting any specific networking requirements flawlessly.

References

[1] Yan Lin Aung, Martín Ochoa, and Jianying Zhou, **"**ATLAS: A Practical Attack Detection and Live Malware Analysis System for IoT Threat Intelligence," *International Conference on Information Security*, vol. 13640, pp. 319-338, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[2] Niraj Singhal, and Arpit Chhabra, "A Novel Learning Approach of Adaptive Cyber Defense System for Smart Cities," *Electronic Systems and Intelligent Computing*, vol. 860, pp. 465-471, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[3] Jinjin Liang, and Yong Nie, "Application of Machine Learning Algorithms in Analysis of Learners' Behaviour Data," *SSRG International Journal of Computer Science and Engineering*, vol. 6, no. 10, pp. 13-17, 2019. [CrossRef] [Publisher Link]

[4] Hannu Turtiainen, Andrei Costin, and Timo Hämäläinen, "Defensive Machine Learning Methods and the Cyber Defence Chain," *Artificial Intelligence and Cybersecurity*, pp. 147-163, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[5] Deepak A. Vidhate, and Parag Kulkarni, "Single Agent Learning Algorithms for Decision making in Diagnostic Applications," *SSRG International Journal of Computer Science and Engineering*, vol. 3, no. 5, pp. 46-52, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] ZebaNaaz, Kauser Fatima, and C.Atheeq, "Performance Based Comparison Study of RSA and Chaotic Maps in MANET," *SSRG International Journal of Electrical and Electronics Engineering*, vol. 4, no. 2, pp. 17-22, 2017. [CrossRef] [Publisher Link]

[7] Manav Bansal, Arpit Chhabra, and Niraj Singhal, "Smart City-Shrewd Vehicle Versatility Utilizing IOT," *International Journal of Engineering Trends and Technology*, vol. 70, no. 3, pp. 29-36, 2022. [CrossRef] [Publisher Link]

[8] Arpit Chhabra et al., "A New Cryptographic Algorithm for Safe Route Transversal of Data in Smart Cities using Rubik Cube," *International Journal of Computer Science and Network Security*, vol. 22, no. 8, pp. 113–122, 2022. [CrossRef] [Publisher Link]

[9] Shakti Chourasiya, and Suvrat Jain, "A Study Review on Supervised Machine Learning Algorithms," *SSRG International Journal of Computer Science and Engineering*, vol. 6, no. 8, pp. 16-20, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[10] D. Rajavel, and S. P. Shantharajah, "Cryptography Based on Combination of Hybridization and Cube's Rotation," *International Journal of Computational Intelligence and Informatics*, vol. 1, no. 4, pp. 294-299, 2012. [Google Scholar] [Publisher Link]

[11] S.Jagadeesan et al., *"*High Level Secure Messages Based on Steganography and Cryptography,*"* *International Journal of Engineering Trends and Technology*, vol. 68, no. 2, pp. 142-145, 2020. [CrossRef] [Publisher Link]

[12] Dalibor Dobrilović, "Networking Technologies for Smart Cities: An Overview," *Interdisciplinary Description of Complex Systems: INDECS*, vol. 16, no. 3-A, pp. 408-416, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[13] Vasiliki Demertzi, Stavros Demertzis, and Konstantinos Demertzis, "An Overview of Cyber Threats, Attacks, and Countermeasures on the Primary Domains of Smart Cities," *Applied Sciences*, vol. 13, no. 2, pp. 790, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[14] K. Jino Abisha et al., "Detection of Twitter Spam's using Machine Learning Algorithm," *SSRG International Journal of Computer Science and Engineering*, vol. 6, no. 3, pp. 10-13, 2019. [CrossRef] [Publisher Link]

[15] Adel S. Elmaghraby, and Michael M. Losavio, "Cyber security challenges in Smart Cities: Safety, security and privacy," *Journal of Advanced Research*, vol. 5, no. 4, pp. 491-497, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[16] J. Jhanavi, and M.Dakshayini, "Blockchain Implementation for Storage," *SSRG International Journal of Mobile Computing and Application*, vol. 5, no. 2, pp. 9-12, 2018. [CrossRef] [Publisher Link]

[17] Carmen Rotună et al., "Smart City Ecosystem Using Blockchain Technology," *Informatica Economica*, vol. 23, no. 4, pp. 41-50, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[18] Manvi Chahar, and Savita, "Implementation and Classification of Anomalous Detection with Varying Parameters," *SSRG International Journal of Computer Science and Engineering*, vol. 6, no. 4, pp. 16-18, 2019. [CrossRef] [Publisher Link]

[19] Elvira Ismagilova et al., "Security, Privacy and Risks within Smart Cities: Literature Review and Development of a Smart City Interaction Framework," *Information Systems Frontiers*, vol. 24, no. 2, pp. 393-414, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[20] Nikhat Naaz Aslam Shaikh, and Vaishali Bagade, "Performance Evaluation and Detection of Grey, Warm, Flooding, Misrouting & Modification of Attacks in Vanet," *SSRG International Journal of Electronics and Communication Engineering*, vol. 8, no. 4, pp. 10-17, 2021. [CrossRef] [Publisher Link]

[21] Sana Aurangzeb et al., "CyberSecurity for Autonomous Vehicles Against Malware Attacks in Smart-Cities," *Research Square*, 2022. [Google Scholar] [Publisher Link]

[22] Mahdi Khosravy et al., "Social IoT Approach to Cyber Defense of a Deep-Learning-Based Recognition System in Front of Media Clones Generated by Model Inversion Attack," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 5, pp. 2694-2704, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[23] Sara N. Matheu et al., "Federated Cyberattack Detection for Internet of Things-Enabled Smart Cities," *Computer*, vol. 55, no. 12, pp. 65-73, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[24] Arpit Chhabra, and Niraj Singhal, "Indicator Based Cyber Threats Detection for Data of Smart Cities Using Bio-Inspired Artificial Algae," *International Journal of Advanced Research in Engineering and Technology (IJARET)*, vol. 11, no. 11, pp. 1530-1536, 2020. [CrossRef] [Google Scholar] [Publisher Link]