A New Cryptographic Algorithm for Safe Route Transversal of Data in Smart Cities using Rubik Cube

Arpit Chhabra[1†], Niraj Singhal[2††], Manav Bansal[3†††] and Syed Vilayat Rizvi[4††††]
arpit.0121@gmail.com drnirajsinghal@gmail.com
Shobhit Institute of Engineering and Technology (Deemed University), Meerut, India, SCRIET, Chaudhary Charan Singh University, Meerut, India.

Abstract
At the point when it is check out ourselves, it might track down various information in each turn or part of our lives. Truth be told, information is the new main thrust of our advanced civilization and in this every day, "information-driven" world, security is the significant angle to consider to guarantee dependability and accessibility of our organization frameworks. This paper includes a new cryptographic algorithm for safe route traversal for data of smart cities which is a contemporary, non-hash, non-straight, 3D encryption execution intended for having information securely scrambled in the interim having a subsequent theoretical layer of safety over it. Encryption generally takes an information string and creates encryption keys, which is the way to unscramble as well. In the interim in another strategy, on the off chance that one can sort out the encryption key, there are opportunities to unravel the information scrambled inside the information string. Be that as it may, in this encryption framework, the work over an encryption key (which is created naturally, henceforth no pre-assurance or uncertainty) just as the calculation produces a "state" in a way where characters are directed into the Rubik block design to disregard the information organization.

Keywords:
Smart cities, Cyber threats, Trusted Partner module (TPM), Advanced persistent threats (APT's), Data transmission.

1. Introduction

In the present day, are experienced digital assaults even on exceptionally modern frameworks and framework models that undermine our information-driven world. Since the more seasoned and surprisingly the most complex advances are normal to cutting-edge lawbreakers and normal day the same, there are dependably chances of hoodlums implying over the escape clauses in such encryption/security calculations. Cybercriminals include running hash tables and building brilliant rainbow tables to go over and unscramble related scrambled information from the objective data set. Having a more current and 2-D encryption style which isn't simply close to difficult to plan yet additionally difficult to track with because of every special encryption strategy for a similar technique for encryption.

Manuscript received August 5, 2022
Manuscript revised August 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.8.15

Aside from that, it's obviously true that the break of safety is normal due to misusing of public encryption keys and the execution of strategies, for example, Trusted Partner Module (TPM's) is hard to carry out and doesn't scale well with the scaling of the application inside. To counter this one should have to handle double security checks in the execution of the encryption program to guarantee the greatest security.

2. Literature Survey

Information encryption is a most customary method that safe exceptionally secret data by utilizing some ordinary calculation, which as of now exist or are prewritten. The most impressive piece of encryption method is key age, which has two sections, one is symmetric key age and another is lopsided key age. These days programmers are effectively able to break the key with the assistance of present-day high figuring machines. The current need is firmly encoded information that can't be decoded through cryptanalysis [1]. In the ECB mode, encryption continues on the information where information is partitioned to squares of size 'n' bits (size of the fundamental square code), and every information block is scrambled freely of any remaining information blocks, utilizing a similar key. This outcome is a basic and quick encryption strategy. Since there are no information conditions between encoded blocks, ECB mode additionally enjoys the benefit that it very well may be carried out on equal processors [2]. Non-Uniform Steps Model, which can be applied as a layer over customary encryption calculations to upgrade their force. Further in this paper, momentarily portray the different encryption calculations that one can use with the Non-Uniform Steps model to upgrade security and issues engaged with the calculation and their conceivable arrangement [3]. Rubik's Cube shows its mechanical workmanship from the parts of beginning and

improvement, qualities, research status, and particularly its mechanical designing plan, just as making a dream for the application in the component. From that point forward, they present investigations of Rubik's Cube are inspected in different disciplines at home and abroad, including the exploration of Rubik's Cube logical representations, decrease calculations, trademark applications, and instrument issues [4]. The encryption key should be erratic and profoundly touchy to tiny changes in its worth. For this reason, confusion maps are so significant because they have these properties. Disarray maps create arbitrary numbers with specific attributes like bifurcation, flightiness, and the sensitivity of the underlying condition [6]. Identification requires setting, for example, realizing danger opens another arrangement of difficulties. For shrewd urban communities to seek innovation-driven savvy arrangements, they need to redesign with location viability observation and basic alarm checking. Danger discovery is to reveal all connected movement, uncovering stages that don't occur ever before which is by all accounts noxious action that might be like the recently found and distinguish the expansiveness of safety incidents [7].

Information security is a significant part of the correspondence framework that has a concentration for trading data among parties in areas truly separated. In the present cutthroat market, various methods have been created to send information safely. Present a mixture strategy that joins the speed of Data Encryption Standard (DES) for encryption of information and Rivest Shamir Adleman (RSA) calculations to scramble DES secret keys for legitimate key dissemination and management [8]. In an ironic twist, smart city technologies are promoted as an effective way to counter and manage uncertainty and risk, yet they paradoxically induce new risks, including making city infrastructure and services more vulnerable and open to extensive forms of digital vandalism, network disruption, and cyber-criminal exploitation [9]. The combined concepts of DES and RSA to form a hybrid technique have been achieved and presented. The performance of the developed hybrid (DES-RSA technique) has and Baras Game-theoretical and compared to other hybrid systems (AES-RC4, SERPENT-RC4, and RC4-AES-SERPENT), using the same packet size of text data samples (ranging from 1KB to 30MB) on the same computer system. The developed hybrid (DES-RSA) technique encrypts data faster than hybrid AES-RC4, SERPENT-RC4, and RC4-AES-SERPENT. It was able to generate secure cipher-text in a short time compared to others [10].

Smart cities are complex systems that are composed of heterogeneous devices that communicate using a plethora of protocols. These devices and protocols are vulnerable to many security and privacy issues. Security and privacy issues in a smart city from the perspective of its architecture. It also discusses the available solutions pertinent to these issues and highlights the open research challenges in the realization of a secure smart city [11]. The multifaceted nature sciences are necessary to their understanding which may be a moving objective; their Game-theoretic urban communities themselves are becoming increasingly amazing through the very advancements for utilizing them [12]. Security includes illegal access to information and attacks causing physical disruptions in commission availability. As digital citizens are more and more instrumented with data available about their location and activities, privacy seems to disappear. Privacy protecting systems that gather data and trigger an emergency response when needed are technological challenges that go hand-in-hand with the continual security challenges [13]. If a network is to work successfully, its users got to collaborate. Collaboration takes the shape of following a network protocol and involves some resource expenditure on the part of the user. Therefore, users cannot automatically be expected to follow the protocol if they're not forced to [14].

3. Problem Identification

From antiquated occasions to current occasions, cryptography has climbed a ton of achievements and acquired up to turn out to be seriously convoluted and inconvenient for arbitrary programmers and digital danger sources. Nonetheless, these strategies are widespread and have been hashed in the past.

In the current situation, users are reliant on cutting-edge encryption frameworks (AES or AES256 to be more explicit). They are intended for putting away private information yet are not implied for getting information over an organization. Let us understand it:

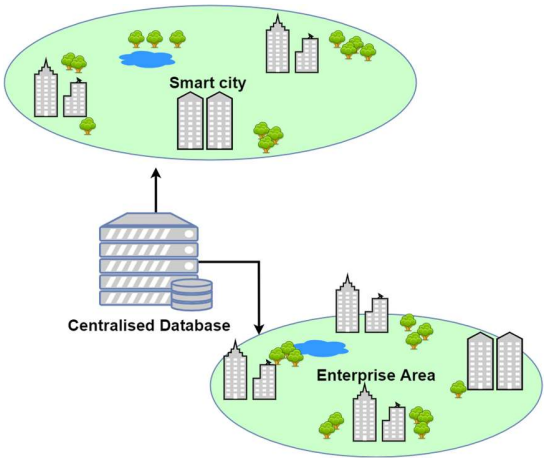


Fig1: Representation of data transmission in smart cities and other WANs.

Thus, in the given figure 1, one expect there is a savvy city, which has around 200,000 occupants who are reliant upon innovations like shrewd homes, workplaces, traffic frameworks, water cleaning frameworks, and so on Which in a roundabout way implies that valuable information from these subsystems is on the steady run from the source (subsystems of the brilliant city) to the data set or some other objective asset inside the savvy city design.

Presently to carry out security with existing cryptographic advancements, for example, AES, Triple DES, RSA security, Blowfish, Two-fish, and so forth need to scramble information (in higher pieces to guarantee the greatest security) every time and move information from one endpoint of the organization to the another. Now, this can happen in either of two ways:

- Right off the starting, the server PC decodes the information and encodes the information with another encryption key.
- On the other hand, the server PC further encodes the information with a current encryption key.
-

In both of two different ways, one at misfortune likewise with the first methodology which will need appropriate synchronization needed for constant information transmission, and in the second methodology won't add any extra security yet entangling the framework for the treatment of information on the server.

Aside from that, these are not guaranteeing any extraordinary component for any potential information spills in the framework.

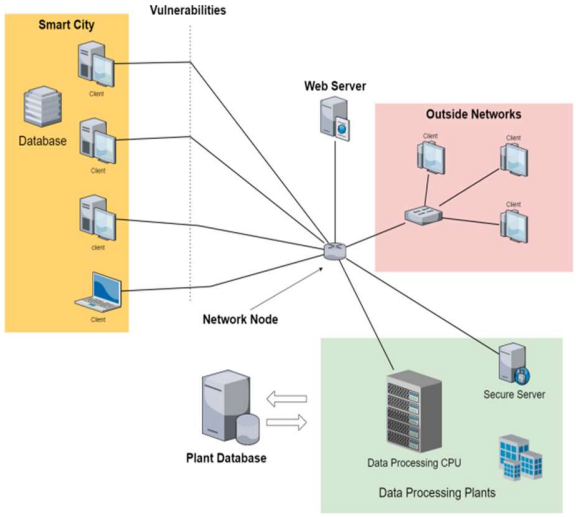


Fig 2: Network diagram related to smart city

Consider the possibility that a digital criminal brings down one of the endpoints of the organization. All things considered, that implies that the server will communicate information to that endpoint and all information parcels will be all around saved with the digital crook. However, the criminal can never really unscramble the information bundles. However, the issue will emerge inside the framework as the framework's information stream is presently at the phase of the information sub-current. Information being sent doesn't reach the target asset. The unfavorable impacts include loss of information concerning a specific subsystem (as information should be appropriate time stepped). This deficiency of information isn't conquerable and over the long haul can influence the framework as the entirety. So another arrangement as far as another cryptographic calculation for safe organization and of course crossing of information has been planned.

4. Cryptographic Algorithm

To take care of this simultaneous issue, consider the possibility of determining a strategy that is uniquely intended to serve progressed server-information crossing. In this way, envision a circumstance. Assume you and your companion are stepping through an exam and you want assistance.

Your companion knows the solutions to every one of the inquiries and you need him to impart the responses to you. Yet, you and your companion are situated a long way from one another in the test community with numerous different understudies in your direction. A severe invigilator is available to impede the method of cheating on the test. Things being what they are, any solid method for conning in the meantime ensuring that the information goes through safe hands (trustful companions who are sitting among you and your companion), and keeping up with security through the span of the test?

Thus, attempting to hold this view, you are having a Rubik's block (a straightforward 3 by 3 Rubik's cube) and you have given that to your companion at the start of the test. You request that your companion compose all replies in each block of the Rubik's 3D shape and blend a long time before transmission. You likewise encourage him to compose the underlying places of the characters alongside the way of blending on a piece of paper.

Also, you educate any remaining companions to follow the briefest conceivable way to pass the block in the interim, blending the solid shape and developing the way on the paper.

Presently your companion composes the responses (inside the person's furthest reaches of 54); blends them well and during the initiation of the test passes the solid shape alongside the paper containing state and encryption keys.

Be that as it may, two issues emerge from here:

Issue One

Consider the possibility that an understudy goes the solid shape through an off-base course; which means the block is lost in the network. What reinforcement component you might perhaps consider to forestall loss of valuable information?

Issue Two

Consider the possibility that rather than the solid shape, the paper containing a succession of blends and encryption gets lost during transmission. Then, at that point, regardless of whether you get the 3D square (information parcel) you will be absent from acquiring valuable data from the obtained information source.

Solution

How conceivably can treat that if even both of the two issues at any point cross the way, actually have the option to get the information back with no security break or break-in information secrecy?

Before going on any further, there is a need to return to our base component for information encryption and how they are

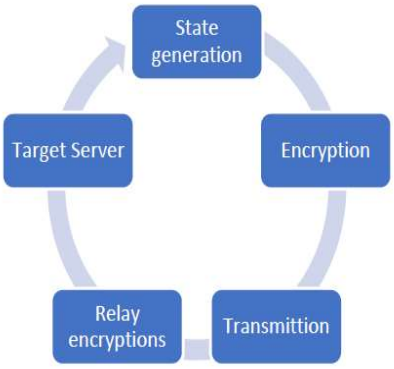


Fig 3: Data transmission cycle.

transmitting it over the network by looking at figure 3.

Along these lines, in reasonable terms, that they are producing an unadulterated pseudorandom state. The state can be characterized as, "The arbitrary way where the embed information into sub-shapes of the Rubik's block".

Blending Rubik's solid shape and on second thought of putting away that in the independent record that can send it as meta information for similar information parcel (Rubik's block) in the meantime encryption key would be able to be made all around safeguarded through a public key.

What's more, after each server endpoint a server-course crossing system will be set off, which will stamp the information parcel professing to cross the particular organization endpoint. If an information bundle is lost or miss found, the information parcels will return to the start server by following converse ways of their stamps.

In this manner, saving the respectability of information over the organization and making information course crossing protected from spillage.

5. Algorithm

Before going any further with the encryption procedure one will first set out our prerequisites, which includes:

- Configuring and setting up the system.
- Encrypting data from the incoming data signals.

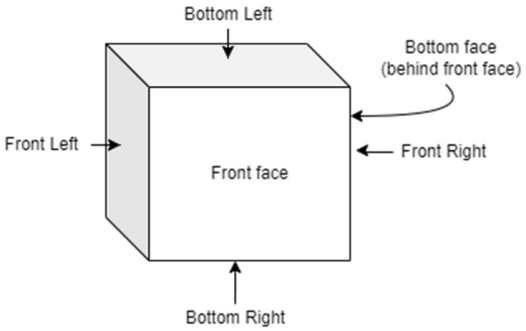
- Decrypting and batching the output at the destination server.
- Maintaining path integrity.

The process of this technique will start when the user on the source end will determine the requirements of the data transmission, like - the amount of data to be transmitted to the unidirectional path to the destination, the number of network nodes in the path, the amount of security that one expects that would promise a better security and latency tradeoff, etc.
In the first stage of the process, the user configures the system for the initial run which can be time-consuming taking into consideration that a lot of hefty tasks will be going underway but since they are onetime processes and will be cached out in the memory for another round of data transmission, it calls for faster transmission of data alongside the encryption. Among the many parameters to set on the onset of initiating the process are the following:

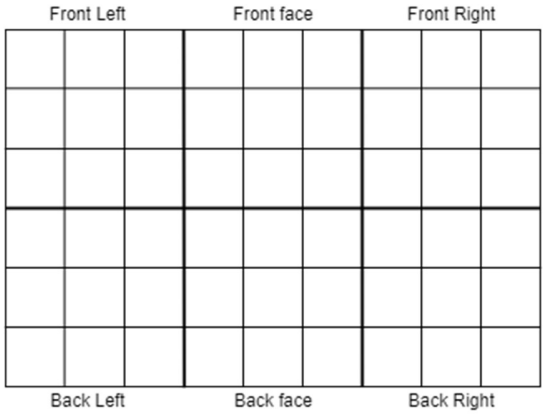
- CubeString: memory is linear or 1D, and hence to mimic that of a Rubik's cube (3D entity) need to model the structure on a linear memory architecture, which will be the base for our whole process and will call it "CubeString" or "cube". In the memory, it will be represented as a string that will represent (cube-Dimension*3) dimensioned separated by logical relations.
- CubeDimension: As one can comprehend from the simple Rubik's cube toy, the complexity of mixing the cube is largely supposed from the dimension i.e. the number of cubes it has on its edges (3x3 or NxN).
- uniform random generator: A uniform random generator is a function that generates numbers between specific ranges with all numbers having an equal probability of occurrence.
- DataQueue: Data queue is the queue for lining up datastreams for encryption.

The representation of the cube in the memory is an important concept in this algorithm. The cube is a 3D entity and hence required to be simplified as a 2D array for easy implementation. The representation is as follow:

- Firstly the name the faces of the cube, like:
  - Front Face
  - Front Left Face
  - Front Right Face
  - Back Face
  - Back Left Face
  - Back Right Face
- The front face is the face on which will apply changes or (mix) with this face in consideration. [*In a Rubik's cube the up-facing face really makes a drastic change when applied to the mixing operations, hence it take only one front-face to remove excess complexity.*]
- The following figure shows the modeling of a 3D cube into a much-relatable 2D rectangle (array).



Fig(4-a) Shows the 3D cube which conceptualize for mixing.



Fig(4-b) Shows the 2D array with 6 sections depicting 6 faces of the Rubik's cube of dimension 3

Fig 4: Diagrammatic representation of the cube

The problems in the average or common encryption methods are as follows:
An asymmetric encryption algorithm based on the factoring problem will have a

public-key calculated using the product of two private keys (large prime numbers).
This calculation is easy to perform, but anyone wanting to derive the private key from the public key will need to factor in, which is much harder.
The difficulty of multiplication grows polynomially with the length of the factors, but the difficulty of factoring grows exponentially. This makes it possible to find a "sweet spot", where a system is usable but essentially unbreakable.
The discrete logarithm problem uses exponentiation and logarithms as its "easy" and "hard" operations. Similar to factoring, the complexity of calculating logarithms grows much more quickly as the size of the exponent increases.
Setting up the System
Firstly it will establish the cubeString for storing and encrypting data. This can be done by running the following piece of code in the source computer.

```
Procedure InitiateProcess(int maxDataSize){

/*
maxDataSize is the integer value stating the maximum character length of the input data and can be computed by thorough data analysis of the data to be transmitted sequentially in the relay format.
*/

int cubeDimension = 3;
int dataSize = maxDataSize;
int cubeLength = ( cubeDimension**2)*6;

/*
The default size of the cube will be three so the starting total possibilities of total combinations start around 40 Quintillion which is relatively high in terms of security-latency tradeoff.
*/

  for(int i=3; i > -1; i++){

/*
 It will run this loop only if the cubeLength is greater than the characterLength of the plainText.
*/

     if(cubeLength>=plainText){
           cubeDimension = i;
           break;
     }
     else{
           i = i+1;
     }
  }
}
```

After this has run, the system is all set and has generated the cubeDimension which is the dimension or edge size of the Rubik's cube. Now will establish the cube String and will cache it out in the system.

```
Procedure setUpCubeString(int cubeDimension){

/*
This procedure is a one-time setup function, meaning it will run for the very first time, during setup and will be cached in the RAM for further use in the relay mechanism.

cubeDimension is the integer variable that stores the dimension of the cube string.
*/

int cubeString = "";

/*
The default size of the cube will be three so the starting total possibilities of total combinations start around 40 Quintillion which is relatively high in terms of security-latency tradeoff.
*/

  for(int i = 0; i < cubeDimension; i++){

/*
  using placeholders characters, (for here: @ is the placeholder in this illustration of the example)
*/

for(var j=0; j< cubeDimension; j++){
  cubeString += "@";
}

return cubeString; // returning for first

cacheOut(cubeString); // caching out

  }
}
```

Encrypting data

The recent algorithm had generated an empty cubeString and has cached it out for future or immediate encryption and transmission. Now one can have to set the State. State. It is how data characters are aligned or placed in the cubeString before the process of mixing and the produced data is called state or state-map.

State is the second security key used for decryption algorithms and adds another layer of security and offers a mechanism for maintaining path integrity.

```
Procedure setState(string cubeString, plainText){

/*
This procedure is an initiation process, meaning it will run every time a new plain data has entered into the queue…

the input parameter cubeString is the object of the cached cubeString.

Assume there exists a function genRandomPos(initialvalue, maxvalue) which uniformly returns an integer between initialvalue and maxvalue
*/

string cubeString = “”, state = “”;

int availPos []; // contains all available indices vacant for data storage

int initvalue = availPos[0];
int maxvalue = availPos[lengthOf(availPos)-1];

 for(int i=0; i< lengthOf(plainText);i++){
    int pos = genRandomPos(initvalue, maxvalue);

   /*
 popping out at index pos as the position is not vacant
*/

   index.pop(pos);

cubeString.concatenate(inputString[availPos[pos]]);

state.concatenate(parse_to_string(pos)); // saving state

/*
Now since the cubeString is a 3D cube representation, hence for calculations will take a 2D rectangle of 3 by 2 squares of dimensions (cubeDimension by cubeDimension) for storing and manipulating data in the cube which then can be stored as a single string in the memory for optimization.
*/

  return cubeString, state;

 }

 }
```

Now one can have the last task of encrypting the message by mixing the data and generating an encryption key. The encryption process in this algorithm is more like mixing of Rubiks randomly with each mix consisting of metadata like- *which row or column to be mixed, how many times should you play the same mixing move,* and *the direction of the manipulation*.

It will be having a class, let’s say named mixRubiks() with the following member Functions:

- mixColumnUp(i, d, f)
- mixColumnDown(i, d, f)
- mixRowLeft(i, d, f)
- mixRowRight(i, d, f)

These member functions will be having the following parameters: position of row or column, the direction of the mix, and frequency of the mix.

These member functions will map out the cubeString (*which will be supplied as the parameter to the constructor of the class mixRubiks()*) mix the cube in a uniformly-random order and will generate an encryption key which can be followed in reverse order to reach the initial-stage or raw cube format which can be decrypted further through the state.

The mixing can be quite difficult to understand so it will take a simpler approach from a pasty box (a 3D entity that is modeled out of a single sheet of cardboard or 2D).

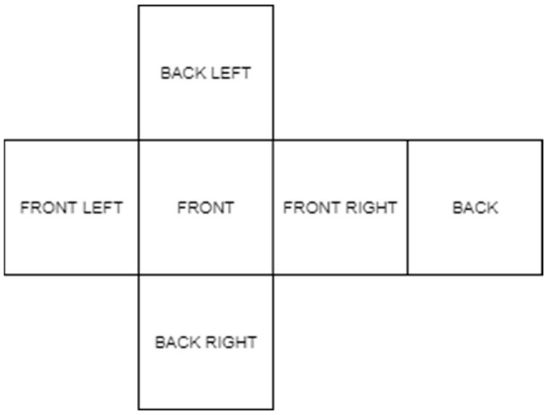


Fig 5: Modelling 3D cube into a 2D entity

The mixing follows the following pattern:

[***Note**: All rows and columns are zero indexed]*

1. For mixing a column: The procedure for mixing the K^{th} column of the FRONT face will be:

    a. K^{th} column of the Front face is mixed UP/DOWN hence mapped on the K^{th} column of the BACK LEFT face.

    b. BACK LEFT face's K^{th} column is mixed in the same direction and hence mapped on the [cubeDimension - 1 - (K mod D)]th column of BACK face, where D is the cubeDimension.

    c. [cubeDimension - 1 - (K mod cubeDimension)]th column of BACK face is mixed in the opposite direction and hence mapped on the Kth column of the BACK RIGHT face.

    d. K^{th} column of the BACK RIGHT face, in the last, will map to the FRONT face.

2. For mixing a row: The procedure for mixing the K^{th} row of the front face will be:

    a. K^{th} row of the Front face is mixed LEFT/RIGHT hence mapped on the K^{th} row of the FRONT LEFT face.

    b. The FRONT LEFT face's K^{th} row is mixed in the same direction and hence mapped on the K^{th} row of BACK face.

    c. K^{th} column of BACK face is mixed in the same direction and hence mapped on the K^{th} row of the FRONT LEFT face.

    d. K^{th} column of the FRONT LEFT face, in the last, will map to the FRONT face.

Decrypting and batching the output at the destination server
The decryption of the mixed cubeString is similar to that of a Rubik's cube and the encryption key is followed in reversed order.
The decryption

Maintaining path integrity.
After it one can have everything set up then arises the question - What if the transmitted data's confidentiality or integrity gets harassed by a network intruder?
The problem is solved in a very clever way. As one can now know the encryption process ends with two byproducts - *encryption key* and *state*, and the cipher data or *cubeString*.

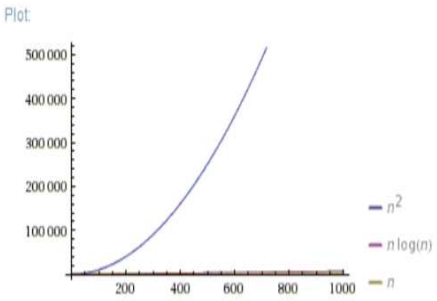
Prerequisites:
The object of the block will be stored and will be replicated for use. This will save time and calculation power. The utility capacities will be running on elective specialist strings to guarantee quicker activity and equal handling of information.
Besides, test results and security investigations are applied to utilize the proposed calculation and its exhibitions are approved with ongoing cryptographic calculations. The security and execution examinations presume that the proposed calculation is secure, quick, and opposes different assaults.

Assumptions:

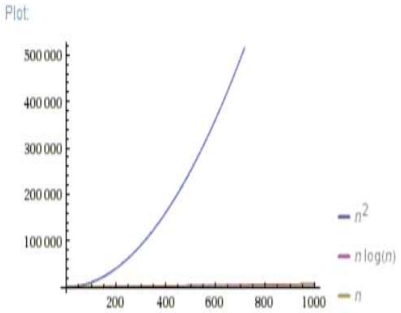
- Caching the making of the cube are truly ignoring it as it will be done once after the server has started rolling for the first time.
- Time Complexity at the best case: If the string is less than the default cached cube dimension.

Let it be D, then the procedure is as follows:
Making of state: Frequency count of the command => n
Filling of the cube: Frequency count of the command => n
Mixing of the cube: Frequency count of the command => n^2
Where n is the total number of string characters of the plain text.
Final best-case complexity: O(n^2+2n) ==(approx.)➔ O(n^2)
X-AXIS: time T

Y-AXIS: input size I



III. Time Complexity at the Worst case:
If the string is more than the default cached cube dimension, the new cube string has to be generated. Let it be D, then the procedure is as follows: Finding the ideal size of the cube: Frequency count of the command => M (a positive integer)
Making of the cube: Frequency count of the command => D
Making of state: Frequency count of the command => N
Filling of the cube: Frequency count of the command => N
Mixing of the cube: Frequency count of the command => N^2 Where n is the total number of string characters of the plain text.
Final best-case complexity: O(N^2+2N+M+D) ➔ (approx.) ➔ O(n^2+2N+A) where A is a large constant integer which can be ignored in analysis Approx ➔ O(N^2)



6. Conclusion

This paper includes a new cryptographic algorithm for safe route traversal for data of smart cities which is a non-traditional, non-hash, non-linear, 3D encryption implementation designed. The security of the proposed algorithm has been proved over many experimental analyses: over time and space complexities. The experimental results have concluded that the proposed algorithm can be used to encrypt and decrypt efficiently. Finally, the encryption algorithm will be proposed to increase the current algorithm's security in the future. This algorithm is a process fit of the data relay and since the cube is cached hence it can be used for real-time data for packet making.

References

[1] Dixit, P., Gupta, A. K., Trivedi, M. C., & Yadav, V. K. (2018). Traditional and hybrid encryption techniques: a survey. In *Networking communication and data knowledge engineering* (pp. 239-248). Springer, Singapore.

[2] Belal, A. A., & Abdel-Gawad, M. A. (2001, March). 2D-encryption mode. In *Second NIST Modes of Operation Workshop*.

[3] Varshney, N., & Qadeer, M. A. (2012). Non-Uniform Steps Model: New Layer to Traditional Security Encryption Algorithms (Next-Generation Data Security Layer). *International Journal of Engineering and Technology*, *4*(6), 808.

[4] Zeng, D. X., Li, M., Wang, J. J., Hou, Y. L., Lu, W. J., & Huang, Z. (2018). Overview of Rubik's cube and reflections on its application in the mechanism. *Chinese Journal of Mechanical Engineering*, *31*(1), 1-12.

[5] M. Bellare, A. Desai, E. Jokipii, P. Rogaway, "A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation", Proceedings of the 38th Symposium on Foundations of Computer Science, IEEE, 1997.

[6] Elghandour, A., Salah, A., & Karawia, A. (2021). A new cryptographic algorithm via a two-dimensional chaotic map. *Ain Shams Engineering Journal*.

[7] Chhabra, A., and Singhal, N, "Indicator-Based Cyber Threats Detection for Data of Smart Cities Using Bio-Inspired Artificial Algae Algorithm", IJARET, Vol. 11, Issue. 11, pp. 1530-1536, (2020).

[8] Adedeji Kazeem, B., & Akinlolu, P. (2014). A new hybrid data encryption and decryption technique to enhance data security in communication networks: algorithm development. *International Journal of Scientific & Engineering Research*, *5*(10).

[9] Dodge M, Kitchin R. The challenges of cybersecurity for smart cities. InCreating Smart Cities 2019 (pp. 205-216). Routledge.

[10] Adedeji Kazeem B, Akinlolu P. A new hybrid data encryption and decryption technique to enhance data security in communication networks: algorithm development. International Journal of Scientific & Engineering Research. 2014 Oct;5(10).

[11] Butt, Talal Ashraf, and Muhammad Afzaal. "Security and privacy in smart cities: issues and current solutions." In *Smart technologies and innovation for a sustainable future*, pp. 317-323. Springer, Cham, 2019.

[12] Eckhoff David and Wanger Isabel, "Privacy in Smart Cities-Applications, Technologies, Challenges and Solutions", IEEE Communications Surveys and Tutorials, Vol. 20, Issue. 1, pp. 489-516, (2017).

[13] Elmaghraby, A. S., andLosavio, M. M., "Cyber Security Challenges in Smart Cities: Safety, security, and privacy", Journal of advanced research, Vol.5, Issue. 4,pp. 491-497, (2014).

[14] Theodorakopoulos, G. and Baras, J. S., "Game-theoretic modeling, of malicious users in collaborative networks", IEEE Journal on selected areas in communications, Vol.26, Issue. 7,pp. 1317-1327, (2008).67.

Arpit Chhabra received his M.C.A degree from B.I.T College, Affiliated from Uttar Pradesh Technical University, and M.Tech(CSE) degree from Shobhit University . Research scholar of Shobhit university in computer science and engineering department. His research interests include Cyber Security, Cloud technology, Information Security techniques.

Dr. Niraj Singhal is Graduate in Computer Science and Engineering, M.Tech. in Computer Engineering and Ph.D. (Computer Engineering & Information Technology). He is Fellow and member of several International/National bodies. He is also serving as reviewer and member of the advisory board for several International/National journals. He has more than one hundred research publications to his credit in National/International journals/conferences of repute. Many of them are in SCI/ICI/SCOPUS/dblp/UGC listed Journals. He has twenty-seven years of rich experience of administration, coordinating and teaching at various levels. His area of interest includes system software, web information retrieval and software agents.