Cost Effective Data Placement in Edge Storage System With Erasure Code

Team Details
1. B. PAVAN(20EG105305)
2. CH. MANIPREETH(20EG105309)
3. VVSS CHANDRA(20EG105351)

Project Supervisor
Mrs ASMA TAHSEEN
ASSISTANT PROFESSOR

Introduction

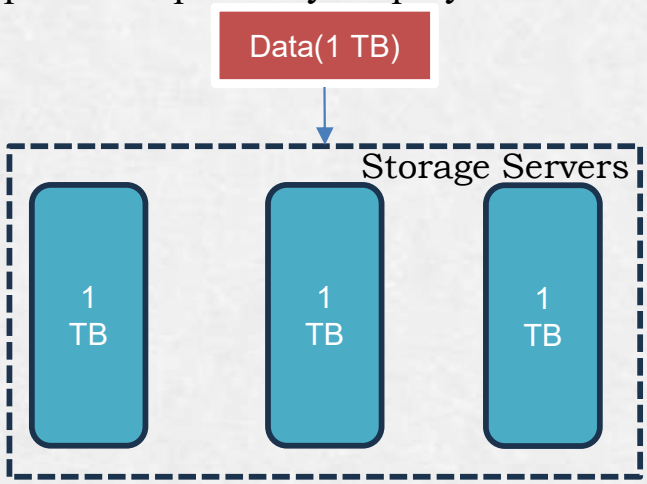
Edge computing is all about bringing computing power and data storage closer to where it's needed. Instead of relying solely on big, distant data centers, edge computing places these resources right at the "edge" of the network, closer to where devices are generating or using data. This setup reduces the time it takes for data to travel back and forth, making processes faster and more responsive. This approach is particularly valuable in scenarios requiring real-time data processing, such as IoT devices, autonomous vehicles, and industrial automation. Edge computing empowers businesses to leverage faster insights and make quicker decisions, ultimately driving innovation and improving user experiences across various industries.

Problem Statement

Most existing studies of Edge Storage System focus on the storage of data replicas in the system to ensure low data retrieval latency for users. However, replica-based edge storage strategies can easily incur high storage costs. It is not cost-effective to store massive replicas of large-size data, especially those that do not require real-time access at the edge e.g., system upgrade files, popular app installation files, videos in online games.. It may not even be possible due to the constrained storage resources on edge servers.

Triple Replication

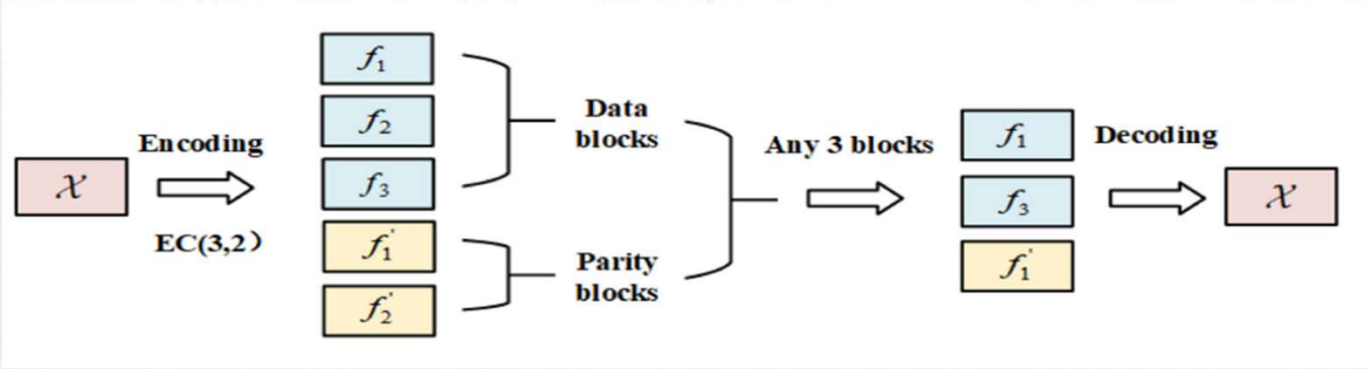
Triple replication typically refers to a data storage or data replication strategy where three copies (replicas) of data are maintained across a distributed system or storage infrastructure. This approach is primarily employed for fault tolerance, high availability, and data durability.



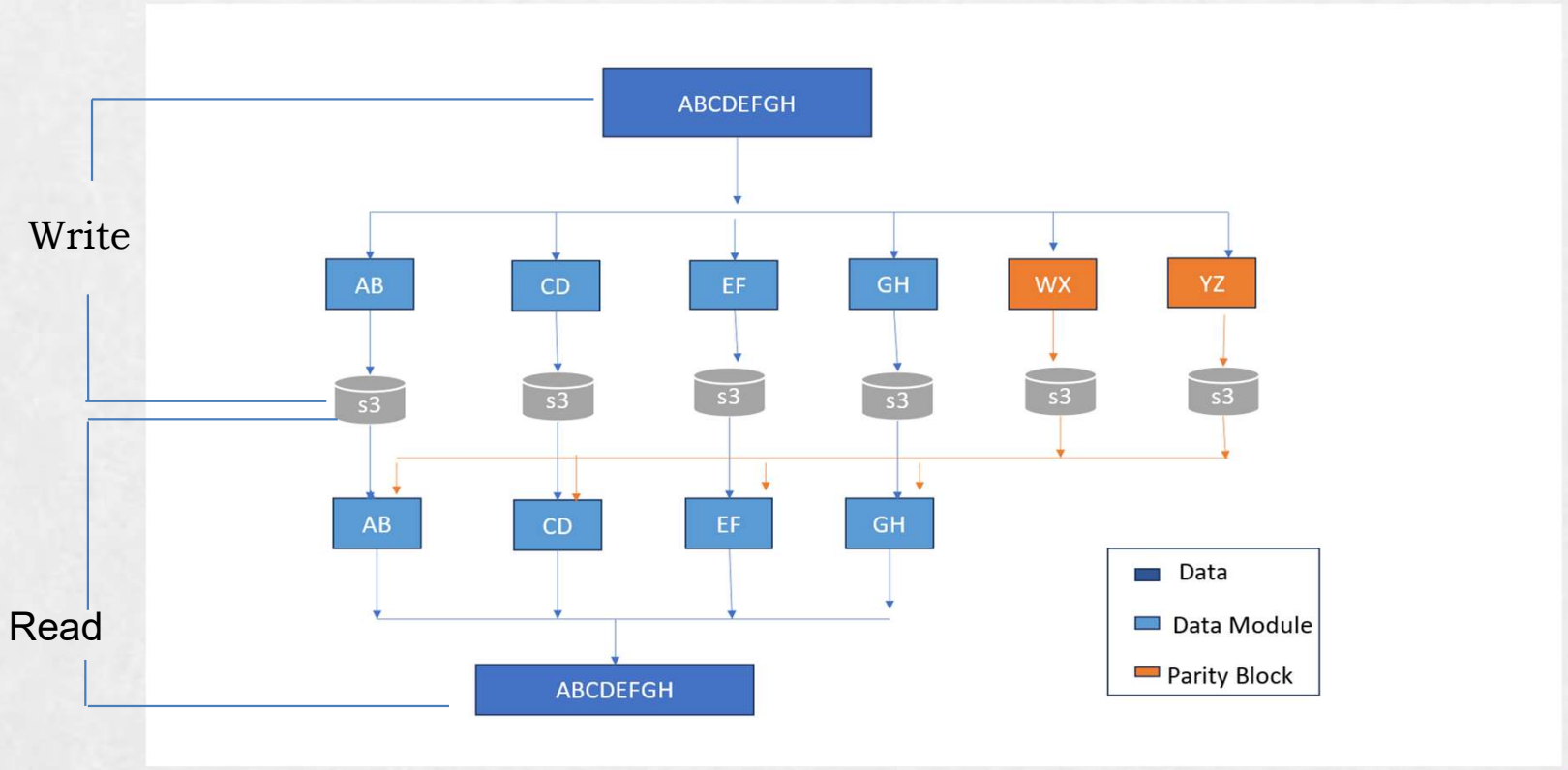
- Above diagram depicts that we used triple replication to store data of size 1-TB which is costing storage of 3-TB.
- Here we can optimize the storage cost by using Erasure Code.

Proposed Method

we implement the use of erasure coding in cost effective storage of large data in Edge Storage System. Under an erasure code scheme, a data X to be stored can be divided into M data blocks and K parity blocks. These data and parity blocks are distributed to be stored on different storage nodes (e.g., edge servers in an ESS) accessible to users. A user can retrieve M data and/or parity blocks (together referred to as coded blocks hereafter) from any accessible edge servers to construct X for use. each block of data which is stored in nodes are encrypted with different algorithms and generate two keys (AES, DES) which will reduce data attacks.



Proposed Method



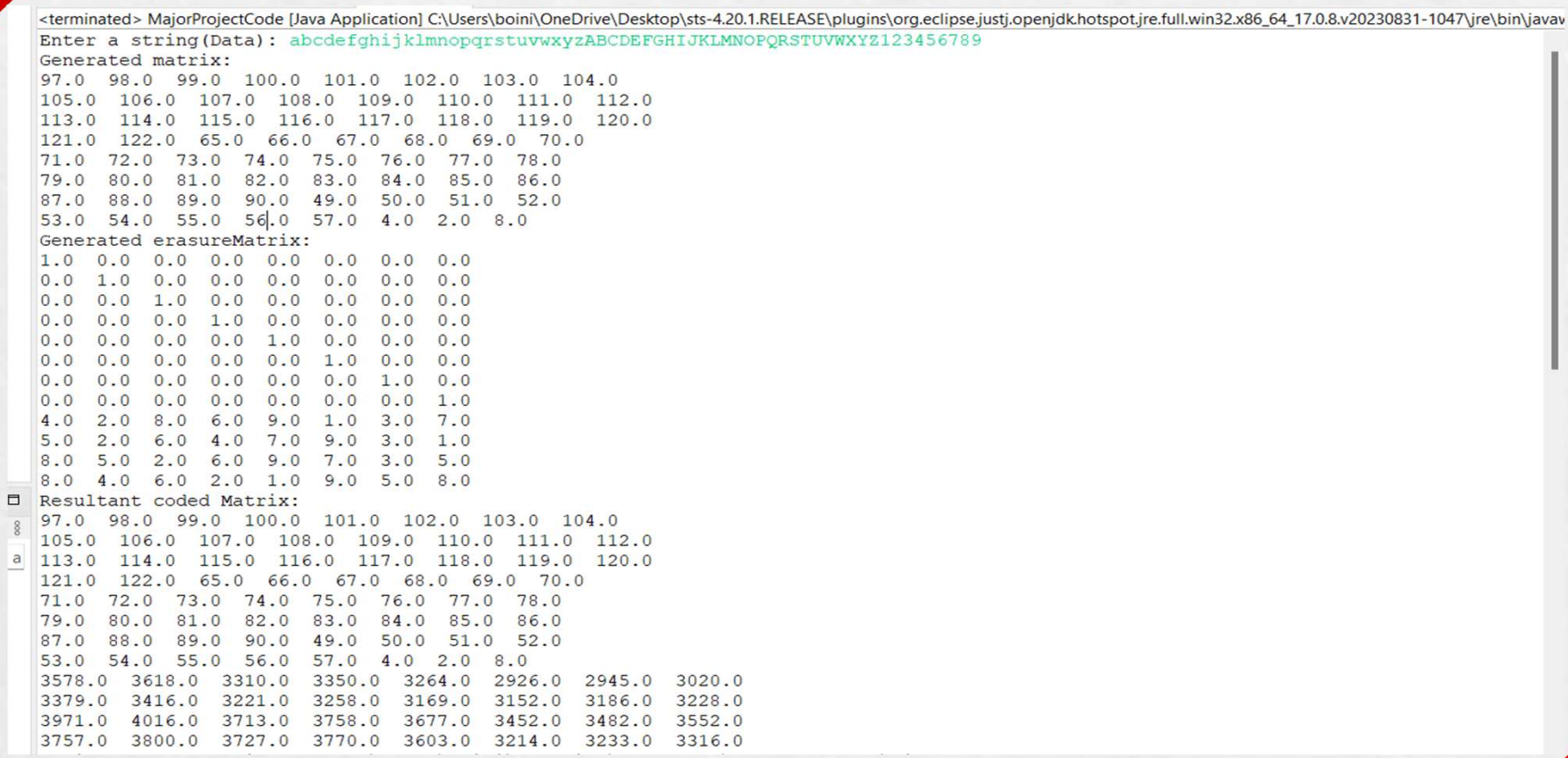
Experiment Environment

+

Coding: Java

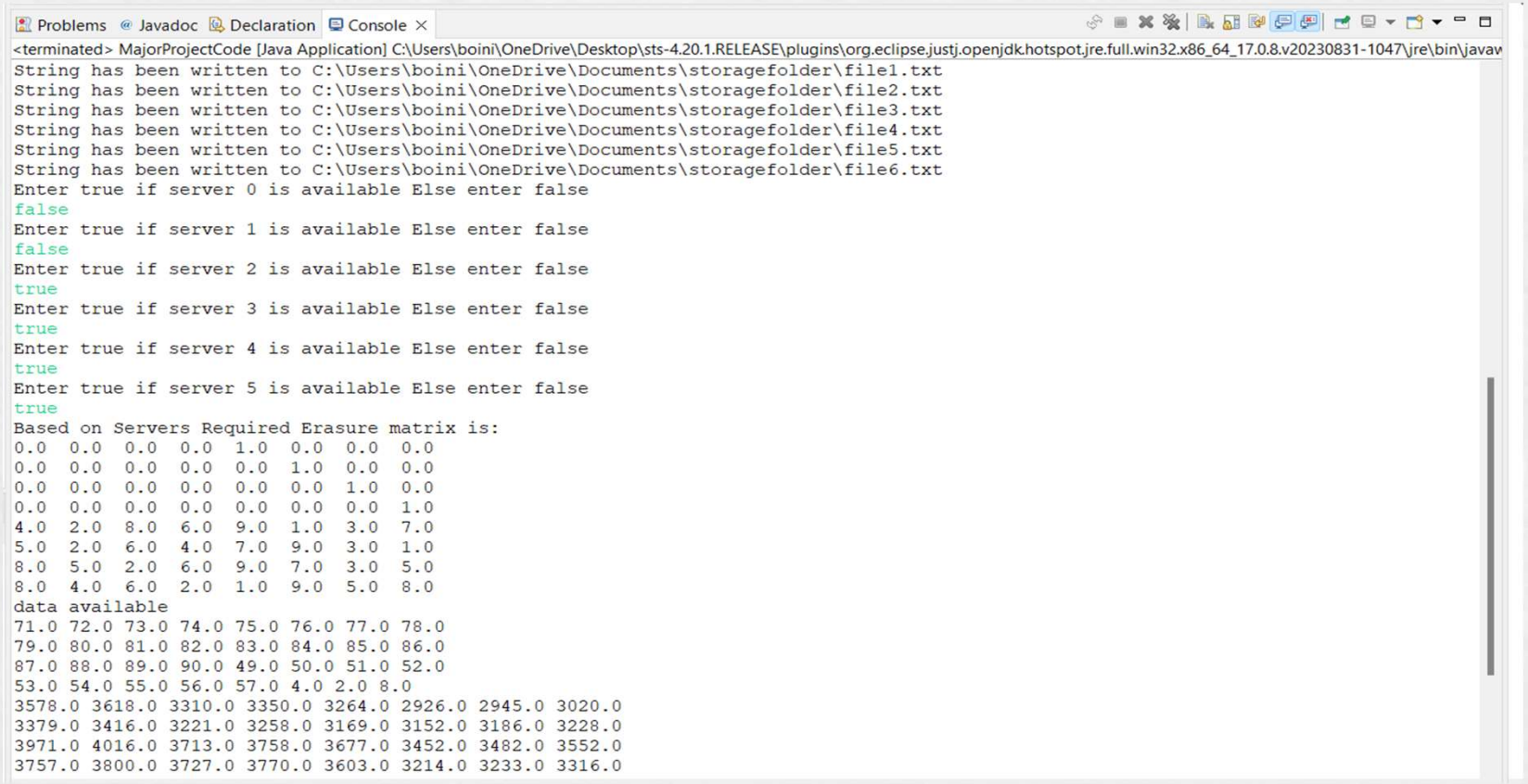
IDE: STS(Spring Suite Tool 4)

Experiment Screen shorts



```
<terminated> MajorProjectCode [Java Application] C:\Users\boini\OneDrive\Desktop\sts-4.20.1.RELEASE\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javav
Enter a string(Data): abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ123456789
Generated matrix:
97.0  98.0  99.0  100.0  101.0  102.0  103.0  104.0
105.0  106.0  107.0  108.0  109.0  110.0  111.0  112.0
113.0  114.0  115.0  116.0  117.0  118.0  119.0  120.0
121.0  122.0  65.0  66.0  67.0  68.0  69.0  70.0
71.0  72.0  73.0  74.0  75.0  76.0  77.0  78.0
79.0  80.0  81.0  82.0  83.0  84.0  85.0  86.0
87.0  88.0  89.0  90.0  49.0  50.0  51.0  52.0
53.0  54.0  55.0  56.0  57.0  4.0  2.0  8.0
Generated erasureMatrix:
1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  1.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  1.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  1.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0
4.0  2.0  8.0  6.0  9.0  1.0  3.0  7.0
5.0  2.0  6.0  4.0  7.0  9.0  3.0  1.0
8.0  5.0  2.0  6.0  9.0  7.0  3.0  5.0
8.0  4.0  6.0  2.0  1.0  9.0  5.0  8.0
Resultant coded Matrix:
97.0  98.0  99.0  100.0  101.0  102.0  103.0  104.0
105.0  106.0  107.0  108.0  109.0  110.0  111.0  112.0
113.0  114.0  115.0  116.0  117.0  118.0  119.0  120.0
121.0  122.0  65.0  66.0  67.0  68.0  69.0  70.0
71.0  72.0  73.0  74.0  75.0  76.0  77.0  78.0
79.0  80.0  81.0  82.0  83.0  84.0  85.0  86.0
87.0  88.0  89.0  90.0  49.0  50.0  51.0  52.0
53.0  54.0  55.0  56.0  57.0  4.0  2.0  8.0
3578.0  3618.0  3310.0  3350.0  3264.0  2926.0  2945.0  3020.0
3379.0  3416.0  3221.0  3258.0  3169.0  3152.0  3186.0  3228.0
3971.0  4016.0  3713.0  3758.0  3677.0  3452.0  3482.0  3552.0
3757.0  3800.0  3727.0  3770.0  3603.0  3214.0  3233.0  3316.0
```

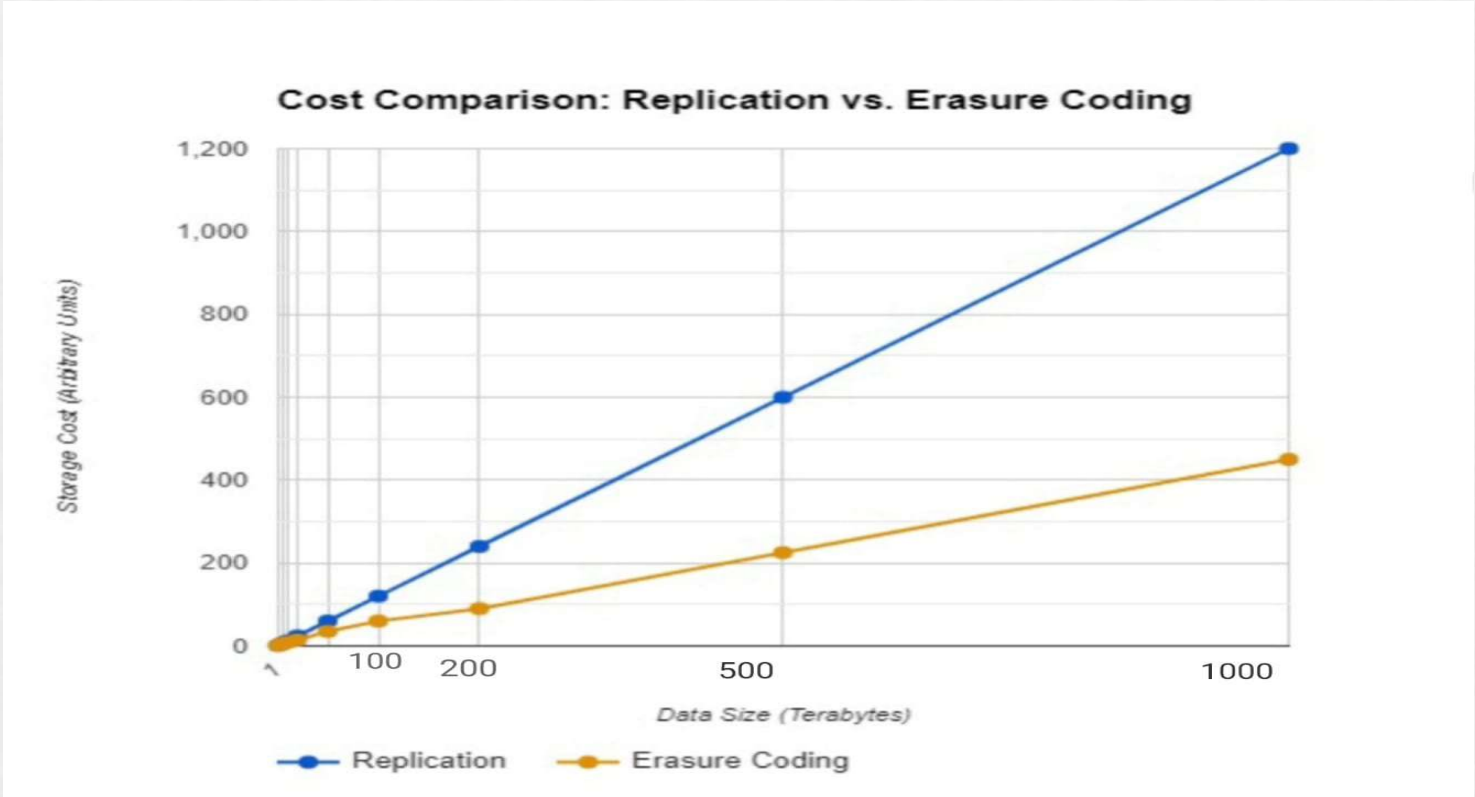
Experiment Screen shorts



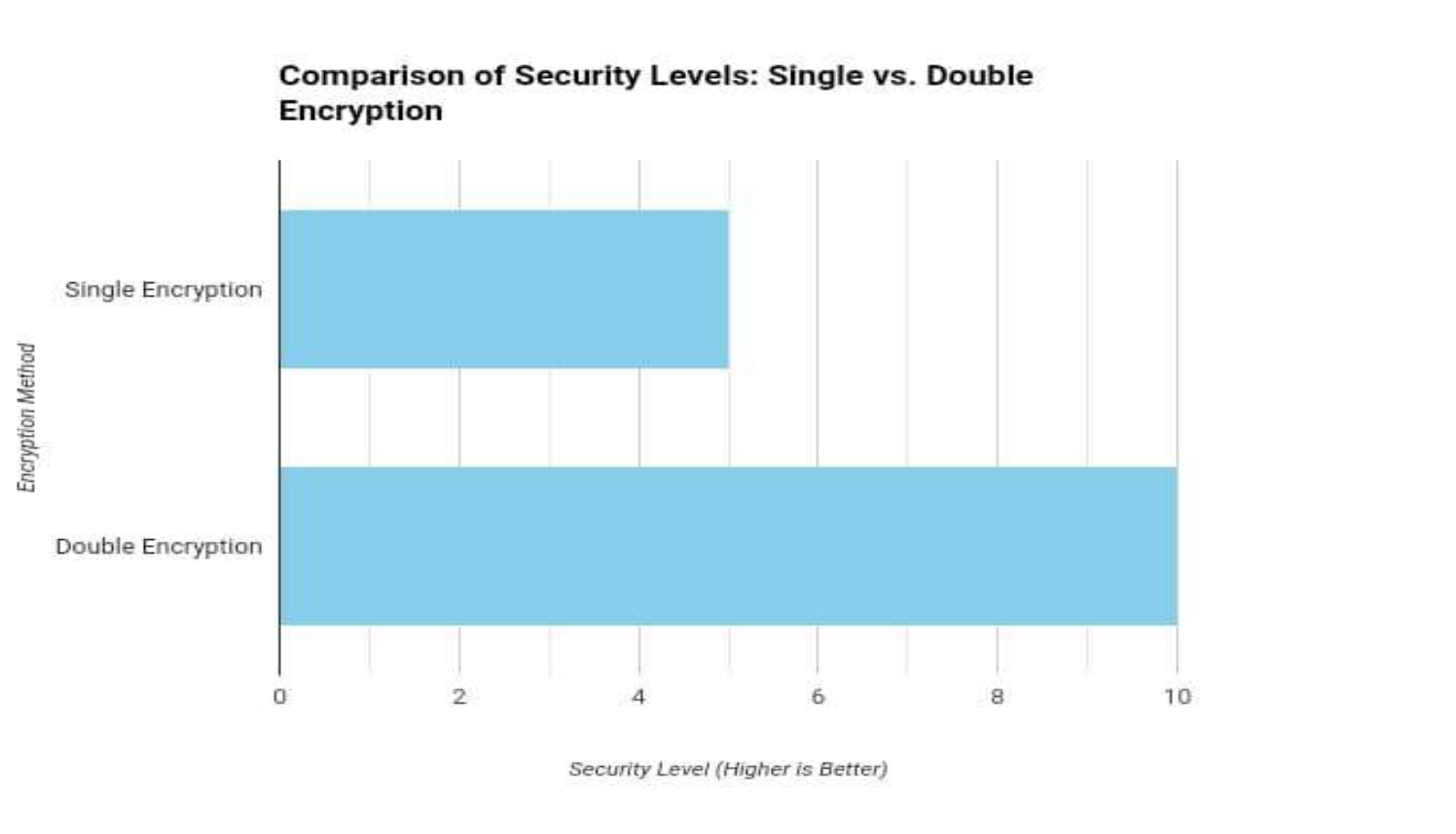
Experiment Screen shorts

```
Based on Servers Required Erasure matrix is:
0.0  0.0  0.0  0.0  1.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0
4.0  2.0  8.0  6.0  9.0  1.0  3.0  7.0
5.0  2.0  6.0  4.0  7.0  9.0  3.0  1.0
8.0  5.0  2.0  6.0  9.0  7.0  3.0  5.0
8.0  4.0  6.0  2.0  1.0  9.0  5.0  8.0
data available
71.0 72.0 73.0 74.0 75.0 76.0 77.0 78.0
79.0 80.0 81.0 82.0 83.0 84.0 85.0 86.0
87.0 88.0 89.0 90.0 49.0 50.0 51.0 52.0
53.0 54.0 55.0 56.0 57.0 4.0 2.0 8.0
3578.0 3618.0 3310.0 3350.0 3264.0 2926.0 2945.0 3020.0
3379.0 3416.0 3221.0 3258.0 3169.0 3152.0 3186.0 3228.0
3971.0 4016.0 3713.0 3758.0 3677.0 3452.0 3482.0 3552.0
3757.0 3800.0 3727.0 3770.0 3603.0 3214.0 3233.0 3316.0
Matrix Multiplication Result or actual matrix after data re-construction
97.0 98.0 99.0 100.0 101.0 102.0 103.0 104.0
105.0 106.0 107.0 108.0 109.0 110.0 111.0 112.0
113.0 114.0 115.0 116.0 117.0 118.0 119.0 120.0
121.0 122.0 65.0 66.0 67.0 68.0 69.0 70.0
71.0 72.0 73.0 74.0 75.0 76.0 77.0 78.0
79.0 80.0 81.0 82.0 83.0 84.0 85.0 86.0
87.0 88.0 89.0 90.0 49.0 50.0 51.0 52.0
53.0 54.0 55.0 56.0 57.0 4.0 2.0 8.0
Data Stored is:abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ123456789
```

Experiment Results



Experiment Results



Finding

1. Cost

Replicating data in edge systems incurs high storage costs due to storing multiple copies. Erasure coding disperses data across nodes without redundant hardware, reducing expenses by up to 68%. This cost-effective method is crucial for minimizing storage costs in edge data storage, offering a compelling solution.

2. Security

Single encryption encrypts data once, offering basic security. Double encryption adds a second layer, requiring decryption of both for access. While single encryption is simpler, double encryption enhances security, suitable for highly sensitive data or environments needing robust protection.

3. Data Integrity Checking

Matrix inversion and multiplication techniques encode data into matrices, detecting errors through operations. Unlike checksum or CRC, they offer robust error detection and correction, enhancing data integrity and reliability significantly.

Justification

The proposed methods offer significant improvements over previous ones in terms of storage, security, and data integrity checking. Erasure coding reduces storage needs from triple replication. Double encryption enhances security compared to single encryption. Matrix inversion and multiplication techniques provide more robust data integrity checking compared to traditional methods like checksum, CRC, and parity checking. Overall, these advancements offer more efficient, secure, and reliable data management solutions.

Parameter	**Previous methods**	**Proposed method**
Storage	Triple Replication	Erasure Coding
Security	Single Encryption	Double Encryption
Data Integrity Checking	Checksum, CRC, parity Checking	Matrix Inversion and Multiplication

THANK YOU