ERASURE CODE OVERVIEW

Understanding Data and Parity Blocks in Erasure Coding:

- **Data Blocks**: These store the actual object data. The more data blocks you have, the more efficient your storage is in terms of space utilization.
- **Parity Blocks**: These provide redundancy to allow the system to reconstruct lost data in case of a disk or node failure. The more parity blocks you have, the higher the fault tolerance (ability to recover from failures).
- MinIO requires a **minimum of 4 disks** to set up erasure coding

How MinIO Decides Data and Parity Blocks:

1. **Total Number of Disks**: MinIO looks at the number of disks (volumes) in the configuration.
2. **Erasure Set Size**: MinIO divides the available disks into erasure sets.
3. **Ratio of Data and Parity Blocks**: MinIO uses an efficient **ratio** of data and parity blocks based on the number of disks. It typically chooses a **balanced distribution** between data and parity, ensuring that the system can tolerate failures while optimizing storage usage.

Erasure Code in Standard Storage

Examples to Illustrate How Data and Parity Blocks are Decided:

1. Minimum Configuration: 4 Disks (2+2 Erasure Set)

- **Disks Available**: 4
- **Erasure Set Configuration**: 2 data blocks + 2 parity blocks
  - **Data Blocks**: 2 disks hold the actual data.
  - **Parity Blocks**: 2 disks store parity information for redundancy.

In this case, you can lose up to **2 disks**, and MinIO will still be able to reconstruct the data using the parity blocks.

2. 6 Disks (4+2 Erasure Set)

- **Disks Available**: 6
- **Erasure Set Configuration**: 4 data blocks + 2 parity blocks
    - **Data Blocks**: 4 disks hold the actual data.
    - **Parity Blocks**: 2 disks store parity information for redundancy.

With this configuration, MinIO can tolerate up to **2 disk failures** while still being able to reconstruct the data.

3. 8 Disks (6+2 Erasure Set)

- **Disks Available**: 8
- **Erasure Set Configuration**: 6 data blocks + 2 parity blocks
    - **Data Blocks**: 6 disks hold the actual data.
    - **Parity Blocks**: 2 disks store parity information.

Here, MinIO can tolerate up to **2 disk failures**. This configuration provides more efficient storage (more disks are used for data).

4. 12 Disks (8+4 Erasure Set)

- **Disks Available**: 12
- **Erasure Set Configuration**: 8 data blocks + 4 parity blocks
    - **Data Blocks**: 8 disks hold the actual data.
    - **Parity Blocks**: 4 disks store parity information.

In this case, MinIO can tolerate **4 disk failures**. It balances redundancy (4 parity disks) and efficiency (8 data disks).

5. 16 Disks (8+8 Erasure Set)

- **Disks Available**: 16
- **Erasure Set Configuration**: 8 data blocks + 8 parity blocks
    - **Data Blocks**: 8 disks hold the actual data.
    - **Parity Blocks**: 8 disks store parity information.

Why 4+4 Isn’t Used for 8 Disks:

A **4+4** configuration for 8 disks would mean that:

- Half the disks are used for **data**, and half are used for **parity**.
- This results in **50% storage efficiency**, which is **less efficient** than the **75% efficiency** of a **6+2** setup.

While **4+4** would give you the same level of fault tolerance (2 disk failures) as **6+2**, it would come at the cost of using more of your storage for parity rather than for actual data. In most scenarios, this is less desirable.

Summary of Key Differences:

Configuration	Data Blocks	Parity Blocks	Usable Storage (%)	Fault Tolerance
6+2	6	2	75%	2 disk failures
4+4	4	4	50%	2 disk failures

Why 2 Disk Failures in 4+4 Setup:

- In a **4+4** setup, there are **4 data blocks** and **4 parity blocks**.
- Even though you have 4 parity blocks, the system can still only tolerate **2 disk failures** because parity blocks can **only reconstruct data blocks**, not other parity blocks.
  In a **4+4** setup, the extra parity blocks don’t provide additional fault tolerance but **more redundancy**. The data blocks are heavily protected, and the data can be reconstructed if up to 2 disks fail. However, losing more than 2 disks means some of the original data blocks are unrecoverable even if you have 4 parity blocks.

Sample Data Example:

Imagine we want to store the string `"Hello, MinIO!"`. For simplicity, let's assume that MinIO divides this string into **8 equal parts** (since we have 8 data blocks). MinIO will then create **8 parity blocks** using erasure coding to provide redundancy.

Step 1: Data Block Creation

We'll split the data string `"Hello, MinIO!"` into 8 parts (one part for each data block):

- **Data Block 1**: `"H"`
- **Data Block 2**: `"e"`
- **Data Block 3**: `"l"`
- **Data Block 4**: `"l"`
- **Data Block 5**: `"o"`
- **Data Block 6**: `","`
- **Data Block 7**: `" "`
- **Data Block 8**: `"MinIO!"`

Now, we have 8 data blocks that will be distributed across 8 disks.

Step 2: Parity Block Creation (Erasure Coding)

MinIO uses **Reed-Solomon erasure coding** to generate parity blocks. Parity blocks are created by applying mathematical transformations (e.g., XOR operations) to the data blocks. These transformations allow the system to reconstruct missing data blocks in case of disk failures.

In this example, we won't get into the mathematical details of how the parity blocks are generated, but let's assume the system creates **8 parity blocks** like this:

- **Parity Block 1**: Generated from Data Blocks 1-8
- **Parity Block 2**: Generated from Data Blocks 1-8 (with different encoding)
- ...
- **Parity Block 8**: Generated from Data Blocks 1-8 (with different encoding)

Step 3: Distribution Across Disks

MinIO will distribute the **8 data blocks** and **8 parity blocks** across the **16 disks**. Each disk stores **one block** (either a data block or a parity block). The distribution could look like this:

Disk	Block Type	Data/Parity Block
Disk 1	Data Block	`"H"`
Disk 2	Data Block	`"e"`
Disk 3	Data Block	`"l"`
Disk 4	Data Block	`"l"`
Disk 5	Data Block	`"o"`
Disk 6	Data Block	`","`
Disk 7	Data Block	`" "`
Disk 8	Data Block	`"MinIO!"`
Disk 9	Parity Block	Parity 1
Disk 10	Parity Block	Parity 2
Disk 11	Parity Block	Parity 3
Disk 12	Parity Block	Parity 4
Disk 13	Parity Block	Parity 5
Disk 14	Parity Block	Parity 6
Disk 15	Parity Block	Parity 7

Disk 16	Parity Block	Parity 8

Step 4: Simulating Disk Failures

Let’s simulate **3 disk failures** (Disk 2, Disk 7, and Disk 14 fail).

- **Disk 2** (Data Block `"e"`) is lost.
- **Disk 7** (Data Block `"  "`) is lost.
- **Disk 14** (Parity Block 6) is lost.

Step 5: Data Recovery

MinIO will use the remaining **5 data blocks** and the available **7 parity blocks** to reconstruct the lost data. Here's how the recovery works:

1. **Reconstruct Data Block `"e"` (Disk 2)**: MinIO uses the remaining data blocks (`"H"`, `"l"`, `"l"`, `"o"`, `","`, `"MinIO!"`) and the parity blocks (Parity 1, 2, 3, 4, 5, 7, 8) to reconstruct the missing Data Block `"e"`. This is possible because each parity block encodes information about the entire data set, allowing MinIO to restore the lost data.
2. **Reconstruct Data Block `"  "` (Disk 7)**: Similarly, MinIO will reconstruct the missing Data Block `"  "` using the available data and parity blocks.
3. **Parity Block (Disk 14)**: Parity blocks are used to reconstruct data blocks, but if we lose a parity block itself (as in the case of Disk 14), it doesn’t affect the ability to recover data. MinIO can continue to function without that particular parity block as long as sufficient data and other parity blocks are available.

Thus, even after losing 3 disks (2 data and 1 parity), MinIO can **fully recover** all the data using the parity blocks.

Step 6: What Happens if More Disks Fail?

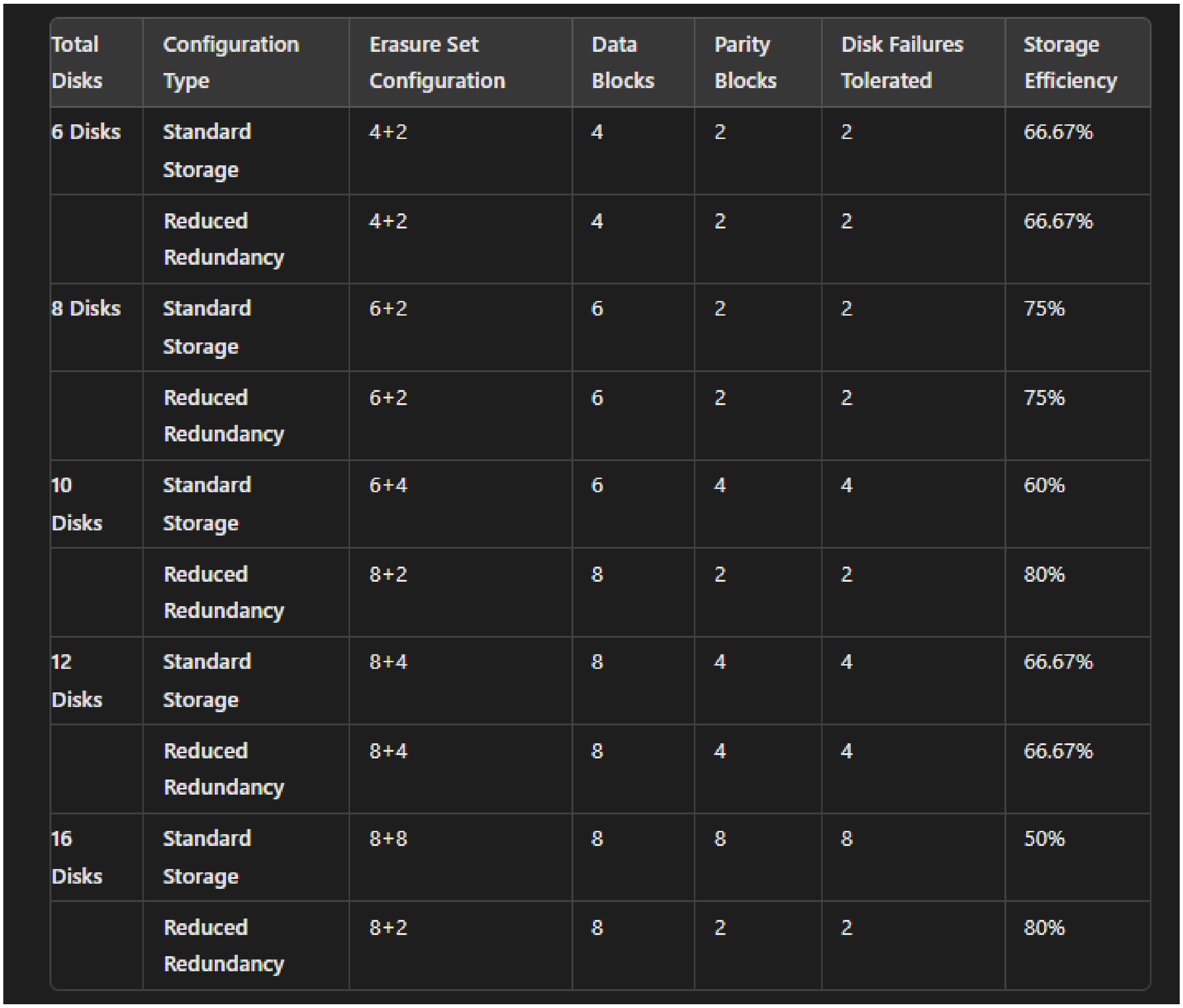
If **more than 8 disks** fail in an **8+8 setup**, MinIO won't have enough parity blocks to reconstruct the lost data. The system can tolerate up to **8 disk failures** in an **8+8 erasure set**, but any more than that would result in **data loss**.

Summary:

- In an **8+8 erasure set**, MinIO splits the data across **8 data blocks** and generates **8 parity blocks**.
- These blocks are distributed across the **16 disks**, with each disk holding one block.

- **Parity blocks** allow MinIO to recover lost data if up to **8 disks** fail.
- Even after multiple disk failures (up to 8), MinIO can **reconstruct the missing data** using the parity information, ensuring **data availability** and **fault tolerance**

Reduced Redundancy versus Standard storage



Total Disks	Configuration Type	Erasure Set Configuration	Data Blocks	Parity Blocks	Disk Failures Tolerated	Storage Efficiency
6 Disks	Standard Storage	4+2	4	2	2	66.67%
	Reduced Redundancy	4+2	4	2	2	66.67%
8 Disks	Standard Storage	6+2	6	2	2	75%
	Reduced Redundancy	6+2	6	2	2	75%
10 Disks	Standard Storage	6+4	6	4	4	60%
	Reduced Redundancy	8+2	8	2	2	80%
12 Disks	Standard Storage	8+4	8	4	4	66.67%
	Reduced Redundancy	8+4	8	4	4	66.67%
16 Disks	Standard Storage	8+8	8	8	8	50%
	Reduced Redundancy	8+2	8	2	2	80%

Standard Storage: Best for **critical data** where **high fault tolerance** is important.

Reduced Redundancy: Best for **non-critical data** where **storage efficiency** is prioritized over fault tolerance.

Custom Settings

MINIO_STORAGE_CLASS_STANDARD="EC:2"

When you specify `EC:2`, MinIO will automatically decide how many data blocks to use based on the total number of disks in the erasure set.

Example Scenarios:

- If you have **6 disks**:
  - With `EC:2`, MinIO will use **4 data blocks** and **2 parity blocks** (4+2 erasure set).
  - This means it can tolerate up to **2 disk failures**.
- If you have **8 disks**:
  - With `EC:2`, MinIO will use **6 data blocks** and **2 parity blocks** (6+2 erasure set).
  - This means it can also tolerate up to **2 disk failures**, but more of the disks are used for data storage.

MINIO_STORAGE_CLASS_STANDARD="EC:4"

When you specify `EC:4`, MinIO will automatically decide how many data blocks to use based on the total number of disks in the erasure set.

1. 6 Disks:

- **Erasure Set**: 2 data blocks + 4 parity blocks (2+4).
- **Disk Failures Tolerated**: 4 disks.
- **Storage Efficiency**: Only **33.33%** of the total disk space is used for data because 4 out of 6 disks are reserved for parity.

2. 8 Disks:

- **Erasure Set**: 4 data blocks + 4 parity blocks (4+4).
- **Disk Failures Tolerated**: 4 disks.
- **Storage Efficiency**: **50%** of the total disk space is used for data since half of the disks are used for parity.

3. 10 Disks:

- **Erasure Set**: 6 data blocks + 4 parity blocks (6+4).
- **Disk Failures Tolerated**: 4 disks.

- **Storage Efficiency**: **60%** of the total disk space is used for data, and 4 disks are used for parity.