AONT-RS: Blending Security and Performance in Dispersed Storage Systems

ENGG5105/CSCI5470 Computer and Network Security

Spring 2014

Patrick P. C. Lee

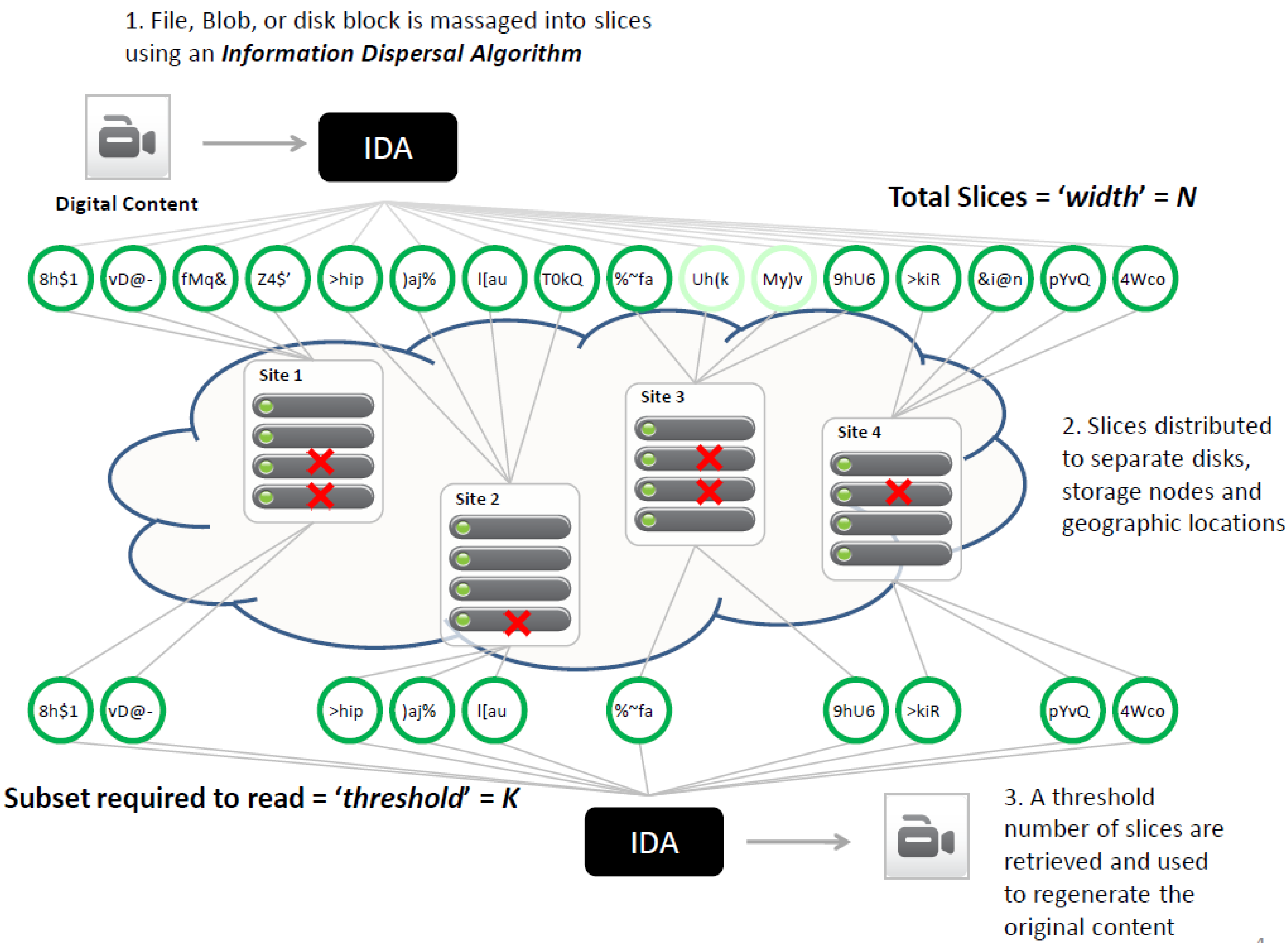
Slides are adapted from the paper presentation at FAST'11
https://www.usenix.org/conference/fast11/aont-rs-blending-security-and-performance-dispersed-storage-systems

Outline

- Appeals of Dispersed Storage
- Methods for Securing Dispersed Data
- A new approach: AONT-RS
- Results on a production system

What is Dispersed Storage?

- Definition:
  - Computationally massaging data into related pieces and storing them to separate locations
- Data resiliency is usually achieved through forward error correction (erasure codes)
- Provides a ***K-of-N*** fault tolerance


1. File, Blob, or disk block is massaged into slices
using an *Information Dispersal Algorithm*
Digital Content
IDA
Total Slices = 'width' = N
8h$1
vD@-
fMq&
Z4$'
>hip
)aj%
l[au
T0kQ
%~fa
Uh(k
My)v
9hU6
>kiR
&i@n
pYvQ
4Wco
Site 1
Site 2
Site 3
Site 4
2. Slices distributed
to separate disks,
storage nodes and
geographic locations
Subset required to read = 'threshold' = K
IDA
3. A threshold
number of slices are
retrieved and used
to regenerate the
original content

Benefits of Dispersing Data

- Data is highly reliable
  - Configurable tolerance for drive, node and site failure
  - Distribution reduces risk of correlated failures
- Data can be efficiently stored
  - Allows for disaster recovery without replication
  - Raw storage requirements often less than 2 copies
- Can also provide a high degree of security

How do I Store Data Securely?

- Usual answer: <u>Encrypt it!</u>

- After encrypting, one has to protect a key
  - How does one store the key privately and reliably?
  - If a key is lost, so is the data that it protects
  - Increasing reliability or availability through replication opens additional vectors for attack and exposure

- In 1979, Adi Shamir and George Blakely independently discovered a better way

Shamir's Secret Sharing

- A secret is divided into **N** shares
  - Any threshold (**K**) number of shares yields the secret
  - Nothing is learned about the secret with < **K** shares
  - (**K**, **N**) threshold scheme
- Allows a high degree of privacy and reliability
  - Exposing the secret requires multiple breaches
  - Shares can be unavailable yet recovery is still possible
- Encryption can be considered as a special case of secret sharing, where **N**= **K**= 2
  - Why?

Drawbacks of Secret Sharing

- For Shamir's scheme, storage and bandwidth requirements are multiplied by **N**
  - E.g., 5 shares for 1 TB of data requires 5 TB raw
- Encoding time per byte grows with **N** · **K**
  - Encoding for 3-of-5 is 10X faster than a 10-of-15
- These forms of secret sharing are unsuitable for performance-or cost-sensitive bulk data storage.

Information Dispersal Algorithm (IDA)

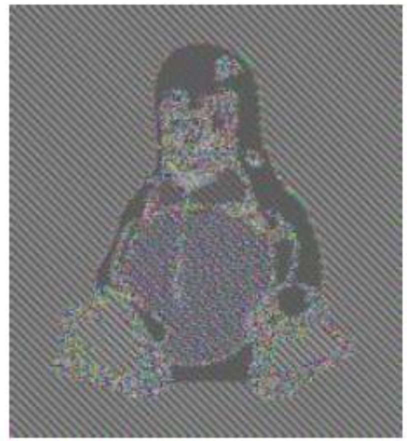
- Proposed by Michael O. Rabin in 1989 as a method to achieve efficiency, security, load balancing and fault tolerance

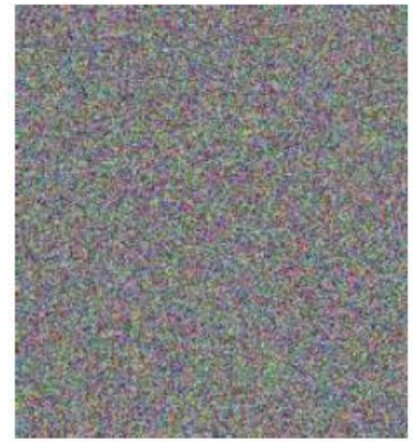
- Raw storage requirements are: (**N**/ **K**) ·Input Size
  - Very efficient since (**N**/ **K**) may be chosen close to 1

- Security of Rabin is not as strong as Shamir
  - Having fewer than **K** shares yields some information
  - Repetitions in input create repetitions in output

Rabin IDA Security Example


Input: a BMP file


Rabin IDA Output


True Security

- ➢ This occurs when the generator matrix is constant
  - Rabin suggested that it could be chosen randomly
  - The problem becomes storing the random matrices:
    - Each matrix is **N** times larger than the input processed per matrix

Secret Sharing made Short

- In 1993, Hugo Krawczyk combined elements of Shamir's Secret Sharing with Rabin's IDA
- The SSMS method:
  - Input is encrypted with a random encryption key
  - Encrypted result is dispersed using Rabin's IDA
  - Random key is dispersed using Shamir's Secret Sharing
- Yields a *computationally secure* secret sharing scheme with good security and efficiency

AONT-RS

- AONT-RS was developed at Cleversafe in 2007
  - Combines Ron Rivest’s All-or-Nothing Transform with Systematic Reed-Solomon encoding
  - Yields a computationally secure secret sharing scheme
- Security and efficiency properties are similar to Secret Sharing made Short, but:
  - Encoding is faster
  - Integrity is protected
  - Output is shorter
  - Rebuilding is simpler

All-or-Nothing Transform

- An unkeyed random transformation that is difficult to invert without all of the output
  - When one has all the output, reversing the transformation is trivial
  - First described by Ron Rivest in 1997

All-or-Nothing Transform

- *(s+1, s+1)* threshold scheme
- Inputs:
  - Data: s words $d_0, d_1, \ldots, d_{s-1}$
  - Random Key: *key*
- Output:
  - Codeword $c_0, c_1, \ldots, c_s$, such that
    - $c_i = d_i \oplus Enc(key, i+1)$, where $0 \le i \le s-1$
      - Enc is an encryption function (e.g., AES)
      - Similar to counter mode (CTR) (see Lecture 1)
    - $c_s = key \oplus hash(c_0, c_1, \ldots, c_{s-1})$
- How to do decryption? Why is it all or nothing?

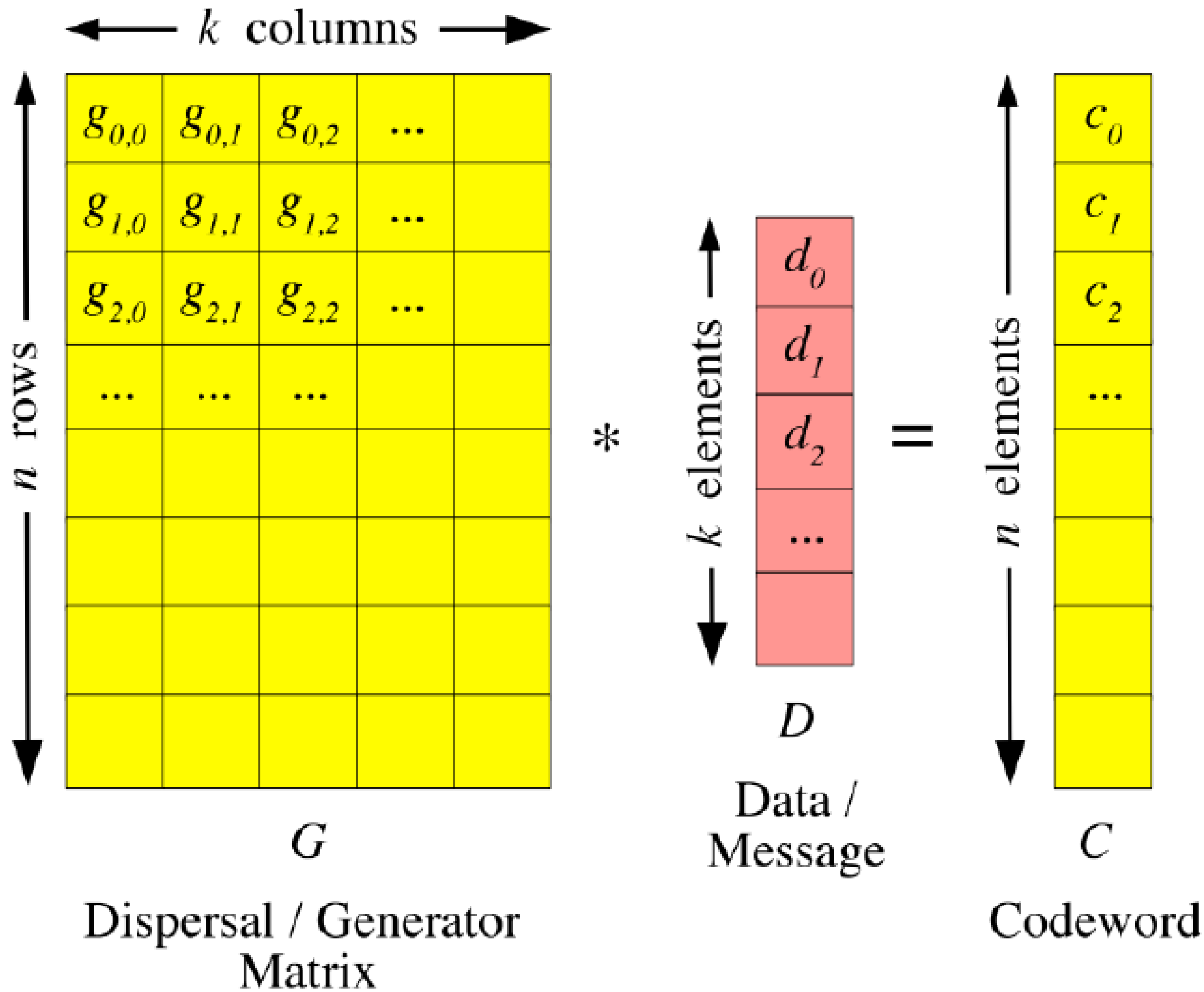
Enhancements to AONT

- AONT-RS enhances AONT in two ways

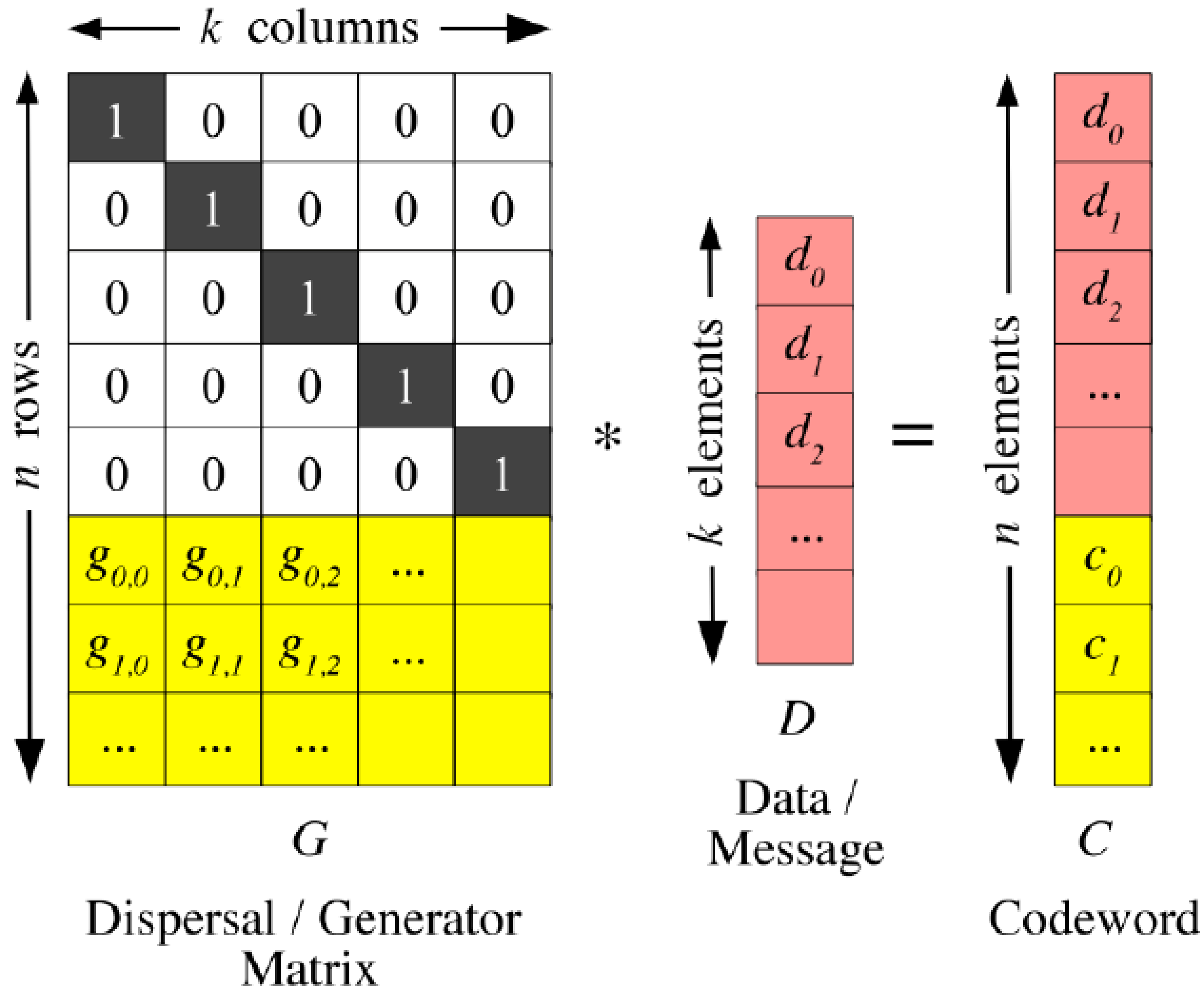
- Add an extra word d_s called **canary**
  - Known, fixed value, for integrity checking

- Employ systematic erasure codes

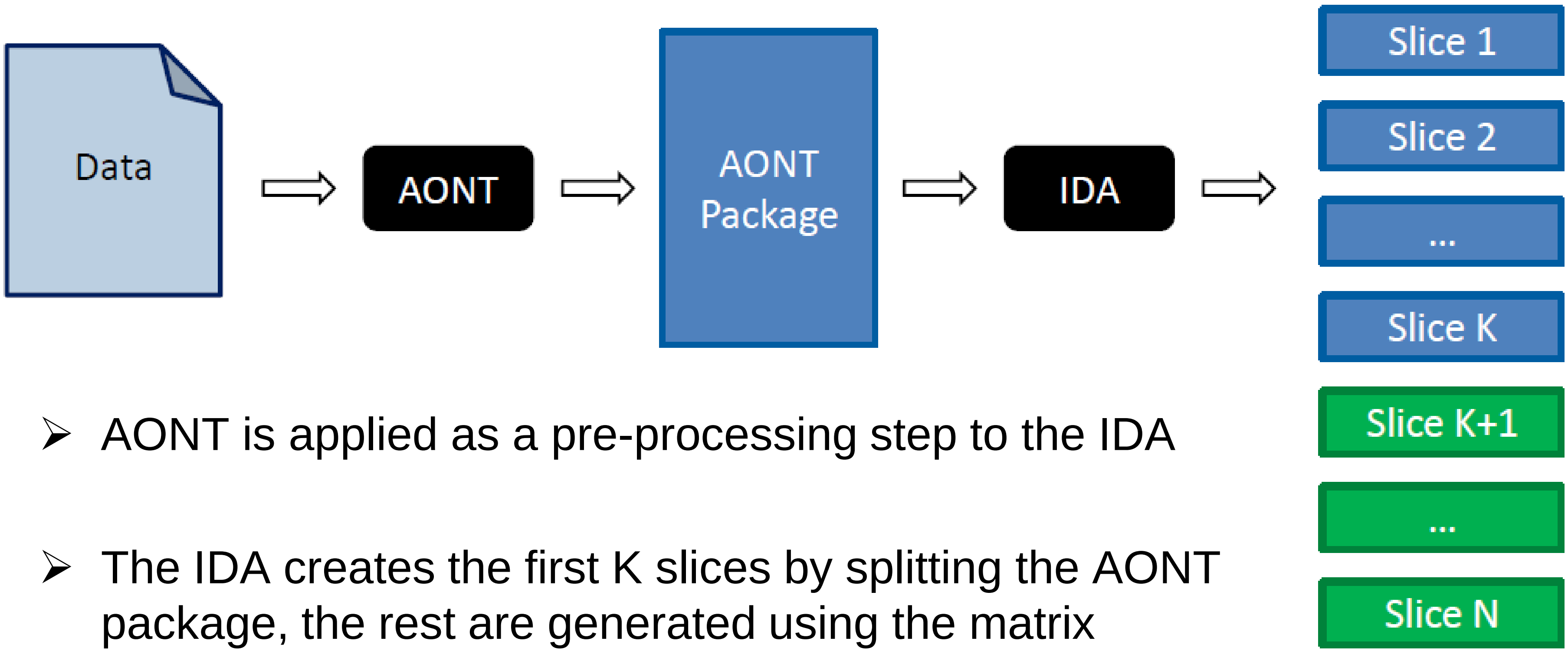
Non-systematic Erasure Codes



Systematic Erasure Codes

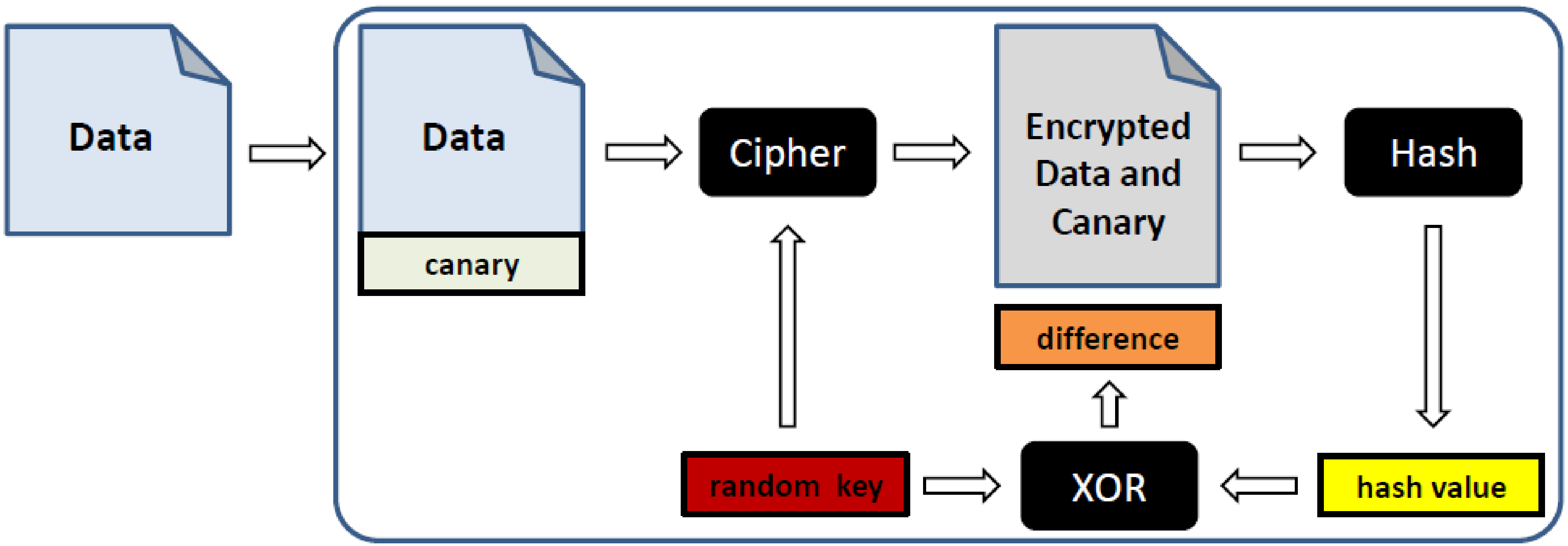


Encoding Data with AONT-RS

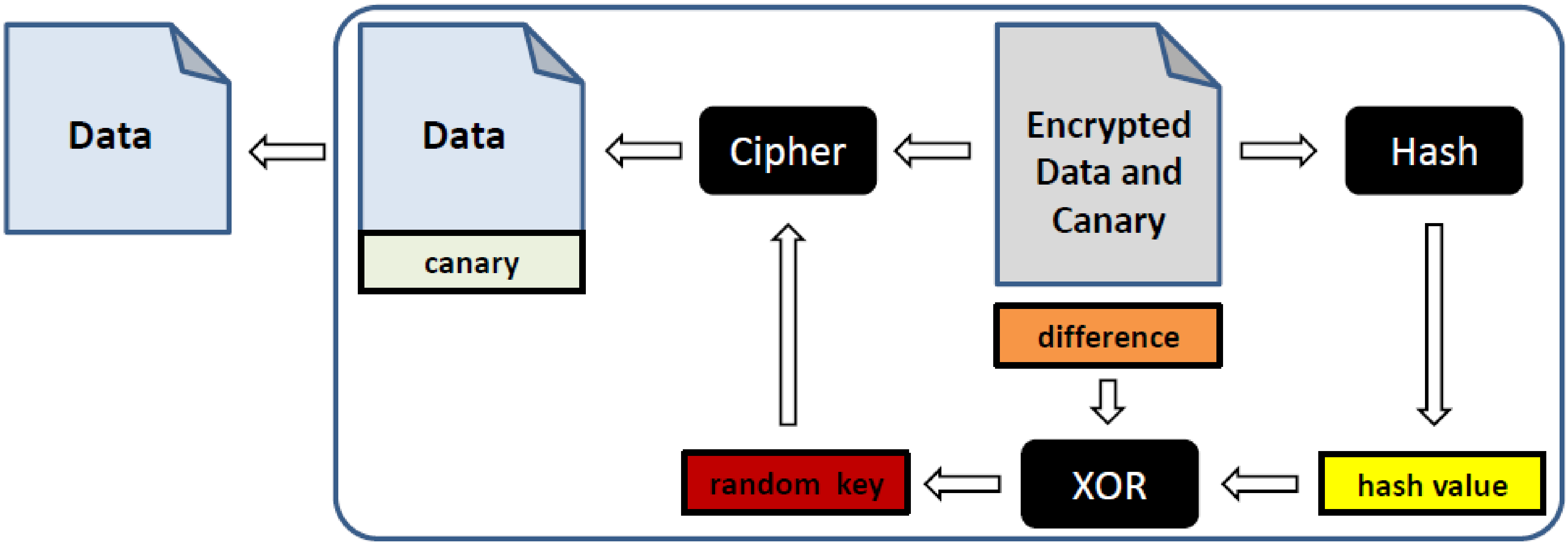


- AONT is applied as a pre-processing step to the IDA
- The IDA creates the first K slices by splitting the AONT package, the rest are generated using the matrix
- Without a threshold number of slices there is not enough information to recreate the AONT package

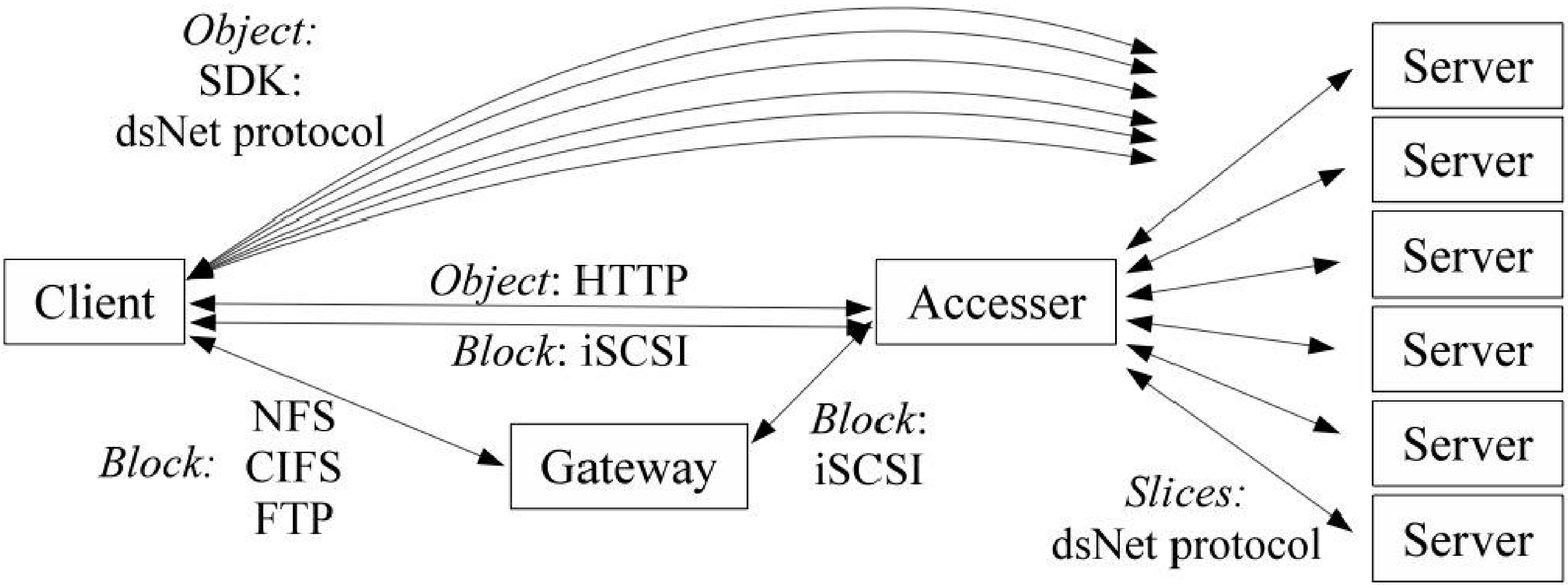
Encoding with AONT



Decoding with AONT



Cleversafe Architecture

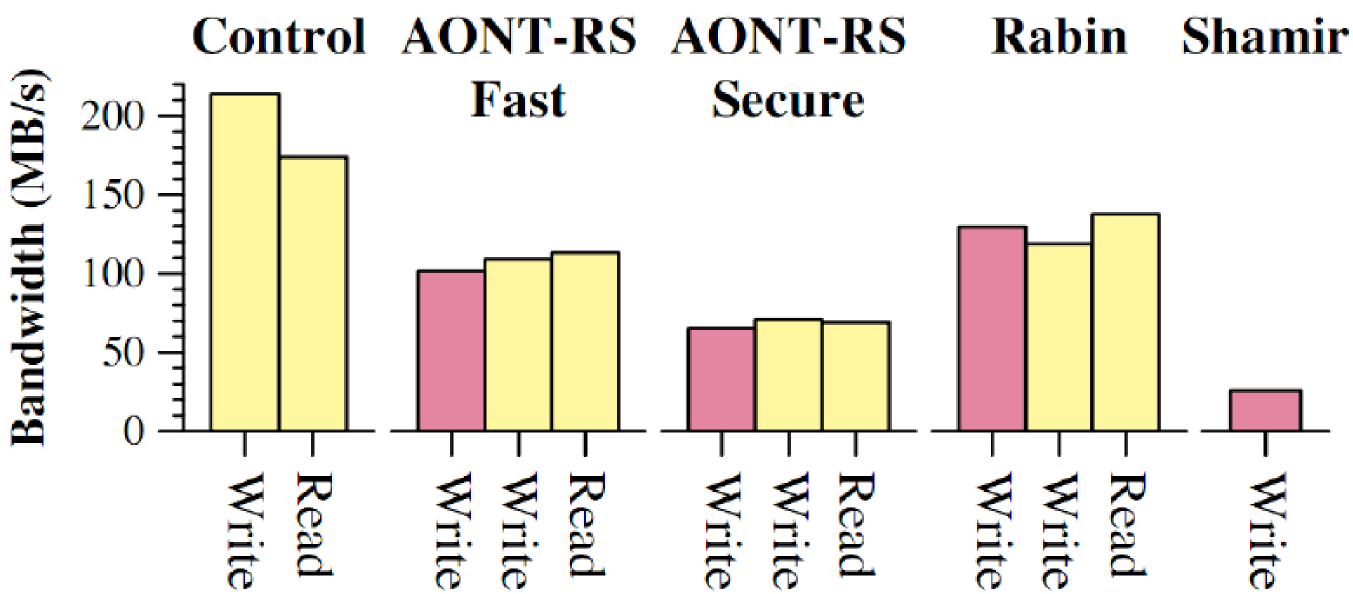


Experiments

- Implementation in Java
- Setup:
  - Tested on Cleversafe's production hardware
  - Consisted of 1 or 2 clients writing to 8 servers
  - Clients had 10 Gbps NICs, servers had 1 Gbps NICs. Bottleneck was CPU
- Schemes:

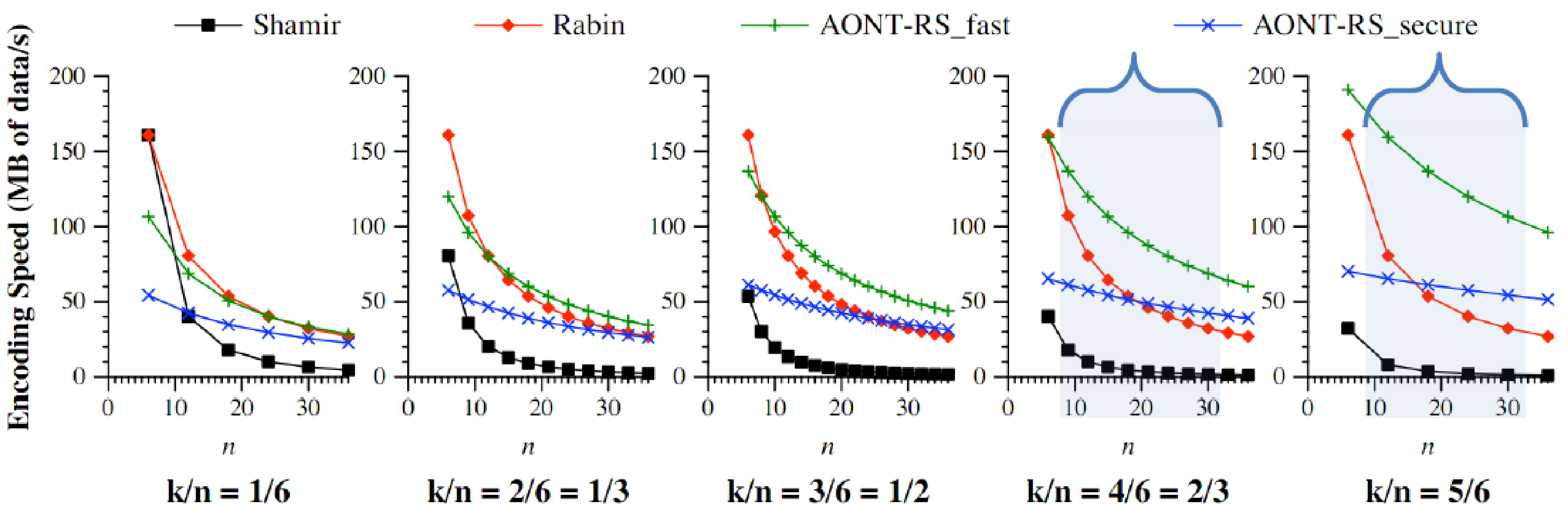
Control 8-of-8	send 8 slices without encoding
AONT-RS fast	AONT-RS using RC4-128 and MD5
AONT-RS secure	AONT-RS using AES-256 and SHA-256
Rabin IDA	IDA protocol

Experimental Results



Algorithm	Write Speed (MB/s)	Read Speed (MB/s)
Control 8-of-8:	214.24	174.31
AONT-RS fast:	109.18	113.38
AONT-RS secure:	70.84	69.18
Rabin IDA:	118.79	137.83

Encoding Speed

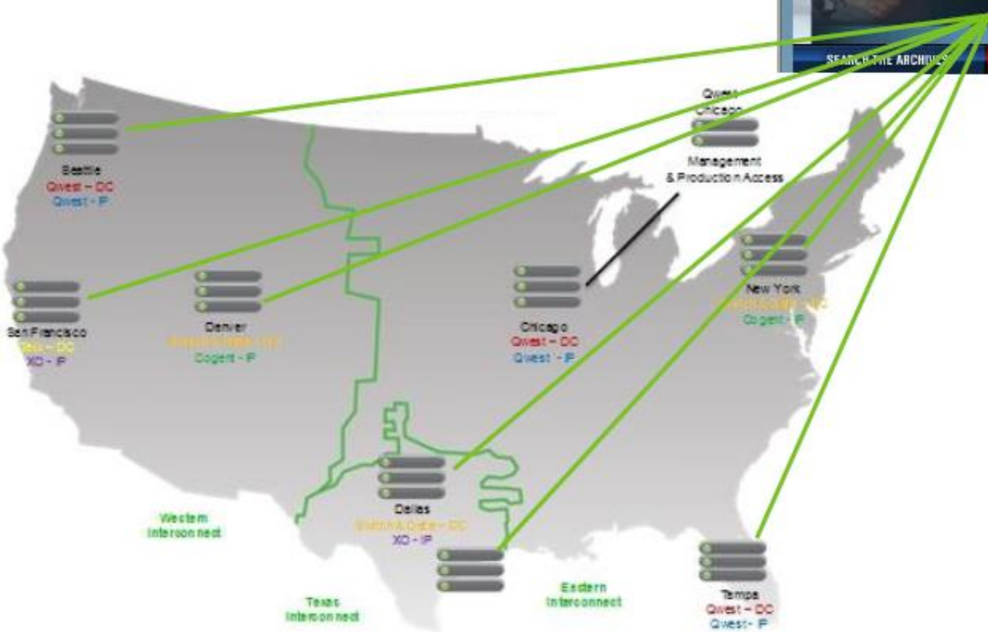


- Typical configurations in deployment:
  - K/ N close to 1 (for higher efficiency)
  - N between 10 and 30
- AONT-RS outperforms Rabin for large n due to systematic nature

Example Deployment

- **Museum of Broadcast Communications**
  - 100,000 hours of historic TV and radio content
  - 50,000 registered users
  - 2.6 million annual visitors

www.museum.tv



- **Deployment details:**
  - 8 sites across US
  - 3 power grids
  - 10-of-16 configuration
  - 40 TB usable, 64 TB raw

Conclusions

- Dispersal offers many benefits for storage:
  - Reliability, efficiency, scalability, and performance
- Dispersal may provide security without the need for a separate key management system
- Presented a new dispersal algorithm with an attractive blend of performance and security
  - Evaluated its theoretical and actual performance
- Described a system in use, relying on this algorithm