SDC
STORAGE DEVELOPER CONFERENCE
SNIA SANTA CLARA, 2014

Next Generation Erasure Coding Techniques

Wesley Leggette
Cleversafe

Topics

- What is Erasure Coded Storage?
- The evolution of Erasure Coded storage
  - From first- to third-generation erasure coding
- Limitations of the current state-of-the-art
- Next (4th) Generation Erasure Coding
  - What it is and how it works
- Conclusions

What is Erasure Coded Storage?

- Erasure codes convert input data into **N** outputs where any **K** ≤ **N** outputs can recover the data
  - Example: in an 8+2 RAID 6 array, a block is converted into 10 outputs (each stored to a different drive), such that any 8 drives contain sufficient information to recover the block
    - K = 8, N = 10

Why use Erasure Codes?

- Unlike replication, erasure codes allow greater fault-tolerance with improved efficiency:

Method	Fault Tolerance	Storage Efficiency
Two Copies	1	50%
Triple Replication	2	33%
RAID 6 (8+2)	2	80%
16-of-20 erasure code	4	80%
30-of-36 erasure code	6	83%
K-of-N erasure code	(N – K)	(K / N)

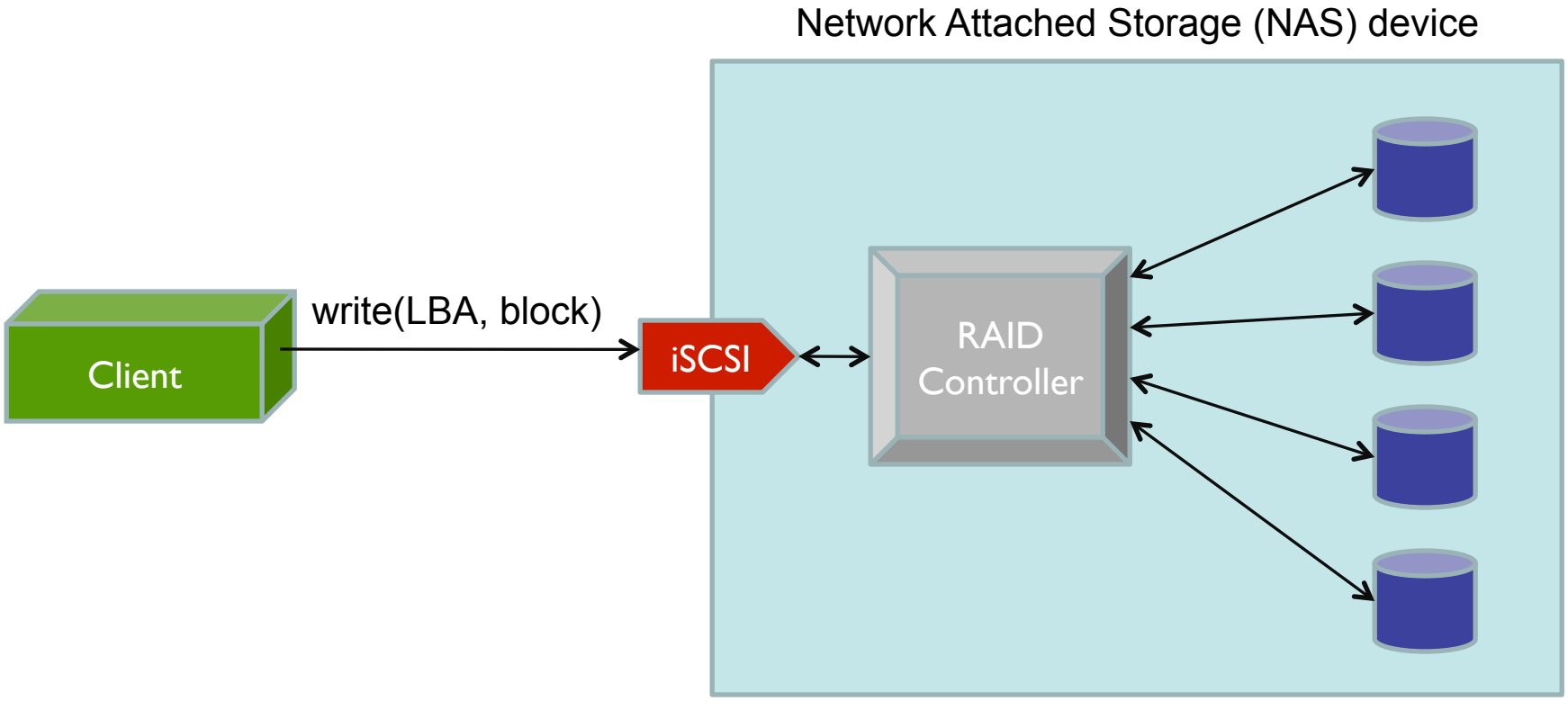
The Evolution of Erasure Coded Storage

- **1st Generation:** RAID 5 / 6
  - Erasure coded data across an array of drives
- **2nd Generation:** RAIN
  - Erasure coded data across networked nodes
- **3rd Generation:** Global Namespace
  - Erasure coded data mapped deterministically with no need to consult a metadata system

1st Generation: RAID 5 / 6

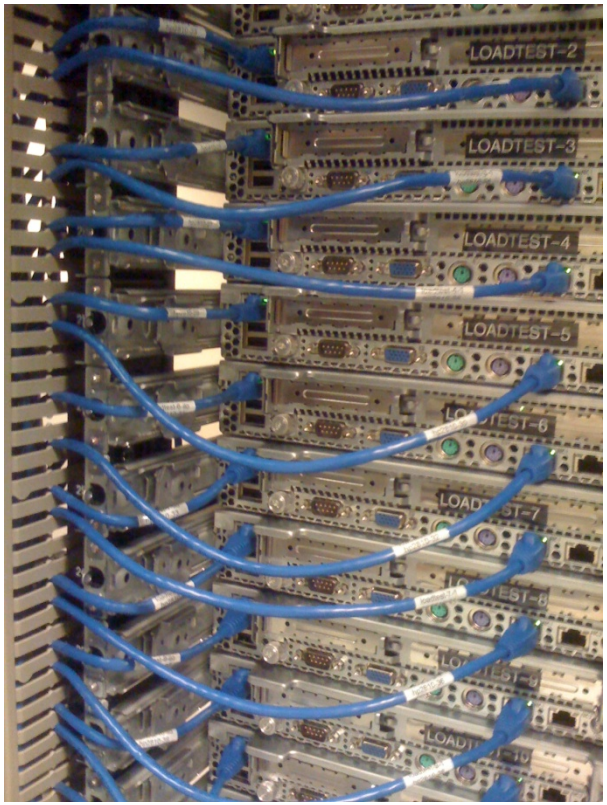
- Advantages:
  - Achieves redundancy without replication
  - Low overhead
- Limitations:
  - Availability limited to availability of a node
  - Tolerates 1-2 failures
  - File system corruption can lead to data loss

RAID 5 / 6 – under the hood

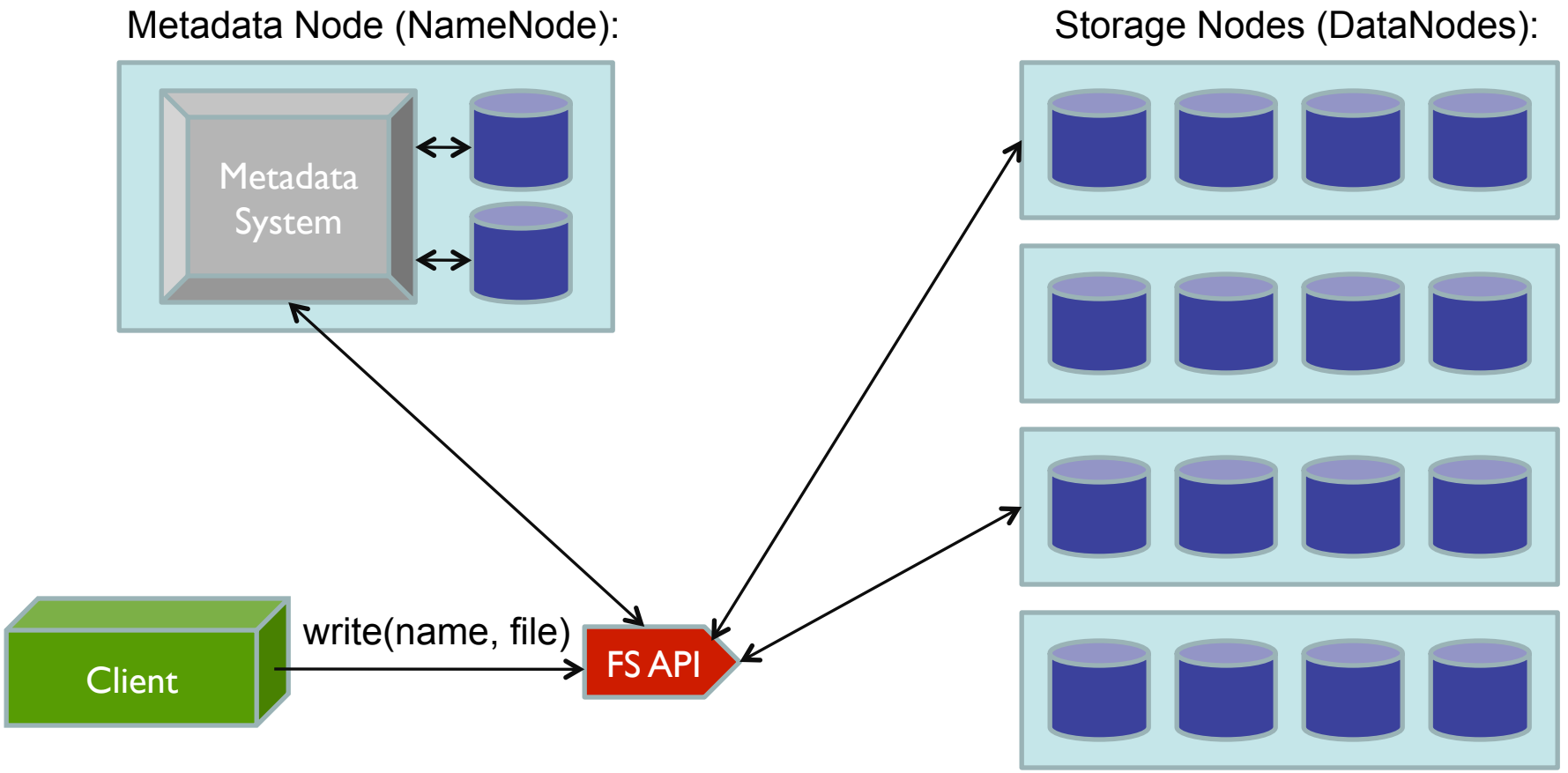


2nd Generation: RAIN

- Advantages:
  - Available despite individual node failures
  - Disaster recovery without replication
- Limitations:
  - Central metadata system required to store or locate data
  - Harder to scale



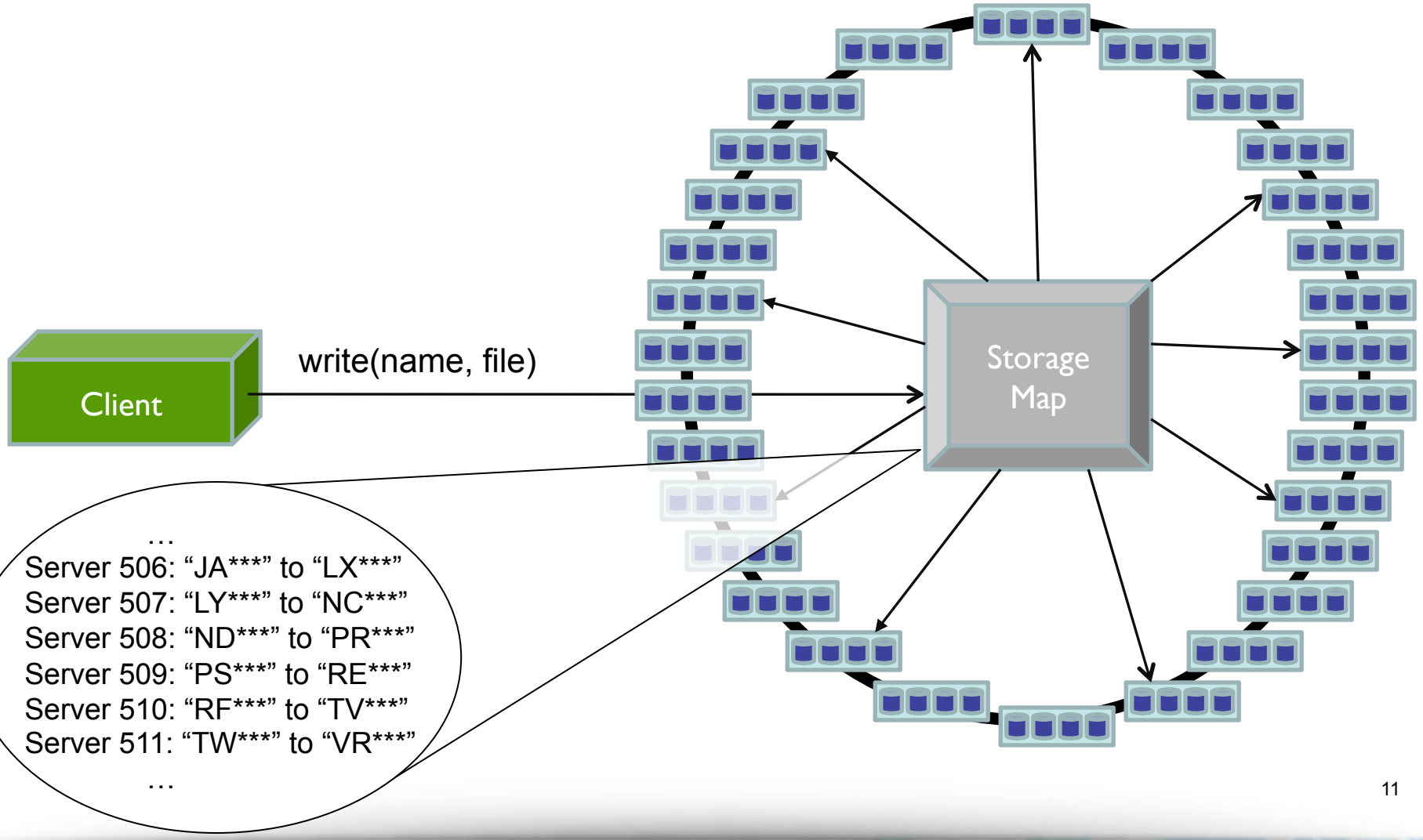
RAIN – under the hood



3rd Generation: Global Namespace

- Advantages:
  - No SPOF
  - Unlimited scalability
- Limitations:
  - Restricted options when storing data
  - Redundancy is over-provisioned to handle node/site outages

Global Namespace – under the hood

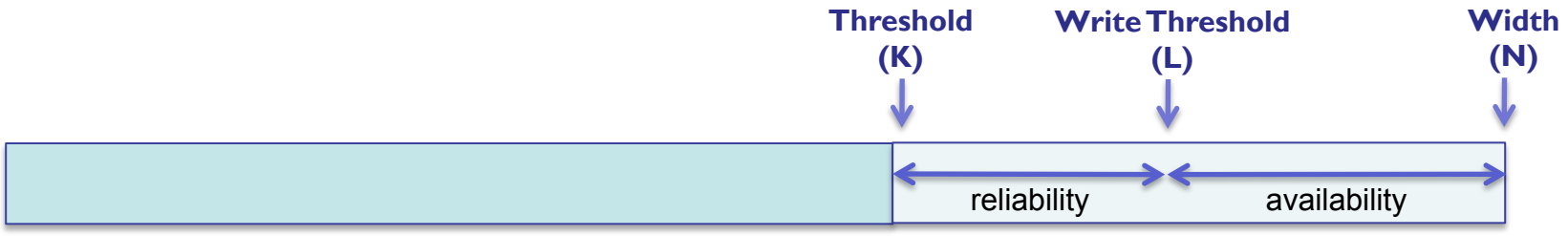


Overcoming limits of a Global Namespace

- Unlike systems with a metadata system:
  - Namespace is deterministic and inflexible
  - Can't adapt to node failures, site outages, performance degradations dynamically
- To improve availability in face of problems:
  - Define **L**, such that **K** ≤ **L** ≤ **N**
  - Operation is successful if **L** outputs are stored

What a Write Threshold (L) provides

- Write thresholds trade reliability for availability:
  - System remains available for writing new data when there are no more than (**N** – **L**) outages
  - System remains free from data loss so long as there are no more than (**L** – **K**) failures



Problems with Write Thresholds

- **N** is increased to handle availability outages
  - Decreases storage efficiency
  - Increases CPU cost of erasure code function
- When there are slow nodes (very common)
  - decrease performance | not store all outputs
- But, when not all outputs are written..
  - Reliability suffers and rebuilding is required

Example situation

- Assume storage system must tolerate 5 node outages and 4 drive failures:
  - If K = 20, then L = K+4, and N = L + 5:
    - K = 20, L = 24, N = 29
    - Storage efficiency: 20/29 = 68.96%
- But what if we could *always* write 24 outputs?
  - K = 20, L=24, N = 24
    - Storage efficiency: 20/24 = 83.33%

4th Generation: Adaptive Placement

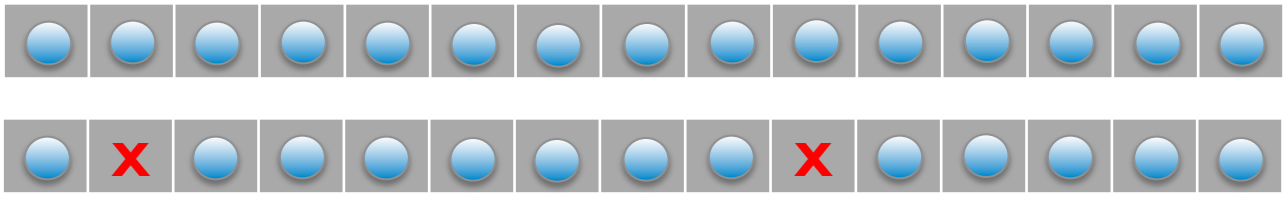
- 4th generation erasure coding combines the scalability of a Global Namespace with the flexibility provided by a metadata system:
  - System can adapt where it stores outputs while retaining the ability to locate them
  - Tolerates performance and availability outages, but not at the expense of efficiency

What's different?

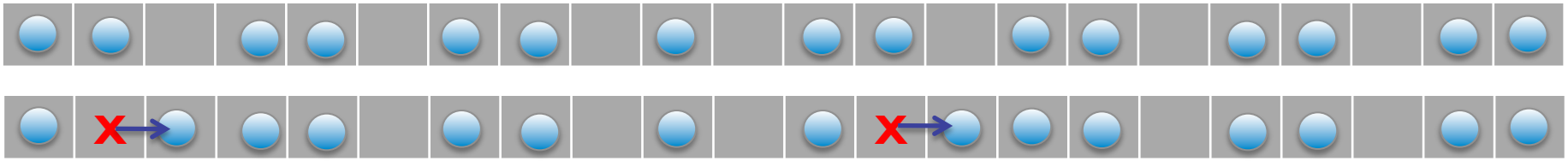
- In 3rd gen storage, there is an assumption that the number of storage locations (*slots*) equals the number of outputs from the erasure code:
  - Slots = Width (**N**)
- The benefits of 4th generation storage follow from breaking this equality, in allowing:
  - Slots ≥ Width (**N**)

Benefits of Extra Slots

10-of-15 (15 slots):



10-of-15 (22 slots):



- In a 10-of-15 configuration across 15 slots, an unavailable slot leads to an unwritten output
- In the same system across 22 slots, up to 7 slots can fail without any impact to reliability

Store Affinity

- Ideally, we would not have to read more than **K** outputs when performing a read operation
  - The erasure code only needs **K** of them
  - Reading extras wastes disk and network IO
- To fix this requires the concept of *Store Affinity*:
  - For any give file, **N** slots are designated as **primary**; the others are made **secondary**
  - When writing, always prefer the primary stores, but permit falling back to secondary

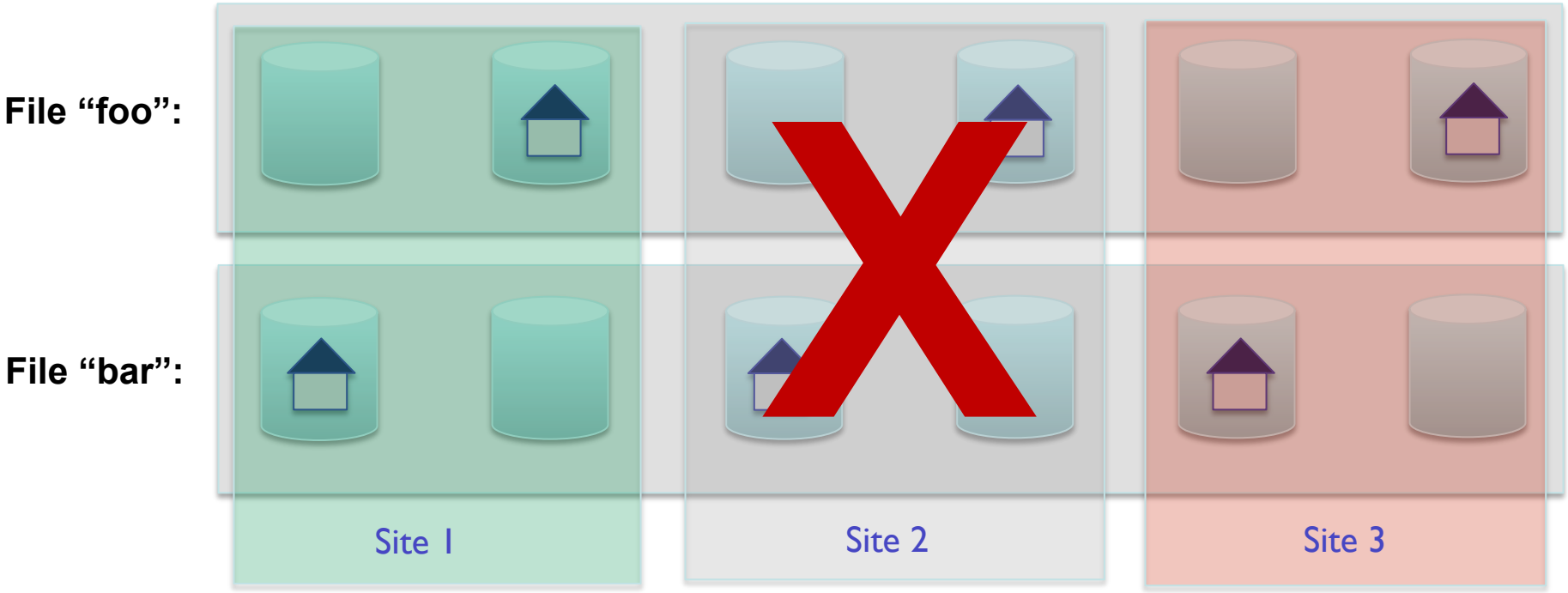
Efficient Reading with Store Affinity

- When reading, select any **K** primary locations to issue read requests to, then issue reads to all of the secondary locations:
  - For every primary slot that does not have an output, a secondary slot will
  - Therefore, no more than **K** outputs are read

Rotating Store Affinity Example

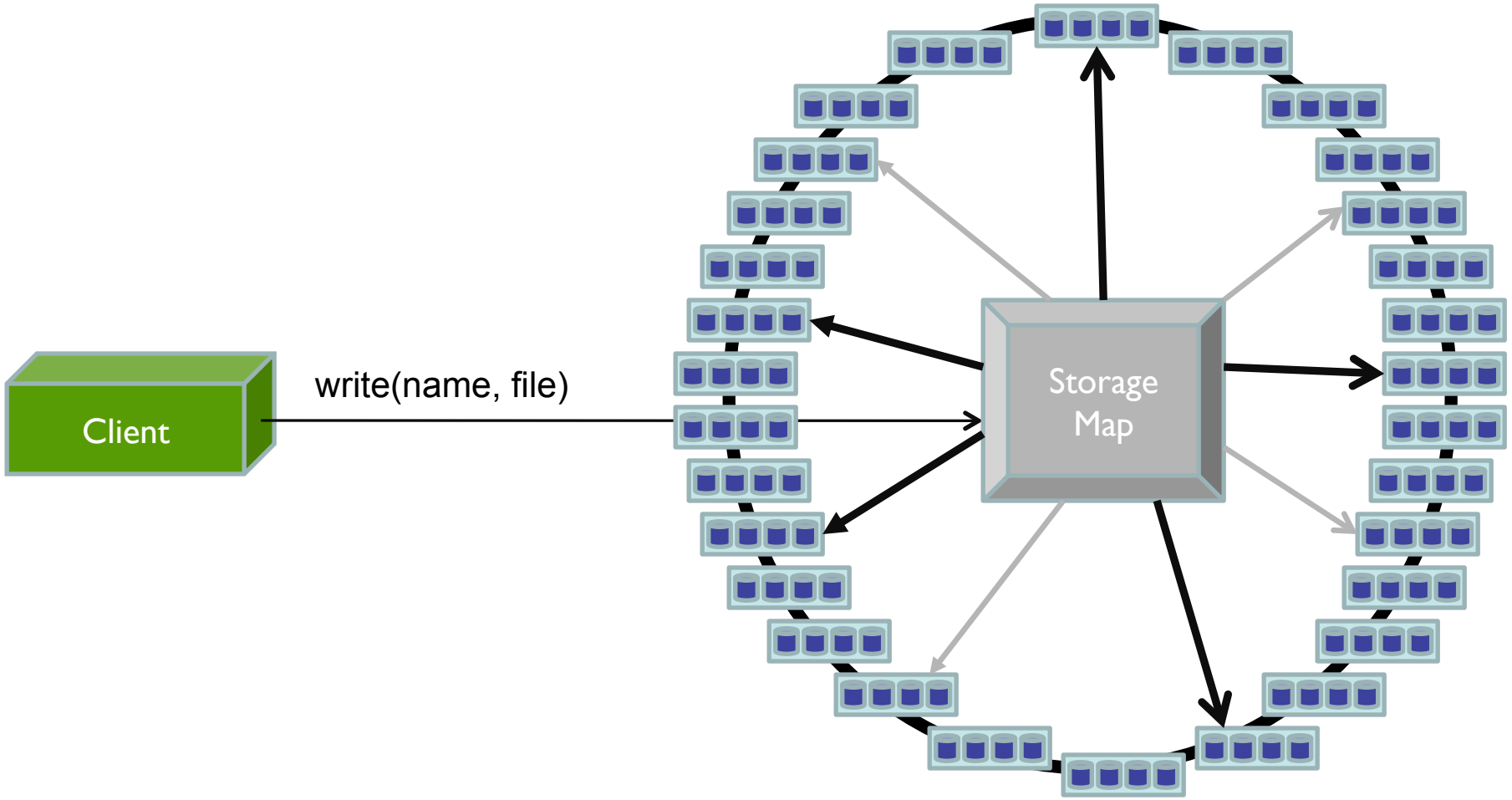
- To equalize utilization, the set of primary slots should rotate deterministically (e.g. by filename)
  - Deterministic function yields which N of the slot-number of stores are primary

Affinity and rotation strategies



- The number of slots and rotation strategy for the affinity algorithm are important to guarantee that reads are supported when a site is down

Adaptive Placement – under the hood



Advantages of Adaptive Placement

- There are many benefits to this approach
  - Rebuilding: No rebuilding due to outages
  - Reliability: Avoids a “worst-case” reliability
  - Availability: Tolerates (*slots* – **N**) failures
  - Performance: Less work for erasure code
  - Optimizations: May select fastest **N** nodes
  - Efficiency: **N** = **L**, so **K**/**N** can be closer to 1

Observations

- Readers and Writers may use deterministic functions to reach agreement on data placement
- Given Adaptive Placement, there is little benefit (and a lot of downsides) to a metadata system
- Erasure coded storage continues to evolve, what might 5th generation systems look like?

Questions & Answers

Wesley Leggette
wleggette@cleversafe.com